HDFS Interview Questions - HDFS

1. What are the different vendor-specific distributions of Hadoop?

The different vendor-specific distributions of Hadoop are Cloudera, MAPR, Amazon EMR, Microsoft Azure, IBM InfoSphere, and Hortonworks (Cloudera).

2. What are the different Hadoop configuration files?

The different Hadoop configuration files include:

- hadoop-env.sh
- mapred-site.xml
- core-site.xml
- yarn-site.xml
- hdfs-site.xml
- Master and Slaves

3. What are the three modes in which Hadoop can run?

The three modes in which Hadoop can run are :

1. Standalone mode: This is the default mode. It uses the local FileSystem and a single Java process to run the Hadoop services.
2. Pseudo-distributed mode: This uses a single-node Hadoop deployment to execute all Hadoop services.
3. Fully-distributed mode: This uses separate nodes to run Hadoop master and slave services.

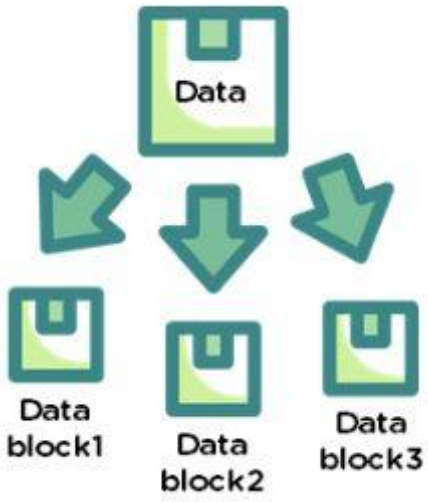
4. What are the differences between regular FileSystem and HDFS?

1. Regular FileSystem: In regular FileSystem, data is maintained in a single system. If the machine crashes, data recovery is challenging due to low fault tolerance. Seek time is more and hence it takes more time to process the data.
2. HDFS: Data is distributed and maintained on multiple systems. If a DataNode crashes, data can still be recovered from other nodes in the

cluster. Time taken to read data is comparatively more, as there is local data read to the disc and coordination of data from multiple systems.

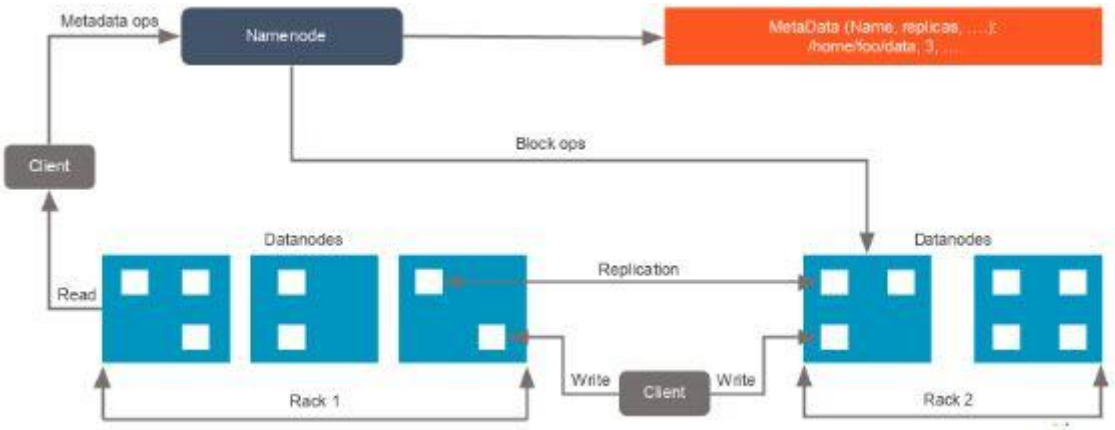
5. Why is HDFS fault-tolerant?

HDFS is fault-tolerant because it replicates data on different DataNodes. By default, a block of data is replicated on three DataNodes. The data blocks are stored in different DataNodes. If one node crashes, the data can still be retrieved from other DataNodes.



6. Explain the architecture of HDFS.

The architecture of HDFS is as shown:



For an HDFS service, we have a NameNode that has the master process running on one of the machines and DataNodes, which are the slave nodes.

NameNode

NameNode is the master service that hosts metadata in disk and RAM. It holds information about the various DataNodes, their location, the size of each block, etc.

DataNode

DataNodes hold the actual data blocks and send block reports to the NameNode every 10 seconds. The DataNode stores and retrieves the blocks when the NameNode asks. It reads and writes the client's request and performs block creation, deletion, and replication based on instructions from the NameNode.

- Data that is written to HDFS is split into blocks, depending on its size. The blocks are randomly distributed across the nodes. With the auto-replication feature, these blocks are auto-replicated across multiple machines with the condition that no two identical blocks can sit on the same machine.
- As soon as the cluster comes up, the DataNodes start sending their heartbeats to the NameNodes every three seconds. The NameNode stores this information; in other words, it starts building metadata in RAM, which contains information about the DataNodes available in the beginning. This metadata is maintained in RAM, as well as in the disk.

7. What are the two types of metadata that a NameNode server holds?

The two types of metadata that a NameNode server holds are:

- Metadata in Disk - This contains the edit log and the FSImage
- Metadata in RAM - This contains the information about DataNodes

8. What is the difference between a federation and high availability?

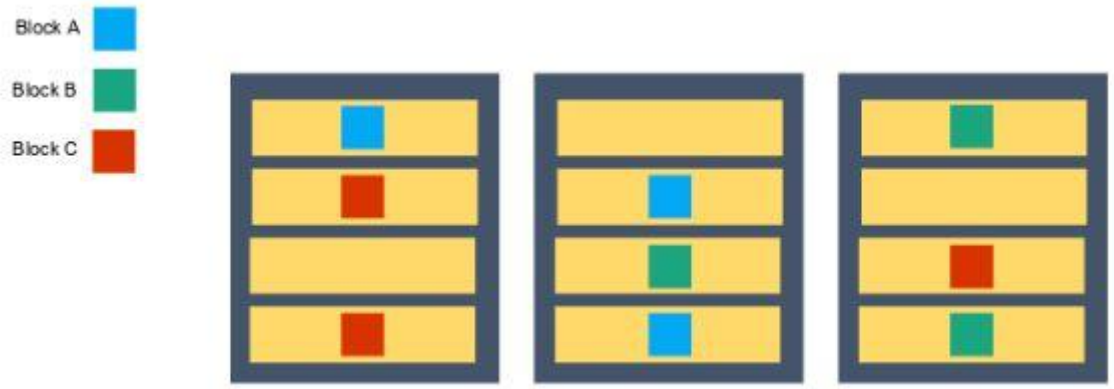
HDFS Federation	HDFS High Availability
• There is no limitation to the number of NameNodes and the NameNodes are not related to each other • All the NameNodes share a pool of metadata in which each NameNode will have its dedicated pool • Provides fault tolerance, i.e., if one NameNode goes down, it will not affect the data of the other NameNode	• There are two NameNodes that are related to each other. Both active and standby NameNodes work all the time • One at a time, active NameNodes will be up and running, while standby NameNodes will be idle and updating its metadata once in a while • It requires two separate machines. First, the active NameNode will be configured, while the secondary NameNode will be configured on the other system

9. If you have an input file of 350 MB, how many input splits would HDFS create and what would be the size of each input split?

