Top Questions for Data Engineer interview

Follow Rahul Agrawal

Let's Go

How to decide executors

How to deploy spark cluster on kubernetes

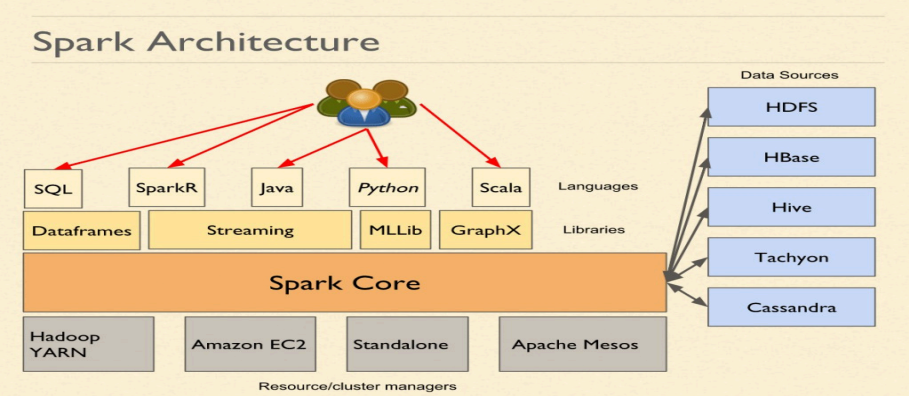
1) Howtoconnects3todatabricks
2) Howtoconnectawsdatacatalogtodatabricks
3) Howtosubmitdatabrickjob
4) Howtocreatejobondatabricks
5) Howtocreatedependentjobsondatabricks
6) Howtocreatepipelineindatabricks

1) Whatisspark?
   - Sparkisanopensourcedistributedcomputingsystemdesignedtoprocesslarge amount of data (structured semi structured and unstructured) in distributed and parallelize manner across the cluster of computers
     - Itsabilitytoprocessdatainmemorymakesitfasterthanotherbigdata processing framework which users disc based storage data processing
   - Unifiedengineforlargescaledataanalytics.
   - SparkalsoprovidesawiderangeoflibrariesforDatascience,ML,DL,streaming and graph processing.

2) DifferenceBetweenSparkandMR?

points	Spark	MR
Data processing model	In-memory Sparksupportbatch processing,streaming processing and interactive proessing	Discbasedstorage MRonlysupportsBatch dataprocessing
Speed	Faster than Map reduce(10 to 100x) ,Spark caches the datainmemorywhich makesitfasterexecution ofcodeandusesDAG execution	Slower in speed . MR used 2 stage processingmodelfor execution(MAPandthen Reduce)
Ease of development	Spark provides wide range ofhighlevelAPI for different Language like (python,scala,java,R) this makeiteasyfor development	MR provides only support forJAVA In MR developer needs to writecodeinLowlevelfor dataprocessing

3) Explainarchitectureofspark?



4) WhatisRDD?

- RDDstandsforresilientdistributeddatasetsitisanfundamentalDatastructure of an apache spark
- RDDisandistributedcollectionofobjectsthatcanbeprocessedinparallel,RDD can be created from data stored in distributed storage such as HDFS,local file system and NOSQL DBs
- RDDsarefaulttolerantmeanstheycanrecoverfromNodefailureandthisfault tolerance is achieved by Maintaining lineage information .which records the information about the transformation applied to RDD.in the case of node failure the lost information can be received from lineage information.
- RDDsupportstwotypeofoperationsactionandtransformations,transformation create new RDD from an existing RDD by applying a function to each element of an RDD
- OntheotherhandactionstriggersthecomputationofanRDDandreturnsa value to the driver program or write the data to external storage

5) DifferenceBetweennarrowandwidetransformation?

