Hadoop

1. **What are the differences between data and information?**

    - **Data** is a collection of facts. Data is unorganized.
    - **Information** is how you understand those facts in context. While information is structured or organized.

2. **Explain big data and what are 5 v's of big data?**

    Big data: is the term for a collection of large and complex data sets, that makes it difficult to process using relational database management tools or traditional data processing applications.

    5v's of big data:

    I. **Volume:** the volume represents the amount of data i.e., in petabytes and exabytes.

    II. **Velocity:** velocity refers to the rate at which data is growing, which is growing at an exponential rate, which is very fast. Today, yesterday's data are considered as old data. Nowadays, social media is a major contributor to the velocity of growing data.

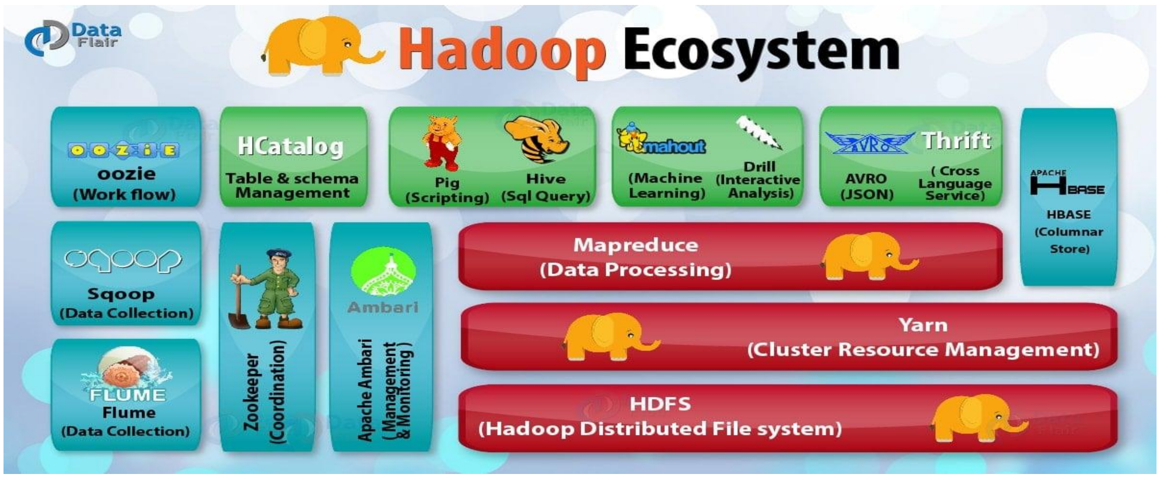
    III. **Variety:** the data which are gathered has a variety of formats like videos, audios, csv, etc. So, these various formats represent the variety of data.

    IV. **Veracity:** veracity refers to the data in doubt or uncertainty of data available due to data inconsistency and incompleteness. Data available can sometimes get messy and may be difficult to trust. With many forms of big data, quality and accuracy are difficult to control. The volume is often the reason behind the lack of quality and accuracy in the data.

    V. **Value:** it is all well and good to have access to big data but unless we can turn it into a value it is useless. By turning it into value I mean, is it adding to the benefits of the organizations? Is the organization working on big data achieving high Roi (return on investment)? Unless it adds to their profits by working on big data, it is useless.

3. **Hadoop ecosystem and components?**
    The Hadoop eco-system consists of 4 core components. Those are.

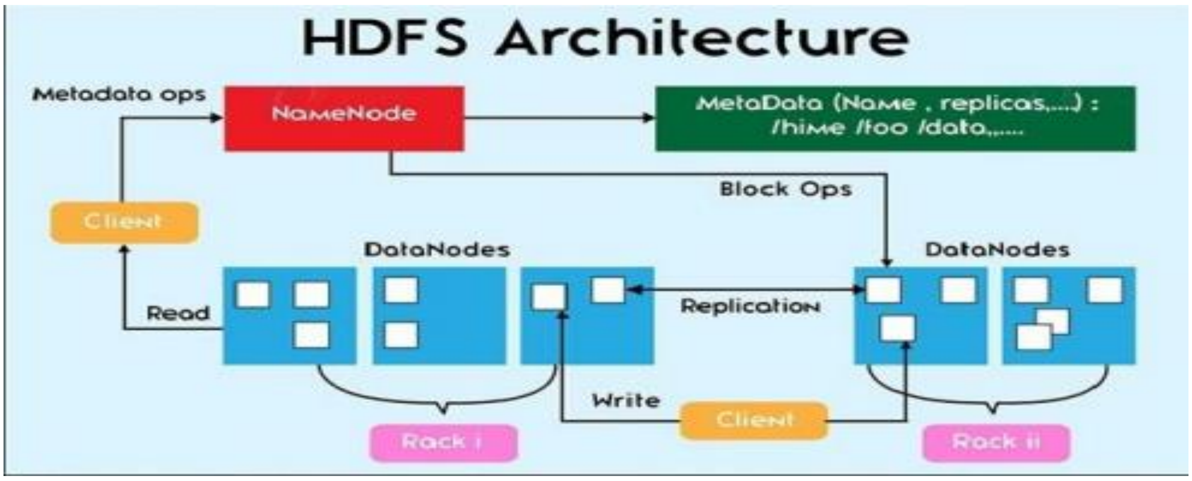
    1) **hdfs:** data storage
    2) **map-reduce** processing the data reside in hdfs.
    3) **yarn:** cluster resource management.
    4) **hadoop components**:
        - **Data access components** - pig and hive
        - **Data integration components** - flume and sqoop
        - **Data storage component**- hbase
        - **Monitoring and management components** - oozie and zookeeper


Data Flair
Hadoop Ecosystem
oozie (Work flow)
HCatalog Table & schema Management
Pig (Scripting)
Hive (Sql Query)
mahout (Machine Learning)
Drill (Interactive Analysis)
AVRO (JSON)
Thrift (Cross Language Service)
HBASE (Columnar Store)
Sqoop (Data Collection)
Flume (Data Collection)
Zookeeper (Coordination)
Apache Ambari (Management & Monitoring)
Mapreduce (Data Processing)
Yarn (Cluster Resource Management)
HDFS (Hadoop Distributed File system)

4. **Explain the storage unit (hdfs) in hadoop?**

Hdfs is the hadoop distributed file system, is the storage layer for hadoop. The files in hdfs are split into blocks. By default, the size of the block is 128 mb and by default replication factor is 3. It follows the master-slave architecture.

