Hadoop

Q & A

COMPILED BY

CHAKRABORTY, SHUBHAM

TURAGA, PRABHANJ

Join our Group

https://goo.gl/forms/Pl1S2qdhnySSYXkG2

Hadoop

1. What is Big Data ?

Big data is vast amount of data (generally in GBs or TBs of size) that exceeds the regular processing capacity of the traditional computing servers and requires special parallel processing mechanism. This data is too big and its rate of increase gets accelerated. This data can be either structural or unstructured data which may not be able to process by legacy databases.

2. What is Hadoop ?

Hadoop is an open source frame work from Apache Software Foundation for storing & processing large-scale data usually called Big Data using clusters of commodity hardware.

3. Who uses Hadoop ?

Big organizations in which data grows exponentially day by day and must require Hadoop platform to process such huge data. For example , Facebook, Google, Amazon, Twitter, IBM, LinkedIn etc… companies uses hadoop technology to solve their big data processing problems.

4. What is commodity hardware?

Commodity hardware is a non-expensive system which is not of high quality or high-availability.

Hadoop can be installed on any commodity hardware. We don't need super computers or high-end hardware to work on Hadoop. Commodity hardware includes RAM because there will be some services which will be running on RAM.

5. 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 amount of data in the distributed file system and process it.

RDBMS will be useful when we want to seek one record from Big data, whereas, Hadoop will be useful when we want Big data in one shot and perform analysis on that later.

6. What are the modes in which Hadoop can run ?

Hadoop can run in three modes.

- **Stand alone or Local mode –** No daemons will be running in this mode and everything runs in a single JVM.
- **Pseudo distributed mode –** All the Hadoop daemons run on a local machine, simulating cluster on a small scale.

- **Fully distributed mode –** A cluster of machines will be setup in master/slaves architecture to distribute and process the data across various nodes of commodity hardware.
-

7. What are main components/projects in Hadoop architecture ?

- **Hadoop Common:** The common utilities that support the other Hadoop modules.
- **HDFS:** Hadoop distributed file system that provides high-throughput access to application data.
- **Hadoop YARN:** A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce:** A YARN-based system for parallel processing of large data sets.

8. List important default configuration files in Hadoop cluster ?

The default configuration files in hadoop cluster are:

- **core-default.xml**
- **hdfs-default.xml**
- **yarn-default.xml**
- **mapred-default.xml**

9. List important site-specific configuration files in Hadoop cluster ?

In order to override any hadoop configuration property's default values, we need to provide configuration values in site-specific configuration files. Below are the four site-specific .xml configuration files and environment variable setup file.

- **core-site.xml** : Common properties are configured in this file.
- **hdfs-site.xml** : Site specific hdfs properties are configured in this file
- **yarn-site.xml** : Yarn specific properties can be provided in this file.
- **mapred-site.xml** : Mapreduce framework specific properties will defined here.
- **hadoop-env.sh** : Hadoop environment variables are setup in this file.

All these configuration files should be placed in hadoop's configuration directory **etc/hadoop** from **hadoop's home directory**.

10. How many hadoop daemon processes run on a Hadoop System ?

As of hadoop-2.5.0 release, three hadoop daemon processes run on a hadoop cluster.

- **NameNode daemon** – Only one daemon runs for entire hadoop cluster.
- **Secondary NameNode daemon** – Only one daemon runs for entire hadoop cluster.
- **DataNode daemon** – One datanode daemon per each datanode in hadoop cluster

11. How to start all hadoop daemons at a time ?

$ **start-dfs.sh** command can be used to start all hadoop daemons from terminal at a time.

12. If some hadoop daemons are already running and if we need to start any one remaining daemon process then what are the commands to use ?

Instead of **start-dfs.sh** which will trigger all the hadoop three daemons at a time, we can also **start running each daemon separately** by the below commands.

Hadoop daemon start commands

```
$ hadoop-daemon.sh start namenode
$ hadoop-daemon.sh start secondarynamenode
$ hadoop-daemon.sh start datanode
```

13. How to stop all the three hadoop daemons at a time ?

By using stop-dfs.sh command, we can stop the above three daemon processes with a single command.

14. What are commands that need to be used to bring down a single hadoop daemon?

Below hadoop-daemon.sh commands can be used to bring down each hadoop daemon separately.

Hadoop daemon start commands

```
$ hadoop-daemon.sh stop namenode
$ hadoop-daemon.sh stop secondarynamenode
$ hadoop-daemon.sh stop datanode
```

15. How many YARN daemon processes run on a cluster ?

Two types of Yarn daemons will be running on hadoop cluster in master/slave fashion.

- **ResourceManager** – Master daemon process
- **NodeManager** – One Slave daemon process per node in a cluster.

16. How to start Yarn daemon processes on a hadoop cluster ?

By running **$ start-yarn.sh** command from terminal on each machine on hadoop cluster, Yarn daemons can be started.

17. How to verify whether the daemon processes are running or not ?

By using java's processes command **$ jps** to check what are all the java processes running on a machine. This command lists down all the daemon processes running on a machine along with their process ids.

18. How to bring down the Yarn daemon processes ?

Using **$ stop-yarn.sh** command, we can bring down both the Yarn daemon processes running on a machine.

19. Can we start both Hadoop daemon processes and Yarn daemon processes with a single command?

Yes, we can start all the above mentioned five daemon processes (3 hadoop + 2 Yarn) with a single command **$ start-all.sh**

20. Can we stop all the above five daemon processes with a single command ?

Yes, by using **$ stop-all.sh** command all the above five daemon processes can be bring down in a single shot.

21. Which operating systems are supported for Hadoop deployment ?

The only supported operating system for hadoop's production deployment is Linux. However, with some additional software Hadoop can be deployed on Windows for test environments.

22. How could be the various components of Hadoop cluster deployed in production?

Both Name Node and Resource Manager can be deployed on a Master Node, and Data nodes and node managers can be deployed on multiple slave nodes.

There is a need for only one master node for namenode and Resource Manager on the system. The number of slave nodes for datanodes & node managers depends on the size of the cluster.

One more node with hardware specifications same as master node will be needed for secondary namenode.

23. 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.

24. Is Namenode also a commodity?

No. Namenode can never be a commodity hardware because the entire HDFS rely on it. It is the single point of failure in HDFS. Namenode has to be a high-availability machine.

25. What is the difference between jps and jps -lm commands ?

jps command returns the process id and short names for running processes. But jps -lm returns long messages along with process id and short names as shown below.

```
hadoop1@ubuntu-1:~$ jps
5314 SecondaryNameNode
5121 DataNode
5458 Jps
4995 NameNode
```

```
hadoop1@ubuntu-1:~$ jps -lm
5314
org.apache.hadoop.hdfs.server.namenode.SecondaryNameNode
5121 org.apache.hadoop.hdfs.server.datanode.DataNode
5473 sun.tools.jps.Jps -lm
4995 org.apache.hadoop.hdfs.server.namenode.NameNode
```

26. What is HDFS ?

HDFS is a distributed file system implemented on Hadoop's framework. It is a block-structured distributed file system designed to store vast amount of data on low cost commodity hardware and ensuring high speed process on data.

HDFS stores files across multiple machines and maintains reliability and fault tolerance. HDFS support parallel processing of data by Mapreduce framework.

27. What are the objectives of HDFS file system?

- Easily Store large amount of data across multiple machines
- Data reliability and fault tolerance by maintaining multiple copies of each block of a file.
- Capacity to move computation to data instead of moving data to computation server. I.e. processing data locally.
- Able to provide parallel processing of data by Mapreduce framework.
-

28. What are the limitations of HDFS file systems?

- HDFS supports file operations reads, writes, appends and deletes efficiently but it doesn't support file updates.
- HDFS is not suitable for large number of small sized files but best suits for large sized files. Because file system namespace maintained by Namenode is limited by it's main memory capacity as namespace is stored in namenode's main memory and large number of files will result in big fsimage file.

29. What is a block in HDFS and what is its size?

It is a fixed size chunk of data usually of size 128 MB. It is the minimum of size of data that HDFS can read/write.

HDFS files are broken into these fixed size chunks of data across multiple machines on a cluster.

Thus, blocks are building bricks of a HDFS file. Each block is maintained in at least 3 copies as mentioned by replication factor in Hadoop configuration to provide data redundancy and maintain fault-tolerance.

30. What are the core components in HDFS Cluster ?

- **Name Node**
- **Secondary Name Node**
- **Data Nodes**
- **Checkpoint Nodes**
- **Backup Node**

31. What is a NameNode ?

Namenode is a dedicated machine in HDFS cluster which acts as a master serve that maintains file system namespace in its main memory and serves the file access requests by users. File system namespace mainly contains fsimage and edits files. Fsimage is a file which contains file names, file permissions and block locations of each file.

Usually only one active namenode is allowed in HDFS default architecture.

32. What is a DataNode ?

DataNodes are slave nodes of HDFS architecture which store the blocks of HDFS files and sends blocks information to namenode periodically through heart beat messages.

Data Nodes serve read and write requests of clients on HDFS files and also perform block creation, replication and deletions.

33. Is Namenode machine same as datanode machine as in terms of hardware?

It depends upon the cluster we are trying to create. The Hadoop VM can be there on the same machine or on another machine. For instance, in a single node cluster, there is only one machine,whereas in the development or in a testing environment, Namenode and datanodes are on different machines.

34. What is a Secondary Namenode?

The Secondary NameNode is a helper to the primary NameNode. Secondary NameNode is a specially dedicated node in HDFS cluster whose main function is to take checkpoints of the file system metadata present on namenode. It is not a backup namenode and doesn’t act as a namenode in case of primary namenode’s failures. It just checkpoints namenode’s file system namespace.

35. What is a Checkpoint Node?

It is an enhanced secondary namenode whose main functionality is to take checkpoints of namenode’s file system metadata periodically. It replaces the role of secondary namenode. Advantage of Checkpoint node over the secondary namenode is that it can upload the result of merge operation of fsimage and edits log files while checkpointing.

36. What is a checkpoint ?

During Checkpointing process, fsimage file is merged with edits log file and a new fsimage file will be created which is usually called as a checkpoint.

37. What is a daemon?

Daemon is a process or service that runs in background. In general, we use this word in UNIX environment. Hadoop or Yarn daemons are Java processes which can be verified with jps command.

38. 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 heart beat then they will decide that there is some problem in datanode or task tracker is unable to perform the assigned task.

39. Are Namenode and Resource Manager run on the same host?

No, in practical environment, Namenode runs on a separate host and Resource Manager runs on a separate host.

40. What is the communication mode between namenode and datanode?

The mode of communication is SSH.

41. If we want to copy 20 blocks from one machine to another, but another machine can copy only 18.5 blocks, can the blocks be broken at the time of replication?

In HDFS, blocks cannot be broken down. Before copying the blocks from one machine to another, the Master node will figure out what is the actual amount of space required, how many block are being used, how much space is available, and it will allocate the blocks accordingly.

42. 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.

43. If a data Node is full, then 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.

44. 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 arise.

45. 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 because it might result in data written by one node can be overwritten by other.

For example, we 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.

46. What is fsimage and edit log in hadoop?

Fsimage is a file which contains file names, file permissions and block locations of each file, and this file is maintained by Namenode for indexing of files in HDFS. We can call it as metadata about HDFS files. The fsimage file contains a serialized form of all the directory and file inodes in the filesystem.

EditLog is a transaction log which contains records for every change that occurs to file system metadata.

Note: Whenever a NameNode is restarted, the latest status of FsImage is built by applying edits records on last saved copy of FsImage

47. What is a Backup Node?

It is an extended checkpoint node that performs checkpointing and also supports online streaming of file system edits.

It maintains an in memory, up-to-date copy of file system namespace and accepts a real time online stream of file system edits and applies these edits on its own copy of namespace in its main memory.

Thus, it maintains always a latest backup of current file system namespace.

48. What are the differences between Backup Node and Checkpoint node or Secondary Namenode ?

- Multiple checkpoint nodes can be registered with namenode but only a single backup node is allowed to register with namenode at any point of time.
- To create a checkpoint, checkpoint node or secondary namenode needs to download fsimage and edits files from active namenode and apply edits to fsimage and saves a copy of new fsimage as a checkpoint.

But in backup node, no need to download fsimage and edits files from active namenode because, it already has an up-to-date copy of fsimage in its main memory and accesses online streaming of edits which are provided by namenode. So, applying these edits into fsimage in its own main memory and saving a copy in local FS.

So, checkpoint creation in backup node is faster than that of checkpoint node or secondary namenode.

- The diff between checkpoint node and secondary namenode is that checkpoint node can upload the new copy of fsimage file back to namenode after checkpoint creation where as a secondary namenode can't upload but can only store in its local FS.
- Backup node provides the option of running namenode with no persistent storage but a checkpoint node or secondary namenode doesn't provide such option.

- In case of namenode failures, data loss in checkpoint node or secondary namenode is certain at least to a minimum amount of data due to time gap between two checkpoints.

But in backup node, data loss is not certain and it maintains namespace which is in sync with namenode at any point of time.

49. What is Safe Mode in HDFS ?

Safe Mode is a maintenance state of NameNode during which Name Node doesn't allow any changes to the file system.

During Safe Mode, HDFS cluster is ready-only and doesn't replicate or delete blocks.

Name Node automatically enters safe mode during its start up and maintain blocks replication value within minimum and maximum allowable replication limit.

50. What is Data Locality in HDFS ?

One of the HDFS design idea is that "Moving Computation is cheaper than Moving data".

If data sets are huge, running applications on nodes where the actual data resides will give efficient results than moving data to nodes where applications are running.

This concept of moving applications to data, is called Data Locality.

This reduces network traffic and increases speed of data processing and accuracy of data since there is no chance of data loss during data transfer through network channels because there is no need to move data.

51. What is a rack?

Rack is a storage area with all the datanodes put together. These datanodes 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.

52. What is Rack Awareness ?

The concept of maintaining Rack Id information by NameNode and using these rack ids for choosing closest data nodes for HDFS file read or writes requests is called Rack Awareness.

By choosing closest data nodes for read/writes request through rack awareness policy, minimizes the write cost and maximizing read speed.

53. How does HDFS File Deletes or Undeletes work?

When a file is deleted from HDFS, it will not be removed immediately from HDFS, but HDFS moves the file into /trash directory. After certain period of time interval, the NameNode deletes the file from the HDFS /trash directory. The deletion of a file releases the blocks associated with the file.

Time interval for which a file remains in /trash directory can be configured with fs.trash.interval property stored in core-site.xml.

As long as a file remains in /trash directory, the file can be undeleted by moving the file from /trash directory into required location in HDFS. Default trash interval is set to 0. So, HDFS Deletes file without storing in trash.

54. What is a Rebalancer in HDFS ?

Rebalancer is a administration tool in HDFS, to balance the distribution of blocks uniformly across all the data nodes in the cluster.

Rebalancing will be done on demand only. It will not get triggered automatically.

HDFS administrator issues this command on request to balance the cluster

```
$ hdfs balancer
```

If a Rebalancer is triggered, NameNode will scan entire data node list and when

- Under-utilized data node is found, it moves blocks from over-utilized data nodes or not-under-utilized data nodes to this current data node
- If Over-utilized data node is found, it moves blocks from this data node to other under-utilized or not-over-utilized data nodes.

55. What is the need for Rebalancer in HDFS ?

Whenever a new data node is added to the existing HDFS cluster or a data node is removed from the cluster then some of the data nodes in the cluster will have more/less blocks compared to other data nodes.

In this unbalanced cluster, data read/write requests become very busy on some data nodes and some data nodes are under utilized.

In such cases, to make all the data nodes space is uniformly utilized for blocks distribution, rebalancing will be done by Hadoop Administrator.

56. When will be a cluster in balanced status ?

A cluster is in a balanced status when, % of space used in each data node is within limits of Average % of space used on data nodes +/- Threshold size .

Percentage space used on a data node should not be less than Average % of space used on data nodes – Threshold size.

Percentage space used on a data node should not be greater than Average % of space used on data nodes + Threshold size.

Here Threshold size is configurable value which is 20 % of used spaced by default.

57. What is Delegation Token in Hadoop ?

Delegation token is an authentication token used to access to secure Namenode from a non-secure client node. HDFS fetchdt command can be used to get the delegation token and store it in a file on the local system. its common syntax is as follows.

fetchdt command

```
$ hadoop fetchdt fileName
```

This fetchdt uses either RPC or HTTPS (over Kerberos) to get the token, and thus it requires kerberos tickets to be present before the fetchdt run.

Once the token is obtained then user can run an HDFS command without having Kerberos tickets, by pointing HADOOP_TOKEN_FILE_LOCATION environmental variable to the delegation token file.

58. What is Hadoop Streaming ?

Streaming is a generic API that allows programs written in virtually any language to be used as Hadoop Mapper and Reducer implementations

59. What is the characteristic of streaming API that makes it flexible run mapreduce jobs in languages like perl, ruby, awk etc. ?

Hadoop Streaming allows to use arbitrary programs for the Mapper and Reducer phases of a Map Reduce job by having both Mappers and Reducers receive their input on stdin and emit output (key, value) pairs on stdout.

60. What is a 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.

61. What is the lowest granularity at which you can apply replication factor in HDFS
– We can choose replication factor per directory
– We can choose replication factor per file in a directory-
– We can choose replication factor per block of a file

True
True
False

62. What is HBase?

A distributed, column-oriented database. HBase uses HDFS for its underlying storage, and supports both batch-style computations using MapReduce and random reads.

63. What is ZooKeeper?

A distributed, highly available coordination service. ZooKeeper provides primitives such as distributed locks that can be used for building distributed applications.

64. What is Chukwa?

A distributed data collection and analysis system. Chukwa runs collectors that store data in HDFS, and it uses MapReduce to produce reports.

65. What is Avro?

A data serialization system for efficient, cross-language RPC, and persistent data storage.

66. What is Sqoop in Hadoop?

It is a tool to transfer the data between Relational database management system(RDBMS) and Hadoop. Thus, we can sqoop the data from RDBMS like mySql or Oracle into HDFS , Hive or HBase as well as exporting data from HDFS, Hive or HBase to RDBMS. Sqoop will read the table row-by-row and the import process is performed in Parallel. Thus, the output may be in multiple files.

67. What is the default block size in HDFS ?

As of Hadoop-2.4.0 release, the default block size in HDFS is 256 MB and prior to that it was 128 MB.

68. What is the benefit of large block size in HDFS ?

Main benefit of large block size is Quick Seek Time. The time to transfer a large file of multiple blocks operates at the disk transfer rate instead of depending much on seek time.

69. What are the overheads of maintaining too large Block size ?

Usually in Mapreduce framework, each map task operate on one block at a time. So, having too few blocks result in too few map tasks running in parallel for longer time which finally results in overall slow down of job performance.

70. If a file of size 10 MB is copied on to HDFS of block size 256 MB, then how much storage will be allocated to the file on HDFS ?

Even though the default HDFS block size is 256 MB, a file which is smaller than a single block doesn’t occupy full block size. So, in this case, the file will occupy just 10 MB but not 256 MB.

71. What are the benefits of block structure concept in HDFS ?

- Main benefit is that the ability to store very large files which can be even larger than the size of single disk (node) as the file is broken into blocks and distributed across various nodes on cluster.
- Another important advantage is simplicity of storage management as the blocks are fixed size, it is easy to calculate how many can be stored on a given disk.
- Blocks replication feature is useful in fault tolerance.
-

72. What if we upgrade our Hadoop version in which, default block size is higher than the current Hadoop version's default block size. Suppose 128 MB (Hadoop 0.20.2) to 256 MB (Hadoop 2.4.0).

All the existing files are maintained at block size of 128 MB but any new files copied on to upgraded hadoop are broken into blocks of size 256 MB.

73. What is Block replication ?

Block replication is a way of maintaining multiple copies of same block across various nodes on cluster to achieve fault tolerance. In this, though one of the data node containing the block becomes dead, the block data can be obtained from other live data nodes which contain the same copy of the block data.

74. What is default replication factor and how to configure it ?

The default replication factor in fully distributed HDFS is 3.

This can be configured with dfs.replication in hdfs-site.xml file at site level.

Replication factor can be setup at file level with below FS command.

```
$ hadoop fs -setrep N /filename
```

In above command 'N' is the new replication factor for the file "/filename".

75. What is HDFS distributed copy (distcp) ?

distcp is an utility for launching MapReduce jobs to copy large amounts of HDFS files within or in between HDFS clusters.

Syntax for using this tool.

```
$ hadoop distcp hdfs://namenodeX/src hdfs://namenodeY/dest
```

76. What is the use of fsck command in HDFS ?

HDFS **fsck** command is a useful to get the files and blocks details of the file system. It's syntax is:

fsck command

$ hadoop fsck<path> [-move|-delete|-openforwrite][-files[-blocks[-locations|-racks]]]

below are the command options and their purpose.

-move	Move corrupted files to /lost+found
-delete	Delete corrupted files.
-openforwrite	Print out files opened for write.
-files	Print out files being checked.
-blocks	Print out block report.
-locations	Print out locations for every block.
-racks	Print out network topology for data-node locations.