By default, each block in HDFS is divided into 128 MB. The size of all the blocks, except the last block, will be 128 MB. For an input file of 350 MB, there are three input splits in total. The size of each split is 128 MB, 128MB, and 94 MB.

10. How does rack awareness work in HDFS?

HDFS Rack Awareness refers to the knowledge of different DataNodes and how it is distributed across the racks of a Hadoop Cluster.



By default, each block of data is replicated three times on various DataNodes present on different racks. Two identical blocks cannot be placed on the same DataNode. When a cluster is “rack-aware,” all the replicas of a block cannot be placed on the same rack. If a DataNode crashes, you can retrieve the data block from different DataNodes.

11. How can you restart NameNode and all the daemons in Hadoop?

The following commands will help you restart NameNode and all the daemons:

You can stop the NameNode with ./sbin /Hadoop-daemon.sh stop NameNode command and then start the NameNode using ./sbin/Hadoop-daemon.sh start NameNode command.

You can stop all the daemons with ./sbin /stop-all.sh command and then start the daemons using the ./sbin/start-all.sh command.

12. Which command will help you find the status of blocks and FileSystem health?

To check the status of the blocks, use the command:

hdfs fsck <path> -files -blocks

To check the health status of FileSystem, use the command:

hdfs fsck / -files –blocks –locations > dfs-fsck.log

13. What would happen if you store too many small files in a cluster on HDFS?

Storing several small files on HDFS generates a lot of metadata files. To store these metadata in the RAM is a challenge as each file, block, or directory takes 150 bytes for metadata. Thus, the cumulative size of all the metadata will be too large.

14. How do you copy data from the local system onto HDFS?

The following command will copy data from the local file system onto HDFS:

hadoop fs –copyFromLocal [source] [destination]

Example: hadoop fs –copyFromLocal /tmp/data.csv /user/test/data.csv

In the above syntax, the source is the local path and destination is the HDFS path. Copy from the local system using a -f option (flag option), which allows you to write the same file or a new file to HDFS.

15. When do you use the dfsadmin -refreshNodes and rmadmin -refreshNodes commands?

The commands below are used to refresh the node information while commissioning, or when the decommissioning of nodes is completed.

dfsadmin -refreshNodes

This is used to run the HDFS client and it refreshes node configuration for the NameNode.

rmadmin -refreshNodes

This is used to perform administrative tasks for ResourceManager.

16. Is there any way to change the replication of files on HDFS after they are already written to HDFS?

Yes, the following are ways to change the replication of files on HDFS:

We can change the dfs.replication value to a particular number in the $HADOOP_HOME/conf/hadoop-site.xml file, which will start replicating to the factor of that number for any new content that comes in.

If you want to change the replication factor for a particular file or directory, use:

$HADOOP_HOME/bin/Hadoop dfs –setrep –w4 /path of the file

Example: $HADOOP_HOME/bin/Hadoop dfs –setrep –w4 /user/temp/test.csv

17. Who takes care of replication consistency in a Hadoop cluster and what do under/over replicated blocks mean?

In a cluster, it is always the NameNode that takes care of the replication consistency. The fsck command provides information regarding the over and under-replicated block.

Under-replicated blocks:

These are the blocks that do not meet their target replication for the files they belong to. HDFS will automatically create new replicas of under-replicated blocks until they meet the target replication.

Consider a cluster with three nodes and replication set to three. At any point, if one of the NameNodes crashes, the blocks would be under-replicated. It means that there was a replication factor set, but there are not enough replicas as per the replication factor. If the NameNode does not get information about the replicas, it will wait for a limited amount of time and then start the re-replication of missing blocks from the available nodes.

Over-replicated blocks:

These are the blocks that exceed their target replication for the files they belong to. Usually, over-replication is not a problem, and HDFS will automatically delete excess replicas.

Consider a case of three nodes running with the replication of three, and one of the nodes goes down due to a network failure. Within a few minutes, the NameNode re-replicates the data, and then the failed node is back with its set of blocks. This is an over-replication situation, and the NameNode will delete a set of blocks from one of the nodes.

MapReduce Interview Questions

18. What is the distributed cache in MapReduce?

A distributed cache is a mechanism wherein the data coming from the disk can be cached and made available for all worker nodes. When a MapReduce program is running, instead of reading the data from the disk every time, it would pick up the data from the distributed cache to benefit the MapReduce processing.

To copy the file to HDFS, you can use the command:

hdfs dfs-put /user/Simplilearn/lib/jar_file.jar

To set up the application's JobConf, use the command:

DistributedCache.addFileToClasspath(newpath(“/user/Simplilearn/lib/jar_file.jar”), conf)

Then, add it to the driver class.

19. What role do RecordReader, Combiner, and Partitioner play in a MapReduce operation?

RecordReader

This communicates with the InputSplit and converts the data into key-value pairs suitable for the mapper to read.

Combiner

This is an optional phase; it is like a mini reducer. The combiner receives data from the map tasks, works on it, and then passes its output to the reducer phase.

Partitioner

The partitioner decides how many reduced tasks would be used to summarize the data. It also confirms how outputs from combiners are sent to the reducer, and controls the partitioning of keys of the intermediate map outputs.

20. Why is MapReduce slower in processing data in comparison to other processing frameworks?

This is quite a common question in Hadoop interviews; let us understand why MapReduce is slower in comparison to the other processing frameworks:

MapReduce is slower because:

- It is batch-oriented when it comes to processing data. Here, no matter what, you would have to provide the mapper and reducer functions to work on data.
- During processing, whenever the mapper function delivers an output, it will be written to HDFS and the underlying disks. This data will be shuffled and sorted, and then be picked up for the reducing phase. The entire process of writing data to HDFS and retrieving it from HDFS makes MapReduce a lengthier process.

- In addition to the above reasons, MapReduce also uses Java language, which is difficult to program as it has multiple lines of code.

21. Is it possible to change the number of mappers to be created in a MapReduce job?

By default, you cannot change the number of mappers, because it is equal to the number of input splits. However, there are different ways in which you can either set a property or customize the code to change the number of mappers.