It is the most important component of hadoop ecosystem. Hdfs is the primary storage system of hadoop. Hadoop distributed file system (hdfs) is a java-based file system that provides scalable, fault tolerance, reliable and cost-efficient data storage for bigdata. Hdfs is a distributed file system that runs on commodity hardware.



Hdfs components:

There are two major components of hadoop hdfs- namenode and datanode. Let's now discuss these hadoop hdfs components.

I. **Namenode**

- It is also known as master node. Namenode does not store actual data or dataset. Namenode stores **metadata i.e. files names, number of blocks, their location, on which rack, which datanode the data is stored,permissions etc.**

- It manages the data nodes.

Tasks of hdfs namenode

- Manage file system namespace.
- Regulates client's access to files.
- Executes file system execution such as naming, closing, opening files and directories.

II. **Datanode**

It is also known as slave. Hdfs datanode is responsible for storing actual data in hdfs. Datanode performs read and write operation as per the request of the clients.
The replica block of datanode consists of 2 files on the file system. The first file is for data and the second file is for recording the block's metadata.

Tasks of hdfs datanode

- Datanode performs operations like block replica creation, deletion, and replication according to the instruction of namenode.
- Datanode manages data storage of the system.

III. **Gateway node:**
Also called edge node.it connects hadoop cluster, the purpose of edge node is to provide an access point to cluster and prevent users/clients direct access of name node and data node.

5. **What is spof in hadoop?**

Before hadoop 2.0.0, the namenode was a single point of failure (spof) in an hadoop cluster.

Each cluster had a single namenode, and if namenode fails, the cluster would be out of service. The cluster will be unavailable until the namenode restarts or brought on a separate machine.

6. **Explain Hadoop hdfs namenode high availability?**

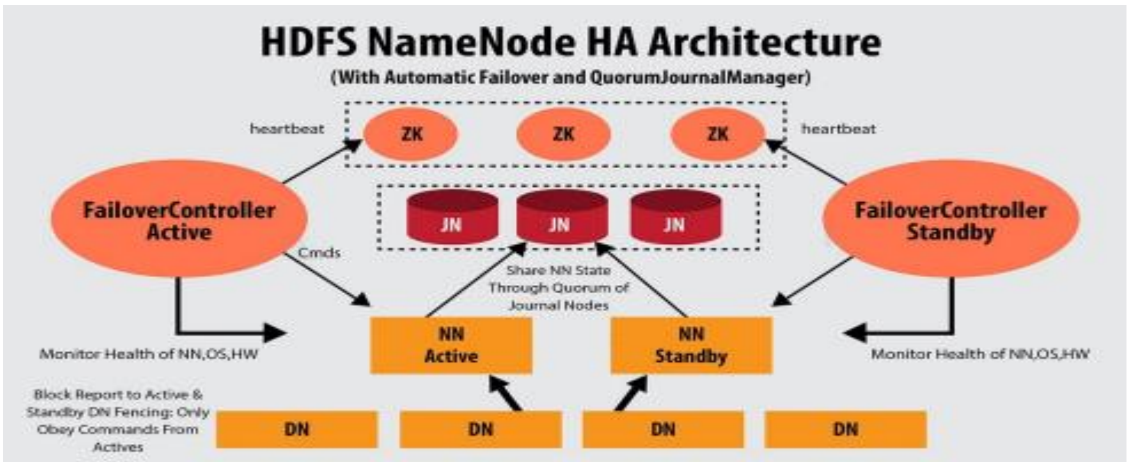
In hadoop 2.0 hdfs namenode high availability architecture, two namenode's runs in same cluster at the same time with a hot standby to avoid spof.

- **Active namenode** – it handles all client operations in the cluster.

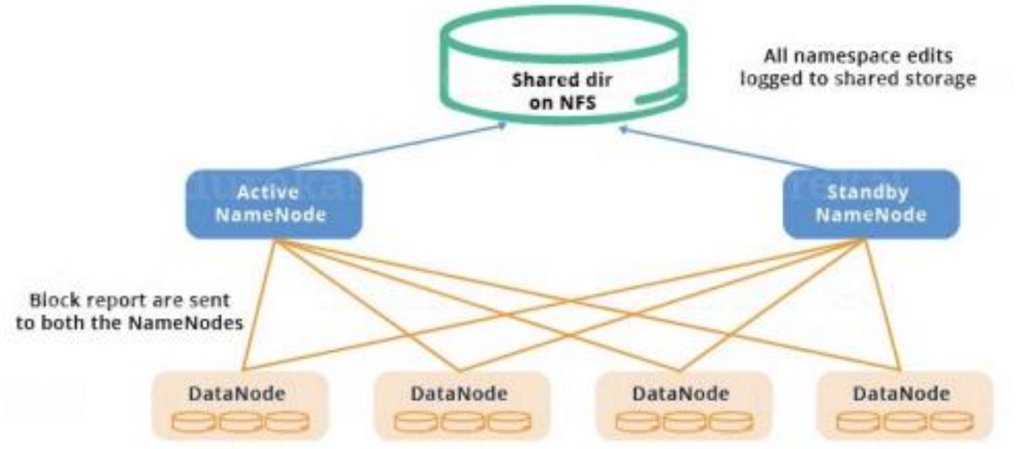
- **Passive/stand by namenode** –it is a standby namenode, which has similar data as active namenode. It acts as a slave.if the active namenode fails, then passive namenode takes all the responsibility of active node and cluster continues to work.

We can implement hdfs high availability architecture with the active and standby namenode configuration in following two ways:

• **Using quorum journal nodes:** for high availability, standby namenode communicates and synchronizes with the active namenode. It happens through a group of nodes or daemons called “journal nodes”. There should be at least three journal nodes.


HDFS NameNode HA Architecture
(With Automatic Failover and QuorumJournalManager)
heartbeat
ZK
ZK
ZK
heartbeat
FailoverController Active
JN
JN
JN
FailoverController Standby
Cmds
Share NN State Through Quorum of Journal Nodes
NN Active
NN Standby
Monitor Health of NN,OS,HW
Monitor Health of NN,OS,HW
Block Report to Active & Standby DN Fencing: Only Obey Commands From Actives
DN
DN
DN
DN

• **Using shared storage**
Standby and active namenode synchronize with each other by using “shared storage device”. For this implementation, both active namenode and standby namenode must have access to the particular directory on the shared storage device (. i.e. network file system).