77. As 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.

78. What happens if you get a ‘connection refused java exception’ when you type hadoop fs -ls /?

It could mean that the Namenode is not working on our hadoop cluster.

79. Can we have multiple entries in the master files?

Yes, we can have multiple entries in the Master files.

80. Why do we need a password-less SSH in Fully Distributed environment?

We need a password-less SSH in a Fully-Distributed environment because when the cluster is LIVE and running in Fully Distributed environment, the communication is too frequent. The Resource Manager/Namenode should be able to send a task to Node manager /Datanodes quickly.

81. Does this lead to security issues?

No, not at all. Hadoop cluster is an isolated cluster. And generally it has nothing to do with an internet. It has a different kind of a configuration. We needn’t worry about that kind of a security breach, for instance, someone hacking through the internet, and so on. Hadoop has implemented Kerberose for strong security way to connect to other machines to fetch and to process data.

82. which port does SSH work on?

SSH works on Port 22, though it can be configured. 22 is the default Port number.

83. If we only had 32 MB of memory how can we sort one terabyte of data?

Take a smaller chunk of data and sort it in memory, so we partition it in lots of little data sets. Then merge those sorted lists into one big list before writing the results back to disk.

84. How can we create an empty file in HDFS ?

We can create empty file with hadoop fs -touchz command. It creates a zero byte file.

$ hadoop fs -touchz /user/hadoop/filename

85. How can we see the contents of a Snappy compressed file or SequenceFile via command line?

With the help of $ hadoop fs -text /user/hadoop/filename command we can see the contents of sequencefile or any compressed format file in text format.

86. How can we check the existence of a file in HDFS ?

The hadoop test is used for file test operations. The syntax is shown below:

hadoop fs -test -[ezd] URI

Here “e” for checking the existence of a file, “z” for checking the file is zero length or not, “d” for checking the path is a directory or not. On success, the test command returns 1 else 0.

87. How can we set the replication factor of directory via command line?

Hadoop setrep is used to change the replication factor of a file. Use the -R option for recursively changing the replication factor.

$ hadoop fs -setrep -w 4 -R /user/hadoop/dir

88. How can we apply the permission to all the files under a HDFS directory recursively?

Using **$ hadoop fs -chmod -R 755 /user/hadoop/dir** command we can set the permissions to a directory recursively.

89. What is hadoop fs -stat command used for?

Hadoop stat returns the stats information of a file. It returns the last updated date and time. The syntax of stat is shown below:

```
$ hadoop fs -stat /filename
2014-09-28 18:34:06
```

90. What is expunge command in HDFS and why is it used for ?

Hadoop fs expunge command is used to empty the trash directory in HDFS. Syntax is:

```
$ hadoop fs –expunge
```

91. How can we see the output of a MR job as a single file if the reducer might have created multiple part-r-0000* files ?

We can use hadoop fs -getmerge command to combine all the part-r-0000* files into single file and this file can be browsed to view the entire results of the MR job at a time. Syntax is:

```
hadoop fs -getmerge &lt;hdfssrc&gt; &lt;localdestination&gt; [addnl]
```

The addnl option is for adding new line character at the end of each file.

92. Which of the following is most important when selecting hardware for our new Hadoop cluster?
1. The number of CPU cores and their speed.
2. The amount of physical memory.
3. The amount of storage.
4. The speed of the storage.
5. It depends on the most likely workload.

Answer 5 – Though some general guidelines are possible and we may need to generalize whether our cluster will be running a variety of jobs, the best fit depends on the anticipated workload.

93. Why would you likely not want to use network storage in your cluster?
1. Because it may introduce a new single point of failure.
2. Because it most likely has approaches to redundancy and fault-tolerance that may
be unnecessary given Hadoop's fault tolerance.
3. Because such a single device may have inferior performance to Hadoop's use of
multiple local disks simultaneously.
4. All of the above.

Answer 4: Network storage comes in many flavors but in many cases we may find a large Hadoop cluster of hundreds of hosts reliant on a single (or usually a pair of) storage devices. This adds a new failure scenario to the cluster and one with a less uncommon likelihood than many others. Where storage technology does look to address failure mitigation it is usually through disk-level redundancy.

94. We will be processing 10 TB of data on our cluster. Our main MapReduce job processes financial transactions, using them to produce statistical models of behavior and future forecasts. Which of the following hardware choices would be our first choice for the cluster?
1. 20 hosts each with fast dual-core processors, 4 GB memory, and one 500 GB
disk drive.
2. 30 hosts each with fast dual-core processors, 8 GB memory, and two 500 GB
disk drives.
3. 30 hosts each with fast quad-core processors, 8 GB memory, and one 1 TB disk drive.
4. 40 hosts each with 16 GB memory, fast quad-core processors, and four 1 TB
disk drives.

Answer 3. Probably! We would suggest avoiding the first configuration as, though it has just enough raw storage and is far from under powered, there is a good chance the setup will provide little room for growth. An increase in data volumes would immediately require new hosts and additional complexity in the MapReduce job could require additional processor power or memory.

Configurations B and C both look good as they have surplus storage for growth and provide similar head-room for both processor and memory. B will have the higher disk I/O and C the better CPU performance. Since the primary job is involved in financial modelling and forecasting, we expect each task to be reasonably heavyweight in terms of CPU and memory needs. Configuration B may have higher I/O

but if the processors are running at 100 percent utilization it is likely the extra disk throughput will not be used. So the hosts with greater processor power are likely the better fit.
Configuration D is more than adequate for the task and we don’t choose it for that very reason; why buy more capacity than we know we need?

95) How Hadoop MapReduce works?

In MapReduce, during the map phase it counts the words in each document, while in the reduce phase it aggregates the data as per the document spanning the entire collection. During the map phase the input data is divided into splits for analysis by map tasks running in parallel across Hadoop framework.

96) Explain what is shuffling in MapReduce ?

The process by which the system performs the sort and transfers the map outputs to the reducer as inputs is known as the shuffle

97) Explain what is distributed Cache in MapReduce Framework ?

Distributed Cache is an important feature provided by map reduce framework. When you want to share some files across all nodes in Hadoop Cluster, DistributedCache is used. The files could be an executable jar files or simple properties file.

98) Explain what is JobTracker in Hadoop? What are the actions followed by Hadoop?

In Hadoop for submitting and tracking MapReduce jobs, JobTracker is used. Job tracker run on its own JVM process

Hadoop performs following actions in Hadoop

- Client application submit jobs to the job tracker
- JobTracker communicates to the Namemode to determine data location
- Near the data or with available slots JobTracker locates TaskTracker nodes
- On chosen TaskTracker Nodes, it submits the work
- When a task fails, Job tracker notify and decides what to do then.
- The TaskTracker nodes are monitored by JobTracker

99) Explain what combiners is and when you should use a combiner in a MapReduce Job?

To increase the efficiency of MapReduce Program, Combiners are used. The amount of data can be reduced with the help of combiner's that need to be transferred across to the reducers. If the operation performed is commutative and associative you can use your reducer code as a combiner. The execution of combiner is not guaranteed in Hadoop

100) What happens when a data node fails ?

When a datanode fails

- Jobtracker and namenode detect the failure
- On the failed node all tasks are re-scheduled
- Namenode replicates the users data to another node

101) Explain what is Speculative Execution?

In Hadoop during Speculative Execution a certain number of duplicate tasks are launched. On different slave node, multiple copies of same map or reduce task can be executed using Speculative Execution. In simple words, if a particular drive is taking long time to complete a task, Hadoop will create a duplicate task on another disk. Disk that finish the task first are retained and disks that do not finish first are killed.

102) Explain what are the basic parameters of a Mapper?

The basic parameters of a Mapper are

- LongWritable and Text
- Text and IntWritable

103) Explain what is the function of MapReducer partitioner?

The function of MapReducer partitioner is to make sure that all the value of a single key goes to the same reducer, eventually which helps evenly distribution of the map output over the reducers

104) Explain what is difference between an Input Split and HDFS Block?

Logical division of data is known as Split while physical division of data is known as HDFS Block

105) Explain what happens in textinformat ?

In textinputformat, each line in the text file is a record. Value is the content of the line while Key is the byte offset of the line. For instance, Key: longWritable, Value: text

106) Mention what are the main configuration parameters that user need to specify to run Mapreduce Job ?

The user of Mapreduce framework needs to specify

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

107) Explain what is WebDAV in Hadoop?

To support editing and updating files WebDAV is a set of extensions to HTTP. On most operating system WebDAV shares can be mounted as filesystems , so it is possible to access HDFS as a standard filesystem by exposing HDFS over WebDAV.

108) Explain how JobTracker schedules a task ?

The task tracker send out heartbeat messages to Jobtracker usually every few minutes to make sure that JobTracker is active and functioning. The message also informs JobTracker about the number of available slots, so the JobTracker can stay upto date with where in the cluster work can be delegated

109) Explain what is Sequencefileinputformat?

Sequencefileinputformat is used for reading files in sequence. It is a specific compressed binary file format which is optimized for passing data between the output of one MapReduce job to the input of some other MapReduce job.

110) Explain what does the conf.setMapper Class do ?

Conf.setMapperclass sets the mapper class and all the stuff related to map job such as reading data and generating a key-value pair out of the mapper

111) Mention Hadoop core components?

Hadoop core components include,

- HDFS
- MapReduce

112) Mention what are the data components used by Hadoop?

Data components used by Hadoop are

- Pig
- Hive

113) Mention what is the data storage component used by Hadoop?

The data storage component used by Hadoop is HBase.

114) Mention what are the most common input formats defined in Hadoop?

The most common input formats defined in Hadoop are;

- TextInputFormat
- KeyValueInputFormat
- SequenceFileInputFormat

115) In Hadoop what is InputSplit?

It splits input files into chunks and assign each split to a mapper for processing.

116) For a Hadoop job, how will you write a custom partitioner?

You write a custom partitioner for a Hadoop job, you follow the following path

- Create a new class that extends Partitioner Class
- Override method getPartition
- In the wrapper that runs the MapReduce

- Add the custom partitioner to the job by using method set Partitioner Class or – add the custom partitioner to the job as a config file

117) For a job in Hadoop, is it possible to change the number of mappers to be created?

No, it is not possible to change the number of mappers to be created. The number of mappers is determined by the number of input splits.

118) Explain what is a sequence file in Hadoop?

To store binary key/value pairs, sequence file is used. Unlike regular compressed file, sequence file support splitting even when the data inside the file is compressed.

119) When Namenode is down what happens to job tracker?

Namenode is the single point of failure in HDFS so when Namenode is down your cluster will set off.

120) Explain how indexing in HDFS is done?

Hadoop has a unique way of indexing. Once the data is stored as per the block size, the HDFS will keep on storing the last part of the data which say where the next part of the data will be.

121) Explain is it possible to search for files using wildcards?

Yes, it is possible to search for files using wildcards.

122) List out Hadoop's three configuration files?

The three configuration files are

- core-site.xml
- mapred-site.xml
- hdfs-site.xml

123) Explain how can you check whether Namenode is working beside using the jps command?

Beside using the jps command, to check whether Namenode are working you can also use

/etc/init.d/hadoop-0.20-namenode status.

124) Explain what is "map" and what is "reducer" in Hadoop?

In Hadoop, a map is a phase in HDFS query solving. A map reads data from an input location, and outputs a key value pair according to the input type.

In Hadoop, a reducer collects the output generated by the mapper, processes it, and creates a final output of its own.

125) In Hadoop, which file controls reporting in Hadoop?

In Hadoop, the hadoop-metrics.properties file controls reporting.

126) For using Hadoop list the network requirements?

For using Hadoop the list of network requirements are:

- Password-less SSH connection
- Secure Shell (SSH) for launching server processes

127) Explain what is a Task Tracker in Hadoop?

A Task Tracker in Hadoop is a slave node daemon in the cluster that accepts tasks from a JobTracker. It also sends out the heartbeat messages to the JobTracker, every few minutes, to confirm that the JobTracker is still alive.

128) Mention what daemons run on a master node and slave nodes?

- Daemons run on Master node is “NameNode”
- Daemons run on each Slave nodes are “Task Tracker” and “Data”

129) Explain how can you debug Hadoop code?

The popular methods for debugging Hadoop code are:

- By using web interface provided by Hadoop framework
- By using Counters
-

130) Explain what is storage and compute nodes?

- The storage node is the machine or computer where your file system resides to store the processing data
- The compute node is the computer or machine where your actual business logic will be executed.

131) Mention what is the use of Context Object?

The Context Object enables 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.

132) Mention what is the next step after Mapper or MapTask?

The next step after Mapper or MapTask is that the output of the Mapper are sorted, and partitions will be created for the output.

133) Mention what is the number of default partitioner in Hadoop?

In Hadoop, the default partitioner is a “Hash” Partitioner.

134) Explain what is the purpose of RecordReader in Hadoop?

In Hadoop, the RecordReader loads the data from its source and converts it into (key, value) pairs suitable for reading by the Mapper.

135) Explain how is data partitioned before it is sent to the reducer if no custom partitioner is defined in Hadoop?

If no custom partitioner is defined in Hadoop, then a default partitioner computes a hash value for the key and assigns the partition based on the result.

136) Explain what happens when Hadoop spawned 50 tasks for a job and one of the task failed?

It will restart the task again on some other TaskTracker if the task fails more than the defined limit.

137) Mention what is the best way to copy files between HDFS clusters?

The best way to copy files between HDFS clusters is by using multiple nodes and the distcp command, so the workload is shared.

138) Mention what is the difference between HDFS and NAS?

HDFS data blocks are distributed across local drives of all machines in a cluster while NAS data is stored on dedicated hardware.

139) Mention how Hadoop is different from other data processing tools?

In Hadoop, you can increase or decrease the number of mappers without worrying about the volume of data to be processed.

140) Mention what job does the conf class do?

Job conf class separate different jobs running on the same cluster. It does the job level settings such as declaring a job in a real environment.

141) Mention what is the Hadoop MapReduce APIs contract for a key and value class?

For a key and value class, there are two Hadoop MapReduce APIs contract

- The value must be defining the org.apache.hadoop.io.Writable interface
- The key must be defining the org.apache.hadoop.io.WritableComparable interface
-

142) Mention what does the text input format do?

The text input format will create a line object that is an hexadecimal number. The value is considered as a whole line text while the key is considered as a line object. The mapper will receive the value as ‘text’ parameter while key as ‘longwriteable’ parameter.

143) Explain how does Hadoop Classpath plays a vital role in stopping or starting in Hadoop daemons?

Classpath will consist of a list of directories containing jar files to stop or start daemons.

144. What does ‘jps’ command do?

It gives the status of the deamons which run Hadoop cluster. It gives the output mentioning the status of namenode, datanode , secondary namenode, Jobtracker and Task tracker.

145. How to restart Namenode?

Step-1. Click on stop-all.sh and then click on start-all.sh OR

Step-2. Write sudo hdfs pressenter, su-hdfs pressenter, /etc/init.d/ha pressenter and then /etc/init.d/hadoop-0.20-namenode start pressenter.

146. 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 nothing to do with Hadoop.

147. What if a Namenode has no data?

It cannot be part of the Hadoop cluster.

148. What happens to job tracker when Namenode is down?

When Namenode is down, your cluster is OFF, this is because Namenode is the single point of failure in HDFS.

149. Replication causes data redundancy, then why is it pursued in HDFS?

HDFS works with commodity hardware systems with average configurations that has high chances of getting crashed any time. Thus, to make the entire system highly fault-tolerant, HDFS replicates and stores data in different places. Any data on HDFS gets stored at least 3 different locations. So, even if one of them is corrupted and the other is unavailable for some time for any reason, then data can be accessed from the third one. Hence, there is no chance of losing the data. This replication factor helps us to attain the feature of Hadoop called Fault Tolerant.

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?

No, 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.

150. Why do we use HDFS for applications having large data sets and not when there are lot of small files?

HDFS is more suitable for large amount of data sets in a single file as compared to small amount of data spread across multiple files. This is because Namenode is a very expensive high performance system, so it is not prudent to occupy the space in the Namenode by unnecessary amount of metadata that is generated for multiple small files. So, when there is a large amount of data in a single file, name node will occupy less space. Hence for getting optimized performance, HDFS supports large data sets instead of multiple small files.

151. What is a 'block' in HDFS?

A 'block' is the minimum amount of data that can be read or written. In HDFS, the default block size is 64 MB as contrast to the block size of 8192 bytes in Unix/Linux. Files in HDFS are broken down into block-sized chunks, which are stored as independent units. HDFS blocks are large as compared to disk blocks, particularly to minimize the cost of seeks. If a particular file is 50 mb, will the HDFS block still consume 64 mb as the default size? No, not at all! 64 mb is just a unit where the data will be stored. In this particular situation, only 50 mb will be consumed by an HDFS block and 14 mb will be free to store something else. It is the MasterNode that does data allocation in an efficient manner.

152. What are the benefits of block transfer?

A file can be larger than any single disk in the network. There's nothing that requires the blocks from a file to be stored on the same disk, so they can take advantage of any of the disks in the cluster. Making the unit of abstraction a block rather than a file simplifies the storage subsystem. Blocks provide fault tolerance and availability. To insure against corrupted blocks and disk and machine failure, each block is replicated to a small number of physically separate machines typicallythree. If a block becomes unavailable, a copy can be read from another location in a way that is transparent to the client?

153. What is the communication channel between client and namenode/datanode?

The mode of communication is SSH.

154. Why 'Reading' is done in parallel and 'Writing' is not in HDFS?

Through mapreduce program the file can be read by splitting its blocks when reading. But while writing as the incoming values are not yet known to the system mapreduce cannot be applied and no parallel writing is possible. Copy a directory from one node in the cluster to another

Use '-**distcp'** command to copy,

Default replication factor to a file is 3.

Use '-**setrep'** command to change replication factor of a file to 2.

hadoop fs -setrep -w 2 apache_hadoop/sample.txt

155. The requirement is to add a new data node to a running Hadoop cluster; how do I start services on just one data node?

You do not need to shutdown and/or restart the entire cluster in this case.

First, add the new node's DNS name to the **conf/slaves** file on the master node.

Then log in to the new slave node and execute −

$ cd path/to/hadoop

$ bin/hadoop-daemon.sh start datanode

$ bin/hadoop-daemon.sh start tasktracker

then issuehadoop dfsadmin -refreshNodes and hadoop mradmin -refreshNodes so that the NameNode and JobTracker know of the additional node that has been added.

156. How do you gracefully stop a running job?

Hadoop job –kill jobid

157. Does the name-node stay in safe mode till all under-replicated files are fully replicated?

No. During safe mode replication of blocks is prohibited. The name-node awaits when all or majority of data-nodes report their blocks.

158. What happens if one Hadoop client renames a file or a directory containing this file while another client is still writing into it?

A file will appear in the name space as soon as it is created. If a writer is writing to a file and another client renames either the file itself or any of its path components, then the original writer will get an IOException either when it finishes writing to the current block or when it closes the file.

159. How to make a large cluster smaller by taking out some of the nodes?

Hadoop offers the decommission feature to retire a set of existing data-nodes. The nodes to be retired should be included into the exclude file, and the exclude file name should be specified as a configuration parameter dfs.hosts.exclude. The decommission process can be terminated at any time by editing the configuration or the exclude files and repeating the -refreshNodes command

160. Can we search for files using wildcards?

Yes. For example, to list all the files which begin with the letter a, you could use the ls command with the * wildcard &minu;

hdfs dfs –ls a*

161. What happens when two clients try to write into the same HDFS file?

HDFS supports exclusive writes only. When the first client contacts the name-node to open the file for writing, the name-node grants a lease to the client to create this file. When the second client tries to open the same file for writing, the name-node will see that the lease for the file is already granted to another client, and will reject the open request for the second client

162. What does "file could only be replicated to 0 nodes, instead of 1" mean?

The namenode does not have any available DataNodes.

163. What is a Combiner?

The Combiner is a ‘mini-reduce’ process which operates only on data generated by a mapper. The Combiner will receive as input all data emitted by the Mapper instances on a given node. The output from the Combiner is then sent to the Reducers, instead of the output from the Mappers Consider case scenario: In M/R system,

164. Suppose Hadoop spawned 100 tasks for a job and one of the task failed. What will Hadoop do?

It will restart the task again on some other TaskTracker and only if the task fails more than four (The default setting and can be changed)times will it kill the job.

165. What are Problems with small files and HDFS?

HDFS is not good at handling large number of small files. Because every file, directory and block in HDFS is represented as an object in the namenode’s memory, each of which occupies approx 150 bytes So 10 million files, each using a block, would use about 3 gigabytes of memory. when we go for a billion files the memory requirement in namenode cannot be met.

166. Can Hadoop handle streaming data?

Yes, through Technologies like Apache Kafka, Apache Flume, and Apache Spark it is possible to do large-scale streaming.

167. Why is Checkpointing Important in Hadoop?

As more and more files are added the namenode creates large edit logs. Which can substantially delay NameNode startup as the NameNode reapplies all the edits. Checkpointing is a process that takes an fsimage and edit log and compacts them into a new fsimage. This way, instead of replaying a potentially unbounded edit log, the NameNode can load the final in-memory state directly from the fsimage. This is a far more efficient operation and reduces NameNode startup time.