- NarrowTransformations:narrowtransformationsaretheonewhichdonot require shuffling and data-exchange between the partitions .they can be executed in each partitions independently (MAP FILTER UNION)
- Widetransformations:widetransformationsarethetransformationsthatrequire data exchange between the partitions and they require shuffling across the network ,which can be expensive operations in terms of time and network bandwidth (group by reduce by join)
- Themaindifferencebetweennarrowandwidetransformationsisthatnarrow transformations can be executed in each partition **independently**, while wide transformations require data exchange between partitions. This means that wide transformations can be slower and more resource-intensive than narrow transformations.
- Itisimportanttonotethatthechoiceoftransformationdependsonthe application requirements and the size of the data being processed. In general, it

is recommended to minimize the use of wide transformations as much as possible to optimize the performance of the Spark application.

6) WhatisthedifferencebetweenDAGandLINEAGE?

- DAGisagraphdatastructurethatrepresentsthedependenciesbetween task or steps in workflow.the edge between the nodes represents the dependencies between tasks
- LINEAGEotherhandsrepresentsthehistoryoftransformationsappliedon a particular piece of data .lienage tracking is important in data processing because it enable data scientist analysts and engineers to understand how particular result was generated and to trace error back to their source

7) Whatispartitionandhowsparkpartitionsthedata?

- Partitionsrefersthеprocessofdividingthelargedatasetintosmaller logical chunks and manageable pieces .in distributed computations it is the common techniques to parallelize the computation of large dataset across the cluster of computers or nodes
- Apachesparkpartitionsthedatabybreakingitintosmallerchunksthat can be process in parallel across the cluster .spark organizes the data into RDDs and uses the partitioner to determine how to divide data into partitions .
- HASH_PARTITIONER(default): assignrecordtoapartitionbasedonhash value the hash function is used to evenly distribute record across the partition based on key value
- RANGE_PARTITIONIER:rangepartitioningisusedwhenthedatais ordered and can be partition based on range of values .this technique is used when you want to maintain ordering of data in each partition
- ThenumberofpartitionsinSparkcanbespecifiedbytheuser,andthe partition size can be adjusted to optimize performance based on the available resources in the cluster.

8) whatissparkdriverordriverprogram?

- Sparkdriverordriverprogramisthemainprogramthatcontrolsthe execution of spark application it is responsible for the coordinating execution of task across the cluster and communicating to the resource manager for resource allocation to the spark application

- Driverprogramisthefirstcomponentthatstartoncesparkapplicationis started and it runs on the machine where application was submitted .it interact with the sparkContext which is the entry point for all spark functionality and creates RDD for data processing .the driver program is also defines the computations to be performed on RDD using actions and transformations
- Thedriverprogramisresponsibleforsettinguptheconfigurationofthe Spark application, including the location of the input data, the number of executors, the amount of memory allocated to each executor, and other parameters. It also monitors the progress of the application and handles any errors or exceptions that occur during the execution.

9) Whatisanexecutor?

- Anexecutorisaworkernodeprocessresponsibleforexecutingthetasks and storing data in memory or on disc and returning the results to driver program .they are responsible for carrying out the actual computation of spark tasks on the data
- Whensparkapplicationislaunchedthedriverprogramcommunicates with the cluster manager (YARN,MESOS OR STANDALONE) to request resources to run the application the cluster manager then launches then executor processes on worker node .
- EachexecutorhasitsownJVMprocessandrunsonaworkernode .executor are launched and managed by cluster manager and are responsible for running the tasks that are assigned to them by driver program
- Oncethecomputationisfinishedexecutorreturntheresulttodriver program

10)What is worker Node?

- Aworkernodeisamachineisclusterthathasresources(CPU,RAM,DISK) allocated to it by cluster manager to run spark task.each worker node typically runs more than one executors ,which are responsible for executing the task assigned to them by the driver program.
- Workernodeareresponsibleforrunningtheactualcomputationonthe data and storing data in memory or on disk .they can also shuffle data between each other during the computation
- Workernodecanbeaddedorremovedbasedonresourceavailableand workload .the cluster manager is responsible for the managing the allocation of resources to the worker nodes and ensuring that the worker node are available and responsive

11)What is the lazy evaluation is spark?