Shared dir on NFS
All namespace edits logged to shared storage
Active NameNode
Standby NameNode
Block report are sent to both the NameNodes
DataNode
DataNode
DataNode
DataNode

7. What is rack awareness in hadoop hdfs?

In a large cluster of hadoop, in order to improve the network traffic while reading/writing in hdfs,namenode maintains the rack id’s of each datanode. Namenode chooses the datanode which is closer to the same rack or nearby rack to read/write request.this concept is called rack awareness in hadoop.

8. Disk balancer?

- Disk balancer is a command-line tool introduced in hadoop 3 for intra-datanode balancing.
- Diskbalancer distributes data within the datanode.
- Hdfs disk balancer operates against a given datanode and moves blocks from one disk to another.
- By default, Disk Balancer is not enabled on a Hadoop cluster. One can enable the Disk Balancer in Hadoop by setting dfs.disk.balancer.enabled true in hdfs-site.xml

9. Explain MapReduce?

MapReduce is the processing layer of Hadoop.

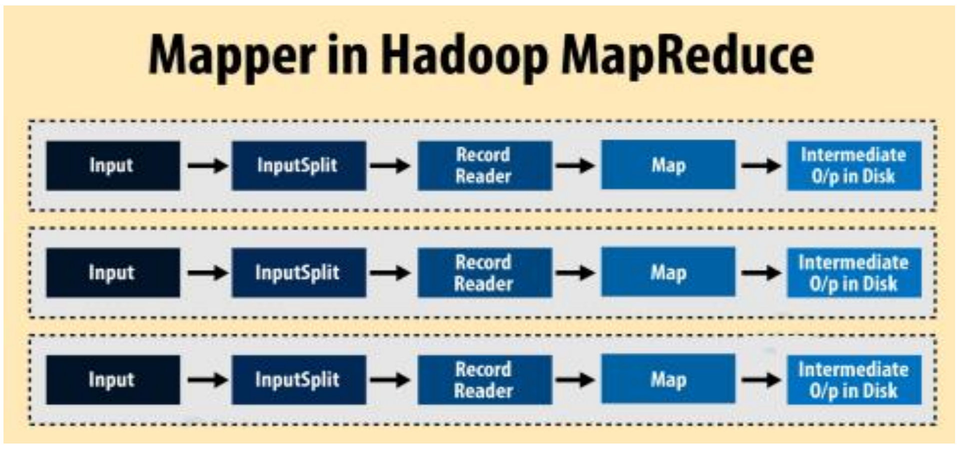
MapReduce is a software framework that processes the vast amount of structured and unstructured data stored in the Hadoop distributed file system.

Working of MapReduce

MapReduce works by breaking the processing into two phases:

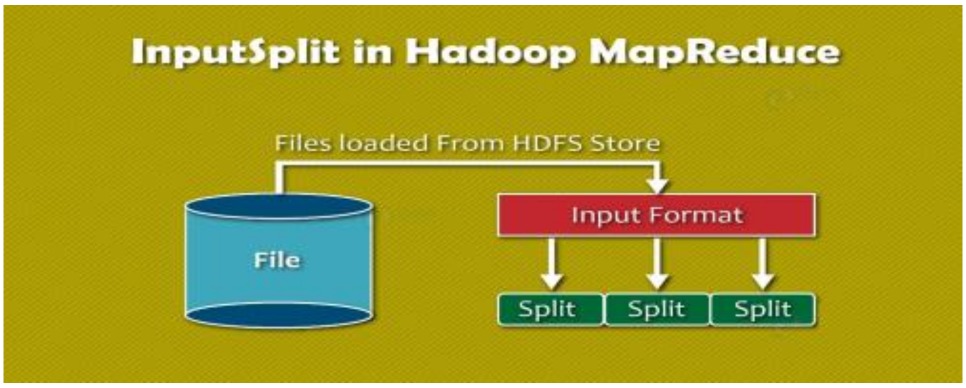
1. Mapper
2. reducer

1. **Mapper**:
   - Mapper task processes each input record from Record Reader and generates an intermediate key-value pair.
   - The key-value pairs can be completely different from the input pair.
   - Hadoop Mapper stores intermediate output on the local disk.
   - MapReduce frame generates one map task for each Input Split.



Mapper Tasks:

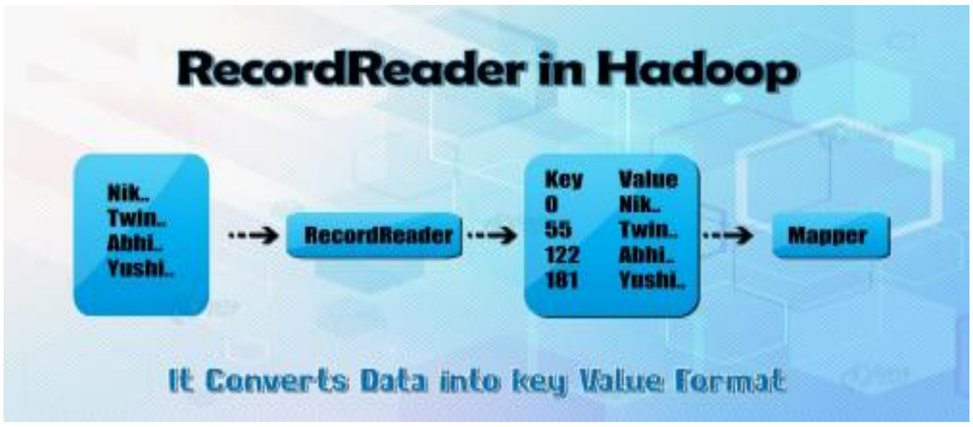
- **Input Files:** The data for a MapReduce task is stored in input files, and input files typically lives HDFS.



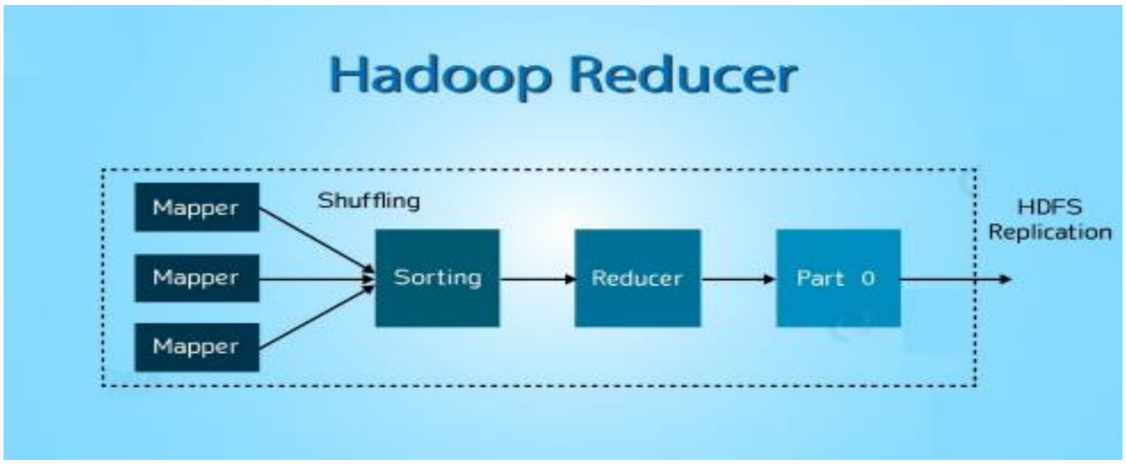
- **Input Format :**Now, InputFormat defines how these input files are split and read. It selects the files or other objects that are used for input. Input Format creates InputSplit.

