GROW DATA SKILLS

Q.1- What are the main components of a Hadoop Application?
Over time, there are various forms in which a Hadoop application is defined. But in most of the cases there are following four core components of Hadoop application:

- HDFS: This is the file system in which Hadoop data is stored. It is a distributed file system with very high bandwidth.
- Hadoop Common_: This is a common set of libraries and utilities used by Hadoop.
- Hadoop MapReduce: This is based on the MapReduce algorithm for providing large-scale data processing.
- Hadoop YARN: This is used for resource management in a Hadoop cluster. It can also schedule tasks for users.

Q.2- What is the core concept behind Apache Hadoop framework?
Apache Hadoop is based on the concept of MapReduce algorithm. In the MapReduce algorithm, Map and Reduce operations are used to process very large data sets.
In this concept, the Map method does the filtering and sorting of data. Reduce method performs the summarising of data.
This is a concept from functional programming.
The key points in this concept are scalability and fault tolerance. In Apache Hadoop these features are achieved by multi-threading and efficient implementation of MapReduce.

Q.3- What is Hadoop Streaming?
Hadoop distribution provides a Java utility called Hadoop Streaming. It is packaged in a jar file. With Hadoop Streaming, we can create and run Map Reduce jobs with an executable script.
We can create executable scripts for Mapper and Reducer functions. These executable scripts are passed to Hadoop Streaming in a command.
Hadoop Streaming utility creates Map and Reduce jobs and submits these to a cluster. We can also monitor these jobs with this utility.

Q.4- What is the difference between Nodes in HDFS?
The differences between NameNode, BackupNode and Checkpoint NameNode are as follows:

- NameNode: NameNode is at the heart of the HDFS file system that manages the metadata i.e. the data of the files is not stored on the NameNode but rather it has the directory tree of all the files present in the HDFS file system on a Hadoop cluster. NameNode uses two files for the namespace:
  fsimage file: This file keeps track of the latest checkpoint of the namespace.
  edits file: This is a log of changes made to the namespace since checkpoint.
- Checkpoint Node: Checkpoint Node keeps track of the latest checkpoint in a directory that has the same structure as that of NameNode's directory.
  Checkpoint node creates checkpoints for the namespace at regular intervals by downloading the edits and fsimage file from the NameNode and merging it locally. The new image is then again updated back to the active NameNode.
- BackupNode: This node also provides check pointing functionality like that of the Checkpoint node but it also maintains its up-to-date in-memory copy of the file system namespace that is in sync with the active NameNode.

Grow Data Skills

Q.5- What is the optimum hardware configuration to run Apache Hadoop?
To run Apache Hadoop jobs, it is recommended to use dual core machines or dual processors. There should be 4GB or 8GB RAM with the processor with Error-correcting code (ECC) memory.
Without ECC memory, there is a high chance of getting checksum errors.
For storage high capacity SATA drives (around 7200 rpm) should be used in Hadoop clusters.
Around 10GB bandwidth Ethernet networks are good for Hadoop.

Q.6- What do you know about Block and Block scanners in HDFS?
A large file in HDFS is broken into multiple parts and each part is stored on a different Block. By default a Block is of 64 MB capacity in HDFS.
Block Scanner is a program that every Data node in HDFS runs periodically to verify the checksum of every block stored on the data node.
The purpose of a Block Scanner is to detect any data corruption errors on a Data node.

Q.7- What are the default port numbers on which Nodes run in Hadoop?
Default port numbers of Name Node, Job Tracker and Task Tracker are as follows:
NameNode runs on port 50070
Task Tracker runs on port 50060
Job Tracker runs on port 50030

Q.8- How will you disable a Block Scanner on HDFS DataNode?
In HDFS, there is a configuration dfs.datanode.scan.period.hours in hdfs-site.xml to set the number of hours interval at which Block Scanner should run.
We can set dfs.datanode.scan.period.hours=0 to disable the Block Scanner. It means it will not run on HDFS DataNode.

Q.9- How will you get the distance between two nodes in Apache Hadoop?
In Apache Hadoop we can use the NetworkTopology.getDistance() method to get the distance between two nodes.
Distance from a node to its parent is considered as 1.

Q.10- Why do we use commodity hardware in Hadoop?
Hadoop does not require a very high-end server with large memory and processing power. Due to this we can use any inexpensive system with average RAM and processor. Such a system is called commodity hardware.
Since there is parallel processing in Hadoop MapReduce, it is convenient to distribute a task among multiple servers and then do the execution. It saves cost as well as it is much faster compared to other options. Another benefit of using commodity hardware in Hadoop is scalability. Commodity hardware is readily available in the market. Whenever we need to scale up our operations in a Hadoop cluster we can obtain more commodity hardware. In the case of high-end machines, we have to raise purchase orders and get them built on demand.

Grow Data Skills

Q.11- How does inter cluster data copying work in Hadoop?
In Hadoop, there is a utility called DistCP (Distributed Copy) to perform large inter/intra-cluster copying of data. This utility is also based on MapReduce. It creates Map tasks for files given as input.
After every copy using DistCP, it is recommended to run cross checks to confirm that there is no data corruption and the copy is complete.

Q.12- How can we update a file at an arbitrary location in HDFS?
This is a trick question. In HDFS, it is not allowed to update a file at an arbitrary location. All the files are written in append only mode. It means all writes are done at the end of a file.
So there is no possibility of updating the files at any random location.

Q.13- What is the Replication factor in HDFS?
Replication factor in HDFS is the number of copies of a file in a file system. A Hadoop application can specify the number of replicas of a file it wants HDFS to maintain. This information is stored in NameNode. We can set the replication factor in following ways:
We can use Hadoop fs shell, to specify the replication factor for a file. Command as follows:

- $hadoop fs –setrep –w 5 /file_name

In the above command, the replication factor of file_name file is set as 5.
We can also use Hadoop fs shell, to specify the replication factor of all the files in a directory.

- $hadoop fs –setrep –w 2 /dir_name

In the above command, the replication factor of all the files under directory dir_name is set as 2.

Q.14- What is the difference between NAS and DAS in a Hadoop cluster?
NAS stands for Network Attached Storage and DAS stands for Direct Attached Storage.
In NAS, compute and storage layers are separated. Storage is distributed over different servers on a network.
In DAS, storage is attached to the node where computation takes place.
Apache Hadoop is based on the principle of moving processing near the location of data. So it needs a storage disk to be local to computation.
With DAS, we get very good performance on a Hadoop cluster. Also DAS can be implemented on commodity hardware. So it is more cost effective.
Only when we have very high bandwidth (around 10 GbE) it is preferable to use NAS storage.

Q.15- What are the two messages that NameNode receives from DataNode?
NameNode receives following two messages from every DataNode:

- Heartbeat: This message signals that DataNode is still alive. Periodic receipt of Heartbeat is very important for NameNode to decide whether to use a DataNode or not.
- Block Report: This is a list of all the data blocks hosted on a DataNode. This report is also very useful for the functioning of NameNode. With this report, NameNode gets information about what data is stored on a specific DataNode.

Grow Data Skills

Q.16- How does indexing work in Hadoop?
Indexing in Hadoop has two different levels.
Index based on File URI: In this case data is indexed based on different files. When we search for data, the index will return the files that contain the data.
Index based on InputSplit: In this case, data is indexed based on locations where input split is located.

Q.17- What data is stored in a HDFS NameNode?
NameNode is the central node of an HDFS system. It does not store any actual data on which MapReduce operations have to be done. But it has all the metadata about the data stored in HDFS DataNodes.
NameNode has the directory tree of all the files in the HDFS filesystem. Using this metadata it manages all the data stored in different DataNodes.

Q.18- What would happen if NameNode crashes in a HDFS cluster?
There is only one NameNode in a HDFS cluster. This node maintains metadata about DataNodes. Since there is only one NameNode, it is the single point of failure in a HDFS cluster. When NameNode crashes, the system may become unavailable.
We can specify a secondary NameNode in the HDFS cluster. The secondary NameNode takes the periodic checkpoints of the file system in HDFS. But it is not the backup of NameNode. We can use it to recreate the NameNode and restart it in case of a crash.

Q.19- What are the main functions of Secondary NameNode?
Main functions of Secondary NameNode are as follows:

- FsImage: It stores a copy of the FsImage file and EditLog.
- NameNode crash: In case NameNode crashes, we can use Secondary NameNode's FsImage to recreate the NameNode.
- Checkpoint: Secondary NameNode runs Checkpoint to confirm that data is not corrupt in HDFS.

Update: It periodically applies the updates from EditLog to the FsImage file. In this way the FsImage file on Secondary NameNode is kept up to date. This helps in saving time during NameNode restart.

Q.20- What happens if an HDFS file is set with a replication factor of 1 and DataNode crashes?
Replication factor is the same as the number of copies of a file on HDFS. If we set the replication factor of 1, it means there is only 1 copy of the file.
In case, DataNode that has this copy of file crashes, the data is lost. There is no way to recover it. It is essential to keep a replication factor of more than 1 for any business critical data.