Map Reduce

1. What is Mapreduce ?

Mapreduce is a framework for processing big data (huge data sets using a large number of commodity computers). It processes the data in two phases namely Map and Reduce phase. This programming model is inherently parallel and can easily process large-scale data with the commodity hardware itself.

It is highly integrated with hadoop distributed file system for processing distributed across data nodes of clusters.

2. What is YARN ?

YARN stands for Yet Another Resource Negotiator which is also called as Next generation Mapreduce or Mapreduce 2 or MRv2.

It is implemented in hadoop 0.23 release to overcome the scalability short come of classic Mapreduce framework by splitting the functionality of Job tracker in Mapreduce frame work into Resource Manager and Scheduler.

3. What is data serialization ?

Serialization is the process of converting object data into byte stream data for transmission over a network across different nodes in a cluster or for persistent data storage.

4. What is deserialization of data ?

Deserialization is the reverse process of serialization and converts byte stream data into object data for reading data from HDFS. Hadoop provides *Writables* for serialization and deserialization purpose.

5. What are the Key/Value Pairs in Mapreduce framework ?

Mapreduce framework implements a data model in which data is represented as key/value pairs. Both input and output data to mapreduce framework should be in key/value pairs only.

6. What are the constraints to Key and Value classes in Mapreduce ?

Any data type that can be used for a Value field in a mapper or reducer must implement org.apache.hadoop.io.Writable Interface to enable the field to be serialized and deserialized.

By default Key fields should be comparable with each other. So, these must implement

hadoop’sorg.apache.hadoop.io.WritableComparable Interface

which in turn extends hadoop's Writable interface
and java.lang.Comparableinterfaces.

7. What are the main components of Mapreduce Job ?

- **Main driver class which provides job configuration parameters.**
- **Mapper class which must extend org.apache.hadoop.mapreduce.Mapper class and provide implementation for map () method.**
- **Reducer class which should extend org.apache.hadoop.mapreduce.Reducer class.**

8. What are the Main configuration parameters that user need to specify to run Mapreduce Job ?

On high level, the user of mapreduce framework needs to specify the following things:

- **The job's input location(s) in the distributed file system.**
- **The job's output location in the distributed file system.**
- **The input format.**
- **The output format.**
- **The class containing the map function.**
- **The class containing the reduce function but it is optional.**
- The JAR file containing the mapper and reducer classes and driver classes.

9. What are the main components of Job flow in YARN architecture ?

Mapreduce job flow on YARN involves below components.

- **A Client node, which submits the Mapreduce job.**
- **The YARN Resource Manager, which allocates the cluster resources to jobs.**
- **The YARN Node Managers, which launch and monitor the tasks of jobs.**
- **The MapReduce Application Master, which coordinates the tasks running in the MapReduce job.**
- **The HDFS file system is used for sharing job files between the above entities.**

10. What is the role of Application Master in YARN architecture ?

Application Master performs the role of negotiating resources from the Resource Manager and working with the Node Manager(s) toexecute and monitor the tasks.

Application Master requests containers for all map tasks and reduce tasks.Once Containers are assigned to tasks, Application Master starts containers by notifying its Node Manager. Application Master collects progress information from all tasks and aggregate values
are propagated to Client Node or user.

Application master is specific to a single application which is a single job in classic mapreduce or a cycle of jobs. Once the job execution is completed, application master will no longer exist.

11. What is identity Mapper ?

Identity Mapper is a default Mapper class provided by hadoop. When no mapper is class is specified in Mapreduce job, then this mapper will be executed.

It doesn’t process/manipulate/ perform any computation on input data rather it simply writes the input data into output. It’s class name is **org.apache.hadoop.mapred.lib.IdentityMapper**.

12. What is identity Reducer ?

It is a reduce phase’s counter part for Identity mapper in map phase. It simply passes on the input key/value pairs into output directory. Its class name is **org.apache.hadoop.mapred.lib.IdentityReducer**.

When no reducer class is specified in Mapreduce job, then this class will be picked up by the job automatically.

13. What is chain Mapper ?

Chain Mapper class is a special implementation of Mapper class through which a set of mapper classes can be run in a chain fashion, within a single map task.

In this chained pattern execution, first mapper output will become input for second mapper and second mappers output to third mapper, and so on until the last mapper.

Its class name is org.apache.hadoop.mapreduce.lib.ChainMapper.

14. What is chain reducer ?

Chain reducer is similar to Chain Mapper class through which a chain of mappers followed by a single reducer can be run in a single reducer task. Unlike Chain Mapper, chain of reducers will not be executed in this, but chain of mappers will be run followed by a single reducer.

Its class name is org.apache.hadoop.mapreduce.lib.ChainReducer.

15. How can we mention multiple mappers and reducer classes in Chain Mapper or Chain Reducer classes ?

In Chain Mapper,

- **ChainMapper.addMapper() method is used to add mapper classes.**

In ChainReducer,

- **ChainReducer.setReducer() method is used to specify the single reducer class.**
- **ChainReducer.addMapper() method can be used to add mapper classes.**

16. What is a combiner ?

Combiner is a semi-reducer in Mapreduce framework. it is an optional component and can be specified with Job.setCombinerClass() method.

Combiner functions are suitable for producing summary information from a large data set. **Hadoop doesn't guarantee on how many times a combiner function will be called** for each map output key. it may call 0 or 1 or many times.

17. What are the constraints on combiner implementation ?

Combiner class must implement Reducer interface and must provide implementation for reduce() method. The combiner class's reduce() method must have same input and output key-value types as the reducer class.

18. What are the advantages of combiner over reducer or why do we need combiner when we are using same reducer class as combiner class ?

The main purpose of Combiner in Mapreduce frame work is to limit the volume of data transfer between map and reduce tasks.

It is a general practice that same reducer class is used as a combiner class, in this case, the only benefit of combiner class is to minimize the input data to reduce phase from data nodes.

19. What are the primitive data types in Hadoop ?

Below are the list of primitive writable data types available in Hadoop.

- **BooleanWritable**

- **ByteWritable**
- **IntWritable**
- **VIntWritable**
- **FloatWritable**
- **LongWritable**
- **VLongWritable**
- **DoubleWritable**
-

20. What is NullWritable and how is it special from other Writable data types ?

NullWritable is a special type of Writable representing a null value. No bytes are read or written when a data type is specified as NullWritable. So, in Mapreduce, a key or a value can be declared as a NullWritable when we don't need to use that field.

21. What is Text data type in Hadoop and what are the differences from String data type in Java ?

Text is a Writable data type for serialization and de-serialization of string data in Hadoop. It can be treated Wrapper class for java.lang.String in Java. Java Strings are immutable where as Hadoop's Text Writable is mutable.

22. What are the uses of GenericWritable class ?

One of the uses is, GenericWritable classes are extended to provide implementations to wrap multiple **value** instances belonging to different data types.

Whenever multiple value types need to be produced from mapper and we need our reducer to process these multiple value types as a single data type because Hadoop reducers do not allow multiple input value types.

In these scenarios a subclass of GenericWritable can be used.

23. How to create multiple value type output from Mapper with IntWritable and Text Writable ?

Write a class extending the **org.apache.hadoop.io.GenericWritable** class. Implement the **getTypes()** method to return an array of the Writable classes.

<u>MultipleValueWritable.java</u>

```
public class MultipleValueWritable extends GenericWritable
{
private static Class[] types = new Class[]
{
IntWritable.class,
Text.class
};
public MultipleValueWritable(Writable value)
{
set(value);
}
protected Class[] getTypes()
{
return types;
}
}
```

24. What is ObjectWritable data type in Hadoop ?

This is a general-purpose generic object wrapper which can be used to achieve the same objective as GenericWritable. org.apache.hadoop.io.ObjectWritable class can handle Java primitive types, strings, and arrays without the need of a Writable wrapper.

25. How do we create Writable arrays in Hadoop ?

Hadoop provides two types of Writable data types for arrays. For one dimensional arrays, **ArrayWritable** and for two dimensional arrays,**TwoDArrayWritable** data types are available.

The elements of these arrays must be other writable objects like IntWritable or LongWritable only but not the java native data types like int or float. For example, below is implementation of array of IntWritables.

```
public static class IntArrayWritable extends ArrayWritable

{

public IntArrayWritable()

{

super(IntWritable.class) ;

}

}
```

26. What are the MapWritable data types available in Hadoop ?

Hadoop provided below MapWritable data types which implement java.util.Map interface

- **AbstractMapWritable** – This is abstract or base class for other MapWritable classes.
- **MapWritable** – This is a general purpose map mapping Writable keys to Writable values.
- **SortedMapWritable** – This is a specialization of the MapWritable class that also implements the SortedMap interface.

27. What is speculative execution in Mapreduce ?

Speculative execution is a mechanism of running multiple copies of same map or reduce tasks on different slave nodes to cope with individual Machine performance.

In large clusters of hundreds of machines, 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 same map or reduce task on different slave nodes. The results from first node to finish are used.

If other copies were executing speculatively, Hadoop tells the Task Trackers to abandon the tasks and discard their outputs.

28. What will happen if we run a Mapreduce job with an output directory that already existing ?

Job will fail with org.apache.hadoop.mapred.FileAlreadyExistsException . In this case, delete the output directory and re-execute the job.

29. What are the naming conventions for output files from Map phase and Reduce Phase ?

Output files from map phase are named as **part-m-xxxxx** and output files from reduce phase are named as **part-r-xxxxx.** These part files are created separately by each individual reducer. Here **xxxxx** is partition number starting from 00000 and increases sequentially by 1 resulting in 00001, 00002 and so on.

30. Where the output does from Map tasks are stored ?

The mapper's output (intermediate data) is stored on the Local file system (not on HDFS) of each individual mapper nodes. This is typically a temporary directory location which can be setup in **mapreduce.cluster.local.dir** configuration property. The intermediate data is deleted after the Hadoop Job completes.

31. When will the reduce() method will be called from reducers in Mapreduce job flow ?

In a MapReduce job, reducers do not start executing the reduce() method until all Map jobs are completed. Reducers start copying intermediate output data from the mappers as soon as they are available. But reduce() method is called only after all the mappers have finished.

32. If reducers do not start before all the mappers are completed then why does the progress on MapReduce job shows something like Map(80%) Reduce(20%) ?

As said above, Reducers start copying intermediate output data from map tasks as soon as they are available and task progress calculation counts this data copying as well. So, even though the actual reduce() method is not triggered to run on map output data, job progress displays completion percentage of reduce phase as 10 % or 20 %. But the actual reduce() method processing starts execution only after completion of map phase by 100 %.

33. Where the output does from Reduce tasks are stored ?

The output from reducers are stored on HDFS cluster but not on local file system. All reducers stores their output part-r-xxxxx files in the output directory specified in the mapreduce job instead of in local FS. But Map tasks output files are not stored on HDFS but they are stored on each individual data nodes local file system.

34. Can we set arbitrary number of Map tasks in a mapreduce job ?

No. We cannot set the number of map tasks in a mapreduce job. But this can be set at site level with **mapreduce.job.maps** configuration property in mapred-site.xml file.

35. Can we set arbitrary number of Reduce tasks in a mapreduce job and if yes, how ?

Yes, we can set the no of reduce tasks at job level in Mapreduce. Arbitrary number of reduce tasks in a job can be setup with **job.setNumReduceTasks(N);**

Here N is the no of reduce tasks of our choice. Reduce tasks can be setup at site level as well with **mapreduce.job.reduces** configuration property in mapred-site.xml file.

36. What happens if we don't override the mapper methods and keep them as it is ?

If we do not override any mapper methods, it will act as the IdentityMapper, directly emits each input record as a output record as it is.

37. What is the use of Context object ?

The job Context object contains configuration data for the job. The Map Context object allows the mapper to interact with the rest of the Hadoop system. It allows mapper to emit output.

38. Can Reducers talk with each other ?

No, Reducers run in isolation. MapReduce programming model does not allow reducers to communicate with each other.

39. What are the primary phases of the Mapper ?

Primary phases of Mapper are: Record
Reader, Mapper, Combiner and Partitioner.

40. What are the primary phases of the Reducer ?

Primary phases of Reducer are: Shuffle, Sort and Reducer and Output Formatter.

41. What are the side effects of not running a secondary name node?

The cluster performance will degrade over time since edit log will grow bigger and bigger. If the secondary namenode is not running at all, the edit log will grow significantly and it will slow the system down. Also, the system will go into safemode for an extended time since the namenode needs to combine the edit log and the current fs checkpoint image.

42. How many racks do you need to create an Hadoop cluster in order to make sure that the cluster operates reliably?

In order to ensure a reliable operation it is recommended to have at least 2 racks with rack placement configured Hadoop has a built-in rack awareness mechanism that allows data distribution between different racks based on the configuration.

43. What is the procedure for namenode recovery?

A namenode can be recovered in two ways:

- Starting new namenode from backup metadata
- Promoting secondary namenode to primary namenode.

44. Hadoop WebUI shows that half of the datanodes are in decommissioning mode. What does that mean? Is it safe to remove those nodes from the network?

This means that namenode is trying retrieve data from those datanodes by moving replicas to remaining datanodes. There is a possibility that data can be lost if administrator removes those datanodes before decommissioning finished .

45. What does the Hadoop administrator have to do after adding new datanodes to the Hadoop cluster?

Since the new nodes will not have any data on them, the administrator needs to start the balancer to redistribute data evenly between all nodes.

46. If the Hadoop administrator needs to make a change, which configuration file does he need to change?

Each node in the Hadoop cluster has its own configuration files and the changes needs to be made in every file. One of the reasons for this is that configuration can be different for every node.

47. Map Reduce jobs take too long. What can be done to improve the performance of the cluster?

One the most common reasons for performance problems on Hadoop cluster is uneven distribution of the tasks. The number tasks has to match the number of available slots on the cluster. Hadoop is not a hardware aware system. It is the responsibility of the developers and the administrators to make sure that the resource supply and demand match.

48. After increasing the replication level, I still see that data is under replicated. What could be wrong?

Data replication takes time due to large quantities of data. The Hadoop administrator should allow sufficient time for data replication depending on the data size. if data size is big enough it is not uncommon that replication will take from a few minutes to a few hours.

49. What is side data distribution in Mapreduce framework ?

The extra read-only data needed by a mapreduce job to process the main data set is called as side data.

There are two ways to make side data available to all the map or reduce tasks.

- **Job Configuration**
- **Distributed cache**

50. How to distribute side data using job configuration ?

Side data can be distributed by setting an arbitrary key-value pairs in the job configuration using the various setter methods onConfiguration object.

In the task, we can retrieve the data from the configuration returned by Context 's

getConfiguration() method.

51. When can we use side data distribution by Job Configuration and when it is not supposed ?

Side data distribution by job configuration is useful only when we need to pass a small piece of meta data to map/reduce tasks.

We shouldn’t use this mechanism for transferring more than a few KB’s of data because it put pressure on the memory usage, particularly in a system running hundreds of jobs.

52. What is Distributed Cache in Mapreduce ?

Distributed cache mechanism is an alternative way of side data distribution by copying files and archives to the task nodes in time for the tasks to use them when they run.

To save network bandwidth, files are normally copied to any particular node once per job.

53. How to supply files or archives to mapreduce job in distributed cache mechanism ?

The files that need to be distributed can be specified as a comma-separated list of URIs as the argument to the -files option in hadoop job command. Files can be on the local file system, on HDFS.

Archive files (ZIP files, tar files, and gzipped tar files) can also be copied to task nodes by distributed cache by using -archives option.these are un-archived on the task node.

The -libjars option will add JAR files to the classpath of the mapper and reducer tasks.

jar command with distributed cache

```
$ hadoop jar example.jar ExampleProgram -files
Inputpath/example.txt input/filename /output/
```

54. How distributed cache works in Mapreduce Framework ?

When a mapreduce job is submitted with distributed cache options, the node managers copies the the files specified by the -files , -archives and -libjars options from distributed cache to a local disk. The files are said to be localized at this point.

local.cache.size property can be configured to setup cache size on local disk of node managers. Files are localized under the**${hadoop.tmp.dir}/mapred/local** directory on the node manager nodes.

55. What will hadoop do when a task is failed in a list of suppose 50 spawned tasks ?

It will restart the map or reduce task again on some other node manager and only if the task fails more than 4 times then it will kill the job. The default number of maximum attempts for map tasks and reduce tasks can be configured with below properties in mapred-site.xml file.

mapreduce.map.maxattempts

mapreduce.reduce.maxattempts

The default value for the above two properties is 4 only.

56. Consider case scenario: In Mapreduce system, HDFS block size is 256 MB and we have 3 files of size 256 KB, 266 MB and 500 MB then how many input splits will be made by Hadoop framework ?

Hadoop will make 5 splits as follows

– 1 split for 256 KB file

– 2 splits for 266 MB file (1 split of size 256 MB and another split of size 10 MB)

– 2 splits for 500 MB file (1 Split of size 256 MB and another of size 244 MB)

57. Why can’t we just have the file in HDFS and have the application read it instead of distributed cache ?

Distributed cache copies the file to all node managers at the start of the job. Now if the node manager runs 10 or 50 map or reduce tasks, it will use the same file copy from distributed cache.

On the other hand, if a file needs to read from HDFS in the job then every map or reduce task will access it from HDFS and hence if a node manager runs 100 map tasks then it will read this file 100 times from HDFS. Accessing the same file from node manager’s Local FS is much faster than from HDFS data nodes.

58. What mechanism does Hadoop framework provides to synchronize changes made in Distribution Cache during run time of the application ?

Distributed cache mechanism provides service for copying just read-only data needed by a mapreduce job but not the files which can be updated. So, there is no mechanism to synchronize the changes made in distributed cache as changes are not allowed to distributed cached files.

A few more hadoop mapreduce interview questions and answers for experienced will be published in the upcoming posts in this category.

59. After restart of namenode, Mapreduce jobs started failing which worked fine before restart. What could be the wrong ?

The cluster could be in a safe mode after the restart of a namenode. The administrator needs to wait for namenode to exit the safe mode before restarting the jobs again. This is a very common mistake by Hadoop administrators.

60. What do you always have to specify for a MapReduce job ?

A. <u>The classes for the mapper and reducer.</u>

B. The classes for the mapper, reducer, and combiner.

C. The classes for the mapper, reducer, partitioner, and combiner.

D. None; all classes have default implementations.

61. How many times will a combiner be executed ?

A. At least once.

B. Zero or one times.

C. <u>Zero, one, or many times.</u>

D. It’s configurable.

62. You have a mapper that for each key produces an integer value and the following set of reduce operations
Reducer A: outputs the sum of the set of integer values.
Reducer B: outputs the maximum of the set of values.
Reducer C: outputs the mean of the set of values.
Reducer D: outputs the difference between the largest and smallest values
in the set.

Which of these reduce operations could safely be used as a combiner?

a. All of them.

b. <u>A and B.</u>

c. A, B, and D.

d. C and D.

e. None of them.

<u>Explanation</u>: Reducer C cannot be used because if such reduction were to occur, the final reducer could receive from the combiner a series of means with no knowledge of how many items were used to generate them, meaning the overall mean is impossible to calculate.

Reducer D is subtle as the individual tasks of selecting a maximum or minimum are safe for use as combiner operations. But if the goal is to determine the overall variance between the maximum and minimum value for each key, this would not work. If the combiner that received the maximum key had values clustered around it, this would generate small results; similarly for the one receiving the minimum value. These sub ranges have little value in isolation and again the final reducer cannot construct the desired result.

63. What is Uber task in YARN ?

If the job is small, the application master may choose to run them in the same JVM as itself, since it judges the overhead of allocating new containers and running tasks in them as outweighing the gain to be had in running them in parallel, compared to running them sequentially on one node. (This is different to Mapreduce
1, where small jobs are never run on a single task tracker.)

Such a job is said to be Uberized, or run as an Uber task.

64. How to configure Uber Tasks ?

By default a job that has less than 10 mappers only and one reducer, and the input size is less than the size of one HDFS block is said to be small job. These values may
be changed for a job by setting

mapreduce.job.ubertask.maxmaps ,

mapreduce.job.ubertask.maxreduces , and mapreduce.job.ubertask.maxbytes

It's also possible to disable Uber tasks entirely by setting mapreduce.job.ubertask.enable to false.

65. What are the ways to debug a failed mapreduce job ?

Commonly there are two ways.

1. **By using mapreduce job counters**
2. **YARN Web UI for looking into syslogs for actual error messages or status.**

66. What is the importance of heartbeats in HDFS/Mapreduce Framework ?

A heartbeat in master/slave architecture is a signal indicating that it is alive. A datanode sends heartbeats to Namenode and node managers send their heartbeats to Resource Managers to tell the master node that these are still alive.

If the Namenode or Resource manager does not receive heartbeat from any slave node then they will decide that there is some problem in data node or node manager and is unable to perform the assigned task, then master (namenode or resource manager) will reassign the same task to other live nodes.

67. Can we rename the output file ?

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

68. What are the default input and output file formats in Mapreduce jobs ?

If input file or output file formats are not specified, then the default file input or output formats are considered as text files.

A few more hadoop mapreduce interview questions and answers for experienced will be published in the upcoming posts in this category.