- **InputSplit –** Amount of data that should go to one mapper at a time.

- **RecordReader –** It communicates with the InputSplit and it converts the data into key-value pairs suitable for reading by the Mapper.By default, it uses TextInputFormat for converting data into the key-value pair. RecordReader communicates with the InputSplit until the file reading is not completed.



2. **Reducer:**
   - Reducer takes the output of the Mapper (intermediate key-value pair) process each of them to generate the output.
   - The output of the reducer is the final output, which is stored in HDFS. Usually, in the Hadoop Reducer, we do aggregation or summation sort of computation.
   - The user decides the number of reducers. By default, the number of reducers is 1.



Phases of Reducer:

- **Shuffle Phase:** Shuffling helps to carry data from the Mapper to the required Reducer. With the help of HTTP, the framework calls for an applicable partition of the output in all Mappers.

- **Sort Phase: In** this phase, the output of the mapper that is actually the key-value pairs will be sorted on the basis of its key value.

- **Reduce Phase:** Once shuffling and sorting is done the Reducer combines the obtained result and performs the computation operation as per the requirement. OutputCollector.collect() property is used for writing the output to the HDFS. Keep remembering that the output of the Reducer will not be sorted.

- **RecordWriter :** collects the output key-value pairs from the Reducer and writes it into the output file.

- **OutputFormat:** Hadoop RecordWriter takes output data from Reducer and writes this data to output files which is determined by the OutputFormat.

10. Types of InputFormat in MapReduce?

InputFormat defines how to split and read input files. InputFormat defines RecordReader. One can also use different types of InputFormats for different purposes.

- FileInputFormat
- TextInputFormat
- KeyValueTextInputFormat
- SequenceFileInputFormat
- SequenceFileAsTextInputFormat
- SequenceFileAsBinaryInputFormat
- NLineInputFormat
- DBInputFormat

11. Types of OutputFormat in MapReduce

- TextOutputFormat

- SequenceFileOutputFormat
- SequenceFileAsBinaryOutputFormat
- MapFileOutputFormat
- MultipleOutputs
- LazyOutputFormat
- DBOutputFormat

12. Difference between split size and block size?

Block size:

In hadoop, the files split into 128 mb blocks and then stored into hadoop file system. The default size of the hdfs block is 128 mb which is configured as per our requirement. All blocks of the file are of the same size except the last block. The last block can be of the same size or smaller.

Split size:

Amount of data that should go to one mapper at a time. Mostly block size and split configure to be same to ensure less data movement across the cluster.

13. Ways to reduce traffic between mapper and reducer?

- By using combiner
- By compression of mapper output
- By writing optimized query to filter out unnecessary data in mapper

14. What is combiner in map reduce?

A combiner, also known as a **semi-reducer,** is an optional class that operates by accepting the inputs from the map class and thereafter passing the output key-value pairs to the reducer class.
The combiner class is used in between the map class and the reduce class to reduce the volume of data transfer between map and reduce. Usually, the output of the map task is large and the data transferred to the reduce task is high.

15. In hadoop mv is faster than cp why?

We can also copy files with in HDFS using commands like cp and mv. mv is faster than cp as mv deals with only metadata where as cp have to copy all the blocks.

16. What is mapper and reducer?

Mapper: multiple mappers work independently on input data, and they produce intermediate results.

Reducer: then the reducer takes all intermediate results from all mapper and aggregates. It produces final output.

17. How do we get the data to mapper?

We have 2 methods to get the data to mapper in MapReduce:

- getsplits()
- createRecordReader()

18. Select * from table_name will trigger how many mappers and reducers?

Zero, this query won't be converted into mapreduce job.a hive query is converted into map reduce job only in case computation is involved in the query.

Select * from table_name; is executed directly with the file system apis.

19. How many mappers for 1 gb file?

Mappers depend on the number of input blocks. Assuming table size is 1024 gb and block size is 128 mb, the number of input blocks is 8. Hence it will launch 8 mappers**.**

20. How do you decide the no.of mappers and reducers in hadoop cluster?

Mapper:

No.of mappers decided on no.of cores and ram

Ex:

- 100 gb of raw data need to be processed. I want to get 10 output files. Data node config is 64 gb ram ,6 cores. How do process?
- I need two data nodes with config 64 gb ram and 6 cores for each node.
- Then one mapper takes 10gb ram and 1 core to process to 10gb of raw data.
- Here 10 mappers uses 100gb ram ,10 cores to process 100gb of raw data and produces 10 ouput files.

Reducer:

No.of reducers decided on per job basis.

Ex:

- If we want to sort data using sort by, we need 10 reducers.
- If we want to sort data using order by, we need 1 reducer.

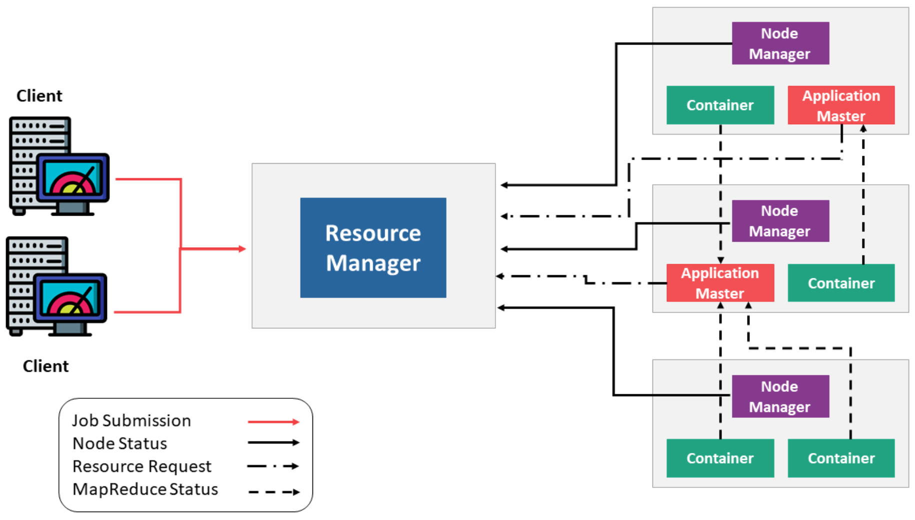
21. What are the different components of yarn architecture?