Grow Data Skills

Q.21- What is the meaning of Rack Awareness in Hadoop?
In Hadoop, most of the components like NameNode, DataNode etc are rack- aware. It means they have the information about the rack on which they exist. The main use of rack awareness is in implementing fault-tolerance.

Any communication between nodes on the same rack is much faster than the communication between nodes on two different racks.

In Hadoop, NameNode maintains information about the rack of each DataNode. While reading/writing data, NameNode tries to choose the DataNodes that are closer to each other. Due to performance reasons, it is recommended to use close data nodes for any operation. So Rack Awareness is an important concept for high performance and fault- tolerance in Hadoop.

If we set Replication factor 3 for a file, does it mean any computation will also take place 3 times?

No. Replication factor of 3 means that there are 3 copies of a file. But computation takes place only one copy of the file. If the node on which the first copy exists does not respond then computation will be done on the second copy.

Q.22- How will you check if a file exists in HDFS?
In Hadoop, we can run hadoop fs command with option e to check the existence of a file in HDFS. This is generally used for testing purposes. Command will be as follows:
+ %>hadoop fs -test -ezd file_uri e is for checking the existence of file z is for checking non-zero size of

File d is for checking if the path is directory

Q.23- Why do we use the fsck command in HDFS?
fsck command is used for getting the details of files and directories in HDFS. Main uses of fsck command in HDFS are as follows:

- delete: We use this option to delete files in HDFS.
- move: This option is for moving corrupt files to lost/found.
- locations: This option prints all the locations of a block in HDFS.
- racks: This option gives the network topology of data-node locations.
- blocks: This option gives the report of blocks in HDFS.

Q.24- What will happen when NameNode is down and a user submits a new job?
Since NameNode is the single point of failure in Hadoop, user jobs cannot execute. The job will fail when the NameNode is down. Users will have to wait for NameNode to restart and come up, before running a job.

Q.25- What are the core methods of a Reducer in Hadoop?
The main task of Reducer is to reduce a larger set of data that shares a key to a smaller set of data. In Hadoop, Reducer has following three core methods:

- setup(): At the start of a task, the setup() method is called to configure various parameters for Reducer.
- reduce(): This is the main operation of Reducer. In the reduce() method we define the task that has to be done for a set of values that share a key.
- cleanup(): Once the reduce() task is done, we can use cleanup() to clean any intermediate data or temporary files.

Grow Data Skills

Q.26- What are the primary phases of a Reducer in Hadoop?
In Hadoop, there are three primary phases of a Reducer:

- Shuffle: In this phase, Reducer copies the sorted output from each Mapper.
- Sort: In this phase, Hadoop framework sorts the input to Reducer by the same key. It uses merge sort in this phase. Sometimes, shuffle and sort phases occur at the same time.
- Reduce: This is the phase in which output values associated with a key are reduced to give output results. Output from Reducer is not re-sorted.

Q.27- What is the use of Context objects in Hadoop?
Hadoop uses Context objects with Mapper to interact with the rest of the system. Context object gets the configuration of the system and job in its constructor.
We use Context objects to pass the information in setup(), cleanup() and map() methods. This is an important object that makes the important information available during the map operations.

Q.28- How does partitioning work in Hadoop?
Partitioning is the phase between Map phase and Reduce phase in Hadoop workflow. Since the partitioner gives output to Reducer, the number of partitions is the same as the number of Reducers.
Partitioner will partition the output from Map phase into distinct partitions by using a user-defined condition.
Partitions can be like Hash based buckets.
E.g. If we have to find the student with the maximum marks in each gender in each subject. We can first use the Map function to map the keys with each gender. Once mapping is done, the result is passed to the Partitioner. Partitioner will partition each row with gender on the basis of subject. For each subject there will be a different Reducer. Reducer will take input from each partition and find the student with the highest marks.

Q.29- What is a Combiner in Hadoop?
Combiner is an optional step between Map and Reduce. Combiner is also called Semi-Reducer. Combiner takes output from Map, creates Key-value pairs and passes these to Reducer.
Combiner's task is to summarise the outputs from Map into summary records with the same key.
By using Combiner, we can reduce the data transfer between Mapper and Reducer. Combiner does the task similar to reduce but it is done on the Map machine itself.

Q.30- What is the default replication factor in HDFS?
Default replication factor in HDFS is 3. It means there will be 3 copies of each data.
We can configure it with dfs.replication in the hdfs-site.xml file.
We can even set it from the command line in Hadoop fs command.

Grow Data Skills

Q.31- How much storage is allocated by HDFS for storing a file of 25 MB size?
This is a question to test the basic concepts of HDFS. In HDFS, all the data is stored in blocks. The size of the block can be configured in HDFS.In Apache Hadoop, the default block size is 64 MB. To store a file of 25 MB size, at least one block will be allocated. This means at least 64 MB will be allocated for the file of 25 MB size.

Q.32- Why does HDFS store data in Block structure?
HDFS stores all the data in terms of Blocks. With Block structure there are some benefits that HDFS gets. Some of these are as follows:

Fault Tolerance: With Block structure, HDFS implements replication. By replicating the same block in multiple locations, fault tolerance of the system increases. Even if some copy is not accessible, we can get the data from another copy.

Large Files: We can store very large files that cannot be even stored on one disk, in HDFS by using Block structure. We just divide the data of the file in multiple Blocks. Each Block can be stored on the same or different machines.

Storage management: With Block storage it is easier for Hadoop nodes to calculate the data storage as well as perform optimization in the algorithm to minimise data transfer across the network.

Q.33- How will you create a custom Partitioner in a Hadoop job?
Partition phase runs between Map and Reduce phase. It is an optional phase. We can create a custom partitioner by extending the org.apache.hadoop.mapreduce.Partitio class in Hadoop. In this class, we have to override the getPartition(KEY key, VALUE value, int numPartitions) method.

This method takes three inputs. In this method, numPartitions is the same as the number of reducers in our job. We pass key and value to get the partition number to which this key,value record will be assigned. There will be a reducer corresponding to that partition. The reducer will further handle summarizing of the data.Once the custom Partitioner class is ready, we have to set it in the Hadoop job. We can use following method to set it:

```
job.setPartitionerClass(CustomPartitioner)
```

Q.34- What is a Checkpoint node in HDFS?
A Checkpoint node in HDFS periodically fetches fsimage and edits from NameNode, and merges them. This merge result is called a Checkpoint. Once a Checkpoint is created, Checkpoint Node uploads the Checkpoint to NameNode. Secondary nodes also take a Checkpoint similar to Checkpoint Node. But it does not upload the Checkpoint to NameNode.

Main benefit of Checkpoint Node is in case of any failure on NameNode. A NameNode does not merge its edits to fsimage automatically during the runtime. If we have a long running task, the edits will become huge. When we restart NameNode, it will take much longer time, because it will first merge the edits. In such a scenario, a Checkpoint node helps for a long running task.

Checkpoint nodes performs the task of merging the edits with fsimage and then uploads these to NameNode. This saves time during the restart of NameNode.

Grow Data Skills

Q.35- What is a Backup Node in HDFS?
Backup Node in HDFS is similar to Checkpoint Node. It takes the stream of edits from NameNode. It keeps these edits in memory and also writes these to storage to create a new checkpoint. At any point of time, Backup Node is in sync with the Name Node.
The difference between Checkpoint Node and Backup Node is that Backup Node does not upload any checkpoints to Name Node. Also Backup node takes a stream instead of periodic reading of edits from Name Node.

Q.36- What is the meaning of the term Data Locality in Hadoop?
In a Big Data system, the size of data is huge. So it does not make sense to move data across the network. In such a scenario, Hadoop tries to move computation closer to data. So the Data remains local to the location wherever it was stored. But the computation tasks will be moved to data nodes that hold the data locally.
Hadoop follows following rules for Data Locality optimization:

- Hadoop first tries to schedule the task on a node that has an HDFS file on a local disk. If it cannot be done, then Hadoop will try to schedule the task on a node on the same rack as the node that has data. If this also cannot be done, Hadoop will schedule the task on the node with the same data on a different rack. The above method works well, when we work with the default replication factor of 3 in Hadoop.

Q.37- What is a Balancer in HDFS?
In HDFS, data is stored in blocks on a DataNode. There can be a situation when data is not uniformly spread into blocks on a DataNode. When we add a new DataNode to a cluster, we can face such a situation. In such a case, HDFS provides a useful tool Balancer to analyze the placement of blocks on a DataNode. Some people call it a Rebalancer also. This is an administrative tool used by admin staff. We can use this tool to spread the blocks in a uniform manner on a DataNode.

Q.38- What are the important points a NameNode considers before selecting the DataNode for placing a data block?
Some of the important points for selecting a DataNode by NameNode are as follows:
NameNode tries to keep at least one replica of a Block on the same node that is writing the block.
It tries to spread the different replicas of the same block on different racks, so that in case of one rack failure, another rack has the data.
One replica will be kept on a node on the same node as the one that is writing it. It is different from point 1. In Point 1, a block is written to the same node. At this point the block is written on a different node on the same rack. This is important for minimizing the network I/O. NameNode also tries to spread the blocks uniformly among all the DataNodes in a cluster.