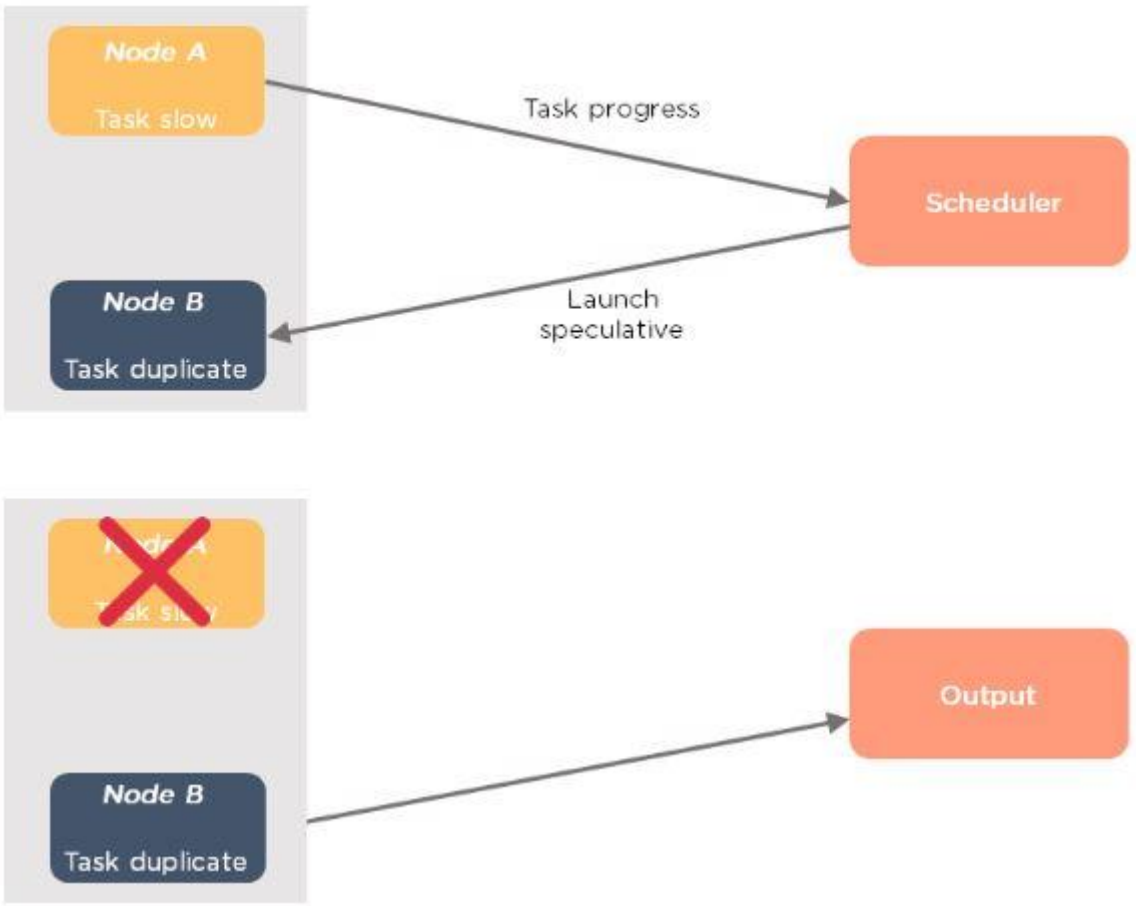
For example, if you have a 1GB file that is split into eight blocks (of 128MB each), there will only be only eight mappers running on the cluster. However, there are different ways in which you can either set a property or customize the code to change the number of mappers.

22. Name some Hadoop-specific data types that are used in a MapReduce program.

This is an important question, as you would need to know the different data types if you are getting into the field of Big Data.

For every data type in Java, you have an equivalent in Hadoop. Therefore, the following are some Hadoop-specific data types that you could use in your MapReduce program:

- IntWritable
- FloatWritable
- LongWritable
- DoubleWritable
- BooleanWritable
- ArrayWritable
- MapWritable
- ObjectWritable



23. What is speculative execution in Hadoop?

If a DataNode is executing any task slowly, the master node can redundantly execute another instance of the same task on another node. The task that finishes first will be accepted, and the other task would be killed. Therefore, speculative execution is useful if you are working in an intensive workload kind of environment.

The following image depicts the speculative execution:

From the above example, you can see that node A has a slower task. A scheduler maintains the resources available, and with speculative execution turned on, a copy of the slower task runs on node B. If node A task is slower, then the output is accepted from node B.

24. How is identity mapper different from chain mapper?

Identity Mapper	Chain Mapper
• This is the default mapper that is chosen when no mapper is specified in the MapReduce driver class. • It implements identity function, which directly writes all its key-value pairs into output. • It is defined in old MapReduce API (MR1) in: org.apache.Hadoop.mapred.lib.package	• This class is used to run multiple mappers in a single map task. • The output of the first mapper becomes the input to the second mapper, second to third and so on. • It is defined in: org.apache.Hadoop.mapreduce.lib.chain.ChainMapperpackage

25. What are the major configuration parameters required in a MapReduce program?

We need to have the following configuration parameters:

- Input location of the job in HDFS
- Output location of the job in HDFS
- Input and output formats
- Classes containing a map and reduce functions
- JAR file for mapper, reducer and driver classes

26. What do you mean by map-side join and reduce-side join in MapReduce?

Map-side join	Reduce-side join
• The mapper performs the join • Each input data must be divided into the same number of partitions • Input to each map is in the form of a structured partition and is in sorted order	• The reducer performs the join • Easier to implement than the map side join, as the sorting and shuffling phase sends the value with identical keys to the same reducer • No need to have the dataset in a structured form (or partitioned)

27. What is the role of the OutputCommitter class in a MapReduce job?

As the name indicates, OutputCommitter describes the commit of task output for a MapReduce job.

Example: org.apache.hadoop.mapreduce.OutputCommitter

public abstract class OutputCommitter extends OutputCommitter

MapReduce relies on the OutputCommitter for the following:

- Set up the job initialization
- Cleaning up the job after the job completion
- Set up the task's temporary output

- Check whether a task needs a commit
- Committing the task output
- Discard the task commit

28. Explain the process of spilling in MapReduce.

Spilling is a process of copying the data from memory buffer to disk when the buffer usage reaches a specific threshold size. This happens when there is not enough memory to fit all of the mapper output. By default, a background thread starts spilling the content from memory to disk after 80 percent of the buffer size is filled.

For a 100 MB size buffer, the spilling will start after the content of the buffer reaches a size of 80 MB.