- Yarn stands for yet another resource negotiator.

- It is the resource management layer of hadoop.
- The yarn was launched in hadoop 2.x.
- Yarn also offers job scheduling. It extends the capability of hadoop to other evolving technologies so that they can take good advantage of hdfs and economic clusters.

Apache Hadoop yarn architecture consists of the following main components:

- **Resource manager:** runs on a master node and manages the resource allocation in the cluster.
- **Node manager:** they run on the slave nodes and are responsible for the execution of a task on every single data node.
- **Application master:** manages the user job lifecycle and resource needs of individual applications. It works along with the node manager and monitors the execution of tasks.
- **Container:** package of resources including ram, cpu, network, hdd etc on a single node.



22. Hcatalog?

It is a table and storage management layer for hadoop. Hcatalog supports different components available in hadoop like mapreduce, hive, and pig to easily read and write data from the cluster. Hcatalog is a key component of hive that enables the user to store their data in any format and structure.

By default, hcatalog supports rcfile, csv, json, sequencefile and orc file formats.

Benefits of hcatalog:

• enables notifications of data availability.

• with the table abstraction, hcatalog frees the user from overhead of data storage.

• provide visibility for data cleaning and archiving tools.

23. What are the basic differences between relational database and hdfs?

Here are the key differences between hdfs and relational database:

	Rdbms	**Hadoop**
Data types	Rdbms relies on the structured data and the schema of the data is always known.	Any kind of data can be stored into hadoop i.e. be it structured, unstructured or semi-structured.
Processing	Rdbms provides limited or no processing capabilities.	Hadoop allows us to process the data which is distributed across the cluster in a parallel fashion.
Schema on read vs. Write	Rdbms is based on ‘schema on write’ where schema validation is done before loading the data.	On the contrary, hadoop follows the schema on read policy.
Read/write speed	In rdbms, reads are fast because the schema of the data is already known.	The writes are fast in hdfs because no schema validation happens during hdfs write.
Cost	Licensed software, therefore, i have to pay for the software.	Hadoop is an open source framework. So, i don’t need to pay for the software.
Best fit use case	Rdbms is used for oltp (online trasanctional processing) system.	Hadoop is used for data discovery, data analytics or olap system.

24. How to solve small file problem in hadoop ?

We group the files in a larger file and for that, we can use hdfs’s sncy() or write a program or we can use methods:

1) har files: it builds a layer system on top of hdfs. Har command will create a har file which then runs a mapreduce job to avoid the files being archived into the small number of hdfs files.

2) hbase: it is a type of storage which creates larger files depending on the access pattern of small files.

3) sequence file: here we use the filename as key and content as value. We write a program to put lots of small files in a single sequence file. They are splittable and so mapreduce can operate each chunk independently and in parallel.

4) filecrush tool: it will turn many small files into fewer larger files. It also changes from text to sequence.

25. What are the challenges that you encountered in your project?

- Data processing time was high when we're using mapreduce execution engine.
- Faced few challenges when we were converting into rc file format into orc file format in hive.
- Faced high latency situation when we were executing hive queries in a small 13 node hadoop cluster.
- We got several times 'memory out of bound exception' when we're using spark as an execution engine. Then we had set memory and disk only serialization instead of memory only serialization.

26. Source database used in your pervious project?

Oracle db or mysql db or mongo db or hbase etc. Please choose db as per your project applications.

27. When you have to restart a job? Will you lose data in spark as it works in memory process?

No data will be lost in memory. But, it override the results when we restart the job.

Apache spark is the capability to operate and to recover loss after a failure occurs. If due to a worker node failure any partition of an rdd is lost, then that partition can be re-computed from the original fault-tolerant dataset using the lineage of operations.

28. Examples on challenges faced in your project and how it was solved?

- Data processing time was high when we're using mapreduce execution engine.
- Faced few challenges when we were converting into rc file format into orc file format in hive.
- Faced high latency situation when we were executing hive queries in a small 13 node hadoop cluster.
- We got several times 'memory out of bound exception' when we're using spark as an execution engine.then we had set memory and disk only serialization instead of memory only serialization.

29. Workflows used in oozie?

Yes. We had created workflows and coordinators to schedule the hive, shell scripts and spark jobs.

30. How you are pulling the data from source?

Structured data by using sqoop & semi structured data by using flume

31. Significance of sqoop query.

By using sqoop we can import data from rdbms to hadoop and export from hadoop to rdbms.we can query data from rdbms directly by using sqoop.

32. How many mappers we can in sqoop maximum?

Sqoop jobs use 4 map tasks by default. It can be modified by passing either -m or --num-mappers argument to the job. There is no maximum limit on number of mappers set by sqoop. Howe ever, we can limit number mappers based on the application.

33. **What is mapsidejoin?**
    Map join is a hive feature that is used to speed up hive queries. It lets a table to be loaded into memory so that a join could be performed within a mapper without using a reduce step.
34. **How can you specify mappers in sqoop?**
    -m,--num-mappers <n>   use n map tasks to import in parallel
35. **How mapreduce works in hive?**
    Hive queries are converted to mapreduce programs in the background by the hive compiler for the jobs to be executed parallel across the hadoop cluster. This helps hadoop developers to focus more on the business problem rather than having to focus on complex programming language logic.
36. **How to call pig or hive scripts from shell?**
    To execute a pig script from the command line we just need to use pig -f somefile.pig
    To execute a pig script from the command line we just need to use hive -f my_script.hql

37. **What are the limitations of hadoop 1.0 ?**
    - Only one namenode is possible to configure.
    - Secondary namenode was to take hourly backup of metadata from namenode. It is only suitable for batch processing of a vast amount of data, which is already in the hadoop system.
    - It is not ideal for real-time data processing.
    - It supports up to 4000 nodes per cluster.
    - It has a single component: jobtracker to perform many activities like resource management, job scheduling, job monitoring, re-scheduling jobs etc. Jobtracker is the single point of failure.
    - It supports only one name no and one namespace per cluster.
    - It does not help the horizontal scalability of namenode.
    - It runs only map/reduce jobs.
38. **Let us take an example of a text file called example_data.txt and understand how mapreduce works.**
    - The content of the example_data.txt file is:
    - Coding,jamming,ice,river,man,driving
    - Now, assume we have to find out the word count on the example_data.txt using mapreduce. So, we will be looking for the unique words and the number of times those unique words appeared.
    - First, we break the input into three divisions, as seen in the figure. This will share the work among all the map nodes.