Grow Data Skills

Q.39- What is Safemode in HDFS?
Safemode is considered as the read-only mode of NameNode in a cluster. During the startup of NameNode, it was in SafeMode. It does not allow writing to the file-system in Safemode. At this time, it collects data and statistics from all the DataNodes. Once it has all the data on blocks, it leaves Safemode.
The main reason for Safemode is to avoid the situation when NameNode starts replicating data in DataNodes before collecting all the information from DataNodes. It may erroneously assume that a block is not replicated well enough, whereas, the issue is that NameNode does not know about the whereabouts of all the replicas of a block. Therefore, in Safemode, NameNode first collects the information about how many replicas exist in a cluster and then tries to create replicas wherever the number of replicas is less than the policy.

Q.40- How will you replace HDFS data volume before shutting down a DataNode?
In HDFS, DataNode supports hot swappable drives. With a swappable drive we can add or replace HDFS data volumes while the DataNode is still running. The procedure for replacing a hot swappable drive is as follows:
First we format and mount the new drive. We update the DataNode configuration dfs.datanode.data.dir to reflect the data volume directories. Run the "dfsadmin -reconfig datanode HOST:PORT start" command to start the reconfiguration process Once the reconfiguration is complete, we just unmount the old data volume After unmount we can physically remove the old disks.

Q.41- What are the important configuration files in Hadoop?
There are two important configuration files in a Hadoop cluster:

- Default Configuration: There are core-default.xml, hdfs-default.xml and mapred-default.xml files in which we specify the default configuration for Hadoop cluster. These are read only files.
- Custom Configuration: We have site-specific custom files like core-site.xml, hdfs-site.xml, mapred-site.xml in which we can specify the site-specific configuration.

All the Jobs in Hadoop and HDFS implementation uses the parameters defined in the above-mentioned files. With customization we can tune these processes according to our use case. In Hadoop API, there is a Configuration class that loads these files and provides the values at run time to different jobs.

Q.42- How will you monitor memory used in a Hadoop cluster?
In Hadoop, TaskTracker is the one that uses high memory to perform a task. We can configure the TastTracker to monitor memory usage of the tasks it creates. It can monitor the memory usage to find the badly behaving tasks, so that these tasks do not bring the machine down with excess memory consumption. In memory monitoring we can also limit the maximum memory used by a task. We can even limit the memory usage per node. So that all the tasks executing together on a node do not consume more memory than a limit. Some of the parameters for setting memory monitoring in Hadoop are as follows:
mapred.cluster.map.memory.mb, mapred.cluster.reduce.memory.mb: This is the size of virtual memory of a single map/reduce slot in a cluster of Map-Reduce framework. mapred.job.map.memory.mb, mapred.job.reduce.memory.mb: This is the default limit of memory

Grow Data Skills

set on each map/reduce task in Hadoop. mapred.cluster.max.map.memory.m mapred.cluster.max.reduce.memory This is the maximum limit of memory set on each map/reduce task in Hadoop.

Q.43- Why do we need Serialization in Hadoop map reduce methods?
In Hadoop, there are multiple data nodes that hold data. During the processing of map and reduce methods data may transfer from one node to another node. Hadoop uses serialization to convert the data from Object structure to Binary format. With serialization, data can be converted to binary format and with de-serialization data can be converted back to Object format with reliability.

Q.44- What is the use of Distributed Cache in Hadoop?
Hadoop provides a utility called Distributed Cache to improve the performance of jobs by caching the files used by applications. An application can specify which file it wants to cache by using JobConf configuration. Hadoop framework copies these files to the nodes one which a task has to be executed. This is done before the start of execution of a task. DistributedCache supports distribution of simple read only text files as well as complex files like jars, zips etc.

Q.45- How will you synchronize the changes made to a file in Distributed Cache in Hadoop?
It is a trick question. In Distributed Cache, it is not allowed to make any changes to a file. This is a mechanism to cache read-only data across multiple nodes.Therefore, it is not possible to update a cached file or run any synchronization in Distributed Cache.

Q.46- Can you elaborate about the Mapreduce job?
Based on the configuration, the MapReduce Job first splits the input data into independent chunks called Blocks. These blocks are processed by Map() and Reduce() functions. First Map function processes the data, then processed by reduce function. The Framework takes care of sorting the Map outputs, scheduling the tasks.

Q.47- Why compute nodes and storage nodes are the same?
Compute nodes for processing the data, Storage nodes for storing the data. By default Hadoop framework tries to minimize the network wastage, to achieve that goal Framework follows the Data locality concept. The Compute code executes where the data is stored, so the data node and compute node are the same.

Q.48- What is the configuration object importance in Mapreduce?
It’s used to set/get of parameter name & value pairs in XML file.It’s used to initialize values, read from external file and set as a value parameter.Parameter values in the program always overwrite with new values which are coming from external configure files.Parameter values received from Hadoop’s default values.

Grow Data Skills

Q.49- Where Mapreduce is not recommended?
Mapreduce is not recommended for Iterative kind of processing. It means repeating the output in a loop manner.To process Series of Mapreduce jobs, MapReduce is not suitable. Each job persists data in a local disk, then again loads to another job. It's a costly operation and not recommended.

Q.50- What is Namenode and its responsibilities?
Namenode is a logical daemon name for a particular node. It's the heart of the entire Hadoop system. Which store the metadata in FsImage and get all block information in the form of Heartbeat.

Q.51- What is Jobtracker's responsibility?
Scheduling the job's tasks on the slaves. Slaves execute the tasks as directed by the JobTracker. Monitoring the tasks, if failed, re-execute the failed tasks.

Q.52- What are Jobtracker and Tasktracker?
MapReduce Framework consists of a single Job Tracker per Cluster, one Task Tracker per node. Usually A cluster has multiple nodes, so each cluster has a single Job Tracker and multiple TaskTrackers.JobTracker can schedule the job and monitor the Task Trackers. If Task Tracker failed to execute tasks, try to re-execute the failed tasks.

TaskTracker follows the JobTracker's instructions and executes the tasks. As a slave node, it reports the job status to Master JobTracker in the form of Heartbeat.

Q.53- What is Job scheduling importance in Hadoop Mapreduce?
Scheduling is a systematic procedure of allocating resources in the best possible way among multiple tasks. Hadoop task trackers performing many procedures, sometimes a particular procedure should finish quickly and provide more priority, to do it few job schedulers come into the picture. Default Schedule is FIFO. Fair scheduling, FIFO and CapacityScheduler are the most popular hadoop scheduling in hadoop.

Q.54- When to use a Reducer?
To combine multiple mapper's output use a reducer. Reducer has 3 primary phases: sort, shuffle and reduce. It's possible to process data without a reducer, but used when the shuffle and sort is required.

Q.55- Where does the Shuffle and Sort process go?
After Mapper generates the output, it temporarily stores the intermediate data on the local File System. Usually this temporary file is configured at coresite.xml in the Hadoop file. Hadoop Framework aggregate and sort this intermediate data, then update into Hadoop to be processed by the Reduce function. The Framework deletes this temporary data in the local system after Hadoop completes the job.

Q.56- Java is mandatory to write Mapreduce Jobs?
No, By default Hadoop is implemented in JavaTM, but MapReduce applications need not be written in Java. Hadoop supports Python, Ruby, C++ and other Programming languages. Hadoop Streaming API allows to create and run Map/Reduce jobs with any executable or script as the mapper and/or the reducer.Hadoop Pipes allows programmers to implement MapReduce applications by using C++ programs.

Grow Data Skills

Q.57- What methods can control the Map And Reduce function's output?
setOutputKeyClass() and setOutputValueClass()
If they are different, then the map output type can be set using the methods.
setMapOutputKeyClass() and setMapOutputValueClass()

Q.58- What is the main difference between Mapper And Reducer?
Map method is called separately for each key/value that has been processed. It processes input key/value pairs and emits intermediate key/value pairs.
The Reduce method is called separately for each key/values list pair. It processes intermediate key/value pairs and emits final key/value pairs.
Both are initialised and called before any other method is called. Both don't have any parameters and no output.

Q.59- Why compute Nodes and Storage Nodes are same?
Compute nodes are logical processing units, Storage nodes are physical storage units (Nodes). Both are running in the same node because of the "data locality" issue. As a result Hadoop minimize the data network wastage and allows it to process quickly.

Q.60- What is the difference between map side join and reduce side join?
Join multiple tables on the mapper side, called map side join. Please note map side join should have a strict format and be sorted properly. If the dataset is smaller tables, go through reducer phrases. Data should be partitioned properly.
Join the multiple tables in the reducer side called reduce side join. If you have a large amount of data tables, plan to join both tables. One table has a large amount of rows and columns, another one has a few tables only, goes through Reduce side join. It's the best way to join the multiple tables

Q.61- What happens if the number of Reducer is 0?
Number of reducers = 0 also valid configuration in MapReduce. In this scenario, No reducer will execute, so mapper output is considered as output, Hadoop stores this information in a separate folder.