69. How can we add the arbitrary key-value pairs in your mapper?

We can set arbitrary (key, value) pairs of configuration data in our Job, with Job.getConfiguration().set("key", "val"), and we can retrieve this data in mapper with Context.getConfiguration().get("key").

This kind of functionality is typically done in the Mapper's setup() method.

70. Which object can be used to get the progress of a particular job?

Context

71. How can we control particular key should go in a specific reducer?

We can control keys (and hence records) to be processed in a particular Reducer by implementing a custom Partitioner class.

72. What are the methods in the Mapper class and order of their invocation?

```
run(Context context)
{
setup(context);
while (context.nextKeyValue())
{
map(context.getCurrentKey(), context.getCurrentValue(), context);
}
cleanup(context);
}
```

```
setup(Context context)
map(Writable key, Writable value, Context context)
cleanup(Context context)
```

The Mapper contains the run() method, which call its own setup() method only once, it also call a map() method for each input and finally calls it cleanup() method. We can override all above methods in our code.

Each of these methods can access the job's configuration data by using Context.getConfiguration().

73. What are the methods in the Reducer class and order of their invocation?

```
run(Context context)
{
setup(context);
while (context.nextKeyValue())
{
reduce(context.getCurrentKey(), context.getCurrentValue(), context);
}
cleanup(context);
}
```

```
setup(Context context)
reduce(Writable key, Writable value, Context context)
cleanup(Context context)
```

The Reducer class contains the run() method, which call its own setup() method only once, it also call a map() method for each input and finally calls it cleanup() method. We can override all above methods in our code.

73. What is Nlineoutputformat?

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

75. What is the difference between an HDFS Block and Input Split?

HDFS Block is the physical division of the data and Input Split is the logical division of the data.

76. What is keyvaluetextinputformat?

In keyvaluetextinputformat, each line in the text file is a ‘record‘. The first separator character divides each line. Everything before the separator is the key and everything after the separator is the value. For instance, Key: text, value: text.

77. Why we cannot do aggregation (addition) in a mapper? Why we require reducer for that?

We cannot do aggregation (addition) in a mapper because, sorting is not done in a mapper. Sorting happens only on the reducer side. Mapper method initialization depends upon each input split. While doing aggregation, we will lose the value of the previous instance. For each row, a new mapper will get initialized. For each row, input split again gets divided into mapper, thus we do not have a track of the previous row value.

78 .Can we process different input file directories with different input formats, like some text files and some sequence files in a single MR job?

Yes, we can implement this by **MultipleInputs.addInputPath()** methods in job driver class. might set up the input as follows:

MultipleInputs.addInputPath(job, inputPath1, TextInputFormat.class, Mapper1.class)

MultipleInputs.addInputPath(job, inputPath2, SequeneFileInputFormat.class, Mapper2.class);

Here Mapper1 class handles TextInputFormat data and Mapper2 class handles SequenceFileInputFormat data.

79. What is the need for serialization in Mapreduce ?

Below are the two necessities for serialization in Hadoop.

- In Hadoop cluster, data is stored in only binary stream format but object structured data can’t be stored directly hadoop data nodes.
- Only Binary stream data can be transferred across data nodes in hadoop cluster. So, Serialization is needed to convert the object structured data into binary stream format.

80. How does the nodes in a hadoop cluster communicate with each other?

- Inter process communication between nodes in a hadoop cluster is implemented using Remote Procedure Calls (RPC).

Inter process communication happens in below three stages.

- RPC protocol uses serialization to convert the message from source data node into a binary stream data.
- Binary stream data is transferred to the remote destination node
- Destination node then use De-serialization to convert the binary stream data into object structured data and then it reads object structured data.

81. What is the Hadoop in built serialization framework ?

Writables are the hadoop's own serialization format which serializes the data into compact size and ensures fast transfer across nodes.Writables are written in Java and supported by Java only.

82. What is Writable and its methods in hadoop library ?

Writable is an Interface in hadoop library and it provides below two methods for serializing and de-serializing the data.

write(DataOutput out) – Writes data into DataOutput binary stream.

readFields(DataInput in) – Reads data from DataInput binary stream.

<u>Writable Interface</u>

```
package org.apache.hadoop.io;
import java.io.DataOutput;
import java.io.DataInput;
import java.io.IOException;
public interface Writable
{
void write(DataOutput out) throws IOException;
void readFields(DataInput in) throws IOException;
}
```

83. What is WritableComparable in hadoop library ?

WritableComparable interface is a sub interface of the Writable and java.lang.comparable interfaces.

WritableComparable Interface

```
package org.apache.hadoop.io;
public interface WritableComparable<T> extends Writable,
Comparable <T>
{
```

}

84. What are the comparators in Mapreduce ?

In Mapreduce framework, during sorting phase, map output keys are compared with each other.

Hadoop provides RawComparator which is an extension of java’s Comparator for comparing binary stream data directly without de-serializing it into objects, thereby avoiding overhead of creation of objects.

```
package org.apache.hadoop.io;

import java.util.Comparator;


public interface RawComparator<T> extends Comparator<T>

{

public int compare(byte[] b1, int s1, int l1, byte[] b2, int s2, int l2);

}
```

The comparator for IntWritables implements the raw compare() method by reading an integer from each of the byte arrays b1 and b2 and comparing them directly, from the given start positions (s1 and s2) and lengths (l1 and l2).

85. What is WritableComparator in Mapreduce ?

WritableComparator is a general implementation of hadoop provided interface RawComparator for WritableComparable classes.

it provides a default implementation of the raw compare() method that deserializes the objects to be compared from the stream and invokes the object compare() method.

WritableComparator can be used to compare two IntWritable objects as shown below.

```
RawComparator<IntWritable> comparator = WritableComparator.get(IntWritable.class);
IntWritable int1 = new IntWritable(150);
IntWritable int2 = new IntWritable(100);
if (comparator.compare(int1, int2) > 0) //true
{
}
```

86. Is output from Mapreduce job is globally sorted if there are more than 1 reducer ?

No, Output from each reducer is sorted in its partition file and after combining all part-r-***** files records will not be in globally sorted order.

87. How to globally sort the output from a Mapreduce Job?

TotalOrderPartitioner class can be used instead of default HashPartitioner class to generate the output from Mapreduce job to be in globally sorted order.

Job.setPartitionerClass(TotalOrderPartitioner.class);

88. How to sort the output from the mapreduce job on a field other than mapper output key field ?

We can write a custom comparator by extending RawComparator class and overriding compare() method based on the field on which we need the output to be sorted. And the custom comparator class needs to be mentioned in the Job's configuration.

Job.setSortComparatorClass(CustomComparator.class);

89. What does Record Reader do in Map phase ?

Record Reader generates the series of map input key/value pairs from input splits. The purpose of record reader is to parse the data into record but doesn’t parse the record itself.

90. What is partitioner in Map Phase ?

Partitioner takes intermediate output from mappers as input and splits into partitions. Each partition will be fed separately into each reducer. The keys are partitioned in such a way that records for any given key are grouped into a single partition.

The partitioned data is written to the local file system for each map task and it will be transferred to its respective reducer.

91. What is Shuffle and Sort phase in Mapreduce Framework ?

In Mapreduce framework, it is guaranteed that every reducer will receive input which is sorted by input key. The process of sorting the map outputs by input key and transferring the sorted output as input to the Reducer is known as shuffle.

92. What is Data Locality Optimization ?

In Hadoop Mapreduce Framework, Hadoop tries its best to run a map task on a node where the input data resides. By this, it reduces the data transfer between nodes on a cluster.

This technique of moving computation to data but not bringing data to computation node is called as Data Locality Optimization.

93. Will data locality optimization possible at reducer phase ?

No, Reduce tasks can not be started on nodes where the map outputs are present on the cluster because usually reduce tasks are lesser in number compared to map tasks and some time a single reducer is required to process all the map tasks output.

So, map outputs need to be transferred to the nodes on which reduce tasks get executed.

94. What is copy phase in Reduce tasks ?

In Mapreduce framework, the map tasks may finish at different times, but the reduce tasks start copying map task outputs as soon as each map task completes. This is known as the copy phase of the reduce task.

The reduce task has five copier threads by default so that it can fetch map outputs in parallel, but this number can be changed by setting the mapred.reduce.parallel.copies property.

95. Is it possible that a Job has 0 reducers ?

Yes, It is legal to set the number of reduce-tasks to zero if no reduction is desired.

96. What happens if number of reducers are 0 ?

In this case, the output of the map tasks go directly onto the HDFS output directory. But the framework does not sort the map outputs before writing them out to the output directory.

97. What is the default input split size in Mapreduce Job ?

The default input split size in Mapreduce job is equal to the size of the HDFS block which is 256 MB as of hadoop-2.4.0 release. Each of these input splits are processed by a separate map task.

98. What are the advantages/disadvantages of small input split size ?

In Mapreduce framework, map tasks are executed in parallel to the process these input splits. if the splits are small, the processing will bebetter load-balanced since a faster node will be able to process proportionally more splits over the course of job than a slower node.

But if the splits are too smaller than the default HDFS block size, then managing splits and creation of map tasks becomes an overhead than the job execution time.

99. What is InputSplit in Hadoop?

When a hadoop job is run, it splits input files into chunks and assign each split to a mapper to process. This is called Input Split.

100. How is the splitting of file invoked in Hadoop Framework ?

It is invoked by the Hadoop framework by
running getInputSplit() method of the Input format class (like FileInputFormat) defined by the user

101. Using command line in Linux, how will we see all jobs running in the hadoop cluster and how will we kill a job?

```

$ hadoop job -list
$ hadoop job -kill jobid
```

102. Is it possible to provide multiple input to Hadoop? If yes then how can we give multiple directories as input to the Hadoop job?

Yes, The input format class provides methods to add multiple directories as
input to a Hadoop job through FileInputFormat.addInputPath() method.

103. Is it possible to have Hadoop job output in multiple directories. If yes then how?

Yes, by using Multiple Outputs class

104. How will we write a custom partitioner for a Hadoop job?

To write a custom partitioner we will have to

- Create a new class that extends Partitioner class
- Override method getPartition()
- In the wrapper that runs the Map Reducer, add the custom partitioner to the job programmatically using method setPartitionerClass()

105. Whether Mapreduce Job mapper output files are replicated ?

No. Mapper output files (part-m-00000) are stored on local file system of data nodes and these are not replicated to provide fault tolerance as these files exist only during job execution. Once the mapreduce job is completed these mapper output files will be flushed out.

106. Whether Mapreduce Job reducer output file blocks are also replicated? If yes how many copies are maintained?

Yes. Since Mapreduce Job reducer output files (part-r-00000) are stored on HDFS instead of on local FS as mapper output, each block of reducer output files are maintained in 3 copies which is equal to default replication factor. For each HDFS block of the reduce output, the first replica is stored on the local node, with other replicas being stored on off-rack nodes.

107. How will be the number of map tasks required to run a map reduce job is determined?

Mapreduce programmer can't specify the number of map tasks to be instantiated in a mapreduce job. The number of map tasks are decided based on the input data in map phase. i.e. no of input splits from the input file.

Each split (which is of one block size generally) will be processed separately by a map task. So, the total number of map tasks are decided no of input splits.

108. are setNumMapTasks(int) & setNumReduceTasks(int) methods supported in Mapreduce JAVA API on job object ?

In Mapreduce Java API, setNumMapTasks(int) method is not supported. When we try to add job.setNumMapTasks(int) to we will get a compile error stating, "setNumMapTasks(int) is undefined for the job type".

However, we can add the reducer equivalent, “job.setNumReduceTasks(int)”.

109. How will be the number of reduce tasks required to run a map reduce job are determined?

The number of reduce tasks is not governed by the size of the input, but is specified independently with job.setNumReduceTasks()method.

110. What are compression codecs available in hadoop ?

DEFLATE – This compression format doesn’t support splitting for mapreduce processing
gzip – This format also doesn’t support splitting for mapreduce processing
bzip2 – It supports splitting for mapreduce processing
LZO – With the help of indexer tool that comes with Hadoop LZO libraries, LZO files can be made splittable.

Snappy – Supports splitting.

111. What is best suitable compression format large sized Log files?

- Use a compression format that supports splitting, like bzip2 (although bzip2 is fairly slow), or one that can be indexed to support splitting, like LZO.
- Use Sequence File, which supports compression and splitting.

- Use an Avro data file, which supports compression and splitting, just like Sequence File, but has the added advantage of being readable and writable from many languages, not just Java.
- For large files, you should not use a compression format that does not support splitting on the whole file, since you lose locality and make MapReduce applications very inefficient.

112. How to compress the output from a mapreduce Job?

Output from a mapreduce job can be compressed by setting the below two properties in mapred-site.xml configurations.

mapreduce.output.compress property to true
mapred.output.compression.codec property to the classname of the compression codec we want to use

Alternatively we can set these properties at job level as well with the help of static methods on FileOutputFormat class.

```
FileOutputFormat.setCompressOutput(job, true);
FileOutputFormat.setOutputCompressorClass(job, GzipCodec.class);
```

113. Is compression possible on Output Sequence Files from Mapreduce job? If yes what are the types allowed?

Yes, Output Sequence Files from map reduce job can be compressed. There are two types compressions are allowed on sequence files.
RECORD — Compresses individual records
BLOCK — Compresses groups of records.

Generally BLOCK compression type is recommended since it compresses better.

These compression types can be configured at site level in mapred-site.xml file with below configuration property.

mapreduce.output.compression.type

The same can be configured at job level as well with below method.

SequenceFileOutputFormat.setOutputCompressionType().

114. Can we compress the output from Map tasks ?

Yes, Mapreduce allows us to compress the intermediate map output files on local file system to reduce network traffic.

Below are the lines to add to enable Snappy map output compression in your job:

```
Configuration conf = new Configuration();
conf.setBoolean("mapred.compress.map.output", true);
conf.setClass("mapred.map.output.compression.codec",
SnappyCodec.class,
CompressionCodec.class);
Job job = new Job(conf);
```

115. Hadoop Performance Tuning ?

There are many ways to improve the performance of Hadoop jobs. In this post, we will provide a few MapReduce properties that can be used at various mapreduce phases to improve the performance tuning.

There is no one-size-fits-all technique for tuning Hadoop jobs, because of the architecture of Hadoop, achieving balance among resources is often more effective than addressing a single problem.

Depending on the type of job you are running and the amount of data you are moving, the solution might be quite different

We encourage you to experiment with these and to report your results.

Bottlenecks

Hadoop resources can be classified into computation, memory, network bandwidth and input and output (I/O). A job can run slowly if any of these resources perform badly. Below are the common resource bottlenecks in hadoop jobs.

- CPU – Key Resource for both Map and Reduce Tasks Computation
- RAM – Main Memory available on the slave (node manager) nodes.
- Network Bandwidth – When large amounts of data sets are being processed, high network utilization occurs among nodes. This may occur when Reduce tasks pull huge data from Map tasks in the Shuffle phase, and also when the job outputs the final results into HDFS.
- Storage I/O – File read write I/O throughput to HDFS. Storage I/O utilization heavily depends on the volume of input, intermediate data, and final output data.

Below are the common issues that may arise in Mapreduce Job Execution flow.Massive I/O Caused by Large Input Data in Map Input Stage.

Problem 1 – Massive I/O Caused by Large Input Data in Map Input Stage

This problem happens most often on jobs with light computation and large volumes of source data. If disk I/O is not fast enough, computation resources will be idle and spend most of the job time waiting for the incoming data. Therefore, performance can be constrained by disk I/O.

We can identify this issue with high values in below job counters.

- **Job counters**: Bytes Read, HDFS_BYTES_READ

Solution 1: Compress Input Data

Compress Input data – Compression of files saves storage space on HDFS and also improves speed of transfer.

We can use any of the below compression techniques on input data sets.

Format	**Codec**	**Extension**	**Splittable**	**Hadoop**
DEFLATE	org.apache.hadoop.io.compress.DefaultCodec	.deflate	N	Y
Gzip	org.apache.hadoop.io.compress.GzipCodec	.gz	N	Y
Bzip2	org.apache.hadoop.io.compress.BZi	.bz2	Y	Y

	p2Codec			
LZO	com.hadoop.compression.lzo.LzopCodec	.lzo	N	Y
LZ4	org.apache.hadoop.io.compress.Lz4Codec	.Lz4	Y	Y
Snappy	org.apache.hadoop.io.compress.SnappyCodec	.Snappy	Y	Y

When we submit a MapReduce job against compressed data in HDFS, Hadoop will determine whether the source file is compressed by checking the file name extension, and if the file name has an appropriate extension, Hadoop will decompress it automatically using the appropriate codec. Therefore, users do not need to explicitly specify a codec in the MapReduce job.

However, if the file name extension does not follow naming conventions, Hadoop will not recognize the format and will not automatically decompress the file. Therefore, to enable self-detection and decompression, we must ensure that the file name extension matches the file name extensions supported by each codec.

<u>Problem 2 – Massive I/O Caused by Spilled Records in Partition and Sort phases</u>

When the map function starts producing output, it is not simply written to disk. Each map task has a circular memory buffer that it writes the output to. The buffer is 100 MB by default. When the contents of the buffer reaches a certain threshold size, a background thread will start to spill the contents to disk. Map outputs will continue to be written to the buffer while the spill takes place, but if the buffer fills up during this time, the map will block until the spill is complete. Spills are written in round-robin fashion to the directories specified by the mapred.local.dir property, in a job-specific sub directory.

To optimize the Map task outputs, we need to ensure that records are spilled (meaning, written to LFS or HDFS file) only once. When records are spilled more than once, the data must be read in and written out multiple times, causing drains on I/O. If buffer size is too small and it is filled up too quickly, then it will lead to multiple spills. Each extra spill generates a large volume of data in intermediate bytes.

We can identify this issue with high values in below job counters.

- **Job counters**: FILE_BYTES_READ, FILE_BYTES_WRITTEN, Spilled Records

<u>Solution 2:</u> Adjust Spill Records and Sorting Buffer

To reduce the amount of data spilled during the intermediate Map phase, we can adjust the following properties for controlling sorting and spilling behavior.

mapreduce.task.io.sort.factor	10	The number of streams to merge at once while sorting files. This determines the number of open file handles.
mapreduce.task.io.sort.mb	100	The total amount of buffer memory to use while sorting files, in megabytes. By default, gives each merge stream 1MB, which should minimize seeks.

mapreduce.map.sort.spill.percent	0.80	The soft limit in the serialization buffer. Once reached, a thread will begin to spill the contents to disk in the background. Note that collection will not block if this threshold is exceeded while a spill is already in progress, so spills may be larger than this threshold when it is set to less than .5

When Map output is being sorted, 16 bytes of
metadata are added immediately before each key-value pair. These 16 bytes include 12 bytes for the key-value offset and 4 bytes for the indirect-sort index. Therefore, the total buffer space defined
in io.sort.mb can be divided into two parts: metadata buffer and key-value buffer.

Formula for io.sort.mb

io.sort.mb = (16 + R) * N / 1,048,576

R – the average length of the key-value pairs (in bytes) and can be calculated by dividing the Map output bytes by the number of Map output records from job counters.

N – calculated by dividing the Map output records by the number of map tasks.

Update the io.sort.spill.percent property to 1.0 to make use of complete buffer space.

Problem 3 – Massive Network Traffic Caused by large Map Output

Large output from the Map phase can cause longer I/O and data transfer time, and in worst cases can raise exceptions, if all the I/O throughput channels are saturated or if network bandwidth is exhausted.

We can identify this issue with high values in below job counters.

- **Job counters**: FILE_BYTES_WRITTEN, FILE_BYTES_READ, Combine Input Records
- **Possible exceptions**: java.io.IOException

<u>Solution 3.1:</u> Compress Map Output

If Map Output is very large, it is always recommended to use compression techniques to reduce the size of intermediate data. By default, Map Output is not compressed but we can enable by setting below properties to true.

mapreduce.map.output.compress	false

mapreduce.map.output.compress.codec	org.apache.hadoop.io. compress.DefaultCodec

Below is the code snippet to enable gzip map output compression in our job:

```
Configuration conf = new Configuration();
conf.setBoolean("mapreduce.map.output.compress", true);
conf.setClass("mapreduce.map.output.compress.codec",
GzipCodec.class,
CompressionCodec.class);
Job job = new Job(conf);
```

<u>Solution 3.2:</u> Implement a Combiner

We can also reduce the network I/O caused by Map Output by implementing Combiner if aggregate operation follows commutative and associative rule.

<u>Problem 4 – Massive Network Traffic Caused by large Reduce Output</u>

Large output from Reducers can cause lot of I/O write operations to HDFS.

We can identify this issue with high values in below job counters.

- **Job counters**: Bytes Written, HDFS_BYTES_WRITTEN
- **Possible exceptions**: java.io.IOException

The above two counters denote the volume of data from Reduce Phase, but these two counters do not include the replication factor. If the replication factor is greater than one, it means that blocks of data will be replicated to different nodes, which requires more I/O for read and write operations, and which also uses network bandwidth.