- Then, all the words are tokenized in each of the mappers, and a hardcoded value (1) to each of the tokens is given. The reason behind giving a hardcoded value equal to 1 is that every word by itself will, at least, occur once.
- Now, a list of key-value pairs will be created where the key is nothing but the individual words and value is one. So, for the first line (coding ice jamming), we have three key-value pairs – coding, 1; ice, 1; jamming, 1.
- The mapping process persists the same on all the nodes.
- Next, a partition process occurs where sorting and shuffling follow so that all the tuples with the same key are sent to the identical reducer.
- Subsequent to the sorting and shuffling phase, every reducer will have a unique key and a list of values matching that very key. For example, coding, [1,1]; ice, [1,1,1].., etc.
- Now, each reducer adds the values which are present in that list of values. As shown in the example, the reducer gets a list of values [1,1] for the key jamming. Then, it adds the number of ones in the same list and gives the final output as – jamming, 2.
- Lastly, all the output key/value pairs are then assembled and written in the output file.

39. What is shuffling in mapreduce?

In hadoop mapreduce, shuffling is used to transfer data from the mappers to the important reducers. It is the process in which the system sorts the unstructured data and transfers the output of the map as an input to the reducer. It is a significant process for reducers. Otherwise, they would not accept any information. Moreover, since this process can begin even before the map phase is completed, it helps to save time and complete the process in a lesser amount of time.

40. What are the three modes that hadoop can run?

Local mode or standalone mode

Hadoop, by default, is configured to run in a no distributed mode. It runs as a single java process. Instead of hdfs, this mode utilizes the local file system. This mode is more helpful for debugging, and there isn't any requirement to configure core-site.xml, hdfs-site.xml, mapred-site.xml, masters & slaves. Stand alone mode is ordinarily the quickest mode in hadoop.

Pseudo-distributed model

In this mode, each daemon runs on a separate java process. This mode requires custom configuration (core-site.xml, hdfs-site.xml, mapred-site.xml). The hdfs is used for input and output. This mode of deployment is beneficial for testing and debugging purposes.

Fully distributed mode

It is the production mode of hadoop. Basically, one machine in the cluster is designated as namenode and another as resource manager exclusively. These are masters. Rest nodes act as data node and node manager. These are the slaves. Configuration parameters and environment need to be defined for hadoop daemons. This mode gives fully distributed computing capacity, security, fault endurance, and scalability.

41. **What is apache zookeeper?**

Apache zookeeper is an open-source service that supports controlling a huge set of hosts. Management and coordination in a distributed environment are complex. Zookeeper automates this process and enables developers to concentrate on building soware features rather than bother about its distributed nature.

Zookeeper helps to maintain configuration knowledge, naming, group services for distributed applications. It implements various protocols on the cluster so that the application should not execute them on its own. It provides a single coherent view of many machines.

42. **What are the benefits of using zookeeper?**

Simple distributed coordination process: the coordination process among all nodes in zookeeper is straightforward.

Synchronization: mutual exclusion and co-operation among server processes. Ordered **messages:** zookeeper tracks with a number by denoting its order with the stamping of each update; with the help of all this, messages are ordered here.

Serialization: encode the data according to specific rules. Ensure your application runs consistently.

Reliability: the zookeeper is very reliable. In case of an update, it keeps all the data until forwarded.

Atomicity: data transfer either succeeds or fails, but no transaction is partial.

43. **Mention the types of znode.**

Persistent znodes:

The default znode in zookeeper is the persistent znode. It permanently stays in the zookeeper server until any other clients leave it apart.

Ephemeral znodes:

These are the temporary znodes. It is smashed whenever the creator client logs out of the zookeeper server. For example, assume client1 created eznode1. Once client1 logs out of the zookeeper server, the eznode1 gets destroyed.

Sequential znodes:

Sequential znode is assigned a 10-digit number in numerical order at the end of its name. Assume client1 produced a sznode1. In the zookeeper server, the sznode1 will be named like this:

Sznode0000000001

If the client1 generates another sequential znode, it will bear the following number in a sequence. So the subsequent sequential znode is <znode name>0000000002.

44. **Mention features of apache sqoop.**

Robust: it is highly robust. It even has community support and contribution and is easily usable.

Full load: sqoop can load the whole table just by a single sqoop command. It also allows us to load all the tables of the database by using a single sqoop command.

Incremental load: it supports incremental load functionality. Using sqoop, we can load parts of the table whenever it is updated.

Parallel import/export: it uses the yarn framework for importing and exporting the data. That provides fault tolerance on the top of parallelism. Import results of sql query: it allows us to import the output from the sql query into the hadoop distributed file system.

45. **Why are blocks in hdfs huge?**

    By default, the size of the hdfs data block is 128 mb. The ideas for the large size of blocks are:

    To reduce the expense of seek: because of the large size blocks, the time consumed to shi the data from the disk can be longer than the usual time taken to commence the block. As a result, the multiple blocks are transferred at the disk transfer rate.

    If there are small blocks, the number of blocks will be too many in hadoop hdfs and too much metadata to store. Managing such a vast number of blocks and metadata will create overhead and head to traffic in a network.

46. **How can you skip the bad records in hadoop?**

    Hadoop provides an option where a particular set of lousy input records can be skipped when processing map inputs. Applications can manage this feature through the skipbadrecords class.

    This feature can be used when map tasks fail deterministically on a particular input. This usually happens due to faults in the map function. The user would have to fix these issues.

47. **Where are the two types of metadata that namenode server stores?**

    The two types of metadata that namenode server stores are in disk and ram. Metadata is linked to two files which are:

    Editlogs: it contains all the latest changes in the file system regarding the last fsimage.

    Fsimage: it contains the whole state of the namespace of the file system from the origination of the namenode.

    Once the file is deleted from hdfs, the namenode will immediately store this in the editlog.

    All the file systems and metadata which are present in the namenode's ram are read by the secondary namenode continuously and later get recorded into the file system or hard disk. Editlogs is combined with fsimage in the namenode. Periodically, secondary namenode downloads the editlogs from the namenode, and then it is implemented to fsimage. The new fsimage is then copied back into the namenode and used only aer the namenode has started the subsequent time.