Q.62- When will we go to Combiner?
Mappers and reducers are independent; they don't talk to each other. When the functions that are commutative(a.b = b.a) and associative {a.(b.c) = (a.b).c} we go to the combiner to optimize the mapreduce process. Many mapreduce jobs are limited by the bandwidth, so by default Hadoop framework minimizes the data bandwidth network wastage. To achieve it's goal, Mapreduce allows user defined "Cominer function" to run on the map output. It's an MapReduce optimization technique, but it's optional.

Q.63- What is the main difference between Mapreduce Combiner and Reducer?
Both Combiner and Reducer are optional, but most frequently used in MapReduce. There are three main differences such as:
combiner will get only one input from one Mapper. While Reducer will get multiple mappers from different mappers.
If aggregation requires a used reducer, but if the function follows commutative (a.b=b.a) and associative a.(b.c)= (a.b).c law, use combiner.

Grow Data Skills

GROW DATA SKILLS

Input and output keys and values types must be the same in the combiner, but the reducer can follow any type input, any output format.

Q.64- What is a partition?
After combiner and intermediate mapoutput the Partitioner controls the keys after sort and shuffle. Partitioner divides the intermediate data according to the number of reducers so that all the data in a single partition gets executed by a single reducer. It means each partition can be executed by only a single reducer. If you call reducer, the partition is called automatically.

Q.65- When will we go to Partition?
By default Hive reads the entire dataset even if the application has a slice of data. It's a bottleneck for mapreduce jobs. So Hive allows special options called partitions. When you are creating a table, hive partitioning the table based on requirement.

Q.66- What are the important steps when you are partitioning a table?
Don't over partition the data with too small partitions, it's overhead to the namenode.
if dynamic partition, at least one static partition should exist and set to strict mode by using given commands.

- SET hive.exec.dynamic.partition = true;
- SET hive.exec.dynamic.partition.mode = nonstrict;

first load data into nonpartitioned table, then load such data into a partitioned table. It's not possible to load data from local to partitioned tables.
insert overwrite table table_name partition(year) select * from nonpartitiontable;

Q.67- Can you elaborate Mapreduce Job architecture?
First Hadoop programmer submit Mapreduce program to JobClient.
Job Client requests the JobTracker to get Job id, Job tracker provides JobID, it's in the form of Job_HadoopStartedtime_00001. It's a unique ID.
Once JobClient receives a Job ID, copy the Job resources (job.xml, job.jar) to File System (HDFS) and submit the job to JobTracker. JobTracker initiates jobs and schedules the job.
Based on configuration, job split the input splits and submit to HDFS. TaskTracker retrieves the job resources from HDFS and launches the Child JVM. In this Child JVM, run the map and reduce tasks and notify to the Job tracker the job status.

Q.68- Why does Task Tracker launch child Jvm?
Most frequently, Hadoop developers mistakenly submit wrong jobs or have bugs. If Task Tracker uses an existing JVM, it may interrupt the main JVM, so other tasks may be influenced. Whereas child JVM if it's trying to damage existing resources, TaskTracker kills that child JVM and retry or relaunch new child JVM.

Grow Data Skills

GROW DATA SKILLS

Q.69- Why does Jobclient and Job Tracker submit job resources to the file system?
Data locality. Moving competition is cheaper than moving Data. So logic/ competition in Jar file and splits. So Where the data is available, in File System Datanodes. So every resource copy where the data is available.

Q.70- How many Mappers and Reducers can run?
By default Hadoop can run 2 mappers and 2 reducers in one datanode. also each node has 2 map slots and 2 reducer slots. It's possible to change the default values in Mapreduce.xml in the conf file.

Q.71- What is Inputsplit?
A chunk of data processed by a single mapper called InputSplit. In other words, a logical chunk of data which is processed by a single mapper called Input split, by default inputSplit = block Size.

Q.72- How to configure the split value?
By default block size = 64mb, but to process the data, the job tracker split the data. Hadoop architects use these formulas to know split size.
+ split size = min (max_splitsize, max (block_size, min_split_size));
+ split size = max(min_split_size, min (block_size, max_split, size));

by default split size = block size. Always No of splits = No of mappers.
Apply above formula:
+ split size = Min (max_splitsize, max (64, 512kB) // max _splitsize = depends on env, may 1gb or 10gb split size = min (10gb (let assume), 64) split size = 64MB.
+ split size = max(min_split_size, min (block_size, max_split, size)); split size = max (512kb, min (64, 10GB)); split size = max (512kb, 64);split size = 64 MB;

Q.73- How much ram is required to process 64mb data?
Leg Assuming. 64 block size, system take 2 mappers, 2 reducers, so 64*4 = 256 MB memory and OS take at least 30% extra space so at least 256 + 80 = 326MB Ram required to process a chunk of data.So in this way requires more memory to process an unstructured process.

Q.74- What is the difference between block And split?
- Block: How much chunk data is stored in the memory called block.
- Split: how much data to process the data called split.

Q.75- Why Hadoop Framework reads a file parallel, why not sequential?
To retrieve data faster, Hadoop reads data parallel, the main reason it can access data faster. While, writes in sequence, but not parallel, the main reason it might result is that one node can be overwritten by another and where the second node. Parallel processing is independent, so there is no relation between two nodes. If you write data in parallel, it's not possible where the next chunk of data has. For example 100 MB data write parallel, 64 MB one block another block 36, if data writes parallel the first block doesn't know where the remaining data is. So Hadoop reads parallel and writes sequentially.

Grow Data Skills

Q.76- If I change block size from 64 to 128?
Even if you have changed block size, it does not affect existing data. After changing the block size, every file chunked after 128 MB of block size. It means old data is in 64 MB chunks, but new data stored in 128 MB blocks.

Q.77- What is Issplitable()?
By default this value is true. It is used to split the data in the input format. if unstructured data, it's not recommendable to split the data, so process the entire file as a one split. to do it first change isSplitable() to false.

Q.78- How much Hadoop allows maximum block size and minimum block size?

- Minimum: 512 bytes. It's local OS file system block size. No one can decrease fewer than block size.
- Maximum: Depends on the environment. There is no upper bound.

Q.79- What are the Job Resource files?
job.xml and job.jar are core resources to process the Job. Job Client copy the resources to the HDFS.

Q.80- What's the Mapreduce Job consist of?
MapReduce job is a unit of work that client wants to be performed. It consists of input data, MapReduce program in Jar file and configuration setting in XML files. Hadoop runs this job by dividing it in different tasks with the help of JobTracker

Q.81- What is the data locality?
Wherever the data is there process the data, computation/process the data where the data available, this process called data locality. "Moving Computation is Cheaper than Moving Data " to achieve this goal following data locality. It's possible when the data is splittable, by default it's true.

Q.82- What is speculative execution?
Hadoop runs the process in commodity hardware, so it's possible to fail if the system also has low memory. So if system failed, process also failed, it's not recommendable.Speculative execution is a process performance optimization technique.Computation/logic distribute to the multiple systems and execute which system execute quickly. By default this value is true. Now even if the system crashes, not a problem, the framework chooses logic from other systems.

- Eg: logic distributed on A, B, C, D systems, completed within a time.

System A, System B, System C, System D systems executed 10 min, 8 mins, 9 mins 12 mins simultaneously. So consider system B and kill remaining system processes, framework take care to kill the other system process.

Grow Data Skills

GROW DATA SKILLS

Q.83- What is a chain Mapper?
Chain mapper class is a special mapper class set which runs in a chain fashion within a single map task. It means, one mapper input acts as another mapper's input, in this way n number of mapper connected in chain fashion.

Q.84- How to do value level comparison?
Hadoop can process key level comparison only but not in the value level comparison.

Q.85- What are the setup and clean up methods?
If you don't know the starting and ending points/lines, it's much more difficult to solve those problems. Setup and clean up can resolve it. N number of blocks, by default 1 mapper called to each split. Each split has one start and clean up methods. N number of methods, number of lines. Setup is initialising job resources.
The purpose of cleaning up is to close the job resources. The map processes the data. Once the last map is completed, cleanup is initialized. It Improves the data transfer performance. All these block size comparisons can be done in reducer as well. If you have any key and value, compare one key value to another key value. If you compare record levels, use these setup and cleanup. It opens once and processes many times and closes once. So it saves a lot of network wastage during the process.

Q.86- How many slots are allocated for each task?
By default each task has 2 slots for mapper and 2 slots for reducer. So each node has 4 slots to process the data.

Q.87- Why does Tasktracker launch a child Jvm to do a task?
Sometimes child threads corrupt parent threads. It means because of a programmer mistake, the MapReduce task disturbed. So task trackers launch a child JVM to process individual mappers or taskers. If tasktracker uses an existing JVM, it might damage the main JVM. If any bugs occur, tasktracker kill the child process and relaunch another child JVM to do the same task. Usually task tracker relaunch and retry the task 4 times.

Q.88- What main configuration parameters are specified in Mapreduce?
The MapReduce programmers need to specify following configuration parameters to perform the map and reduce jobs:

- The input location of the job in HDFs.
- The output location of the job in HDFS.
- The input and output's format.
- The classes contain map and reduce functions, respectively.
- The .jar file for mapper, reducer and driver classes

Q.89- What is identity Mapper?
Identity Mapper is the default Mapper class provided by Hadoop. when no other Mapper class is defined, Identify will be executed. It only writes the input data into output and does not perform computations and calculations on the input data. The class name is org.apache.hadoop.mapred.lib.IdentityMapper.