- Lazy evaluation is the key feature of spark that allows it to optimize data processing by delaying the evaluation of transformation until the results are actually needed .
- Lazy evaluation means that spark doesn't execute the transformations right away ,but instead it bulbs the logical execution plan(DAG) that represent the sequence of transformations to be executed when an action is called
- DAG is optimized by catalyst optimizer to reduce the number of shuffle and expensive operations by delaying the execution of transformation until an action is called ,spark can optimize the execution plan and minimize the amount of data shuffled across the network ,this can result in significant performance gain especially when dealing with large datasets
- Lazy evaluation also allows spark to handle cyclic dependencies between RDDs

12)What is the pair RDD in spark?

- PairRDDisanRDDwhereeachelementisakey-valuepair,pairRDDused extensively in spark for operations that require grouping aggregating or joining data based on keys

13) Difference between persist() and cache() in spark?

- Bothpersist()andcache()aremethodsusedtostoreRDD,DF,DSin memory to speed up subsequent computations .
- cache()=persist(StorageLevel.MEMORY_ONLY)
- Persistallowsyoutospecifythestoragelevelexplicitlywhichdetermines where and how RDD will be cached (MEMORY_ONLY_SER ,MEMORY_AND_DISK ,DISK_ONLY) where as cache() always stored the cached data in memory using the default storage level
- cache()islazyotherhandpersist()isanegeroperationwhichimmediately caches the RDD in memory

14) What is serialization and Deserialization?

- serialization andDeserializationareprocessofconvertingdatastructure or objects into a format that can be transmitted over a network or stored in a file and then converting them back into the original form
- Objecttostreamofbytes=>serialization andreverseisDeserialization.

- Spark uses serialization and deserialization extensively to transmit the data between the nodes in a cluster ,by default spark uses java serialization whichisnotveryefficientandcanleadtoperformance issues ,however spark also supports other serialization formats like KYRO and AVRO which is more efficient and faster than java serialization

15)Difference between MAP and FLATMAP?

- BotharetransformationsthatcanbeappliedonRDDandDFs,themain difference between them is how the handle nested data

```
val rdd = sc.parallelize(Seq("Hello World", "Apache Spark"))

val mapRDD = rdd.map(line => line.split(" "))

// Result: [["Hello", "World"], ["Apache", "Spark"]]

val flatMapRDD = rdd.flatMap(line => line.split(" "))

// Result: ["Hello", "World", "Apache", "Spark"]
```

16) Various levels of persistence spark?

- MEMORY_ONLY:storesthedatainmemoryasdeserializedjavaobjects.if the data does not fit in the memory some partitions will not be cached and will be computed on the fly when needed
- MEMORY_ONLY_SER:storesthedatainmemoryasserializedjavaobject s ,this save the memory compared to first one but serialization and deserialization is compute expensive
- MEMORY_AND_DISK:storesthedatainmemoryasdeserializedjava object but splits the data to disk id the memory is full .this can be useful if the data is too large to fit in memory.
- MEMORY_AND_DISK_SER:sameasaboveonlydifferenceisitis serialized
- DISK_ONLY:storesdataindiskonly

- MEMORY_ONLY_2 : same as MEMORY_ONLY,MEMORY_ONLY_SER but only with 2 replicas
- OFF_HEAP : stores the RDD and DF on off-heap memory

17)What is an accumulator in spark ?

18) What is the broadcast variable in spark ?

- Broadcastvariableisareadonlyvariablethatcanbeusedtocachevalue or data structure on each worker node rather than sending a copy of it with every task .broadcast variable are typically used for sharing large read only data structures such as lookup tables or ML models with the task that need to use them
- Broadcastvariablearecreatedbycallingthe "sparkContext.broadcast(value)" method ,the broadcast() method returns a "Broadcast[T]" object where T is type of the value
- Onceabroadcastvariableiscreatedbyitcanbeusedinaclosure(a function that captures a variables from the enclosing scope) that is executed on each worked node .the value of broadcast variable is cached on each worker node so that it can be accessed quickly by the task that use it

19) What is checkpointing in spark ?