29. How can you set the mappers and reducers for a MapReduce job?

The number of mappers and reducers can be set in the command line using:

-D mapred.map.tasks=5 –D mapred.reduce.tasks=2

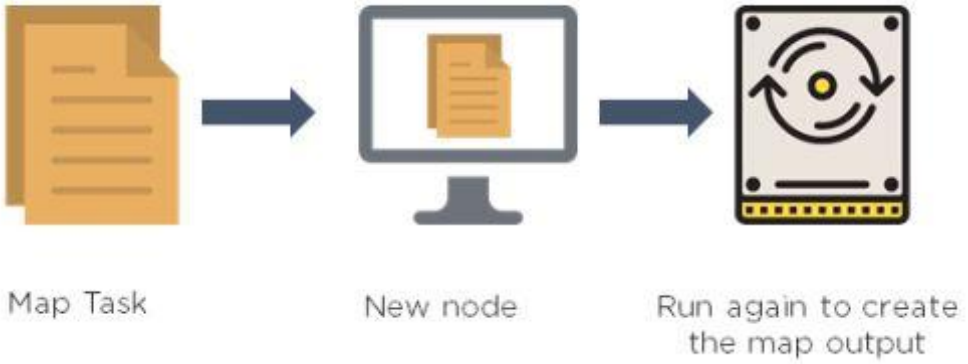
In the code, one can configure JobConf variables:

job.setNumMapTasks(5); // 5 mappers

job.setNumReduceTasks(2); // 2 reducers

30. What happens when a node running a map task fails before sending the output to the reducer?

If this ever happens, map tasks will be assigned to a new node, and the entire task will be rerun to re-create the map output. In Hadoop v2, the YARN framework has a temporary daemon called application master, which takes care of the execution of the application. If a task on a particular node failed due to the unavailability of a node, it is the role of the application master to have this task scheduled on another node.



31. Can we write the output of MapReduce in different formats?

Yes. Hadoop supports various input and output File formats, such as:

- TextOutputFormat - This is the default output format and it writes records as lines of text.
- SequenceFileOutputFormat - This is used to write sequence files when the output files need to be fed into another MapReduce job as input files.
- MapFileOutputFormat - This is used to write the output as map files.
- SequenceFileAsBinaryOutputFormat - This is another variant of SequenceFileInputFormat. It writes keys and values to a sequence file in binary format.
- DBOutputFormat - This is used for writing to relational databases and HBase. This format also sends the reduce output to a SQL table.

Hadoop Interview Questions - YARN

Now, let's learn about resource management and the job scheduling unit in Hadoop, which is YARN (Yet Another Resource Negotiator).

32. What benefits did YARN bring in Hadoop 2.0 and how did it solve the issues of MapReduce v1?

In Hadoop v1, MapReduce performed both data processing and resource management; there was only one master process for the processing layer known as JobTracker. JobTracker was responsible for resource tracking and job scheduling.

Managing jobs using a single JobTracker and utilization of computational resources was inefficient in MapReduce 1. As a result, JobTracker was overburdened due to handling, job scheduling, and resource management. Some of the issues were scalability, availability issue, and resource utilization. In addition to these issues, the other problem was that non-MapReduce jobs couldn't run in v1.

To overcome this issue, Hadoop 2 introduced YARN as the processing layer. In YARN, there is a processing master called ResourceManager. In Hadoop v2, you have ResourceManager running in high availability mode. There are node managers running on multiple machines, and a temporary daemon called application master. Here, the ResourceManager is only handling the client connections and taking care of tracking the resources.

In Hadoop v2, the following features are available:

- Scalability - You can have a cluster size of more than 10,000 nodes and you can run more than 100,000 concurrent tasks.
- Compatibility - The applications developed for Hadoop v1 run on YARN without any disruption or availability issues.
- Resource utilization - YARN allows the dynamic allocation of cluster resources to improve resource utilization.
- Multitenancy - YARN can use open-source and proprietary data access engines, as well as perform real-time analysis and run ad-hoc queries.

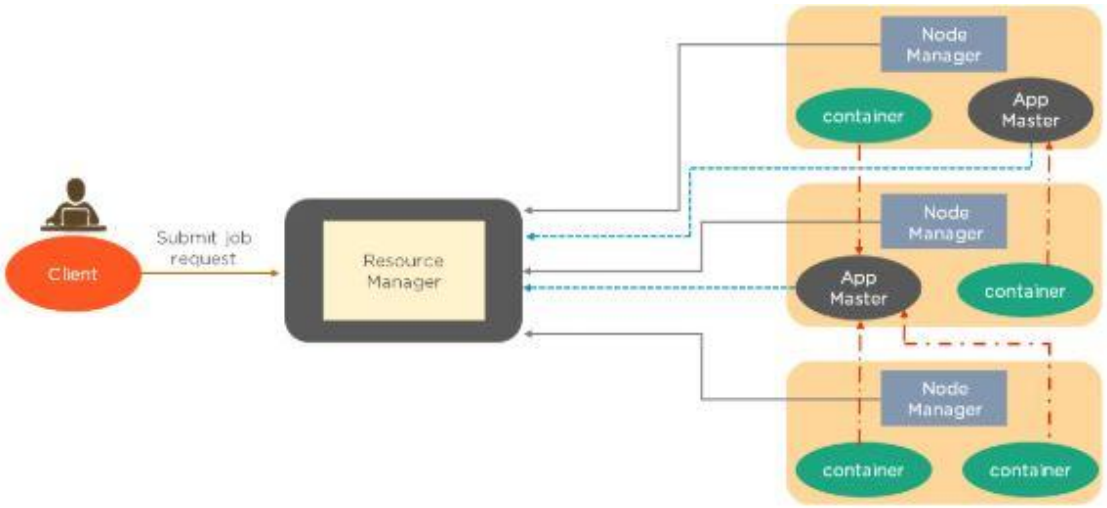
33. Explain how YARN allocates resources to an application with the help of its architecture.

There is a client/application/API which talks to ResourceManager. The ResourceManager manages the resource allocation in the cluster. It has two internal components, scheduler, and application manager. The ResourceManager is aware of the resources that are available with every node manager. The scheduler allocates resources to various running applications when they are running in parallel. It schedules resources based on the requirements of the applications. It does not monitor or track the status of the applications.

Applications Manager is what accepts job submissions. It monitors and restarts the application masters in case of failures. Application Master manages the resource needs of individual applications. It interacts with the scheduler to acquire the required resources, and with NodeManager to execute and monitor tasks, which tracks the jobs running. It monitors each container's resource utilization.

A container is a collection of resources, such as RAM, CPU, or network bandwidth. It provides the rights to an application to use a specific amount of resources.

Let us have a look at the architecture of YARN:



Whenever a job submission happens, ResourceManager requests the NodeManager to hold some resources for processing. NodeManager then guarantees the container that would be available for processing. Next, the ResourceManager starts a temporary daemon called application master to take care of the execution. The App Master, which the applications manager launches, will run in one of the containers. The other containers will be utilized for execution. This is briefly how YARN takes care of the allocation.

34. Which of the following has replaced JobTracker from MapReduce v1?

1. NodeManager
2. ApplicationManager
3. ResourceManager
4. Scheduler

The answer is ResourceManager. It is the name of the master process in Hadoop v2.

35. Write the YARN commands to check the status of an application and kill an application.

The commands are as follows:

a) To check the status of an application:

yarn application -status ApplicationID

b) To kill or terminate an application:

yarn application -kill ApplicationID

36. Can we have more than one ResourceManager in a YARN-based cluster?

Yes, Hadoop v2 allows us to have more than one ResourceManager. You can have a high availability YARN cluster where you can have an active ResourceManager and a standby ResourceManager, where the ZooKeeper handles the coordination.