Grow Data Skills

GROW DATA SKILLS

Q.90- What is a RecordReader in MapReduce?
RecordReader is used to read key/value pairs from the InputSplit by converting the byte-oriented view and presenting record-oriented view to Mapper.

Q.91- What is OutputCommitter?
OutPutCommitter describes the commit of MapReduce task. FileOutputCommitter is the default available class available for OutputCommitter in MapReduce. It performs the following operations:
Create a temporary output directory for the job during initialization.
Then, it cleans the job as it removes the temporary output directory post job completion.
Set up the task temporary output. Identifies whether a task needs commitment. The commit is applied if required.
JobSetup, JobCleanup and TaskCleanup are important tasks during the output commit.

Q.92- What are the parameters of Mappers and Reducers?
The four parameters for mappers are:

- LongWritable (input)
- text (input)
- text (intermediate output)
- IntWritable (intermediate output)

The four parameters for reducers are:

- Text (intermediate output)
- IntWritable (intermediate output)
- Text (final output)
- IntWritable (final output)

Q.93- Explain Jobconf in Mapreduce?
It is a primary interface to define a map-reduce job in Hadoop for job execution. JobConf specifies mapper, Combiner, partitioner, Reducer,InputFormat , OutputFormat implementations and other advanced job faets like Comparators.

Q.94- Explain Job scheduling through Jobtracker?
JobTracker communicates with NameNode to identify data location and submits the work to TaskTracker node. The TaskTracker plays a major role as it notifies the JobTracker for any job failure. It actually refers to the heartbeat reporter reassuring the JobTracker that it is still alive. Later, the JobTracker is responsible for the actions as it may either resubmit the job or mark a specific record as unreliable or blacklist it.

Q.95- What is SequenceFileInputFormat?
A compressed binary output file format to read in sequence files and extends the FileInputFormat.It passes data between output-input (between output of one MapReduce job to input of another MapReduce job)phases of MapReduce jobs.

Grow Data Skills

Q.96- Explain how the input and output data format of the Hadoop Framework?
The MapReduce framework operates exclusively on pairs, that is, the framework views the input to the job as a set of pairs and produces a set of pairs as the output of the job, conceivably of different types.
See the flow mentioned below:

- (input) -> map -> -> combine/sorting -> -> reduce -> (output)

Q.97- What are the restrictions to the Key and Value Class?
The key and value classes have to be serialized by the framework. To make them serializable Hadoop provides a Writable interface. As you know from the java itself that the key of the Map should be comparable, hence the key has to implement one more interface Writable Comparable.

Q.98- Explain the wordcount implementation via Hadoop Framework?
We will count the words in all the input file flow as below
input Assume there are two files each having a sentence Hello World Hello World (In file 1) Hello World Hello World (In file 2)
Mapper : There would be each mapper for the a file For the given sample input the first map output:
< Hello, 1>
< World, 1>
< Hello, 1>
< World, 1>
The second map output:
< Hello, 1>
< World, 1>
< Hello, 1>
< World, 1>
Combiner/Sorting (This is done for each individual map) So output looks like this The output of the first map:
< Hello, 2>
< World, 2>
The output of the second map:
< Hello, 2>
< World, 2>
Reducer : It sums up the above output and generates the output as below
< Hello, 4>
< World, 4>

Output
Final output would look like
Hello 4 times
World 4 times

[Table of Contents](#HADOOP)

Grow Data Skills

Q.98- How Mapper is instantiated in a running Job?
The Mapper itself is instantiated in the running job, and will be passed a MapContext object which it can use to configure itself.

Q.99- Which are the methods in the Mapper Interface?
The Mapper contains the run() method, which calls its own setup() method only once, it also calls a map() method for each input and finally calls it cleanup() method. All above methods you can override in your code.

Q.100- What happens if You don't Override the Mapper methods and keep them as it is?
If you do not override any methods (leaving even map as-is), it will act as the identity function, emitting each input record as a separate output.

Q.101- What is the use of context objects?
The Context object allows the mapper to interact with the rest of the Hadoop system. It Includes configuration data for the job, as well as interfaces which allow it to emit output.

Q.102- How can you Add the arbitrary Key value pairs in your Mapper?
You can set arbitrary (key, value) pairs of configuration data in your Job, e.g. with
Job.getConfiguration().set("myKey", "myVal"), and then retrieve this data in your mapper with
Context.getConfiguration().get("myKey"). This kind of functionality is typically done in the Mapper's setup() method.

Q.103- How does Mapper's run method work?
The Mapper.run() method then calls map(KeyInType, ValInType, Context) for each key/value pair in the InputSplit for that task

Q.104- Which Object can be used to get the progress of a particular Job?
Context

Q.105- What is the next step after Mapper Or Maptask?
The output of the Mapper is sorted and Partitions will be created for the output. Number of partitions depends on the number of reducers.

Q.106- How can we control which particular Key should go in a specific Reducer?
Users can control which keys (and hence records) go to which Reducer by implementing a custom Partitioned.

Q.107- What is the use of Combiner?
It is an optional component or class, and can be specified via Job.setCombinerClass(ClassName), to perform local aggregation of the intermediate outputs, which helps to cut down the amount of data transferred from the Mapper to the Reducer.

Grow Data Skills

Q.107- How many Maps are there in a particular Job?
The number of maps is usually driven by the total size of the inputs, that is, the total number of blocks of the input files.
Generally it is around 10-100 maps per-node. Task setup takes a while, so it is best if the maps take at least a minute to execute.
Suppose, if you expect 10TB of input data and have a block size of 128MB, you'll end up with 82,000 maps, to control the number of blocks you can use the mapreduce.job.maps parameter (which only provides a hint to the framework). Ultimately, the number of tasks is controlled by the number of splits returned by the InputFormat.getSplits() method (which you can override).

Q.108- What is the Reducer used for?
Reducer reduces a set of intermediate values which share a key to a (usually smaller) set of values. The number of reductions for the job is set by the user via Job.setNumReduceTasks(int).

Q.109- Explain the core methods of the Reducer?
The API of Reducer is very similar to that of Mapper, there's a run() method that receives a Context containing the job's configuration as well as interfacing methods that return data from the reducer itself back to the framework. The run() method calls setup() once, reduce() once for each key associated with the reduce task, and cleanup() once at the end. Each of these methods can access the job's configuration data by using Context.getConfiguration().
As in Mapper, any or all of these methods can be overridden with custom implementations. If none of these methods are overridden, the default reducer operation is the identity function; values are passed through without further processing.
The heart of Reducer is its reduce() method. This is called once per key; the second argument is an Iterable which returns all the values associated with that key.

Q.110- What are the primary phases of the Reducer?
Shuffle, Sort and Reduce.

Q.111- Explain the Shuffle?
Input to the Reducer is the sorted output of the mappers. In this phase the framework fetches the relevant partition of the output of all the mappers, via HTTP.

Q.112- Explain the Reducer's sort phase?
The framework groups Reducer inputs by keys (since different mappers may have output the same key) in this stage. The shuffle and sort phases occur simultaneously; while map-outputs are being fetched they are merged (It is similar to merge-sort).

Q.113- Explain the Reducer's reduce phase?
In this phase the reduce(MapOutKeyType, Iterable, Context) method is called for each pair in the grouped inputs. The output of the reduce task is typically written to the FileSystem via Context.write (ReduceOutKeyType, ReduceOutValType). Applications can use the Context to report progress, set application-level status messages and update Counters, or just indicate that they are alive. The output of the Reducer is not sorted.

Grow Data Skills

GROW DATA SKILLS

Q.114- How many Reducers should be configured?
The right number of reduces seems to be 0.95 or 1.75 multiplied by
(<no.of nades> * mapreduce.tasktracker.reduce.tasks.maximum).
With 0.95 all of the reducers can launch immediately and start transfering map outputs as the maps finish. With 1.75 the faster nodes will finish their first round of reductions and launch a second wave of reductions doing a much better job of load balancing. Increasing the number of reductions increases the framework overhead, but increases load balancing and lowers the cost of failures.

Q.115- It can be possible that a Job has 0 Reducers?
It is legal to set the number of reduce-tasks to zero if no reduction is desired.

Q.116- What happens if the number of Reducers are 0?
In this case the outputs of the map-tasks go directly to the FileSystem, into the output path set by setOutputPath(Path). The framework does not sort the map-outputs before writing them out to the FileSystem.