48. **Explain the purpose of the dfsadmin tool?**

    The dfsadmin tools are a specific set of tools designed to help you root out information about your hadoop distributed file system (hdfs). As a bonus, you can use them to perform some administration operations on hdfs as well.

49. **Explain the actions followed by a jobtracker in hadoop.**

    The client application is used to submit the jobs to the jobtracker. The jobtracker associates with the namenode to determine the data location. With the help of available slots and the near the data, jobtracker locates tasktracker nodes.

It submits the work on the selected tasktracker nodes.

When a task fails, jobtracker notifies and decides the further steps. Jobtracker monitors the tasktracker nodes.

50. **List the actions that happen when a datanode fail**

    Both the jobtracker and the name node detect the failure on which blocks were the datanode failed.

    On the failed node all the tasks are rescheduled by locating other datanodes with copies of these blocks

    User's data will be replicated to another node from namenode to maintain the configured replication factor.

51. **What are the basic parameters of a mapper?**

    The primary parameters of a mapper are text, longwritable, text, and intwritable. The initial two represent input parameters, and the other two signify intermediate output parameters.

52. **Mention the main configuration parameters that has to be specified by the user to run mapreduce.**

    The chief configuration parameters that the user of the mapreduce framework needs to mention is:

    Job's input location

    Job's output location

    The input format

    The output format

    The class including the map function

    The class including the reduce function

    Jar file, which includes the mapper, the reducer, and the driver classes. 35. Explain the resilient distributed datasets in spark.

    Resilient distributed datasets is the basic data structure of apache spark. It is installed in the spark core. They are immutable and fault-tolerant. Rdds are generated by transforming already present rdds or storing an outer dataset from well-built storage like hdfs or hbase.

    Since they have distributed collections of objects, they can be operated in parallel. Resilient distributed datasets are divided into parts such that they can be executed on various nodes of a cluster.

53. **How can you restart namenode and all the daemons in hadoop?**

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

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

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

54. **If the source data gets updated every now and then, how will you synchronize the data in hdfs that is imported by sqoop?**

    If the source data gets updated in a very short interval of time, the synchronization of data in hdfs that is imported by sqoop is done with the help of incremental parameters.

We should use incremental import along with the append choice even when the table is refreshed continuously with new rows. Principally where values of a few columns are examined, and if it encounters any revised value for those columns, only a new row will be inserted. Similar to incremental import, the origin has a date column examined for all the records that have been modified aer the last import, depending on the previous revised column in the beginning. The values would be modernized.

55. What is the default file format to import data using apache sqoop?

There are basically two file formats sqoop allos to import data they are:

Delimited text file format

Sequence file format

56. What is apache flume in hadoop ?

Apache flume is a tool/service/data ingestion mechanism for assembling, aggregating, and carrying huge amounts of streaming data such as record files, f i f li d d

Flume is a very stable, distributed, and configurable tool. It is generally designed to copy streaming data (log data) from various web servers to hdfs.

57. Compare hdfs with network attached storage (nas).

In this question, first explain nas and hdfs, and then compare their features as follows:

Network-attached storage (nas) is a file-level computer data storage server connected to a computer network providing data access to a heterogeneous group of clients. Nas can either be a hardware or software which provides services for storing and accessing files. Whereas hadoop distributed file system (hdfs) is a distributed filesystem to store data using commodity hardware.

In hdfs data blocks are distributed across all the machines in a cluster. Whereas in nas data is stored on a dedicated hardware.

Hdfs is designed to work with mapreduce paradigm, where computation is moved to the data. Nas is not suitable for mapreduce since data is stored separately from the computations.

Hdfs uses commodity hardware which is cost-effective, whereas a nas is a high-end storage devices which includes high cost.

58. List the difference between hadoop 1 and hadoop 2.

This is an important question and while answering this question, we have to mainly focus on two points i.e. passive namenode and yarn architecture.

In hadoop 1.x, “namenode” is the single point of failure. In hadoop 2.x, we have active and passive “namenodes”. If the active “namenode” fails, the passive “namenode” takes charge. Because of this, high availability can be achieved in hadoop 2.x.

Also, in hadoop 2.x, yarn provides a central resource manager. With yarn, you can now run multiple applications in hadoop, all sharing a common resource. Mrv2 is a particular type of distributed application that runs the mapreduce framework on top of yarn. Other tools can also perform data processing via yarn, which was a problem in hadoop 1.x.

	Hadoop 1.x	**Hadoop 2.x**
Passive namenode	Namenode is a single point of failure	Active & passive namenode

Processing	Mrv1 (job tracker & task tracker)	Mrv2/yarn (resourcemanager & nodemanager)

59. Why does one remove or add nodes in a hadoop cluster frequently?

One of the most attractive features of the hadoop framework is its *utilization of commodity hardware*. However, this leads to frequent "datanode" crashes in a hadoop cluster. Another striking feature of hadoop framework is the *ease of scale* in accordance with the rapid growth in data volume. Because of these two reasons, one of the most common task of a hadoop administrator, as learned from the hadoop admin training, is to commission (add) and decommission (remove) "data nodes" in a hadoop cluster.

60. What happens when two clients try to access the same file in the hdfs?

Hdfs supports exclusive write only.

When the first client contacts the "namenode" to open the file for writing, the "namenode" grants a lease to the client to create this file. When the second client tries to open the same file for writing, the "namenode" will notice that the lease for the file is already granted to another client, and will reject the open request for the second client.

61. How does namenode tackle datanode failures?

Namenode periodically receives a heartbeat (signal) from each of the datanode in the cluster, which implies datanode is functioning properly.

A block report contains a list of all the blocks on a datanode. If a datanode fails to send a heartbeat message, after a specific period of time it is marked dead.

The namenode replicates the blocks of dead node to another datanode using the replicas created earlier.

62. What will you do when namenode is down?

The namenode recovery process involves the following steps to make the hadoop cluster up and running:

Use the file system metadata replica (fsimage) to start a new namenode.

Then, configure the datanodes and clients so that they can acknowledge this new namenode, that is started.

Now the new namenode will start serving the client after it has completed loading the last checkpoint fsimage (for metadata information) and received enough block reports from the datanodes.

Whereas, on large hadoop clusters this namenode recovery process may consume a lot of time and this becomes even a greater challenge in the case of the routine maintenance.