<u>Solution 4.1:</u> Compress Reducer/Final Output

We can enable compression on Mapreduce job's output by setting below properties to true at site level for all jobs.

mapreduce.output.fileoutputformat.compress	false	Compress?
mapreduce.output.fileoutputfo	RECORD	If SequenceFiles,

rmat.compress.type		then it Should be one of NONE, RECORD or BLOCK.
mapreduce.output.fileoutputformat.compress.codec	org.apache.hadoop.io.compress.DefaultCodec	

Set the above properties either in Job driver using code snippet like below (Snappy compression with Block Mode) or in mapred-site.xml file.

- **<u>If Output Files are Not Sequence Files</u>**

```
FileOutputFormat.setCompressOutput(job, true);
FileOutputFormat.setOutputCompressorClass(job,
GzipCodec.class);
```

- **<u>If Output Files are Sequence Files</u>**

```

job.setOutputFormatClass(SequenceFileOutputFormat.class);
SequenceFileOutputFormat.setCompressOutput(job, true);
SequenceFileOutputFormat.setOutputCompressorClass(job,
SnappyCodec.class);
SequenceFileOutputFormat.setOutputCompressionType(job,
CompressionType.BLOCK);
```

- <u>**To Make Global Changes to cluster**</u>

mapred-site.xml

```
<property>
<name>mapreduce.output.fileoutputformat.compress</name>
<value>true</value>
</property>
<property>
<name>mapreduce.output.fileoutputformat.compress.codec</na
me>
<value>SnappyCodec.class</value>
</property>
<property>
<name>mapreduce.output.fileoutputformat.compress.type</nam
e>
<value>BLOCK</value>
</property>
```

<u>Solution 4.2:</u> Adjust Replication Factor

By reducing the replication factor to 1 when more replications are needed we can improve the job performance as , copying data to multiple nodes will be reduced. Set dfs.replication property to 1 using conf object in Job Driver program.

<u>**Problem 4 – Insufficient Parallel Tasks**</u>

If the number of parallel tasks running concurrently is insufficient to run the job, the job can leave many resources idle. Increasing the number of parallel tasks helps to accelerate the overall job execution by better utilizing resources.

Incorrect configurations may degrade the MR job performances some times. For example, if our data node machine has 16 CPU cores but we configured only 8 mappers on each machine, then remaining 8 cores will be idle because no work load will be assigned to them; As a result, only a limited portion of I/O throughput and network bandwidth will be actually utilized, because there are no requests coming from the other CPU cores.

Try to use all the available Map and Reduce task slots on the cluster across all the current running jobs. From YARN Web UI we can verify below counters to identify this issue.

- **Task Summary List**: Num Tasks, Running, Map Task Capacity, Reduce Task Capacity

Observe the cluster's total available Map and Reduce slots and Job's currently assigned no of Map and Reduce tasks to identify incorrect configuration.

<u>Solution 5:</u> Adjust Number of Map Tasks & Reduce Tasks & Memory

Both over allocation and Under allocation of Map Tasks & Reduce Tasks will degrade the performance. So we need to find out the optimized values by trial and error methods to keep the cluster resource utilization in balanced.

mapreduce.tasktracker.map.tasks.maximum	2	The maximum number of map tasks that will be run simultaneously by a task tracker.
mapreduce.tasktracker.reduce.tasks.maximum	2	The maximum number of reduce tasks that will be run simultaneously by a task tracker.
mapreduce.map.memory.mb	1024	The amount of memory to request from the scheduler for each map task.
mapreduce.map.cpu.vcores	1	The number of virtual cores to request from the scheduler for each map task.
mapreduce.reduce.memory.mb	1024	The amount of memory to request from the scheduler for each reduce task.
mapreduce.reduce.cpu.vcores	1	The number of virtual cores to request from

		the scheduler for each reduce task.

yarn.app.mapreduce.am.resource.mb	1536	The amount of memory the MR AppMaster needs.
yarn.app.mapreduce.am.resource.cpu-vcores	1	The number of virtual CPU cores the MR AppMaster needs.

The above are the default values in mapred-default.xml file and these can be overridden in mapred-site.xml to better utilize node managers resources completely.

We can also adjust the Memory for tasks with property mapred.child.java.opts = -Xmx2048M in mapred-site.xml

If we have 16 CPU cores and 32 GB RAM on Node Managers, then we can tune these properties upto 8 Map Tasks and 4 Reduce Tasks with memory 2048 MB allocated to each task at the maximum and leaving 4 cpu cores in buffer for other tasks/operations running on same Node Manager.

We can also set these properties at Job level on Configuration Object.

Below are some additional Reduce Side Tuning Properties

mapreduce.reduce.shuffle.parallelcopies	5	The default number of parallel transfers run by

		reduce during the copy(shuffle) phase.

mapreduce.shuffle.max.threads	0	Max allowed threads for serving shuffle connections. Set to zero to indicate the default of 2 times the number of available processors (as reported by Runtime.availableProcessors ()). Netty is used to serve requests, so a thread is not needed for each connection.
mapreduce.shuffle.transferTo.allowed		This option can enable/disable using nio transferTo method in the shuffle phase. NIO transferTo does not perform well on windows in the shuffle phase. Thus, with this configuration property it is possible to disable it, in which case custom transfer method will be used. Recommended value is false when running Hadoop on Windows. For Linux, it is recommended to set it to

		true. If nothing is set then the default value is false for Windows, and true for Linux.
mapreduce.shuffle.transfer.buffer.size	131072	This property is used only if mapreduce.shuffle.transferTo.allowed is set to false. In that case, this property defines the size of the buffer used in the buffer copy code for the shuffle phase. The size of this buffer determines the size of the IO requests.
mapreduce.reduce.markreset.buffer.percent	0.0	The percentage of memory - relative to the maximum heap size- to be used for caching values when using the mark-reset functionality.
mapreduce.map.speculative	true	If true, then multiple instances of some map tasks may be executed in parallel.
mapreduce.reduce.speculative	true	If true, then multiple instances of some reduce tasks may be executed in parallel.

mapreduce.job.speculative.speculative-cap-running-tasks	0.1	The max percent (0-1) of running tasks that can be speculatively re-executed at any time.
mapreduce.job.speculative.speculative-cap-total-tasks	0.01	The max percent (0-1) of all tasks that can be speculatively re-executed at any time.
mapreduce.job.speculative.minimum-allowed-tasks	10	The minimum allowed tasks that can be speculatively re-executed at any time.
mapreduce.job.speculative.retry-after-no-speculate	1000	The waiting time(ms) to do next round of speculation if there is no task speculated in this round.
mapreduce.job.speculative.retry-after-speculate	15000	The waiting time(ms) to do next round of speculation if there are tasks speculated in this round.

HIVE

Q & A

CHAKRABORTY, SHUBHAM

HIVE

1. What is Metadata?

Data about Data.

2. What is Hive?

Hive is one of the important tool in Hadoop eco system and it provides an SQL like dialect to Hadoop distributed file system.

3. What are the features of Hive?

Hive provides,

- Tools to enable easy data extract/transform/load (ETL)
- A mechanism to project structure on a variety of data formats
- Access to files stored either directly in HDFS or other data storage systems as HBase
- Query execution through MapReduce jobs.
- SQL like language called HiveQL that facilitates querying and managing large data sets residing in hadoop.

4. What are the limitations of Hive?

Below are the limitations of Hive:

- Hive is best suited for data warehouse applications, where a large data set is maintained and mined for insights, reports, etc.
- Hive does not provide record-level update, insert, nor delete.
- Hive queries have higher latency than SQL queries, because of start-up overhead for MapReduce jobs submitted for each hive query.
- As Hadoop is a batch-oriented system, Hive doesn't support OLTP (Online Transaction Processing).
- Hive is close to OLAP (Online Analytic Processing) but not ideal since there is significant latency between issuing a query and receiving a reply, both due to the overhead of Mapreduce jobs and due to the size of the data sets Hadoop was designed to serve.
- If we need OLAP, we need to use NoSQL databases like HBase that can be integrated with Hadoop.

5. What is the differences Between Hive and HBase?

Hive is not a database but a data warehousing frame work. Hive doesn't provide record level operations on tables.

- HBase is a NoSQL Database and it provides record level updates, inserts and deletes to the table data.
- HBase doesn't provide a query language like SQL, but Hive is now integrated with HBase.

6. What is Hive Metastore?

The metastore is the central repository of Hive metadata. The metastore is divided into two pieces: a service and the backing store for the data. By default, the metastore is run in the same process as the Hive service. Using this service, it is possible to run the metastore as a standalone (remote) process. Set the METASTORE_PORT environment variable to specify the port the server will listen on.

7. Wherever (Different Directory) we run hive query, it creates new metastore_db, please explain the reason for it?

Whenever we run the hive in embedded mode, it creates the local metastore. And
before creating the metastore it looks whether metastore already exist or not. This property is defined in configuration file hive-site.xml.

Property is "javax.jdo.option.ConnectionURL" with default value "*jdbc:derby:;databaseName=metastore_db;create=true*".

So to change the behavior change the location to absolute path, so metastore will be used from that location.

8. What are the different types of Hive Metastore?

Below are three different types of metastore.

- Embedded Metastore
- Local Metastore
- Remote Metastore

9. What is the default Hive warehouse directory?

It is /user/hive/warehouse directory in local file system.

10. How to start Hive Thrift server?

We can issue below command from terminal to start Hive thrift server.

$ hive –service hiveserver

11. How to start Hive metastore service as a background process?

We can start hive metastore service as a background process with below command.

$ hive --service metastore & By using kill -9 <process id> we can stop this service.

12. How to configure hive remote metastore in hive-site.xml file?

We can configure remote metastore in hive-site.xml file with the below property.

hive-site.xml file in node

<property>

<name>hive.metastore.uris</name>

<value>thrift://node1(or IP Address):9083</value>

<description>IP address (or fully-qualified domain name) and port of the metastore host</description>

</property>

13. What is the need for partitioning in Hive?

Partitioning is mainly intended for quick turn around time for queries on hive tables.

14. We have already 3 tables named US,UK,IND in Hive. Now we have one more JPN created using *hadoop fs -mkdir JPN*. Can we move the content in IND to JPN directly?

Yes, we can copy contents from hive warehouse directory table IND into JPN.

15. Now we have to display the contents in US,UK,IND,JPN. By using SELECT * FROM TABLES is it possible to display?

No, Because JPN is created by using fs -mkdir command. It is not part of metadata.

16. Is it possible to use same metastore by multiple users, in case of embedded hive?

No, it is not possible to use metastore in sharing mode. It is recommended to use
standalone “real” database like MySQL or PostGreSQL.

17. What is HCatalog and how to use it?

HCatalog is a Table and Storage Management tool to Hadoop/HDFS. In MR, we use it by specifying InputOutput Formats
i.e.HCatInputFormat and HCatOutputFormat.

In Pig, we use it by specifying Storage types
i.e HCatLoader and HCatStorer.

18. If we run hive as a server, what are the available mechanisms for connecting it from application?

Below are following ways by which we can connect with the Hive Server:

- Thrift Client: Using thrift we can call hive commands from a various programming
  languages e.g: Java, PHP, Python and Ruby.
- JDBC Driver : It supports the Type 4 (pure Java) JDBC Driver
- ODBC Driver: It supports ODBC protocol.

19. Is multi line comment supported in Hive Script ?

No.

20. What is SerDe in Apache Hive?

A SerDe is a Serializer Deserializer. Hive uses SerDe to read and write data from tables. An important concept behind Hive is that it DOES NOT own the Hadoop File System (HDFS) format that data is stored in. Users are able to write files to HDFS with whatever tools/mechanism takes their fancy(“CREATE EXTERNAL TABLE” or “LOAD DATA INPATH,”) and use Hive to correctly “parse” that file format in a way that can be used by Hive. A SerDe is a powerful and customizable mechanism that Hive uses to “parse” data stored in HDFS to be used by Hive.

21. Which classes are used by the Hive to Read and Write HDFS Files?

Following classes are used by Hive to read and write HDFS files

- *TextInputFormat/HiveIgnoreKeyTextOutputFormat*: These 2 classes read/write data in plain text file format.
- *SequenceFileInputFormat/SequenceFileOutputFormat*: These 2 classes read/write data in hadoop SequenceFile format.

22. What are the examples of the SerDe classes which hive uses to Serialize and Deserialize data?

Hive currently use below SerDe classes to serialize and deserialize data:

- *MetadataTypedColumnsetSerDe*: This SerDe is used to read/write delimited records like CSV, tab-separated control-A separated records (quote is not supported yet.)
- *ThriftSerDe*: This SerDe is used to read/write thrift serialized objects. The class file for the Thrift object must be loaded first.
- *DynamicSerDe*: This SerDe also read/write thrift serialized objects, but it understands thrift DDL so the schema of the object can be provided at run time. Also it supports a lot of different protocols, including TBinaryProtocol, TJSONProtocol, TCTLSeparatedProtocol

23. How do we write our own custom SerDe ?

In most cases, users want to write a Deserializer instead of a SerDe, because users just want to read their own data format instead of writing to it.

- For example, the *RegexDeserializer* will deserialize the data using the configuration
  parameter 'regex', and possibly a list of column names
- If your SerDe supports DDL (basically, SerDe with parameterized columns and column types), you probably want to implement a Protocol based on DynamicSerDe, instead of writing a SerDe from scratch.
- The reason is that the framework passes DDL to SerDe through"thrift DDL" format, and it's non-trivial to write a "thrift DDL" parser.

24. What is the functionality of Query Processor in Apache Hive ?

This component implements the processing framework for converting SQL to a graph of map/reduce jobs and the execution time framework to run those jobs in the order of dependencies.

25. What is ObjectInspector functionality ?

Hive uses *ObjectInspector* to analyze the internal structure of the row object and also
the structure of the individual columns. ObjectInspector provides a

uniform way to access complex objects that can be stored in multiple formats in the memory, including:

- Instance of a Java class (Thrift or native Java)
- A standard Java object (we use java.util.List to represent Struct and Array, and use
  java.util.Map to represent Map)
- A lazily-initialized object (For example, a Struct of string fields stored in a single Java string object with starting offset for each field)
- A complex object can be represented by a pair of ObjectInspector and Java Object.
- The ObjectInspector not only tells us the structure of the Object, but also gives us ways to access the internal fields inside the Object.

26. What are the types of tables in Hive?

There are two types of tables.

- Managed tables
- External tables

Only while dropping tables these two differentiates. Otherwise both type of tables are very similar.

27. What kind of data warehouse application is suitable for Hive?

Hive is not a full database. The design constraints and limitations of Hadoop and HDFS
impose limits on what Hive can do.
Hive is most suited for data warehouse applications, where

- Relatively static data is analyzed,
- Fast response times are not required, and
- When the data is not changing rapidly.

28. Does Hive provide OLTP or OLAP?

Hive doesn’t provide crucial features required for OLTP, Online Transaction Processing.
It’s closer to being an OLAP tool, Online Analytic Processing. So, Hive is best suited for
data warehouse applications, where a large data set is maintained and mined for insights, reports, etc.

29. Does Hive support record level Insert, delete or update?

No. Hive does not provide record-level update, insert, or delete. Henceforth, Hive does not
provide transactions too. However, users can go with CASE statements and built in functions of Hive to satisfy the above DML operations. Thus, a complex update query in
a RDBMS may need many lines of code in Hive.

30. How can we change a column data type in Hive?

We can use below command to alter data type of a column in hive.

ALTER TABLE table_name CHANGE column_name column_name new_datatype;

Example: If we want to change the data type of *empid* column from integer to bigint in a
table called employee.

ALTER TABLE employee CHANGE empid empid BIGINT;

31. How can we copy the columns of a hive table into a file?

By using *awk* command in shell, the output from HiveQL Describe command can be written to a file.

$ hive -S -e "describe table_name;" | awk -F" " ’{print 1}’ > ~/output.

32. How to rename a table in Hive?

Using ALTER command with RENAME, we can rename a table in Hive.

ALTER TABLE hive_table_name RENAME TO new_name;

33. Is there any alternative way to rename a table without ALTER command?

By using Import and export options we can be rename a table as shown below. Here we are saving the hive data into HDFS and importing back to new table like below.

EXPORT TABLE tbl_name TO 'HDFS_location';

IMPORT TABLE new_tbl_name FROM 'HDFS_location';

If we prefer to just preserve the data, we can create a new table from old table like
below.

CREATE TABLE new_tbl_name AS SELECT * FROM old_tbl_name;

DROP TABLE old_tbl_name;

34. What is the difference between order by and sort by in hive?

- SORT BY will sort the data within each reducer. We can use any number of reducers
for SORT BY operation.
- ORDER BY will sort all of the data together, which has to pass through one reducer.
Thus, ORDER BY in hive uses single reducer.
- ORDER BY guarantees total order in the output while SORT BY only guarantees
ordering of the rows within a reducer. If there is more than one reducer, SORT BY may give partially ordered final results

35. What is Double data type in Hive?

Double data type in Hive will present the data differently unlike RDBMS.
See the double type data below:
14324.0
342556.0
1.28893E4

E4 represents 10^4 here. So, the value1.28893E4 represents 12889.3. All the
calculations will be accurately performed using double type

It is crucial while exporting the double type data to any RDBMS since the type may be
wrongly interpreted. So, it is advised to cast the double type into appropriate type before
exporting.

36. What is the Hive configuration precedence order?

There is a precedence hierarchy to setting properties. In the following list, lower numbers take precedence over higher numbers:

1. **The Hive SET command**
2. **The command line -hiveconf option**
3. **hive-site.xml**
4. **hive-default.xml**

5. **hadoop-site.xml (or, equivalently, core-site.xml, hdfs-site.xml, and mapred-site.xml)**
6. **hadoop-default.xml (or, equivalently, core-default.xml, hdfs-default.xml, and mapred-default.xml)**

37. How do change settings within Hive Session?

We can change settings from within a session, too, using the SET command. This is useful for changing Hive or MapReduce job settings for a particular query. For example, the following command ensures buckets are populated according to the table definition.

```
hive> SET hive.enforce.bucketing=true;
```

To see the current value of any property, use SET with just the property name:

```
hive> SET hive.enforce.bucketing;
hive.enforce.bucketing=true
```

By itself, SET will list all the properties and their values set by Hive. This list will not include Hadoop defaults, unless they have been explicitly overridden in one of the ways covered in the above answer. Use SET -v to list all the properties in the system, including Hadoop defaults.

38. How to print header on Hive query results?

We need to use following set command before our query to show column headers in STDOUT.

hive> set hive.cli.print.header=true;

39. How to get detailed description of a table in Hive?

Use below hive command to get a detailed description of a hive table.

hive> describe extended <tablename>;

40. How to access sub directories recursively in Hive queries?

To process directories recursively in Hive, we need to set below two commands in hive session. These two parameters work in conjunction.

hive> Set mapred.input.dir.recursive=true;

hive> Set hive.mapred.supports.subdirectories=true;

Now hive tables can be pointed to the higher level directory. This is suitable for a scenario where the directory structure is as following:*/data/country/state/city*

41. How to skip header rows from a table in Hive?

Suppose while processing some log files, we may find header records.

System=….
Version=…
Sub-version=….

Like above, It may have 3 lines of headers that we do not want to include in our Hive query. To skip header lines from our tables in Hive we can set a table property that will allow us to skip the header lines.

```
CREATE EXTERNAL TABLE userdata (
name STRING,
job STRING,
dob STRING,
id INT,
salary INT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ' ' STORED AS TEXTFILE
LOCATION '/user/data'
TBLPROPERTIES("skip.header.line.count"="3");
```

42. Is it possible to create multiple table in hive for same data?

As hive creates schema and append on top of an existing data file. One can have multiple schema for one data file, schema will be saved in hive's metastore and data will not be parsed or serialized to disk in given schema. When we will try to retrieve data, schema will be used. For example if we have 5 column (name, job, dob, id, salary) in the data file present in hive metastore then, we can have multiple schema by choosing any number of columns from the above list. (Table with 3 columns or 5 columns or 6 columns).

But while querying, if we specify any column other than above list, will result in NULL values.

43. What is the maximum size of string data type supported by Hive?

Maximum size is 2 GB.

44. What are the Binary Storage formats supported in Hive?

By default Hive supports text file format, however hive also supports below binary formats.

Sequence Files, Avro Data files, RCFiles, ORC files, Parquet files

Sequence files: General binary format. splittable, compressible and row oriented. a typical example can be. if we have lots of small file, we may use sequence file as a container, where file name can be a key and content could stored as value. it support compression which enables huge gain in performance.

Avro datafiles: Same as Sequence file splittable, compressible and row oriented except support of schema evolution and multilingual binding support.

RCFiles: Record columnar file, it’s a column oriented storage file. it breaks table in row split. in each split stores that value of first row in first column and followed sub subsequently.

ORC Files: Optimized Record Columnar files

45. is HQL case sensitive?

HQL is not case sensitive.

46. Describe CONCAT function in Hive with Example?

CONCAT function will concatenate the input strings. We can specify any number of strings separated by comma.

Example: CONCAT ('Hive','-','is','-','a','-','data warehouse','-','in Hadoop');
Output: Hive-is-a-data warehouse-in Hadoop

So, every time we delimit the strings by '-'. If it is common for all the strings, then Hive provides another command CONCAT_WS. Here you have to specify the delimit operator first.