Q.117- How many instances of Jobtracker can run on a Hadoop Cluster?
Only one

Q.118- What is the Jobtracker and what it performs in a Hadoop Cluster?
JobTracker is a daemon service which submits and tracks the MapReduce tasks to the Hadoop cluster. It runs its own JVM process. And usually it runs on a separate machine, and each slave node is configured with a job tracker node location. The JobTracker is a single point of failure for the Hadoop MapReduce service. If it goes down, all running jobs are halted.
JobTracker in Hadoop performs following actions
Client applications submit jobs to the Job tracker.
The JobTracker talks to the NameNode to determine the location of the data
The JobTracker locates TaskTracker nodes with available slots at or near the data
The JobTracker submits the work to the chosen TaskTracker nodes.
A TaskTracker will notify the JobTracker when a task fails. The JobTracker decides what to do then: it may resubmit the job elsewhere, it may mark that specific record as something to avoid, and it may even blacklist the TaskTracker as unreliable.
When the work is completed, the JobTracker updates its status.
The TaskTracker nodes are monitored. If they do not submit heartbeat signals often enough, they are deemed to have failed and the work is scheduled on a different TaskTracker.
A TaskTracker will notify the JobTracker when a task fails. The JobTracker decides what to do then: it may resubmit the job elsewhere, it may mark that specific record as something to avoid, and it may even blacklist the TaskTracker as unreliable.
When the work is completed, the JobTracker updates its status.
Client applications can poll the JobTracker for information.

Grow Data Skills

Q.119- How is a task scheduled by a Jobtracker?
The TaskTrackers send out heartbeat messages to the JobTracker, usually every few minutes, to reassure the JobTracker that it is still alive. These messages also inform the JobTracker of the number of available slots, so the JobTracker can stay up to date with where in the cluster work can be delegated. When the JobTracker tries to find somewhere to schedule a task within the MapReduce operations, it first looks for an empty slot on the same server that hosts the DataNode containing the data, and if not, it looks for an empty slot on a machine in the same rack.

Q.120- How many instances of Tasktracker run on a Hadoop Cluster?
There is one Daemon Tasktracker process for each slave node in the Hadoop cluster.

Q.121- How many maximum Jvm can run on a Slave Node?
One or Multiple instances of Task Instance can run on each slave node. Each task instance is run as a separate JVM process. The number of Task instances can be controlled by configuration. Typically a high end machine is configured to run more task instances.

Q.122- What is Nas?
It is one kind of file system where data can reside on one centralized machine and all the cluster members will read and write data from that shared database, which would not be as efficient as HDFS.

Q.123- How do Hdfs differ from Nfs?
Following are differences between HDFS and NAS
In HDFS Data Blocks are distributed across local drives of all machines in a cluster. Whereas in NAS data is stored on dedicated hardware.
HDFS is designed to work with MapReduce System, since computation is moved to data. NAS is not suitable for MapReduce since data is stored separately from the computations.
HDFS runs on a cluster of machines and provides redundancy using replication protocol. Whereas NAS is provided by a single machine therefore does not provide data redundancy.

Q.124- How does a NameNode handle the failure of the Data Nodes?
HDFS has master/slave architecture. An HDFS cluster consists of a single
NameNode, a master server that manages the file system namespace and regulates access to files by clients.
In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. The NameNode and DataNode are pieces of software designed to run on commodity machines.
NameNode periodically receives a Heartbeat and a Block report from each of the DataNodes in the cluster. Receipt of a Heartbeat implies that the DataNode is functioning properly. A Blockreport contains a list of all blocks on a DataNode. When NameNode notices that it has not received a heartbeat message from a data node after a certain amount of time, the data node is marked as dead. Since blocks will be under replication the system begins replicating the blocks that were stored on the dead DataNode. The NameNode Orchestrates the replication of data blocks from one DataNode to another. The replication data transfer happens directly between DataNode and the data never passes through the NameNode.

Grow Data Skills

GROW DATA SKILLS

Q.125- Can Reducer talk with each other?
No, Reducer runs in isolation.

Q.126- Where the Mapper's intermediate data will be stored?
The mapper output (intermediate data) is stored on the Local file system (NOT HDFS) of each individual mapper node. This is typically a temporary directory location which can be setup in config by the Hadoop administrator. The intermediate data is cleaned up after the Hadoop Job completes.

Q.127- What is the Hadoop Mapreduce api contract for a Key and Value Class?
The Key must implement the org.apache.hadoop.io.WritableComparable interface.
The value must implement the org.apache.hadoop.io.Writable interface.

Q.128- What is an IdentityMapper and IdentityReducer in Mapreduce?

- org.apache.hadoop.mapred.lib.IdentityMapper: Implements the identity function, mapping inputs directly to outputs. If a MapReduce programmer does not set the Mapper Class using JobConf.setMapperClass then IdentityMapper.class is used as a default value.
- org.apache.hadoop.mapred.lib.IdentityReducer : Performs no reduction, writing all input values directly to the output. If a MapReduce programmer does not set the Reducer Class using JobConf.setReducerClass then IdentityReducer.class is used as a default value.

Q.129- What is the meaning of Speculative Execution in Hadoop?
Speculative execution is a way of coping with individual Machine performance. In large clusters where hundreds or thousands of machines are involved there may be machines which are not performing as fast as others.
This may result in delays in a full job due to only one machine not performing well. To avoid this, speculative execution in hadoop can run multiple copies of the same map or reduce tasks on different slave nodes. The results from first node to finish are used.

Q.130- How is HDFS different from traditional File Systems?
HDFS, the Hadoop Distributed File System, is responsible for storing huge data on the cluster. This is a distributed file system designed to run on commodity hardware.
It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant.
HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware.
HDFS provides high throughput access to application data and is suitable for applications that have large data sets.
HDFS is designed to support very large files. Applications that are compatible with HDFS are those that deal with large data sets. These applications write their data only once but they read it one or more times and require these reads to be satisfied at streaming speeds. HDFS supports write-once-read-many semantics on files.

Grow Data Skills

GROW DATA SKILLS

Q.131- What is Hdfs block size and how is it different from Traditional File System block size?
In HDFS data is split into blocks and distributed across multiple nodes in the cluster. Each block is typically 64Mb or 128Mb in size. Each block is replicated multiple times. Default is to replicate each block three times. Replicas are stored on different nodes. HDFS utilizes the local file system to store each HDFS block as a separate file. HDFS Block size can not be compared with the traditional file system block size.

Q.132- What is a NameNode and how many instances of NameNode run on a Hadoop Cluster?
The NameNode is the centrepiece 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.
There is only One NameNode process run on any hadoop cluster. NameNode runs on its own JVM process. In a typical production cluster it runs on a separate machine.
The NameNode is a Single Point of Failure for the HDFS Cluster. When the NameNode goes down, the file system goes offline.
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 to successful requests by returning a list of relevant DataNode servers where the data lives.

Q.133- How does the client communicate with Hdfs?
The Client communication to HDFS happens using Hadoop HDFS API. Client applications talk to the NameNode whenever they wish to locate a file, or when they want to add/copy/move/delete a file on HDFS. The NameNode responds the successful requests by returning a list of relevant DataNode servers where the data lives. Client applications can talk directly to a DataNode, once the NameNode has provided the location of the data.

Q.134- How the Hdfs blocks are replicated?
HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks; all blocks in a file except the last block are the same size. The blocks of a file are replicated for fault tolerance. The block size and replication factor are configurable per file. An application can specify the number of replicas of a file. The replication factor can be specified at file creation time and can be changed later. Files in HDFS are write-once and have strictly one writer at any time.
The NameNode makes all decisions regarding replication of blocks. HDFS uses a rack-aware replica placement policy. In default configuration there are a total 3 copies of a data block on HDFS, 2 copies are stored on datanodes on the same rack and 3rd copy on a different rack.

Q.135- Can you give some examples of Big Data?
There are many real life examples of Big Data! Facebook is generating 500+ terabytes of data per day, NYSE (New York Stock Exchange) generates about 1 terabyte of new trade data per day, a jet airline collects 10 terabytes of censor data for every 30 minutes of flying time. All these are day to day examples of Big Data!

Grow Data Skills

Q.136- What is the basic difference between traditional Rdbms and Hadoop?
Traditional RDBMS is used for transactional systems to report and archive the data, whereas Hadoop is an approach to store huge amounts of data in the distributed file system and process it. RDBMS will be useful when you want to seek one record from Big data, whereas, Hadoop will be useful when you want Big data in one shot and perform analysis on that later.

Q.137- What is structured and unstructured Data?
Structured data is the data that is easily identifiable as it is organized in a structure. The most common form of structured data is a database where specific information is stored in tables, that is, rows and columns. Unstructured data refers to any data that cannot be identified easily. It could be in the form of images, videos, documents, email, logs and random text. It is not in the form of rows and columns.

Q.138- Since the data is replicated thrice in Hdfs, does it mean that any calculation done on One Node will also be replicated on the other Two?
Since there are 3 nodes, when we send the MapReduce programs, calculations will be done only on the original data. The master node will know which node exactly has that particular data. In case, if one of the nodes is not responding, it is assumed to be failed. Only then, the required calculation will be done on the second replica.

Q.139- What is throughput and how does Hdfs get a good throughput?
Throughput is the amount of work done in a unit time. It describes how fast the data is getting accessed from the system and it is usually used to measure performance of the system. In HDFS, when we want to perform a task or an action, then the work is divided and shared among different systems. So all the systems will be executing the tasks assigned to them independently and in parallel. So the work will be completed in a very short period of time. In this way, the HDFS gives good throughput. By reading data in parallel, we decrease the actual time to read data tremendously.