63. What is a checkpoint?

In brief, "checkpointing" is a process that takes an fsimage, edit log and compacts them into a new fsimage. Thus, instead of replaying an 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. Checkpointing is performed by secondary namenode.

64. How is hdfs fault tolerant?

When data is stored over hdfs, namenode replicates the data to several datanode. The default replication factor is 3. You can change the configuration factor as per your need. If a datanode goes down, the namenode will automatically copy the data to another node from the replicas and make the data available. This provides fault tolerance in hdfs.

65. **Can namenode and datanode be a commodity hardware?**
The smart answer to this question would be, datanodes are commodity hardware like personal computers and laptops as it stores data and are required in a large number. But from your experience, you can tell that, namenode is the master node and it stores metadata about all the blocks stored in hdfs. It requires high memory (ram) space, so namenode needs to be a high-end machine with good memory space.
66. **Why do we use hdfs for applications having large data sets and not when there are a lot of small files?**
Hdfs is more suitable for large amounts of data sets in a single file as compared to small amount of data spread across multiple files. As you know, the namenode stores the metadata information regarding the file system in the ram. Therefore, the amount of memory produces a limit to the number of files in my hdfs file system. In other words, too many files will lead to the generation of too much metadata. And, storing these metadata in the ram will become a challenge. As a thumb rule, metadata for a file, block or directory takes 150 bytes.
67. **How do you define "block" in hdfs? What is the default block size in hadoop 1 and in hadoop 2? Can it be changed?**
Blocks are the nothing but the smallest continuous location on your hard drive where data is stored. Hdfs stores each as blocks, and distribute it across the hadoop cluster. Files in hdfs are broken down into block-sized chunks, which are stored as independent units.
Hadoop 1 default block size: 64 mb
Hadoop 2 default block size: 128 mb
Yes, blocks can be configured. The dfs.block.size parameter can be used in the hdfs-site.xml file to set the size of a block in a hadoop environment.
68. **What does 'jps' command do?**
The 'jps' command helps us to check if the hadoop daemons are running or not. It shows all the hadoop daemons i.e namenode, datanode, resourcemanager, nodemanager etc. That are running on the machine.
69. **How do you define "rack awareness" in hadoop?**
Rack awareness is the algorithm in which the "namenode" decides how blocks and their replicas are placed, based on rack definitions to minimize network traffic between "datanodes" within the same rack. Let's say we consider replication factor 3 (default), the policy is that "for every block of data, two copies will exist in one rack, third copy in a different rack". This rule is known as the "replica placement policy".
70. **What is "speculative execution" in hadoop?**
If a node appears to be executing a task slower, the master node can redundantly execute another instance of the same task on another node. Then, the task which finishes first will be accepted and the other one is killed. This process is called "speculative execution".
71. **How can i restart "namenode" or all the daemons in hadoop?**
This question can have two answers, we will discuss both the answers. We can restart namenode by following methods:

You can stop the namenode individually using***. /sbin /hadoop-daemon.sh stop namenode*** command and then start the namenode using***. /sbin/hadoop-daemon.sh start namenode*** command.

To stop and start all the daemons, use*. /sbin/stop-all.sh* and then use *./sbin/start-all.sh* command which will stop all the daemons first and then start all the daemons.

These script files reside in the sbin directory inside the hadoop directory.

72. **Name the three modes in which hadoop can run.**
The three modes in which hadoop can run are as follows:

Standalone (local) mode: this is the default mode if we don't configure anything. In this mode, all the components of hadoop, such namenode, datanode, resourcemanager, and nodemanager, run as a single java process. This uses the local filesystem.

Pseudo-distributed mode: a single-node hadoop deployment is considered as running hadoop system in pseudo-distributed mode. In this mode, all the hadoop services, including both the master and the slave services, were executed on a single compute node.

Fully distributed mode: a hadoop deployments in which the hadoop master and slave services run on separate nodes, are stated as fully distributed mode.

73. **What is "mapreduce"? What is the syntax to run a "mapreduce" program?**
It is a framework/a programming model that is used for processing large data sets over a cluster of computers using parallel programming. The syntax to run a mapreduce program is **hadoop_jar_file.jar /input_path /output_path**.

74. **What are the main configuration parameters in a "mapreduce" program?**
The main configuration parameters which users need to specify in "mapreduce" framework are:
Job's input locations in the distributed file system
Job's output location in the distributed file system
Input format of data
Output format of data
Class containing the map function
Class containing the reduce function
Jar file containing the mapper, reducer and driver classes

75. **State the reason why we can't perform "aggregation" (addition) in mapper? Why do we need the "reducer" for this?**
This answer includes many points, so we will go through them sequentially.

We cannot perform "aggregation" (addition) in mapper because sorting does not occur in the "mapper" function. Sorting occurs only on the reducer side and without sorting aggregation cannot be done.

During "aggregation", we need the output of all the mapper functions which may not be possible to collect in the map phase as mappers may be running on the different machine where the data blocks are stored.

And lastly, if we try to aggregate data at mapper, it requires communication between all mapper functions which may be running on different machines. So, it will consume high network bandwidth and can cause network bottlenecking.

76. **Explain “distributed cache” in a “mapreduce framework”.**
    Distributed cache can be explained as, a facility provided by the mapreduce framework to cache files needed by applications. Once you have cached a file for your job, hadoop framework will make it available on each and every data nodes where you map/reduce tasks are running. Then you can access the cache file as a local file in your mapper or reducer job.
77. **How do “reducers” communicate with each other?**
    This is a tricky question. The “mapreduce” programming model does not allow “reducers” to communicate with each other. “reducers” run in isolation.
78. **What does a “mapreduce partitioner” do?**
    A “mapreduce partitioner” makes sure that all the values of a single key go to the same “reducer”, thus allowing even distribution of the map output over the “reducers”. It redirects the “mapper” output to the “reducer” by determining which “reducer” is responsible for the particular key.
79. **How will you write a custom partitioner?**
    Custom partitioner for a hadoop job can be written easily by following the below steps:

    Create a new class that extends partitioner class

    Override method – getpartition, in the wrapper that runs in the mapreduce.

    Add the custom partitioner to the job by using method set partitioner or add the custom partitioner to the job as a config file.

80. **What do you know about “sequencefileinputformat”?**
    “sequencefileinputformat” is an input format for reading within sequence files. It is a specific compressed binary file format which is optimized for passing the data between the outputs of one “mapreduce” job to the input of some other “mapreduce” job.
    Sequence files can be generated as the output of other mapreduce tasks and are an efficient intermediate representation for data that is passing from one mapreduce job to another.