Syntax: CONCAT_WS ('-','Hive','is','a','data warehouse','in Hadoop');
Output: Hive-is-a-data warehouse-in Hadoop

47. Describe REPEAT function in Hive with example?

REPEAT function will repeat the input string n times specified in the command.

Example: REPEAT('Hive',3);
Output: HiveHiveHive.

48. Describe REVERSE function in Hive with example?

REVERSE function will reverse the characters in a string.

Example: REVERSE('Hive');
Output: eviH

49. Describe TRIM function in Hive with example?

TRIM function will remove the spaces associated with a string.

Example: TRIM(' Hadoop ');
Output: Hadoop.

If we want to remove only leading or trailing spaces then we can specify the below commands respectively.

LTRIM(' Hadoop');
RTRIM('Hadoop ');

50. Describe RLIKE in Hive with an example?

RLIKE (Right-Like) is a special function in Hive where if any substring of A matches with B then it evaluates to true. It also obeys Java regular expression pattern. Users don't need to put % symbol for a simple match in RLIKE.

Examples: 'Express' RLIKE 'Exp' –> True
'Express' RLIKE '^E.*' –> True (Regular expression)

Moreover, RLIKE will come handy when the string has some spaces. Without using TRIM function, RLIKE satisfies the required scenario. Suppose if A has value 'Express ' (2 spaces additionally) and B has value

‘Express’. In these situations, RLIKE will work better without using TRIM.

‘Express ‘ RLIKE ‘Express’ –> True

Note: RLIKE evaluates to NULL if A or B is NULL.

51. What kind of datawarehouse application is suitable for Hive?

Hive is not a full database. The design constraints and limitations of Hadoop and HDFS impose limits on what Hive can do.

Hive is most suited for data warehouse applications, where

1) Relatively static data is analyzed,

2) Fast response times are not required, and

3) When the data is not changing rapidly.

Hive doesn’t provide crucial features required for OLTP, Online Transaction Processing. It’s closer to being an OLAP tool, Online Analytic Processing.So, Hive is best suited for data warehouse applications, where a large data set is maintained and mined for insights, reports, etc.

52. How can the columns of a table in hive be written to a file?

By using awk command in shell, the output from HiveQL (Describe) can be written to a file.

53. LOWER or LCASE function in Hive with example?

LOWER or LCASE function will convert the input string to lower case characters.

Example:

LOWER('Hadoop');

LCASE('Hadoop');

Output:

hadoop

Note:

If the characters are already in lower case then they will be preserved.

54. UPPER or UCASE function in Hive with example?

UPPER or UCASE function will convert the input string to upper case characters.

Example:

UPPER('Hadoop');

UCASE('Hadoop');

Output:

HADOOP

Note:

If the characters are already in upper case then they will be preserved.

55. Can we change the data type of a column in a hive table?

Using REPLACE column option

ALTER TABLE table_name REPLACE COLUMNS ……

56. What is the need for custom Serde?

Depending on the nature of data the user has, the inbuilt SerDe may not satisfy the format of the

data. SO users need to write their own java code to satisfy their data format requirements.

57. What is the default location where hive stores table data?

hdfs://namenode_server/user/hive/warehouse

58. What are the three different modes in which hive can be run?

Local mode

Distributed mode

Pseudodistributed mode

59. Is there a date data type in Hive?

Yes. The TIMESTAMP data types stores date in java.sql.timestamp format

60. What are collection data types in Hive?

There are three collection data types in Hive.

ARRAY

MAP

STRUCT

61. Can we run unix shell commands from hive? Give example.

Yes, using the ! mark just before the command.

For example !pwd at hive prompt will list the current directory.

62. What is a Hive variable? What for we use it?

The hive variable is variable created in the Hive environment that can be referenced by Hive

scripts. It is used to pass some values to the hive queries when the query starts executing.

63. Can hive queries be executed from script files? How?

Using the source command.

Example –

Hive> source /path/to/file/file_with_query.hql

64. What is the importance of .hiverc file?

It is a file containing list of commands needs to run when the hive CLI starts. For example setting

the strict mode to be true etc.

65. What are the default record and field delimiter used for hive text files?

The default record delimiter is − \n

And the filed delimiters are − \001,\002,\003

66. What do you mean by schema on read?

The schema is validated with the data when reading the data and not enforced when writing data.

67. How do you list all databases whose name starts with p?

SHOW DATABASES LIKE ‘p.*’

68. What does the “USE” command in hive do?

With the use command you fix the database on which all the subsequent hive queries will run.

69. How can you delete the DBPROPERTY in Hive?

There is no way you can delete the DBPROPERTY.

What is the significance of the line

set hive.mapred.mode = strict;

It sets the mapreduce jobs to strict mode.By which the queries on partitioned tables can not run

without a WHERE clause. This prevents very large job running for long time.

70. How do you check if a particular partition exists?

This can be done with following query

SHOW PARTITIONS table_name PARTITION(partitioned_column=’partition_value’)

71. Which java class handles the Input record encoding into files which store the tables in Hive?

org.apache.hadoop.mapred.TextInputFormat

72. Which java class handles the output record encoding into files which result from Hive queries?

org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat

73. What is the significance of 'IF EXISTS" clause while dropping a table?

When we issue the command DROP TABLE IF EXISTS table_name

Hive throws an error if the table being dropped does not exist in the first place.

74. When you point a partition of a hive table to a new directory, what happens to the data?

The data stays in the old location. It has to be moved manually.

Write a query to insert a new columnnewcolINT into a hive table htab at a position before an existing

column xcol

ALTER TABLE table_name

CHANGE COLUMN new_col INT

BEFORE x_col

75. Does the archiving of Hive tables give any space saving in HDFS?

No. It only reduces the number of files which becomes easier for namenode to manage.

76. How can you stop a partition form being queried?

By using the ENABLE OFFLINE clause with ALTER TABLE statement.

77. While loading data into a hive table using the LOAD DATA clause, how do you specify it is a hdfs file and not a local file ?

By Omitting the LOCAL CLAUSE in the LOAD DATA statement.

78. If you omit the OVERWRITE clause while creating a hive table,what happens to file which are new and files which already exist?

The new incoming files are just added to the target directory and the existing files are simply overwritten. Other files whose name does not match any of the incoming files will continue to exist.

If you add the OVERWRITE clause then all the existing data in the directory will be deleted before new data is written.

79. What does the following query do?

```
INSERT OVERWRITE TABLE employees

PARTITION (country, state)

SELECT ..., se.cnty, se.st

FROM staged_employees se;
```

It creates partition on table employees with partition values coming from the columns in the select clause. It is called Dynamic partition insert.

80. What is a Table generating Function on hive?

A table generating function is a function which takes a single column as argument and expands it to multiple column or rows. Example explode

81. How can Hive avoid mapreduce?

If we set the property hive.exec.mode.local.auto to true then hive will avoid mapreduce to fetch query results.

82. Is it possible to create Cartesian join between 2 tables, using Hive?

No. As this kind of Join can not be implemented in mapreduce

83. As part of Optimizing the queries in HIve, what should be the order of table size in a join query?

In a join query the smallest table to be taken in the first position and largest table should be taken in the last position.

84. What is the usefulness of the DISTRIBUTED BY clause in Hive?

It controls how map output is divided among reducers. It is useful in case of streaming data

85. How will you convert the string '51.2' to a float value in the price column?

Select cast price as FLOAT

86. What will be the result when you do cast 'abc' as INT?

Hive will return NULL

87. Can the name of a view be same as the name of a hive table?

No. The name of a view must be unique when compared to all other tables and views present in the same database.

88. Can we LOAD data into a view?

No. A view can not be the target of a INSERT or LOAD statement.

89. What types of costs are associated in creating index on hive tables?

Indexes occupies space and there is a processing cost in arranging the values of the column on which index is created.

Give the command to see the indexes on a table.

SHOW INDEX ON table_name

This will list all the indexes created on any of the columns in the table table_name.

90. What is bucketing ?

The values in a column are hashed into a number of buckets which is defined by user. It is a way to

avoid too many partitions or nested partitions while ensuring optimizes query output.

91. What does streamtable(tablename) do?

It is query hint to stream a table into memory before running the query. It is a query optimization

Technique.

92. Can a partition be archived? What are the advantages and Disadvantages?

Yes. A partition can be archived. Advantage is it decreases the number of files stored in namenode

and the archived file can be queried using hive. The disadvantage is it will cause less efficient

query and does not offer any space savings.

93. What is a generic UDF in hive?

It is a UDF which is created using a java program to server some specific need not covered under

the existing functions in Hive. It can detect the type of input argument programmatically and

provide appropriate response.

94. The following statement failed to execute. What can be the cause?

LOAD DATA LOCAL INPATH '${env:HOME}/country/state/'

OVERWRITE INTO TABLE address;

The local inpath should contain a file and not a directory. The $env:HOME is a valid variable

available in the hive environment.

95. How do you specify the table creator name when creating a table in Hive?

The TBLPROPERTIES clause is used to add the creator name while creating a table.

The TBLPROPERTIES is added like –

TBLPROPERTIES('creator'= 'Joan')

96. Hive Performance Tuning ?

Below are the list of practices that we can follow to optimize Hive Queries.

1. Enable Compression in Hive

By enabling compression at various phases (i.e. on final output, intermediate data), we achieve the performance improvement in Hive Queries. For further details on how to enable compression Hive refer the post Compression in Hive.

2. Optimize Joins

We can improve the performance of joins by enabling Auto Convert Map Joins and enabling optimization of skew joins.

Auto Map Joins

Auto Map-Join is a very useful feature when joining a big table with a small table. if we enable this feature, the small table will be saved in the local cache on each node, and then joined with the big table in the Map phase. Enabling Auto Map Join provides two advantages. First, loading a small table into cache will save read time on each data node.

Second, it avoids *skew joins* in the Hive query, since the join operation has been already done in the Map phase for each block of data.

To enable the Auto Map-Join feature, we need to set below properties.

hive-site.xml

```
<property>

<name>hive.auto.convert.join</name>

<value>true</value>

<description>Whether Hive enables the optimization about
converting common join into mapjoin based on the input file
size</description>

</property>

<property>

<name>hive.auto.convert.join.noconditionaltask</name>

<value>true</value>

<description>

Whether Hive enables the optimization about converting common
join into mapjoin based on the input file size.

If this parameter is on, and the sum of size for n-1 of the
tables/partitions for a n-way join is smaller than the

specified size, the join is directly converted to a mapjoin (there is no
conditional task).

</description>

</property>
```

```
<property>

<name>hive.auto.convert.join.noconditionaltask.size</name>

<value>10000000</value>

<description>

If hive.auto.convert.join.noconditionaltask is off, this parameter does
not take affect.

However, if it is on, and the sum of size for n-1 of the tables/partitions
for a n-way join is smaller than this size,

the join is directly converted to a mapjoin(there is no conditional
task). The default is 10MB

</description>

</property>

<property>

<name>hive.auto.convert.join.use.nonstaged</name>

<value>false</value>

<description>

For conditional joins, if input stream from a small alias can be directly
applied to join operator without

filtering or projection, the alias need not to be pre-staged in
distributed cache via mapred local task.

Currently, this is not working with vectorization or tez execution
engine.

</description>

</property>
```

Skew Joins

We can enable optimization of skew joins, i.e. imbalanced joins by setting hive.optimize.skewjoin property to true either via SET command in hive shell or hive-site.xml file. Below are the list of properties that can be fine tuned to better optimize the skew joins.

hive-site.xml

```
<property>
<name>hive.optimize.skewjoin</name>
<value>true</value>
<description>
Whether to enable skew join optimization.
The algorithm is as follows: At runtime, detect the keys with a large skew. Instead of
processing those keys, store them temporarily in an HDFS directory. In a follow-up map-reduce
job, process those skewed keys. The same key need not be skewed for all the tables, and so,
the follow-up map-reduce job (for the skewed keys) would be much faster, since it would be a  map-join.
</description>
</property>
<property>
```

```
<name>hive.skewjoin.key</name>

<value>100000</value>

<description>

Determine if we get a skew key in join. If we see more than the
specified number of rows with the same key in join operator,

we think the key as a skew join key.

</description>

</property>

<property>

<name>hive.skewjoin.mapjoin.map.tasks</name>

<value>10000</value>

<description>

Determine the number of map task used in the follow up map join job
for a skew join.

It should be used together with hive.skewjoin.mapjoin.min.split to
perform a fine grained control.

</description>

</property>

<property>

<name>hive.skewjoin.mapjoin.min.split</name>

<value>33554432</value>

<description>
```

Determine the number of map task at most used in the follow up map join job for a skew join by specifying

the minimum split size. It should be used together with hive.skewjoin.mapjoin.map.tasks to perform a fine grained control.

</description>

</property>

Enable Bucketed Map Joins

If tables are bucketed by a particular column and these tables are being used in joins then we can enable bucketed map join to improve the performance. To do this, we can set below properties in **hive-site.xml** or hive shell.

```
<property>

<name>hive.optimize.bucketmapjoin</name>

<value>true</value>

<description>Whether to try bucket mapjoin</description>

</property>

<property>

<name>hive.optimize.bucketmapjoin.sortedmerge</name>

<value>true</value>

<description>Whether to try sorted bucket merge map
join</description>

</property>
```

3. Avoid Global Sorting in Hive

Global Sorting in Hive can be achieved in Hive with **ORDER BY** clause but this comes with a drawback. ORDER BY produces a result by setting the number of reducers to one, making it very inefficient for large datasets.

When a globally sorted result is not required, then we can use **SORT BY** clause. SORT BY produces a sorted file per reducer.

If we need to control which reducer a particular row goes to, we can use **DISTRIBUTE BY** clause, for example,

```
SELECT id, name, salary, dept FROM employee
DISTRIBUTE BY dept
SORT BY id ASC, name DESC;
```

Each *dept* will be processed separately by a reducer and records will be sorted by *id* and *name* fields within each *dept* separately.

4. Enable Tez Execution Engine

Instead of running Hive queries on venerable Map-reduce engine, we can improve the performance of hive queries at least by 100% to 300 % by running on Tez execution engine. We can enable the Tez engine with below property from hive shell.

```
hive> set hive.execution.engine=tez;
```

5. Optimize LIMIT operator

By default LIMIT operator still executes the entire query, then only returns a limited results. Because this behavior is generally wasteful, it can be avoided by setting below properties.

```
<property>
<name>hive.limit.optimize.enable</name>
<value>true</value>
<description>Whether to enable to optimization to trying a smaller subset of data for simple LIMIT first.</description>
</property>
<property>
<name>hive.limit.row.max.size</name>
<value>100000</value>
<description>When trying a smaller subset of data for simple LIMIT, how much size we need to guarantee each row to have at least.</description>
</property>
<property>
<name>hive.limit.optimize.limit.file</name>
<value>10</value>
<description>When trying a smaller subset of data for simple LIMIT, maximum number of files we can sample.</description>
</property>
```

<property>

<name>hive.limit.optimize.fetch.max</name>

<value>50000</value>

<description>

Maximum number of rows allowed for a smaller subset of data for simple LIMIT, if it is a fetch query.

Insert queries are not restricted by this limit.

</description>

</property>

6. Enable Parallel Execution

Hive converts a query into one or more stages. Stages could be a MapReduce stage, sampling stage, a merge stage, a limit stage. By default, Hive executes these stages one at a time. A particular job may consist of some stages that are not dependent on each other and could be executed in

parallel, possibly allowing the overall job to complete more quickly. Parallel execution can be enabled by setting below properties.

<property>

<name>hive.exec.parallel</name>

<value>true</value>

<description>Whether to execute jobs in parallel</description>

</property>

<property>

```
<name>hive.exec.parallel.thread.number</name>
<value>8</value>
<description>How many jobs at most can be executed in parallel</description>
</property>
```

7. Enable Mapreduce Strict Mode

we can enable mapreduce strict mode by setting below property to strict.

```
<property>
<name>hive.mapred.mode</name>
<value>nonstrict</value>
<description>
The mode in which the Hive operations are being performed.
In strict mode, some risky queries are not allowed to run. They include:
Cartesian Product.
No partition being picked up for a query.
Comparing bigints and strings.
Comparing bigints and doubles.
Orderby without limit.
</description>
```

8. Single Reduce for Multi Group BY

By enabling single reducer task for multi group by operations, we can combine multiple GROUP BY operations in a query into a single MapReduce job.

```
<property>
<name>hive.multigroupby.singlereducer</name>
<value>true</value>
<description>
Whether to optimize multi group by query to generate single M/R  job
plan. If the multi group by query has
common group by keys, it will be optimized to generate single M/R
job.
</description>
</property>
```

9. Controls Parallel Reduce Tasks

We can control the number of parallel reduce tasks that can be run for a given hive query with below properties.

```
<property>
<name>hive.exec.reducers.bytes.per.reducer</name>
<value>256000000</value>
<description>size per reducer.The default is 256Mb, i.e if the input
size is 1G, it will use 4 reducers.</description>
```

```
</property>
<property>
<name>hive.exec.reducers.max</name>
<value>1009</value>
<description>
max number of reducers will be used. If the one specified in the configuration parameter mapred.reduce.tasks is
negative, Hive will use this one as the max number of reducers when automatically determine number of reducers.
</description>
</property>
```

we can also set the parallel reduce tasks to a fixed value with below property.

```
hive> set mapred.reduce.tasks=32;
```

10. Enable Vectorization

Vectorization feature is introduced into hive for the first time in hive-0.13.1 release only. By vectorized query execution, we can improve performance of operations like scans, aggregations, filters and joins, by performing them in batches of 1024 rows at once instead of single row each time.

We can enable vectorized query execution by setting below three properties in either hive shell or hive-site.xml file.

```
hive> set hive.vectorized.execution.enabled = true;
hive> set hive.vectorized.execution.reduce.enabled = true;
hive> set hive.vectorized.execution.reduce.groupby.enabled = true;
```

11. Enable Cost Based Optimization

Recent Hive releases provided the feature of cost based optimization, one can achieve further optimizations based on query cost, resulting in potentially different decisions: how to order joins, which type of join to perform, degree of parallelism and others.

cost based optimization can be enabled by setting below properties in hive-site.xml file.

hive-site.xml

```
<property>
<name>hive.cbo.enable</name>
<value>true</value>
<description>Flag to control enabling Cost Based Optimizations using Calcite framework.</description>
</property>
<property>
<name>hive.compute.query.using.stats</name>
<value>true</value>
```

```
<description>
When set to true Hive will answer a few queries like count(1) purely using stats
stored in metastore. For basic stats collection turn on the config hive.stats.autogather to true.
For more advanced stats collection need to run analyze table queries.
</description>
</property>
<property>
<name>hive.stats.fetch.partition.stats</name>
<value>true</value>
<description>
Annotation of operator tree with statistics information requires partition level basic
statistics like number of rows, data size and file size. Partition statistics are fetched from
metastore. Fetching partition statistics for each needed partition can be expensive when the
number of partitions is high. This flag can be used to disable fetching of partition statistics
from metastore. When this flag is disabled, Hive will make calls to filesystem to get file sizes
and will estimate the number of rows from row schema.
</description>
```

```
</property>

<property>

<name>hive.stats.fetch.column.stats</name>

<value>true</value>

<description>

Annotation of operator tree with statistics information requires
column statistics.

Column statistics are fetched from metastore. Fetching column
statistics for each needed column

can be expensive when the number of columns is high. This flag can
be used to disable fetching

of column statistics from metastore.

</description>

</property>

<property>

<name>hive.stats.autogather</name>

<value>true</value>

<description>A flag to gather statistics automatically during the
INSERT OVERWRITE command.</description>

</property>

<property>

<name>hive.stats.dbclass</name>

<value>fs</value>
```

<description>

Expects one of the pattern in [jdbc(:.*), hbase, counter, custom, fs].

The storage that stores temporary Hive statistics. In filesystem based statistics collection ('fs'),

each task writes statistics it has collected in a file on the filesystem, which will be aggregated

after the job has finished. Supported values are fs (filesystem), jdbc:database (where database

can be derby, mysql, etc.), hbase, counter, and custom as defined in StatsSetupConst.java.