There can only be one active ResourceManager at a time. If an active ResourceManager fails, then the standby ResourceManager comes to the rescue.

37. What are the different schedulers available in YARN?

The different schedulers available in YARN are:

- FIFO scheduler - This places applications in a queue and runs them in the order of submission (first in, first out). It is not desirable, as a long-running application might block the small running applications
- Capacity scheduler - A separate dedicated queue allows the small job to start as soon as it is submitted. The large job finishes later compared to using the FIFO scheduler
- Fair scheduler - There is no need to reserve a set amount of capacity since it will dynamically balance resources between all the running jobs

38. What happens if a ResourceManager fails while executing an application in a high availability cluster?

In a high availability cluster, there are two ResourceManagers: one active and the other standby. If a ResourceManager fails in the case of a high availability cluster, the standby will be elected as active and instructs the ApplicationMaster to abort. The ResourceManager recovers its running state by taking advantage of the container statuses sent from all node managers.

39. In a cluster of 10 DataNodes, each having 16 GB RAM and 10 cores, what would be the total processing capacity of the cluster?

Every node in a Hadoop cluster will have one or multiple processes running, which would need RAM. The machine itself, which has a Linux file system, would have its own processes that need a specific amount of RAM usage. Therefore, if you have 10 DataNodes, you need to allocate at least 20 to 30 percent towards the overheads, Cloudera-based services, etc. You could have 11 or 12 GB and six or seven cores available on every machine for processing. Multiply that by 10, and that's your processing capacity.

40. What happens if requested memory or CPU cores go beyond the size of container allocation?

If an application starts demanding more memory or more CPU cores that cannot fit into a container allocation, your application will fail. This happens because the requested memory is more than the maximum container size.

Now that you have learned about HDFS, MapReduce, and YARN, let us move to the next section. We'll go over questions about Hive, Pig, HBase, and Sqoop.

Hadoop Interview Questions - HIVE

Before moving into the Hive interview questions, let us summarize what Hive is all about. Facebook adopted the Hive to overcome MapReduce's limitations. MapReduce proved to be difficult for users as they found it challenging to code because not all of them were well-versed with the coding languages. Users required a language similar to SQL, which was well-known to all the users. This gave rise to Hive.

Hive is a data warehouse system used to query and analyze large datasets stored in HDFS. It uses a query language called HiveQL, which is similar to SQL. Hive works on structured data. Let us now have a look at a few Hive questions.

41. What are the different components of a Hive architecture?

The different components of the Hive are:

- User Interface: This calls the execute interface to the driver and creates a session for the query. Then, it sends the query to the compiler to generate an execution plan for it
- Metastore: This stores the metadata information and sends it to the compiler for the execution of a query
- Compiler: This generates the execution plan. It has a DAG of stages, where each stage is either a metadata operation, a map, or reduces a job or operation on HDFS
- Execution Engine: This acts as a bridge between the Hive and Hadoop to process the query. Execution Engine communicates bidirectionally with Metastore to perform operations, such as create or drop tables.

42. What is the difference between an external table and a managed table in Hive?

External Table	Managed Table
• External tables in Hive refer to the data that is at an existing location outside the warehouse directory • Hive deletes the metadata information of a table and does not change the table data present in HDFS	• Also known as the internal table, these types of tables manage the data and move it into its warehouse directory by default • If one drops a managed table, the metadata information along with the table data is deleted from the Hive warehouse directory

43. What is a partition in Hive and why is partitioning required in Hive

Partition is a process for grouping similar types of data together based on columns or partition keys. Each table can have one or more partition keys to identify a particular partition.

Partitioning provides granularity in a Hive table. It reduces the query latency by scanning only relevant partitioned data instead of the entire data set. We can partition the transaction data for a bank based on month — January, February, etc. Any operation regarding a particular month, say February, will only have to scan the February partition, rather than the entire table data.

44. Why does Hive not store metadata information in HDFS?

We know that the Hive's data is stored in HDFS. However, the metadata is either stored locally or it is stored in RDBMS. The metadata is not stored in HDFS, because HDFS read/write operations are time-consuming. As such, Hive stores metadata

information in the metastore using RDBMS instead of HDFS. This allows us to achieve low latency and is faster.

45. What are the components used in Hive query processors?

The components used in Hive query processors are:

- Parser
- Semantic Analyzer
- Execution Engine
- User-Defined Functions
- Logical Plan Generation
- Physical Plan Generation
- Optimizer
- Operators
- Type checking

46. Suppose there are several small CSV files present in /user/input directory in HDFS and you want to create a single Hive table from these files. The data in these files have the following fields: {registration_no, name, email, address}. What will be your approach to solve this, and where will you create a single Hive table for multiple smaller files without degrading the performance of the system?

Using SequenceFile format and grouping these small files together to form a single sequence file can solve this problem. Below are the steps:

```
> read.table(file = "data.table", header = TRUE)
   name age gender
1  john  23   male
2  mary  21 female
3 jacob  18   male
4 nancy  25 female
```

47. Write a query to insert a new column(new_col INT) into a hive table (h_table) at a position before an existing column (x_col).

The following query will insert a new column:

ALTER TABLE h_table

CHANGE COLUMN new_col INT

BEFORE x_col

48. What are the key differences between Hive and Pig?

Hive	Pig
• It uses a declarative language, called HiveQL, which is similar to SQL for reporting. • Operates on the server-side of the cluster and allows structured data. • It does not support the Avro file format by default. This can be done using "Org.Apache.Hadoop.Hive.serde2.Avro" • Facebook developed it and it supports partition	• Uses a high-level procedural language called Pig Latin for programming • Operates on the client-side of the cluster and allows both structured and unstructured data • Supports Avro file format by default. • Yahoo developed it, and it does not support partition

Hadoop Interview Questions - PIG

Pig is a scripting language that is used for data processing. Let us have a look at a few questions involving Pig:

49. How is Apache Pig different from MapReduce?

Pig	MapReduce
• It has fewer lines of code compared to MapReduce. • A high-level language that can easily perform join operation. • On execution, every Pig operator is converted internally into a MapReduce job • Works with all versions of Hadoop	• Has more lines of code. • A low-level language that cannot perform join operation easily. • MapReduce jobs take more time to compile. • A MapReduce program written in one Hadoop version may not work with other versions

50. What are the different ways of executing a Pig script?

The different ways of executing a Pig script are as follows:

- Grunt shell
- Script file
- Embedded script

51. What are the major components of a Pig execution environment?

The major components of a Pig execution environment are:

- Pig Scripts: They are written in Pig Latin using built-in operators and UDFs, and submitted to the execution environment.
- Parser: Completes type checking and checks the syntax of the script. The output of the parser is a Directed Acyclic Graph (DAG).
- Optimizer: Performs optimization using merge, transform, split, etc. Optimizer aims to reduce the amount of data in the pipeline.
- Compiler: Converts the optimized code into MapReduce jobs automatically.
- Execution Engine: MapReduce jobs are submitted to execution engines to generate the desired results.