Q.140- What is streaming access?
As HDFS works on the principle of 'Write Once, Read Many', the feature of streaming access is extremely important in HDFS. HDFS focuses not so much on storing the data but how to retrieve it at the fastest possible speed, especially while analyzing logs. In HDFS, reading the complete data is more important than the time taken to fetch a single record from the data.

Q.141- What is a Commodity Hardware so does Commodity Hardware include Ram?
Commodity hardware is a non-expensive system which is not of high quality or high-availability. Hadoop can be installed in any average commodity hardware. We don't need supercomputers or high-end hardware to work on Hadoop. Yes, Commodity hardware includes RAM because there will be some services which will be running on RAM.

Q.142- Is NameNode also a Commodity?
No. Namenode can never be a commodity hardware because the entire HDFS relies on it. It is the single point of failure in HDFS. Namenode has to be a high-availability machine.

Grow Data Skills

GROW DATA SKILLS

Q.143- What is Metadata?
Metadata is the information about the data stored in data nodes such as location of the file, size of the file and so on.

Q.144- What is a Daemon?
Daemon is a process or service that runs in the background. In general, we use this word in the UNIX environment. The equivalent of Daemon in Windows is “services” and in Dos is ” TSR”.

Q.145- What is a Heartbeat in Hdfs?
A heartbeat is a signal indicating that it is alive. A datanode sends heartbeat to Namenode and task tracker will send its heart beat to job tracker. If the Namenode or job tracker does not receive a heartbeat then they will decide that there is some problem in the datanode or task tracker is unable to perform the assigned task.

Q.146- How indexing is done in Hdfs?
Hadoop has its own way of indexing. Depending upon the block size, once the data is stored, HDFS will keep on storing the last part of the data which will say where the next part of the data will be. In fact, this is the base of HDFS.

Q.147- If a Data Node is full, how is it identified?
When data is stored in datanode, then the metadata of that data will be stored in the Namenode. So Namenode will identify if the data node is full.

Q.148- If DataNodes increase then do we need to upgrade NameNode?
While installing the Hadoop system, Namenode is determined based on the size of the clusters. Most of the time, we do not need to upgrade the Namenode because it does not store the actual data, but just the metadata, so such a requirement rarely arises.

Q.149- Are Job Tracker and Task Trackers present in separate machines?
Yes, job tracker and task tracker are present in different machines. The reason is that the job tracker is a single point of failure for the Hadoop MapReduce service. If it goes down, all running jobs are halted.

Q.150- On what basis NameNode will decide which DataNode to write on?
As the Namenode has the metadata (information) related to all the data nodes, it knows which datanode is free.

Q.151- Who is a user in Hdfs?
A user is like you or me, who has some query or who needs some kind of data.

Q.152- Is the client the end user in Hdfs?
No, Client is an application which runs on your machine, which is used to interact with the Namenode (job tracker) or datanode (task tracker).

Grow Data Skills

Q.153- What is the Communication Channel between client and NameNode/DataNode?
The mode of communication is SSH.

Q.154- What is a Rack?
Rack is a storage area with all the datanodes put together. These data nodes can be physically located at different places. Rack is a physical collection of datanodes which are stored at a single location. There can be multiple racks in a single location.

Q.155- On what basis Data will be stored on a Rack?
When the client is ready to load a file into the cluster, the content of the file will be divided into blocks. Now the client consults the Namenode and gets 3 datanodes for every block of the file which indicates where the block should be stored. While placing the datanodes, the key rule followed is "for every block of data, two copies will exist in one rack, and a third copy in a different rack". This rule is known as "Replica Placement Policy".

Q.156- Do we need to place 2nd and 3rd Data in Rack 2 only?
Yes, this is to avoid datanode failure.

Q.157- What if Rack 2 and DataNode fails?
If both rack2 and datanode present in rack 1 fails then there is no chance of getting data from it. In order to avoid such situations, we need to replicate that data more number of times instead of replicating only thrice. This can be done by changing the value in the replication factor which is set to 3 by default.

Q.158- What is the difference between Gen1 and Gen2 Hadoop with regards to the NameNode?
In Gen 1 Hadoop, Namenode is the single point of failure. In Gen 2 Hadoop, we have what is known as Active and Passive Namenodes kind of a structure. If the active Namenode fails, the passive Namenode takes over the charge.

Q.159- Do we require two servers for the NameNode and the DataNodes?
Yes, we need two different servers for the Namenode and the datanodes. This is because Namenode requires a highly configurable system as it stores information about the location details of all the files stored in different data nodes and on the other hand, datanodes require a low configuration system.

Q.160- Why are the number of splits equal to the number of Maps?
The number of maps is equal to the number of input splits because we want the key and value pairs of all the input splits.

Q.161- Is a Job split into maps?
No, a job is not split into maps. Spilt is created for the file. The file is placed on datanodes in blocks. For each split, a map is needed.

Grow Data Skills

Q.162- Which are the two types of writes In Hdfs?
There are two types of writes in HDFS:

- Posted and non-posted write. Posted Write is when we write it and forget about it, without worrying about the acknowledgement.
- It is similar to our traditional Indian post.
- Non-posted Write, we wait for the acknowledgement. It is similar to today's courier services. Naturally, non-posted write is more expensive than the posted write. It is much more expensive, though both writes are asynchronous.

Q.163- Why reading is done in parallel and writing is not in Hdfs?
Reading is done in parallel because by doing so we can access the data fast. But we do not perform the write operation in parallel. The reason is that if we perform the write operation in parallel, then it might result in data inconsistency. For example, you have a file and two nodes are trying to write data into the file in parallel, then the first node does not know what the second node has written and vice-versa. So, this makes it confusing which data to be stored and accessed.

Q.164- Can Hadoop be compared to a Nosql Database like Cassandra?
Though NOSQL is the closest technology that can be compared to Hadoop, it has its own pros and cons. There is no DFS in NOSQL. Hadoop is not a database. It's a file system (HDFS) and distributed programming framework (MapReduce).

Q.165- How does JobTracker schedule a task?
The TaskTrackers send out heartbeat messages to the JobTracker, usually every few minutes, to reassure the JobTracker that it is still alive. These messages also inform the JobTracker of the number of available slots, so the JobTracker can stay up to date with where in the cluster work can be delegated. When the JobTracker tries to find somewhere to schedule a task within the MapReduce operations, it first looks for an empty slot on the same server that hosts the DataNode containing the data, and if not, it looks for an empty slot on a machine in the same rack.

Q.166- What is a Task Tracker in Hadoop and how many instances of Task Tracker run on a Hadoop Cluster?
A TaskTracker is a slave node daemon in the cluster that accepts tasks (Map, Reduce and Shuffle operations) from a JobTracker. There is only One Task Tracker process run on any hadoop slave node. Task Tracker runs on its own JVM process. Every TaskTracker is configured with a set of slots, these indicate the number of tasks that it can accept. The TaskTracker starts a separate JVM process to do the actual work (called as Task Instance) this is to ensure that process failure does not take down the task tracker. The TaskTracker monitors these task instances, capturing the output and exit codes. When the Task instances finish, successfully or not, the task tracker notifies the JobTracker. The TaskTrackers also send out heartbeat messages to the JobTracker, usually every few minutes, to reassure the JobTracker that it is still alive. These messages also inform the JobTracker of the number of available slots, so the JobTracker can stay up to date with where in the cluster work can be delegated.

Grow Data Skills

Q.167- What is a task instance in Hadoop and where does it run?
Task instances are the actual MapReduce jobs which are run on each slave node. The TaskTracker starts a separate JVM process to do the actual work (called as Task Instance) this is to ensure that process failure does not take down the task tracker. Each Task Instance runs on its own JVM process. There can be multiple processes of task instances running on a slave node. This is based on the number of slots configured on the task tracker. By default a new task instance JVM process is spawned for a task.

Q.168- What is the configuration of a typical Slave Node on a Hadoop Cluster and how many Jvms run on a Slave Node?
Single instance of a Task Tracker is run on each Slave node. Task tracker is run as a separate JVM process.
Single instance of a DataNode daemon is run on each Slave node. DataNode daemon is run as a separate JVM process.
One or Multiple instances of Task Instance is run on each slave node. Each task instance is run as a separate JVM process. The number of Task instances can be controlled by configuration. Typically a high end machine is configured to run more task instances.

Q.169- How does NameNode handle DataNode failures?
NameNode periodically receives a Heartbeat and a Blockreport from each of the DataNodes in the cluster. Receipt of a Heartbeat implies that the DataNode is functioning properly. A Blockreport contains a list of all blocks on a DataNode. When NameNode notices that it has not received a heartbeat message from a data node after a certain amount of time, the data node is marked as dead. Since blocks will be under replication the system begins replicating the blocks that were stored on the dead datanode. The NameNode Orchestrates the replication of data blocks from one datanode to another. The replication data transfer happens directly between datanodes and the data never passes through the namenode.

Q.170- Does Mapreduce programming model provide a way for Reducers to communicate with each other and in a Mapreduce Job can a Reducer communicate with another Reducer?
Nope, MapReduce programming model does not allow reducers to communicate with each other. Reducers run in isolation.