</description>

</property>

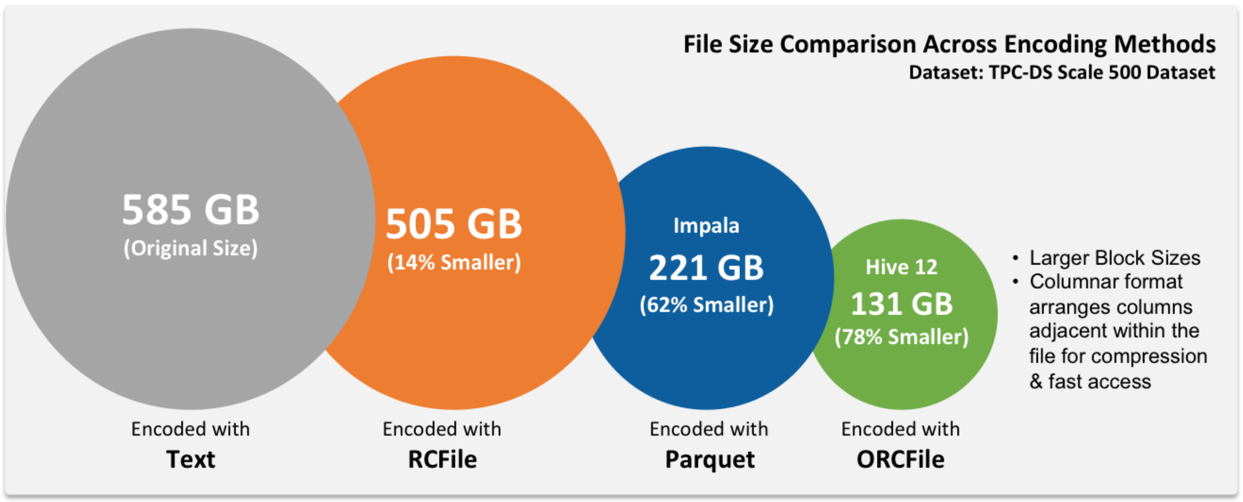
And we can gather basic statistics about all columns in an employee table with below command in hive shell.

hive> ANALYZE TABLE employee COMPUTE STATISTICS FOR COLUMNS;

hive> ANALYZE TABLE employee COMPUTE STATISTICS FOR COLUMNS id, dept;

12. Use ORC File Format

Using ORC (Optimized Record Columnar) file format we can improve the performance of Hive Queries very effectively. Below picture on file format best depicts the power of ORC file file over other formats.



Create Tables with ORC File Format

We can create new hive table with ORC file format with just by adding STORED AS ORC clause to CREATE TABLE command in hive. Optionally we can provide compression techniques in TBLPROPERTIES clause.

Convert Existing Tables to ORC

Create a table with the same schema as the source table and STORED AS ORC, then we can submit below command to copy data from regular old table new ORC formatted table.

hive> INSERT OVERWRITE TABLE orc_emp SELECT * FROM emp;

Key	Default	Notes
orc.compress	ZLIB	Compression to use in addition to columnar compression (one of NONE, ZLIB, SNAPPY)

orc.compress.size	262,144 (= 256 KiB)	Number of bytes in each compression chunk
orc.stripe.size	268,435,456 (= 256 MiB)	Number of bytes in each stripe
orc.row.index.stride	10,000	Number of rows between index entries (must be >= 1,000)
orc.create.index	true	Whether to create inline indexes

<u>Example ORC table creation:</u>

hive> CREATE TABLE EMP_ORC (id int, name string, age int, address string)

> STORED AS ORC tblproperties (“orc.compress" = “SNAPPY”);

hive> INSERT OVERWRITE TABLE EMP_ORC SELECT * FROM EMP;

Thus by using these 12 techniques we can improve the performance of Hive Queries.

97. Enable Compression in Hive ?

Find Available Compression Codecs in Hive

To enable compression in Hive, first we need to find out the available compression codes on hadoop cluster, and we can use below setcommand to list down the available compression codecs.

Compression Codecs in Hive

```
hive> set io.compression.codecs;
io.compression.codecs=
    org.apache.hadoop.io.compress.GzipCodec,
    org.apache.hadoop.io.compress.DefaultCodec,
    org.apache.hadoop.io.compress.BZip2Codec,
    org.apache.hadoop.io.compress.SnappyCodec
hive>
```

Enable Compression on Intermediate Data

A complex Hive query is usually converted to a series of multi-stage MapReduce jobs after submission, and these jobs will be chained up by the Hive engine to complete the entire query. So "intermediate output" here refers to the output from the previous MapReduce job, which will be used to feed the next MapReduce job as input data.

We can enable compression on Hive Intermediate output by setting the property hive.exec.compress.intermediate either from Hive Shell using set command or at site level in hive-site.xml file.

hive-site.xml

```
  <property>
    <name>hive.exec.compress.intermediate</name>
    <value>true</value>
    <description>
      This controls whether intermediate files produced by Hive
between multiple map-reduce jobs are compressed.
      The compression codec and other options are determined from
Hadoop config variables mapred.output.compress*
    </description>
  </property>
  <property>
    <name>hive.intermediate.compression.codec</name>
    <value>org.apache.hadoop.io.compress.SnappyCodec</value>
    <description/>
  </property>
  <property>
    <name>hive.intermediate.compression.type</name>
    <value>BLOCK</value>
    <description/>
  </property>
```

Or we can set these properties in hive shell as shown below with set commands.

```
hive> set hive.exec.compress.intermediate=true;
hive> set hive.intermediate.compression.codec=org.apache.hadoop.io.compress.SnappyCodec;
hive> set hive.intermediate.compression.type=BLOCK;
hive>
```

Enable Compression on Final Output

We can enable compression on final output in hive shell by setting below properties.

```
 <property>
   <name>hive.exec.compress.output</name>
   <value>true</value>
   <description>
     This controls whether the final outputs of a query (to a local/HDFS file or a Hive table) is compressed.
     The compression codec and other options are determined from Hadoop config variables mapred.output.compress*
   </description>
 </property>
```

Or

```
hive> set
mapreduce.output.fileoutputformat.compress.codec=org.apache.had
oop.io.compress.GzipCodec;

hive> set mapreduce.output.fileoutputformat.compress.type=BLOCK;

hive>
```

Example Table Creation with Compression Enabled

In the below shell snippet we are creating a new table compressed_emp from existing testemp table in hive after setting the compression properties to true in the hive shell.

Source Table: testemp contents

```
hive> select * from testemp;

OK

123	Ram	Team Lead

345	Siva	Member

678	Krishna		Member

Time taken: 0.096 seconds, Fetched: 3 row(s)

hive>
```

Setting Compression properties in Hive Shell:

```
hive> set hive.exec.compress.output=true;

hive> set mapreduce.output.fileoutputformat.compress=true;
hive> set
mapreduce.output.fileoutputformat.compress.codec=org.apache.had
oop.io.compress.GzipCodec;

hive> set mapreduce.output.fileoutputformat.compress.type=BLOCK;

hive> set hive.exec.compress.intermediate=true;
```

<u>**Target Table compressed_emp Creation:**</u>

```
hive> CREATE TABLE compressed_emp ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'

    > AS SELECT * FROM testemp;

Query ID = hadoop1_20150502214545_8eb6915e-8d0d-4109-b743-
cb6505dfa26b

Total jobs = 3

Launching Job 1 out of 3

Number of reduce tasks is set to 0 since there's no reduce operator

Starting Job = job_1430579477500_0002, Tracking URL =
http://localhost:8088/proxy/application_1430579477500_0002/

Kill Command = /usr/lib/hadoop/hadoop-2.3.0/bin/hadoop job  -kill
job_1430579477500_0002

Hadoop job information for Stage-1: number of mappers: 1; number
of reducers: 0

2015-05-02 21:45:28,983 Stage-1 map = 0%,  reduce = 0%

2015-05-02 21:45:34,437 Stage-1 map = 100%,  reduce = 0%,
Cumulative CPU 1.19 sec
```

```
MapReduce Total cumulative CPU time: 1 seconds 190 msec

Ended Job = job_1430579477500_0002

Stage-4 is selected by condition resolver.

Stage-3 is filtered out by condition resolver.

Stage-5 is filtered out by condition resolver.

Moving data to:
hdfs://localhost:9000/tmp/mydir/hadoop1/af416484-b7cb-4036-
bfbd-5942c09fcfd9/hive_2015-05-02_21-45-
19_676_3039038823977237962-1/-ext-10001

Moving data to:
hdfs://localhost:9000/user/hive/warehouse/compressed_emp

Table default.compressed_emp stats: [numFiles=1, numRows=3,
totalSize=66, rawDataSize=50]

MapReduce Jobs Launched:

Stage-Stage-1: Map: 1   Cumulative CPU: 1.19 sec   HDFS Read: 268
HDFS Write: 144 SUCCESS

Total MapReduce CPU Time Spent: 1 seconds 190 msec

OK

Time taken: 16.422 seconds

hive> dfs -ls -R /user/hive/warehouse/compressed_emp;

-rw-r--r--   1 hadoop1 supergroup         66 2015-05-02 21:45
/user/hive/warehouse/compressed_emp/000000_0.gz

hive> dfs -cat /user/hive/warehouse/compressed_emp/000000_0.gz;

342?
```

```
    J??
      IM?U?IML?261?
                ?,K??M?MJ-?23???.?,??
                              ???5hive> ;
hive> dfs -text
/user/hive/warehouse/compressed_emp/000000_0.gz;
123   Ram  Team Lead
345   Siva   Member
678   Krishna      Member
hive>
```

Thus we can create the output files in gzipped format and we can view the contents of this file with dfs -text command.

98. Hive on Tez – Hive Integration with Tez

<u>Tez Advantages:</u>

- Tez offers a customizable execution architecture that allows us to express complex computations as data flow graphs and allows for dynamic performance optimizations based on real information about the data and the resources required to process it.
- Tez increases the processing speed from GB’s to PB’s of data and 10’s to 1000’s of nodes when compared to mapreduce framework.

- The Apache Tez library allows developers to create Hadoop applications that integrate with YARN and perform well with Hadoop clusters.

Benefits of Integrating Hive with Tez:

- Tez can translate complex SQL statements into highly optimized, purpose-built data processing graphs that strike the right balance between performance, throughput, and scalability across a wide range of use cases and data set sizes.
- Tez helps Hive in becoming into interactive from batch mode.
- Till hive-0.12 release, there is only mapreduce framework available in hive to convert hive queries into execution jobs on hadoop clusters. But first time in hive-0.13.1 release Tez execution engine framework is embedded into hive to improve the performance of complex hive queries.

Hive on Tez:

By default, execution engine in hive is mapreduce (mr), so we don't need to specify it explicitly to submit mapreduce jobs from our hive queries. To setup hive on tez, we need below components at the minimum.

Prerequisite:

- Running Hadoop 2 cluster with YARN framework
- Hive-0.13.1 installed on hadoop cluster
- Tez installed and configured on Hadoop successfully.

For tez installation and configuration hadoop2 we can refer our previous post on Tez framework.

We assume all the above three installations are done already and running fine.

<u>Hive setup for Tez:</u>

As Tez is already installed successfully and we are able to sample Tez DAG jobs successfully on hadoop cluster, now we can easily setup Hive for Tez engine.

We need to perform below list of activities in the same order.

- As of Hive 0.13.1 release, Hive embeds Tez, we need to copy hive-exec-0.13.1.jar file from $HIVE_HOME/lib directory into HDFS directory specified in tez.lib.uris property in tez-site.xml file in ${TEZ_CONF_DIR}. In this post, it is /apps/tez-0.4.1 is the HDFS directory. Use below command to copy this jar.

$ hadoop fs -put $HIVE_HOME/lib/hive-exec-0.13.1.jar /apps/tez-0.4.1/

To run query on Tez engine, we need whether to set hive.execution.engine=tez; each time for hive session or change this value permanently in hive-site.xml. In this post, we will simply set this hive variable for each session to compare results of mr and tezframeworks.

```
hive> set hive.execution.engine=tez;
```

That's it we are done with hive setup for Tez.

PIG

1. What is Apache Pig?

Pig is a scripting language for exploring huge data sets of size gigabytes or terabytes very easily. Pig provides an engine for executing data flows in parallel on Hadoop

Apache Pig is top level project in Apache Software foundation for analyzing large data sets that consists of a high-level language for expressing data analysis programs.

2. What is Pig Latin?

Pig Latin is a data flow Scripting Language like Perl for exploring large data sets. A Pig Latin program is made up of a series of operations, or transformations, that are applied to the input data to produce output.

3. What is Pig Engine?

Pig Engine is an execution environment to run Pig Latin programs. It converts these Pig Latin operators or transformations into a series of MapReduce jobs.

4. What are the modes of Pig Execution?

Pig execution can be done in two modes.

- **Local Mode**: Local execution in a single JVM, all files are installed and run using local host and file system.
- **Mapreduce Mode**: Distributed execution on a Hadoop cluster, it is the default mode.

5. What is bag?

A bag is one of the data models present in Pig. It is an unordered collection of tuples with possible duplicates. Bags are used to store collections while grouping. The size of bag is the size of the local disk, this means that the size of the bag is limited. When the bag is full, then Pig will spill this bag into local disk and keep only some parts of the bag in memory. There is no necessity that the complete bag should fit into memory. We represent bags with "{}".

6. What are the Pig Latin Features?

- Pig Latin script is made up of a series of operations, or transformations, that are
  applied to the input data to produce output
- Pig Latin programs can be executed either in Interactive
  mode through Grunt shellor in Batch mode via Pig Latin Scripts.

- Pig Latin includes operators for many of the traditional data operations (join, sort, filter, etc.)
- User Defined Functions (UDF)
- Debugging Environment

7. What are the advantages of using Pig over Mapreduce?

In Mapreduce,

- Development cycle is very long. Writing mappers and reducers, compiling
  and packaging the code, submitting jobs, and retrieving the results is a time
  Consuming process.
- Performing Data set joins is very difficult
- Low level and rigid, and leads to a great deal of custom user code that is hard to maintain and reuse is complex.

In pig,

- No need of compiling or packaging of code. Pig operators will be converted into map or reduce tasks internally.
- Pig Latin provides all of the standard data-processing operations, such as join, filter, group by, order by, union, etc
- high level of abstraction for processing large data sets

8. What is the difference between Pig Latin and HiveQL ?

Pig Latin:

- Pig Latin is a Procedural language
- Nested relational data model
- Schema is optional

HiveQL:

- HiveQL is Declarative
- HiveQL flat relational
- Schema is required

9. What are the common features in Pig and Hive?

- Both provide high level abstraction on top of Mapreduce
- Both convert their commands internally into Mapreduce jobs
- Both doesn’t support low-latency queries and thus OLAP or OLTP are not supported.

10. What is the difference between logical and physical plans?

Pig undergoes some steps when a Pig Latin Script is converted into MapReduce jobs. After performing the basic parsing and semantic checking, it produces a logical plan. The logical plan describes the logical operators that have to be executed by Pig during execution. After this, Pig produces a physical plan. The physical plan describes the physical operators that are needed to execute the script.

11. Why do we need MapReduce during Pig programming?

Pig is a high-level platform that makes many Hadoop data analysis issues easier to execute. A program written in Pig Latin is like a query written in SQL, where we need an
execution engine to execute the query. So, Pig engine will convert the program into MapReduce jobs. Here, MapReduce acts as the execution engine.

12. How many ways we can run Pig programs?

Pig programs or commands can be executed in three ways.

- **Script – Batch Method**
- **Grunt Shell – Interactive Method**
- **Embedded mode**

All these ways can be applied to both Local and Mapreduce modes of execution.

13. What is Grunt in Pig?

Grunt is an Interactive Shell in Pig, and below are its major features:

- Ctrl-E key combination will move the cursor to the end of the line.
- Grunt remembers command history, and can recall lines in the history buffer using up or down cursor keys.
- Grunt supports Auto completion mechanism, which will try to complete
  Pig Latin keywords and functions when you press the Tab key

14. Is Pig Latin Case Sensitive?

The names (aliases) of relations and fields are case sensitive. The names of Pig Latin
functions are case sensitive. The names of parameters and all other Pig Latin keywords are case insensitive.

15. What is bag?

A bag is one of the data models present in Pig. It is an un-ordered collection of tuples with possible duplicates. Bags are used to store collections while grouping. The size of bag is the size of the local disk, this means that the size of the bag is limited. When the bag is full, then Pig will spill this bag into local disk and keep only some parts of the bag in memory. There is no necessity that the complete bag should fit into memory. We represent bags
with “{}”.

16. What is a tuple?

A tuple is an ordered set of fields and A field is a piece of data.

17. What is a relation in Pig?

A Pig relation is a bag of tuples. A Pig relation is similar to a table in a relational database,
where the tuples in the bag correspond to the rows in a table. Unlike a relational table,
however, Pig relations don't require that every tuple contain the same number of fields or that the fields in the same position (column) have the same type.

18. What does mean by unordered collection in a bag or in a relation?

Relations are unordered means there is no guarantee that tuples
are processed in any particular order. Furthermore, processing may be paralleled in which case tuples are not processed according to any total ordering.

19. How the fields are referenced in a relation?

Fields in a relation can be referenced in two ways, by positional notation or by name (alias)

- Positional notation is generated by the system. Positional notation is indicated with the dollar sign ($) and begins with zero (0); for example, $0, $1, $2.
- Names are assigned by user using schema (or, in the case of the GROUP operator and some functions, by the system). We can use any name that is not a Pig keyword.

20. What are the simple data types supported by Pig?

Simple Types	Description	Example
int	Signed 32-bit	integer 10
long	Signed 64-bit	integer Data:10L or 10l
float	32-bit floating point	Data: 10.5F or 10.5f or 10.5e2f
double	64-bit floating point	Data: 10.5 or 10.5e2 or 10.5E2
chararray	Character array	hello world
bytearray	Byte array	
boolean	boolean	true/false (case insensitive)
datetime	datetime	1970-01-01T00:00:00.000+00:00
biginteger	Java BigInteger	200000000000
bigdecimal	Java BigDecimal	33.4567833213

21. What are the complex data types supported in Pig Latin?

Data Types	Description	Example
tuple	An ordered set of fields.	(19,2)
bag	A collection of tuples.	{(19,2), (18,1)}
map	A set of key value pairs.	[open#apache]

22. What are the features of bag?

- A bag can have duplicate tuples.
- A bag can have tuples with differing numbers of fields. However, if Pig tries to access a field that does not exist, a null value is substituted.
- A bag can have tuples with fields that have different data types. However, for Pig to
  effectively process bags, the schemas of the tuples within those bags should be the same.

23. What is an outer bag?

An outer bag is nothing but a relation.

24. What is an inner bag?

An inner bag is a relation inside any other bag.

Example: (4,{(4,2,1),(4,3,3)})

In the above example, the complete relation is an outer bag
and {(4,2,1),(4,3,3)} is an inner bag.

25. What is a Map?

A map is a set of key/value pairs. Key values within a relation must be unique.

26. What does FOREACH do?

FOREACH is used to apply transformations to the data and to generate new data items. The name itself is indicating that for each element of a data bag, the respective action will be performed.

Syntax: FOREACH bagname GENERATE expr1, expr2, …..

The meaning of this statement is that the expressions mentioned after GENERATE will be applied to the current record of the data bag.

27. What does DISTINCT operator will do in Pig?

Removes duplicate tuples in a relation.

Syntax: alias = DISTINCT alias;

28. What is FILTER operator in Pig?

Selects tuples from a relation based on some condition.

Syntax: alias = FILTER alias BY expression;

29. What does GROUP operator will do in Pig?

Groups the data in one or more relations.

Syntax: alias = GROUP alias { ALL | BY expression} [, alias ALL | BY expression …] [USING 'collected' | 'merge'] [PARTITION BY partitioner] [PARALLEL n];

30. What is difference between GROUP and COGROUP?

The GROUP and COGROUP operators are identical. Both operators work with one or more relations. For readability GROUP is used in statements involving one relation and COGROUP is used in statements involving two or more relations. We can COGROUP up to
but no more than 127 relations at a time.

31. Can you give us some examples how Hadoop is used in real time environment?

Let us assume that we have an exam paper consisting of 10 Multiple-choice questions and 20 students appear for that exam. Every student will attempt each question. For each question and each answer option, a key will be generated. So we have a set of *key-value pairs* for all the questions and all the answer options for every student. Based on the options that the students have selected, you have to analyze and find out how many students have answered correctly. This isn't an easy task. Here Hadoop comes into picture! Hadoop helps you in solving these problems quickly and without much effort. You may also take the case of how many students have wrongly attempted a particular question.

32. What is BloomMapFile used for?

The *BloomMapFile* is a class that extends *MapFile*. So its functionality is similar to MapFile. BloomMapFile uses dynamic Bloom filters to provide quick membership test for the keys. It is used in Hbase table format.

33. What is the difference between logical and physical plans?

Pig undergoes some steps when a Pig Latin Script is converted into MapReduce jobs. After performing the basic parsing and semantic checking, it produces a logical plan. The *logical plan* describes the logical operators that have to be executed by Pig during execution. After this, Pig produces a physical plan. The *physical plan* describes the physical operators that are needed to execute the script.

34. Does ‘ILLUSTRATE’ run MR job?

No, illustrate will not pull any MR, it will pull the internal data. On the console, illustrate will not do any job. It just shows output of each stage and not the final output.

35. Is the keyword ‘DEFINE’ like a function name?

Yes, the keyword ‘DEFINE’ is like a function name. Once you have registered, you have to define it. Whatever logic you have written in Java program, you have an exported jar and also a jar registered by you. Now the compiler will check the function in exported jar. When the function is not present in the library, it looks into your jar.

36. Is the keyword ‘FUNCTIONAL’ a User Defined Function (UDF)?

No, the keyword ‘FUNCTIONAL’ is not a User Defined Function (UDF). While using UDF, we have to override some functions. Certainly you have to do your job with the help of these functions only. But the keyword ‘FUNCTIONAL’ is a built-in function i.e a pre-defined function, therefore it does not work as a UDF.