52. Explain the different complex data types in Pig.

Pig has three complex data types, which are primarily Tuple, Bag, and Map.

Tuple

A tuple is an ordered set of fields that can contain different data types for each field. It is represented by braces ().

Example: (1,3)

Bag

A bag is a set of tuples represented by curly braces {}.

Example: {(1,4), (3,5), (4,6)}

Map

A map is a set of key-value pairs used to represent data elements. It is represented in square brackets [ ].

Example: [key#value, key1#value1,….]

53. What are the various diagnostic operators available in Apache Pig?

Pig has Dump, Describe, Explain, and Illustrate as the various diagnostic operators.

Dump

The dump operator runs the Pig Latin scripts and displays the results on the screen.

Load the data using the “load” operator into Pig.

Display the results using the “dump” operator.

Describe

Describe operator is used to view the schema of a relation.

Load the data using “load” operator into Pig

View the schema of a relation using “describe” operator

Explain

Explain operator displays the physical, logical and MapReduce execution plans.

Load the data using “load” operator into Pig

Display the logical, physical and MapReduce execution plans using “explain” operator

Illustrate

Illustrate operator gives the step-by-step execution of a sequence of statements.

Load the data using “load” operator into Pig

Show the step-by-step execution of a sequence of statements using “illustrate” operator

54. State the usage of the group, order by, and distinct keywords in Pig scripts.

The group statement collects various records with the same key and groups the data in one or more relations.

Example: Group_data = GROUP Relation_name BY AGE

The order statement is used to display the contents of relation in sorted order based on one or more fields.

Example: Relation_2 = ORDER Relation_name1 BY (ASC|DSC)

Distinct statement removes duplicate records and is implemented only on entire records, and not on individual records.

Example: Relation_2 = DISTINCT Relation_name1

55. What are the relational operators in Pig?

The relational operators in Pig are as follows:

COGROUP

It joins two or more tables and then performs GROUP operation on the joined table result.

CROSS

This is used to compute the cross product (cartesian product) of two or more relations.

FOREACH

This will iterate through the tuples of a relation, generating a data transformation.

JOIN

This is used to join two or more tables in a relation.

LIMIT

This will limit the number of output tuples.

SPLIT

This will split the relation into two or more relations.

UNION

It will merge the contents of two relations.

ORDER

This is used to sort a relation based on one or more fields.

56. What is the use of having filters in Apache Pig?

FilterOperator is used to select the required tuples from a relation based on a condition. It also allows you to remove unwanted records from the data file.

Example: Filter the products with a whole quantity that is greater than 1000

A = LOAD '/user/Hadoop/phone_sales' USING PigStorage(',') AS (year:int, product:chararray, quantity:int);

B = FILTER A BY quantity > 1000

57. Suppose there's a file called "test.txt" having 150 records in HDFS. Write the PIG command to retrieve the first 10 records of the file.

To do this, we need to use the limit operator to retrieve the first 10 records from a file.

Load the data in Pig:

test_data = LOAD "/user/test.txt" USING PigStorage(',') as (field1, field2,….);

Limit the data to first 10 records:

Limit_test_data = LIMIT test_data 10;

Hadoop Interview Questions - HBase

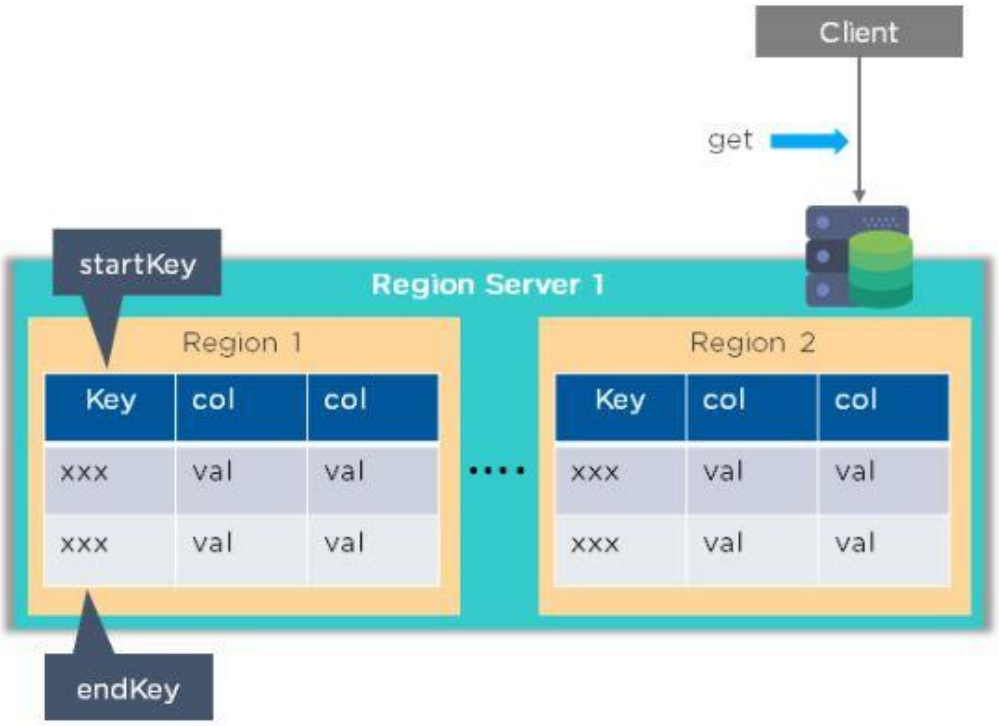
Now let's have a look at questions from HBase. HBase is a NoSQL database that runs on top of Hadoop. It is a four-dimensional database in comparison to RDBMS databases, which are usually two-dimensional.

58. What are the key components of HBase?

This is one of the most common interview questions.

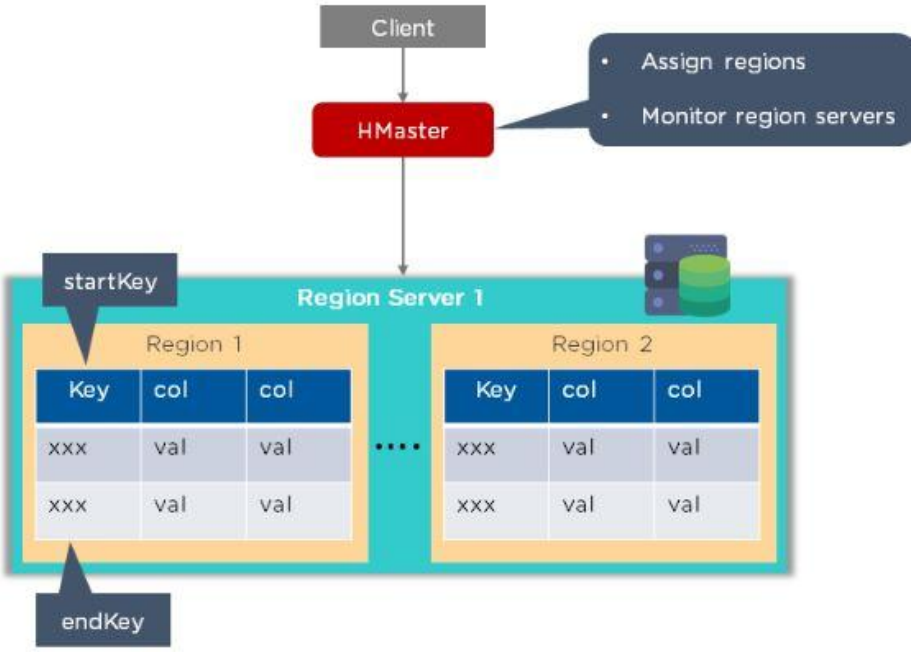
Region Server

Region server contains HBase tables that are divided horizontally into "Regions" based on their key values. It runs on every node and decides the size of the region. Each region server is a worker node that handles read, writes, updates, and delete request from clients.