Q.171- Can I set the number of Reducers to Zero?
Yes, Setting the number of reducers to zero is a valid configuration in Hadoop. When you set the reducers to zero no reducers will be executed, and the output of each mapper will be stored to a separate file on HDFS.This is different from the condition when reducers are set to a number greater than zero and the Mappers output (intermediate data) is written to the Local file system(NOT HDFS) of each master slave node.

Q.172- Where is the Mapper Output intermediate Key value data stored?
The mapper output (intermediate data) is stored on the Local file system (NOT HDFS) of each individual mapper node. This is typically a temporary directory location which can be setup in config by the hadoop administrator. The intermediate data is cleaned up after the Hadoop Job completes.

Grow Data Skills

Q.173- If Reducers do not start before all Mappers finish then why does the progress on Mapreduce Job shows something like Map 50 percent Reduce 10 percent and why Reducers progress percentage is displayed when Mapper is not Finished yet?
Reducers start copying intermediate key-value pairs from the mappers as soon as they are available. The progress calculation also takes into account the processing of data transfer which is done by reduce process, therefore the reduced progress starts showing up as soon as any intermediate key-value pair for a mapper is available to be transferred to reducer. Though the reducer progress is still updated, the programmer-defined reduce method is called only after all the mappers have finished.

Q.174- Explain in brief the three Modes in which Hadoop can be run?
The three modes in which Hadoop can be run are:

- Standalone (local) mode - No Hadoop daemons running, everything runs on a single Java Virtual machine only.
- Pseudo-distributed mode - Daemons run on the local machine, thereby simulating a cluster on a smaller scale.
- Fully distributed mode - Runs on a cluster of machines.

Q.175- Explain what are the features of Standalone local Mode?
In stand-alone or local mode there are no Hadoop daemons running, and everything runs on a single Java process. Hence, we don't get the benefit of distributing the code across a cluster of machines. Since it has no DFS, it utilizes the local file system. This mode is suitable only for running MapReduce programs by developers during various stages of development. It's the best environment for learning and good for debugging purposes.

Q.176- What are the features of fully distributed mode?
In Fully Distributed mode, the clusters range from a few nodes to 'n' number of nodes. It is used in production environments, where we have thousands of machines in the Hadoop cluster. The daemons of Hadoop run on these clusters. We have to configure separate masters and separate slaves in this distribution, the implementation of which is quite complex. In this configuration, Namenode and Datanode run on different hosts and there are nodes on which the task tracker runs. The root of the distribution is referred to as HADOOP_HOME.

Q.177- Explain what are the main features Of pseudo mode?
In Pseudo-distributed mode, each Hadoop daemon runs in a separate Java process, as such it simulates a cluster though on a small scale. This mode is used both for development and QA environments. Here, we need to do the configuration changes.

Q.178- What are the port numbers of NameNode and JobTracker and TaskTracker?
The port number for Namenode is ’70′, for job tracker is ’30′ and for task tracker is ’60′.

Q.179- Tell us what is a spill factor with respect to the ram?
Spill factor is the size after which your files move to the temp file. Hadoop-tmp directory is used for this. Default value for io.sort.spill.percent is 0.80. A value less than 0.5 is not recommended.

Grow Data Skills

Q.180- Is fs.mapr working for a single directory?
Yes, fs.mapr.working.dir is just one directory.

Q.181- Which are the three main Hdfs site.xml properties?
The three main hdfs-site.xml properties are:

- Dfs.name.dir which gives you the location on which metadata will be stored and where DFS is located – on disk or onto the remote.
- Dfs.data.dir which gives you the location where the data is going to be stored.
- Fs.checkpoint.dir which is for secondary Namenode.

Q.182-. How can I restart Namenode?
Click on stop-all.sh and then click on start-all.sh OR
Write sudo hdfs (press enter), su-hdfs (press enter), /etc/init.d/ha (press enter) and then /etc/init.d/hadoop-0.20-namenode start (press enter).

Q.183- How can we check whether Namenode is working or not?
To check whether Namenode is working or not, use the command /etc/init.d/hadoop-0.20-namenode status or as simple as jps'.

Q.184- At times you get a connection refused Java Exception when you run the file system check command Hadoop fsck?
The most possible reason is that the Namenode is not working on your VM.

Q.185- What is the use of the command Mapred.job.tracker?
The command mapred.job.tracker is used by the Job Tracker to list out which host and port that the MapReduce job tracker runs at. If it is "local", then jobs are run in-process as a single map and reduce tasks.

Q.186- What does etc.init.d do?
/etc /init.d specifies where daemons (services) are placed or to see the status of these daemons. It is very LINUX specific, and has nothing to do with Hadoop.

Q.187- How can we look for the Namenode in the browser?
If you have to look for Namenode in the browser, you don't have to give localhost: 8021, the port number to look for Namenode in the browser is 50070.

Q.188- What do masters and slaves consist of?
Masters contain a list of hosts, one per line, that are to host secondary namenode servers. Slaves consist of a list of hosts, one per line, that host datanode and task tracker servers.

Q.189- What is the function Of Hadoop-env.sh and where is it present?
This file contains some environment variable settings used by Hadoop; it provides the environment for Hadoop to run. The path of JAVA_HOME is set here for it to run properly. Hadoop-env.sh file is present in the conf/hadoop-env.sh location. You can also create your own custom configuration file conf/hadoop-user-env.sh, which will allow you to override the default Hadoop settings.

Grow Data Skills

Q.190- Can we have multiple entries in the master files?
Yes, we can have multiple entries in the Master files.

Q.191- In Hadoop_pid_dir and what does pid stands for?
PID stands for 'Process ID'.

Q.192- What do Hadoop metrics and properties files do?
Hadoop-metrics Properties is used for 'Reporting Purposes. It controls the reporting for hadoop. The default status is 'not to report'.

Q.193- What are the network requirements for hadoop?
The Hadoop core uses Shell (SSH) to launch the server processes on the slave nodes. It requires password-less SSH connection between the master and all the slaves and the Secondary machines.

Q.194- Why do we need a passwordless ssh in a fully distributed environment?
We need a password-less SSH in a Fully-Distributed environment because when the cluster is LIVE and running in a Fully Distributed environment, the communication is too frequent. The job tracker should be able to send a task to the task tracker quickly.

Q.195- What will happen if a NameNode has no data?
If a Namenode has no data it cannot be considered as a Namenode. In practical terms, Namenode needs to have some data.

Q.196- What happens to the job tracker when NameNode is down?
Namenode is the main point which keeps all the metadata, keeping track of failure of datanode with the help of heart beats. As such when a namenode is down, your cluster will be completely down, because Namenode is the single point of failure in a Hadoop Installation.

Q.197- Explain what you mean by formatting the Dfs?
Like we do in Windows, DFS is formatted for proper structuring of data. It is not usually recommended to do as it formats the Namenode too in the process, which is not desired.

Q.198- We use Unix variants for hadoop and can we use Microsoft Windows for the same?
In practicality, Ubuntu and Red Hat Linux are the best Operating Systems for Hadoop. On the other hand, Windows can be used but it is not used frequently for installing Hadoop as there are many support problems related to it. The frequency of crashes and the subsequent restarts makes it unattractive. As such, Windows is not recommended as a preferred environment for Hadoop Installation, though users can give it a try for learning purposes in the initial stage.

Q.199- Which one decides the input split hdfs client or NameNode?
The HDFS Client does not decide. It is already specified in one of the configurations through which input split is already configured.

Grow Data Skills

Q.200- Can you tell me if we can create a hadoop cluster from scratch?
Yes, we can definitely do that. Once we become familiar with the Apache Hadoop environment, we can create a cluster from scratch.

Q.201- Explain the significance of ssh and what is the port on which port does ssh work and why do we need password in ssh localhost?
SSH is a secure shell communication, is a secure protocol and the most common way of administering remote servers safely, relatively very simple and inexpensive to implement. A single SSH connection can host multiple channels and hence can transfer data in both directions. SSH works on Port No. 22, and it is the default port number. However, it can be configured to point to a new port number, but it's not recommended. In a local host, password is required in SSH for security and in a situation where password less communication is not set.

Q.202- What is ssh and explain in detail about ssh communication between masters and the slaves?
Secure Socket Shell or SSH is a password-less secure communication that provides administrators with a secure way to access a remote computer and data packets are sent across the slave. This network protocol also has some format into which data is sent across. SSH communication is not only between masters and slaves but also between two hosts in a network. SSH appeared in 1995 with the introduction of SSH - 1. Now SSH 2 is in use, with the vulnerabilities coming to the fore when Edward Snowden leaked information by decrypting some SSH traffic.

Q.203- Can you tell what will happen to a NameNode and when the Job tracker is not up and running?
When the job tracker is down, it will not be in functional mode, all running jobs will be halted because it is a single point of failure. Your whole cluster will be down but still Namenode will be present. As such the cluster will still be accessible if Namenode is working, even if the job tracker is not up and running. But you cannot run your Hadoop job.

Grow Data Skills