37. Why do we need MapReduce during Pig programming?

Pig is a high-level platform that makes many Hadoop data analysis issues easier to execute. The language we use for this platform is: *Pig Latin*. A program written in *Pig Latin* is like a query written in SQL, where we need an execution engine to execute the query. So, when a program is written in Pig Latin, Pig compiler will convert the program into MapReduce jobs. Here, *MapReduce* acts as the execution engine.

38. Are there any problems which can only be solved by MapReduce and cannot be solved by PIG? In which kind of scenarios MR jobs will be more useful than PIG?

Let us take a scenario where we want to count the population in two cities. I have a data set and sensor list of different cities. I want to count the population by using one mapreduce for two cities. Let us assume that one is Bangalore and the other is Noida. So I need to consider key of Bangalore city similar to Noida through which I can bring the population data of these two cities to one reducer. The idea behind this is some how I have to instruct map reducer program – whenever you find city with the name '*Bangalore*' and city with the name '*Noida'*, you create the alias name which will be the common name for these two cities so that you create a common key for both the cities and it get passed to the same reducer. For this, we have to write *custom partitioner*.

In mapreduce when you create a '*key'* for city, you have to consider *'city'* as the key. So, whenever the framework comes across a different city, it considers it as a different key. Hence, we need to use customized partitioner. There is a provision in mapreduce only, where you can write your custom partitioner and mention if city = bangalore or noida then pass similar hashcode. However, we cannot create custom partitioner in Pig. As Pig is not a framework, we cannot direct execution engine to customize the partitioner. In such scenarios, MapReduce works better than Pig.

39. Does Pig give any warning when there is a type mismatch or missing field?

No, Pig will not show any warning if there is no matching field or a mismatch. If you assume that Pig gives such a warning, then it is difficult to find in log file. If any mismatch is found, it assumes a null value in Pig.

40. What is the function of co-group in Pig?

Co-group joins the data set by grouping one particular data set only. It groups the elements by their common field and then returns a set of records containing two separate bags. The first bag consists of the record of the first data set with the common data set and the second bag consists of the records of the second data set with the common data set.

41. Can we say co-group is a group of more than 1 data set?

Co-group is a group of one data set. But in the case of more than one data sets, co-group will group all the data sets and join them based on the common field. Hence, we can say that co-group is a *group* of more than one data set and *join* of that data set as well.

42. What does FOREACH do?

FOREACH is used to apply transformations to the data and to generate new data items. The name itself is indicating that for each element of a data bag, the respective action will be performed.

***Syntax** :* FOREACH bagname GENERATE expression1, expression2, …..
The meaning of this statement is that the expressions mentioned after GENERATE will be applied to the current record of the data bag.

SQOOP

1. What is Sqoop?

Sqoop is an open source tool that enables users to transfer bulk data between Hadoop eco system and relational databases.

2. What are the relational databases supported in Sqoop?

Below are the list of RDBMSs that are supported by Sqoop Currently.

- MySQL
- PostGreSQL
- Oracle
- Microsoft SQL
- IBM’s Netezza
- Teradata

3. What are the destination types allowed in Sqoop Import command?

Currently Sqoop Supports data imported into below services.

- HDFS
- Hive
- HBase
- HCatalog
- Accumulo

4. Is Sqoop similar to distcp in hadoop?

Partially yes, hadoop's distcp command is similar to Sqoop Import command. Both submits parallel map-only jobs but distcp is used to copy any type of files from Local FS/HDFS to HDFS and Sqoop is for transferring the data records only between RDMBS and Hadoop eco system services, HDFS, Hive and HBase.

5. What are the majorly used commands in Sqoop?

In Sqoop Majorly Import and export commands are used. But below commands are also useful some times.

- codegen
- eval
- import-all-tables
- job

- list-databases
- list-tables
- merge
- metastore

6. When Importing tables from MySQL to what are the precautions that needs to be taken care w.r.t to access?

In MySQL, we need to make sure that we have granted all privileges on the databases, that needs to be accessed, should be given to all users at destination hostname. If Sqoop is being run under localhost and MySQL is also present on the same then we can grant the permissions with below two commands from MySQL shell logged in with ROOT user.

MySQL:

```
$ mysql -u root -p
mysql> GRANT ALL PRIVILEGES ON *.* TO '%'@'localhost';
mysql> GRANT ALL PRIVILEGES ON *.* TO ''@'localhost';
```

7. What if my MySQL server is running on MachineA and Sqoop is running on MachineB for the above question?

From MachineA login to MySQL shell and perform the below command as root user. If using hostname of second machine, then that should be added to /etc/hosts file of first machine.

MySQL:

$ mysql -u root -p

mysql> GRANT ALL PRIVILEGES ON *.* TO '%'@'MachineB hostname or Ip address';

mysql> GRANT ALL PRIVILEGES ON *.* TO ''@'MachineB hostname or Ip address';

8. How Many Mapreduce jobs and Tasks will be submitted for Sqoop copying into HDFS?

For each sqoop copying into HDFS only one mapreduce job will be submitted with 4 map tasks. There will not be any reduce tasks scheduled.

9. How can we control the parallel copying of RDBMS tables into hadoop ?

We can control/increase/decrease speed of copying by configuring the number of map tasks to be run for each sqoop copying process. We can do this by providing argument -m 10 or –num-mappers 10 argument to sqoop import command. If we specify -m 10 then it will submit 10 map tasks parallel at a time. Based on our requirement we can increase/decrease this number to control the copy speed.

10. What is the criteria for specifying parallel copying in Sqoop with multiple parallel map tasks?

To use multiple mappers in Sqoop, RDBMS table must have one primary key column (if present) in a table and the same will be used as split-by column in Sqoop process. If primary key is not present, we need to provide any unique key column or set of columns to form unique valuesand these should be provided to -split-by column argument.

11. While loading tables from MySQL into HDFS, if we need to copy tables with maximum possible speed, what can you do ?

We need to use –direct argument in import command to use direct import fast path and this –direct can be used only with MySQL and PostGreSQL as of now.

12. What is the example connect string for Oracle database to import tables into HDFS?

We need to use Oracle JDBC Thin driver while connecting to Oracle database via Sqoop. Below is the sample import command to pull tableemployees from oracle database testdb.

MySQL:

```
sqoop import \
--connect jdbc:oracle:thin:@oracle.example.com/testdb \
--username SQOOP \
--password sqoop \
--table employees
```

13. While connecting to MySQL through Sqoop, I am getting Connection Failure exception what might be the root cause and fix for this error scenario?

This might be due to insufficient permissions to access your MySQL database over the network. To confirm this we can try the below command to connect to MySQL database from Sqoop's client machine.

```
$ mysql --host=MySql node&gt; --database=test --user= --password=
```

If this is the case then we need grant permissions user @ sqoop client machine as per the answer to Question 6 in this post.

14. While importing tables from Oracle database, Sometimes I am getting java.lang.IllegalArgumentException: Attempted to generate class with no columns! or NullPointerException what might be the root cause and fix for this error scenario?

While dealing with Oracle database from Sqoop, Case sensitivity of table names and user names matters highly. Most probably by specifying these two values in UPPER case will solve the issue unless actual names are mixed with Lower/Upper cases. If these are mixed, then we need to provide them within double quotes.

In case, the source table is created under different user namespace, then we need to provide table name as USERNAME.TABLENAME as shown below.

MySQL:

```
sqoop import \
--connect jdbc:oracle:thin:@oracle.example.com/ORACLE \
--username SQOOP \
--password sqoop \
--table SIVA.EMPLOYEES
```

15. What is the best way to provide passwords to user accounts of RDBMS in Sqoop import/export commands?

We can use –password argument to provide password of user account but it is not secure. The -P argument (prompts for user password) will read a password from a console prompt ,and is the preferred method of entering credentials. Credentials may still be transferred between nodes of the MapReduce cluster using insecure means. The most secure way is to use, –password-file <file containing the

password>method. Set authentication password in this file on the users home directory with 400 permissions.

16. If HDFS target directory already exists, then we how can we overwrite the RDBMS table into it via Sqoop?

We can use –delete-target-dir argument to sqoop import command, so that target directory will be deleted prior to copying the new table contents into that directory.

17. Can we provide free-form SQL queries in Sqoop Import commands ? If yes, what is the criteria to use them?

Yes, We can use free-form SQL queries with the help of -e, –query arguments to Sqoop Import command, but we must specify the –target-dirargument along with this.

18. Can we append the tables data to an existing target directory?

If the destination directory already exists in HDFS, Sqoop will refuse to import. If we use the –append argument, Sqoop will import data to a temporary directory and then rename the files into normal target directory in a manner that, it does not conflict with existing file names in that directory.

19. When RDBMS table is only getting new rows and the existing rows are not changed, then how can we pull only the new rows into HDFS via sqoop?

We can use –incremental append argument to pull only the new rows from RDBMS table into HDFS directory.

20. When existing rows are also being updated in addition to new rows then how can we bring only the updated records into HDFS ?

In this case we need to use –incremental lastmodified argument with two additional mandatory arguments –check-column <col> and –last-value (value).

21. What is the default file format that is used to store HDFS files or Hive tables when and RDBMS file is imported via Sqoop?

Default file type is text file format. It is same as specifying –as-textfile clause to sqoop import command.

22. Can we save RDBMS table into Hive table with Avro file format and compression enabled? If yes how can we do that ?

Yes, we can use –as-avrodatafile clause to create the target file as avro file and -z,–compress clause as argument, optionally we can specify –compression-codec <c> to provide the compression codec class name.

23. What is the difference between –target-dir and –warehouse-dir arguments while importing and can we use both in the same import command?

target-dir is used to create the directory with the name provided in HDFS but warehouse-dir additionally creates two more sub level directories as HDFS path +/warehouse/dir/(tablename)/+. No we cannot use these two clauses in the same command. These are mutually exclusive.

24. Can we write any default value for all the string NULLs in an RDBMS table while importing into Hive table?

Yes, we can provide our own meaningful default value to all NULL strings in source RDBMS table while loading into Hive Table with –null-string argument. Below is the sample sqoop import command

```
sqoop import \
--connect jdbc:mysql://mysql.server.com/sqoop \
--table emp \
--null-string '\\N' \
--null-non-string '\\N'
```

25. Can we load RDBMS tables data into an Hive Partition directly ? If Yes how can we achieve it?

Sqoop can import data for Hive into a particular partition by specifying the –hive-partition-keyand –hive-partition-value arguments.

26. Can we import RDBMS tables into HCatalog directly, If yes, what are the limitations in it?

Yes by using –hcatalog-database option with –hcatalog-table we can create Hcatalog tables directly but importing/exporting into/from HCatalog has limitations as of now and doesn’t below arguments.

- –direct
- –export-dir
- –target-dir
- –warehouse-dir
- –append
- –as-sequencefile
- –as-avrofile

27. Does Sqoop import direct fast importing technique while importing into Hive/HBase/Hcatalog?

No, fast direct import is supported only for HDFS and that too from MySQL and PostGreSQL only.

28. Can we bulk load HBase tables with RDBMS tables via Sqoop?

use of –hbase-bulkload argument enable bulk loading into HBase table while importing from RDBMS table via Sqoop.

29. In destination HBase table how will the row keys be maintained uniquely?

HBase tables maintain row keys uniquely and it is the primary key of input RDBMS table if it is present, otherwise it will be split-by column from input import command.

30. What are the various table creation arguments available in Sqoop?

Sqoop supports direct table creations in Hive, HBase and HCatalog as of now, and below are the corresponding create table arguments respectively.

- –create-hive-table argument used with –hive-import
- –hbase-create-table along with –column-family <family> optionally –hbase-row-key <col>
- –create-hcatalog-table

31. What is the role of JDBC driver in a Sqoop set up?

To connect to different relational databases sqoop needs a connector. Almost every DB vendor makes this connecter available as a JDBC driver which is specific to that DB. So Sqoop needs the JDBC driver of each of the database it needs to inetract with.

32. Is JDBC driver enough to connect sqoop to the databases?

No. Sqoop needs both JDBC and connector to connect to a database.

33. When to use --target-dir and when to use --warehouse-dir while importing data?

To specify a particular directory in HDFS use --target-dir but to specify the parent directory of all the sqoop jobs use --warehouse-dir. In this case under the parent directory sqoop will cerate a directory with the same name as th e table.

34. How can you import only a subset of rows form a table?

By using the WHERE clause in the sqoop import statement we can import only a subset of rows.

35. How can we import a subset of rows from a table without using the where clause?

We can run a filtering query on the database and save the result to a temporary table in database.

Then use the sqoop import command without using the --where clause

36. What is the advantage of using --password-file rather than -P option while preventing the display of password in the sqoop import statement?

The --password-file option can be used inside a sqoop script while the -P option reads from standard input , preventing automation.

37.What is the default extension of the files produced from a sqoop import using the --compress parameter?

.gz

38. What is the significance of using --compress-codec parameter?

To get the out file of a sqoop import in formats other than .gz like .bz2 we use the --compress -code parameter.

39. What is a disadvantage of using --direct parameter for faster data load by sqoop?

The native utilities used by databases to support faster laod do not work for binary data formats like SequenceFile

40. How can you control the number of mappers used by the sqoop command?

The Parameter --num-mapers is used to control the number of mappers executed by a sqoop command. We should start with choosing a small number of map tasks and then gradually scale up as choosing high number of mappers initially may slow down the performance on the database side.

41. How can you avoid importing tables one-by-one when importing a large number of tables from a database?

Using the command

```
sqoop import-all-tables
--connect
--usrename
--password
--exclude-tables table1,table2 ..
```

This will import all the tables except the ones mentioned in the exclude-tables clause.

42. When the source data keeps getting updated frequently, what is the approach to keep it in sync with the data in HDFS imported by sqoop?

sqoop can have 2 approaches.

a − To use the --incremental parameter with append option where value of some columns are checked and only in case of modified values the row is imported as a new row.

b − To use the --incremental parameter with lastmodified option where a date column in the source is checked for records which have been updated after the last import.

43. What is the usefulness of the options file in sqoop.

The options file is used in sqoop to specify the command line values in a file and use it in the sqoop commands.

For example the --connect parameter's value and --user name value scan be stored in a file and used again and again with different sqoop commands.

44. Is it possible to add a parameter while running a saved job?

Yes, we can add an argument to a saved job at runtime by using the --exec option

sqoop job --exec jobname -- -- newparameter

45. How do you fetch data which is the result of join between two tables?

By using the --query parameter in place of --table parameter we can specify a sql query. The result of the query will be imported

46. How can we slice the data to be imported to multiple parallel tasks?

Using the --split-by parameter we specify the column name based on which sqoop will divide the data to be imported into multiple chunks to be run in parallel.

47. How can you choose a name for the mapreduce job which is created on submitting a free-form query import?

By using the --mapreduce-job-name parameter. Below is a example of the command.

```
sqoop import \
--connect jdbc:mysql://mysql.example.com/sqoop \
--username sqoop \
--password sqoop \
--query 'SELECT normcities.id, \
countries.country, \
normcities.city \
FROM normcities \
JOIN countries USING(country_id) \
WHERE $CONDITIONS' \
--split-by id \
--target-dir cities \
--mapreduce-job-name normcities
```

48. Before starting the data transfer using mapreduce job, sqoop takes a long time to retrieve the minimum and maximum values of columns mentioned in –split-by parameter. How can we make it efficient?

We can use the --boundary –query parameter in which we specify the min and max value for the column based on which the split can happen into multiple mapreduce tasks. This makes it faster as the query inside the –boundary-query parameter is executed first and the job is ready with the information on how many mapreduce tasks to create before executing the main query.

49. What is the difference between the parameters sqoop.export.records.per.statement and sqoop.export.statements.per.transaction ?

The parameter "sqoop.export.records.per.statement" specifies the number of records that will be used in each insert statement.

But the parameter "sqoop.export.statements.per.transaction" specifies how many insert statements can be processed parallel during a transaction.

50. How will you implement all-or-nothing load using sqoop?

Using the staging-table option we first load the data into a staging table and then load it to the final target table only if the staging load is successful.

51. How do you clear the data in a staging table before loading it by Sqoop?

By specifying the –clear-staging-table option we can clear the staging table before it is loaded. This can be done again and again till we get proper data in staging.

52. How will you update the rows that are already exported?

The parameter --update-key can be used to update existing rows. In it a comma-separated list of columns is used which uniquely identifies a row. All of these columns is used in the WHERE clause of the generated UPDATE query. All other table columns will be used in the SET part of the query.

53. How can you sync a exported table with HDFS data in which some rows are deleted?

Truncate the target table and load it again.

54. How can you export only a subset of columns to a relational table using sqoop?

By using the –column parameter in which we mention the required column names as a comma separated list of values.

55. How can we load to a column in a relational table which is not null but the incoming value from HDFS has a null value?

By using the –input-null-string parameter we can specify a default value and that will allow the row to be inserted into the target table.

56. How can you schedule a sqoop job using Oozie?

Oozie has in-built sqoop actions inside which we can mention the sqoop commands to be executed.

57. Sqoop imported a table successfully to HBase but it is found that the number of rows is fewer than expected. What can be the cause?

Some of the imported records might have null values in all the columns. As Hbase does not allow all null values in a row, those rows get dropped.

58. Give a sqoop command to show all the databases in a MySql server.

```
$ sqoop list-databases --connect jdbc:mysql://database.example.com/
```

59. What do you mean by Free Form Import in Sqoop?

Sqoop can import data form a relational database using any SQL query rather than only using table and column name parameters.

60. How can you force sqoop to execute a free form Sql query only once and import the rows serially.

By using the –m 1 clause in the import command, sqoop cerates only one mapreduce task which will import the rows sequentially.

61. In a sqoop import command you have mentioned to run 8 parallel Mapreduce task but sqoop runs only 4. What can be the reason?

The Mapreduce cluster is configured to run 4 parallel tasks. So the sqoop command must have number of parallel tasks less or equal to that of the MapReduce cluster.

62. What is the importance of --split-by clause in running parallel import tasks in sqoop?

The –split-by clause mentions the column name based on whose value the data will be divided into groups of records. These group of records will be read in parallel by the mapreduce tasks.

63. What does this sqoop command achieve?

```
$ sqoop import --connnect <connect-str> --table foo --target-dir /dest \
```

It imports data from a database to a HDFS file named foo located in the directory /dest

64. What happens when a table is imported into a HDFS directory which already exists using the –apend parameter?

Using the --append argument, Sqoop will import data to a temporary directory and then rename the files into the normal target directory in a manner that does not conflict with existing filenames in that directory.

65. How can you control the mapping between SQL data types and Java types?

By using the --map-column-java property we can configure the mapping between.

Below is an example

$ sqoop import ... --map-column-java id = String, value = Integer

66. How to import only the updated rows form a table into HDFS using sqoop assuming the source has last update timestamp details for each row?

By using the lastmodified mode. Rows where the check column holds a timestamp more recent than the timestamp specified with --last-value are imported.

67. What are the two file formats supported by sqoop for import?

Delimited text and Sequence Files.

68.Give a sqoop command to import the columns

employee_id,first_name,last_name from the MySql table Employee

```
$ sqoop import --connect jdbc:mysql://host/dbname --table
EMPLOYEES \
   --columns "employee_id,first_name,last_name"
```

69. Give a sqoop command to run only 8 mapreduce tasks in parallel

```
$ sqoop import --connect jdbc:mysql://host/dbname --table
table_name\
    -m 8
```

70. What does the following query do?
**$ sqoop import --connect jdbc:mysql://host/dbname --table EMPLOYEES **
--where "start_date > '2012-11-09'

It imports the employees who have joined after 9-NOv-2012.

71. Give a Sqoop command to import all the records from employee table divided into groups of records by the values in the column department_id.

```
$ sqoop import --connect jdbc:mysql://db.foo.com/corp --table
EMPLOYEES \
   --split-by dept_id
```

72. What does the following query do?
**$ sqoop import --connect jdbc:mysql://db.foo.com/somedb --table sometable **
--where "id > 1000" --target-dir /incremental_dataset –append

It performs an incremental import of new data, after having already imported the first 100,0rows of a table

73. Give a sqoop command to import data from all tables in the MySql DB DB1.

```
sqoop import-all-tables --connect jdbc:mysql://host/DB1
```

74. Give a command to execute a stored procedure named proc1 which exports data to from MySQL db named DB1 into a HDFS directory named Dir1.

```
$ sqoop export --connect jdbc:mysql://host/DB1 --call proc1 \
    --export-dir /Dir1
```

75. What is a sqoop metastore?

It is a tool using which Sqoop hosts a shared metadata repository. Multiple users and/or remote users can define and execute saved jobs createdwithsqoopjobcreatedwithsqoopjob defined in this metastore.

Clients must be configured to connect to the metastore in sqoop-site.xml or with the --meta-connect argument.

76. What is the purpose of sqoop-merge?

The merge tool combines two datasets where entries in one dataset should overwrite entries of an older dataset preserving only the newest version of the records between both the data sets.

77. How can you see the list of stored jobs in sqoop metastore?

sqoop job –list

78. Give the sqoop command to see the content of the job named myjob?

Sqoop job –show myjob

79. Which database the sqoop metastore runs on?

Running sqoop-metastore launches a shared HSQLDB database instance on the current machine.

80. Where can the metastore database be hosted?

The metastore database can be hosted anywhere within or outside of the Hadoop cluster.