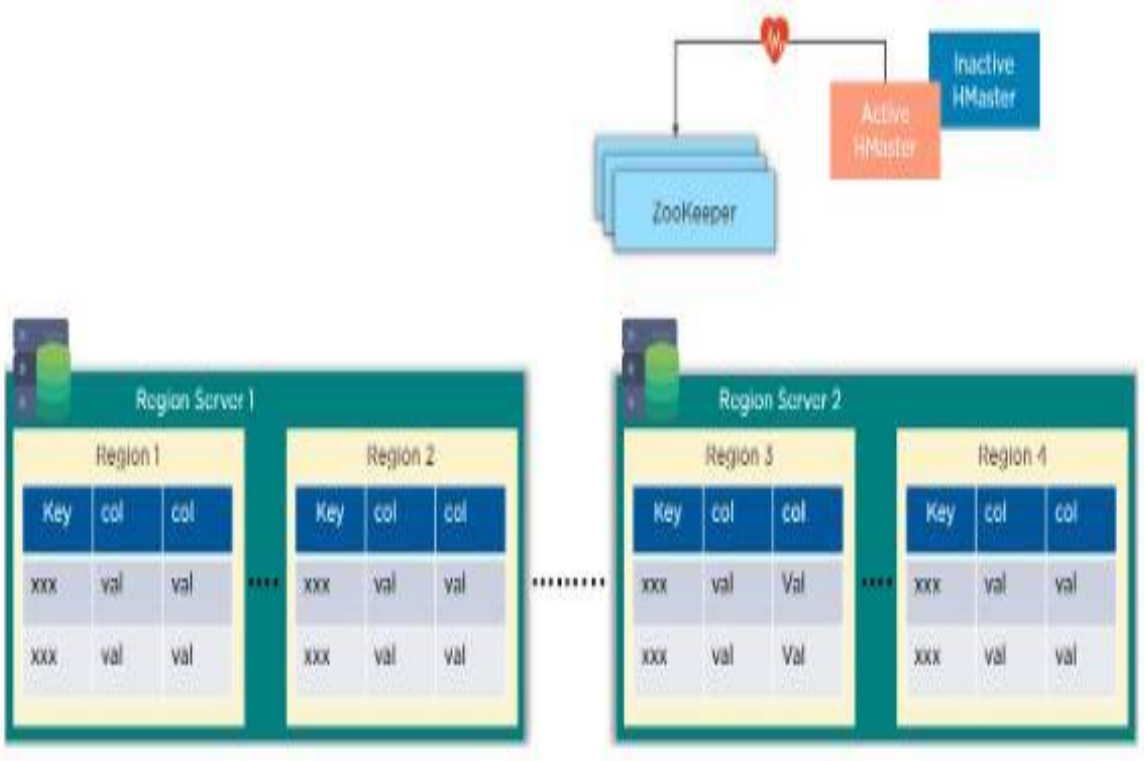
HMaster

This assigns regions to RegionServers for load balancing, and monitors and manages the Hadoop cluster. Whenever a client wants to change the schema and any of the metadata operations, HMaster is used.



ZooKeeper

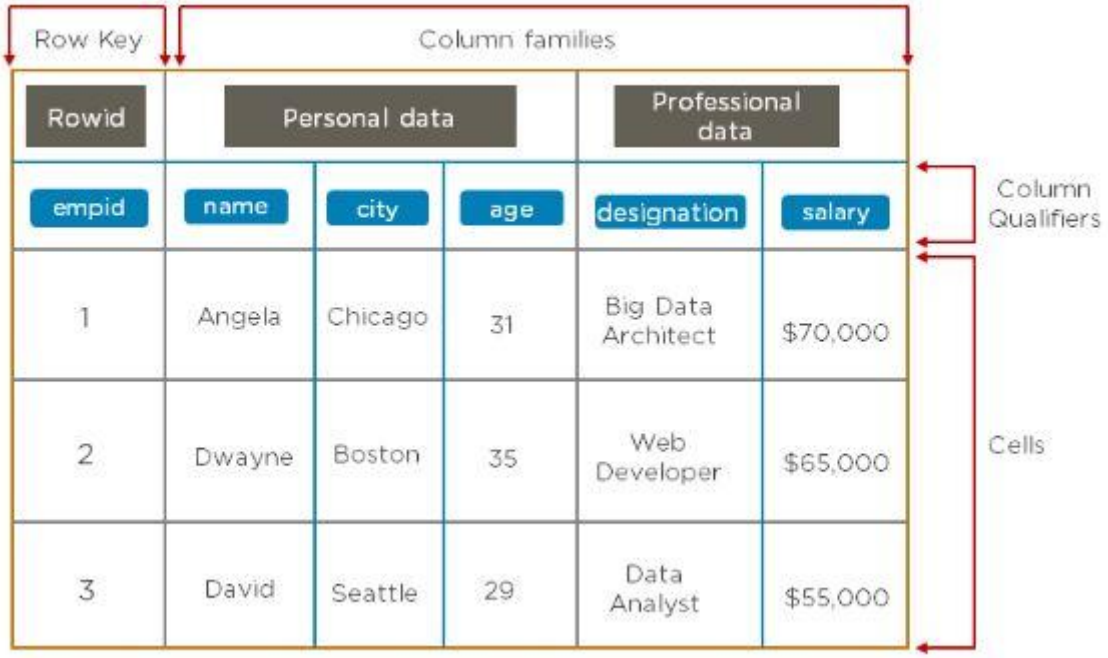
This provides a distributed coordination service to maintain server state in the cluster. It looks into which servers are alive and available, and provides server failure notifications. Region servers send their statuses to ZooKeeper indicating if they are ready to reading and write operations.



59. Explain what row key and column families in HBase is.

The row key is a primary key for an HBase table. It also allows logical grouping of cells and ensures that all cells with the same row key are located on the same server.

Column families consist of a group of columns that are defined during table creation, and each column family has certain column qualifiers that a delimiter separates.



60. Why do we need to disable a table in HBase and how do you do it?

The HBase table is disabled to allow modifications to its settings. When a table is disabled, it cannot be accessed through the scan command.

Employee_table

Rowid	Personal data			Professional data	
empid	name	city	age	designation	salary
1	Angela	Chicago	31	Big Data Architect	$70,000
2	Dwayne	Boston	35	Web Developer	$65,000
3	David	Seattle	29	Data Analyst	$55,000

To disable the employee table, use the command:

disable ‘employee_table’

To check if the table is disabled, use the command:

is_disabled ‘employee_table’

61. Write the code needed to open a connection in HBase.

The following code is used to open a connection in HBase:

Configuration myConf = HBaseConfiguration.create();

HTableInterface usersTable = new HTable(myConf, “users”);

62. What does replication mean in terms of HBase?

The replication feature in HBase provides a mechanism to copy data between clusters. This feature can be used as a disaster recovery solution that provides high availability for HBase.

The following commands alter the hbase1 table and set the replication_scope to 1. A replication_scope of 0 indicates that the table is not replicated.

disable ‘hbase1’

alter ‘hbase1’, {NAME => ‘family_name’, REPLICATION_SCOPE => ‘1’}

enable ‘hbase1’

63. Can you import/export in an HBase table?

Yes, it is possible to import and export tables from one HBase cluster to another.

HBase export utility:

hbase org.apache.hadoop.hbase.mapreduce.Export “table name” “target export location”

Example: hbase org.apache.hadoop.hbase.mapreduce.Export “employee_table” “/export/employee_table”

HBase import utility:

create ‘emp_table_import’, {NAME => ‘myfam’, VERSIONS => 10}

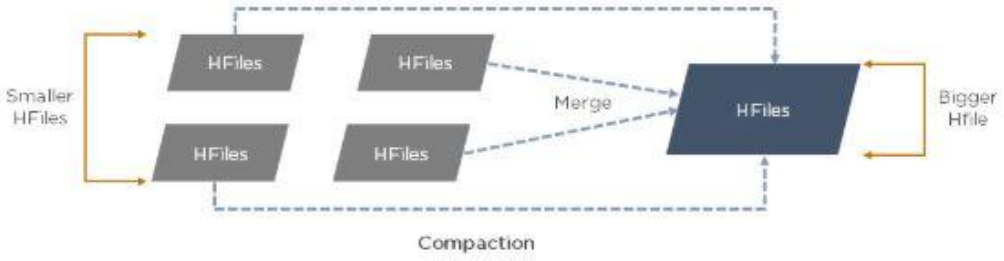
hbase org.apache.hadoop.hbase.mapreduce.Import “table name” “target import location”

Example: create ‘emp_table_import’, {NAME => ‘myfam’, VERSIONS => 10}

hbase org.apache.hadoop.hbase.mapreduce.Import “emp_table_import” “/export/employee_table”

64. What is compaction in HBase?

Compaction is the process of merging HBase files into a single file. This is done to reduce the amount of memory required to store the files and the number of disk seeks needed. Once the files are merged, the original files are deleted.



65. How does Bloom filter work?

The HBase Bloom filter is a mechanism to test whether an HFile contains a specific row or row-col cell. The Bloom filter is named after its creator, Burton Howard Bloom. It is a data structure that predicts whether a given element is a member of a set of data. These filters provide an in-memory index structure that reduces disk reads and determines the probability of finding a row in a particular file.

66. Does HBase have any concept of the namespace?

A namespace is a logical grouping of tables, analogous to a database in RDBMS. You can create the HBase namespace to the schema of the RDBMS database.

To create a namespace, use the command:

create_namespace ‘namespace name’

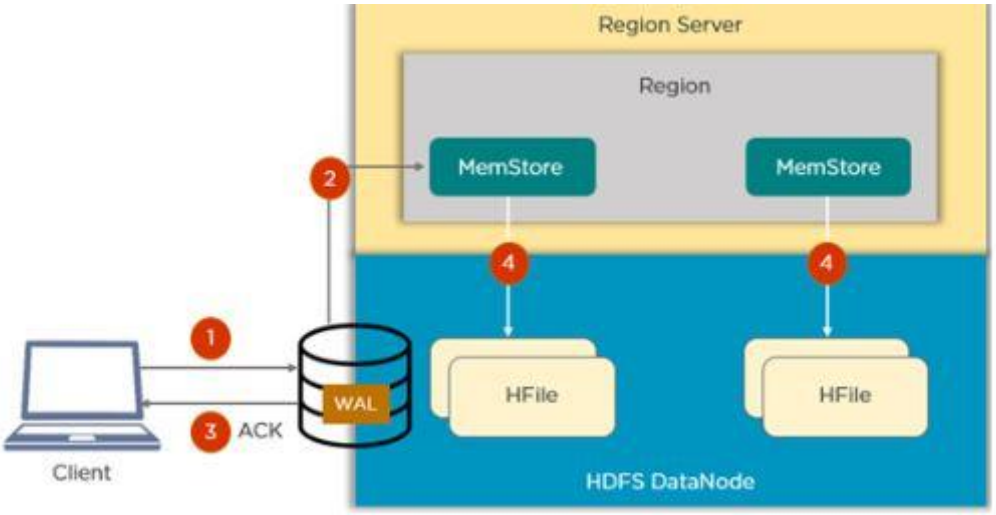
To list all the tables that are members of the namespace, use the command:
list_namespace_tables ‘default’

To list all the namespaces, use the command:

list_namespace

67. How does the Write Ahead Log (WAL) help when a RegionServer crashes?

If a RegionServer hosting a MemStore crash, the data that existed in memory, but not yet persisted, is lost. HBase recovers against that by writing to the WAL before the write completes. The HBase cluster keeps a WAL to record changes as they happen. If HBase goes down, replaying the WAL will recover data that was not yet flushed from the MemStore to the HFile.



68. Write the HBase command to list the contents and update the column families of a table.

The following code is used to list the contents of an HBase table:

scan ‘table_name’

Example: scan ‘employee_table’

To update column families in the table, use the following command:

alter ‘table_name’, ‘column_family_name’

Example: alter ‘employee_table’, ‘emp_address’

69. What are catalog tables in HBase?

The catalog has two tables: hbasemeta and -ROOT-

The catalog table hbase:meta exists as an HBase table and is filtered out of the HBase shell’s list command. It keeps a list of all the regions in the system and the location of hbase:meta is

stored in ZooKeeper. The -ROOT- table keeps track of the location of the .META table.

70. What is hotspotting in HBase and how can it be avoided?

In HBase, all read and write requests should be uniformly distributed across all of the regions in the RegionServers. Hotspotting occurs when a given region serviced by a single RegionServer receives most or all of the read or write requests.

Hotspotting can be avoided by designing the row key in such a way that data is written should go to multiple regions across the cluster. Below are the techniques to do so:

- Salting
- Hashing
- Reversing the key