- CheckpointingisamechanisminsparkthatallowstheRDDlineagetobe truncated and saved to a reliable storage system such as HDFS. to prevent recomputing the RDD from scratch in case of a mode failure or other issued ,checkpointing can improve the reliability and performance of a spark application ,especially for complex DAGs with many stages and log lineage chains
- Whencheckpointing istriggeredsparkwritesthecurrentRDDlineageto disk and replaces it with placeholder RDD tha points to the checkpointing data .subsequent transformations and actions on RDD will use the checkpointing data instead of original data reducing the amount of computation needed to receiver from failure
- Setcheckpointing directoryusingthe "SparkContext.setCheckpointDir(path)" where path is the path to the checkpoint directory on a reliable storage system
- ThencallRDD.checkpoint()

- Notethatcheckpointingincursaperformanceoverheadassparkneedsto write the data to disk and read it back therefore checkpointing should be used judiciously and only for RDD with long lineage or complex DAGs

20) What is sparkContext ?

- sparkContext istheentrypointtoasparkapplicationandrepresentthe connection to a spark cluster ,it is a client side object that coordinates the execution of tasks on the spark cluster.
- WhenasparkapplicationstartsitcreatesasparkContextthatmanages the allocation of resources for the application ,the sparkContext also provides APIs for creating RDDs.
- ThesparkContextistypicallycreatedonlyoncerperapplicationandall subs

21)What is executor memory ?

- Executormemoryreferstotheamountofmemoryallocatedtoeach executor process running on worker node .this executor memory is used to store data and perform computations on the data
- Executormemoryisdividedintotwoparts
  - Storagememory:thisistheamountofmemoryusedtocachedata in memory that will be reused across multiple stages of spark job .this is used to reduce the time it takes to access the frequently accessed data and can be configured using the "**spark.storage.memoryFraction**" parameter
- Executionmemory:thisistheamountofmemoryusedtoexecutetasks within a stage of a spark job.this included memory for storing intermediate results and data structures used during the computation.this can be configured using the "**spark.executor.memory**" parameter

22) Static resource allocation and dynamic resource allocation?

- **Staticresourceallocation:inthismodetheresourceareallocatedto spark application in a fixed and static manner .this means that resources like CPU memory and cores are assigned to the application at the time of launching the application.**Static resource allocation is suitable for running applications that have predictable resource requirements and run for a long time .
- Tousestaticyouneedtospecify "spark.executor.instance"config para,aetr to the number of executors you want to allocate and the "spark.executors.cores" and "spark.executor.memory" per executor respectively
- dynamicresourceallocation:inthismoderesourcesareallocatedtothe spark application dynamically based on the workload of the application

.When a spark application starts it requests a minimum number of executors and can dynamically acquire additional executors as needed based on the workloads.

23) Explain spark stages?

- Astageissetofparalleltasksthatareexecutedondatapartitionsina single executor. Each spark job is divided into multiple stages which are the executed sequentially or in parallel depending on the dependencies between them
- Astageinsparkconsistsoftwotypesoftasks
- SHUFFLEMAPTASKS:thesearethetasksthatreaddatafrominput sources and apply transformations on the data to generate output records .the output record then partitioned based on a hash function and shuffle across the network to be consumed by the reduce tasks in next stage
- Reducetasks : thesearethetasksthatreceiveandshuffledatafromthe previous stage and apply a reduction operation on the data to produce the final output stage.

24) Difference between coalesce() and repartition() in spark ?

- BothareusedtochangethenumberofpartitionsofanRDDorDF
- coalesce()isusedtodecreasethenumberofpartitionswhilereparitiion() is used to increase or decrease the number of partitions
- Coalesce()avoidstheshuffling itisfasterthan**repartition**
- coalesce(numpartition:int,shuffle:boolean(false))shuffleparameter controls the shuffle across the network

25)What is spark SQL?

- SparkSQLisamoduleinsparkthatprovidesaprogramminginterfacefor working with structured and semi-structured data using SQL syntax .it allow you to perform data analysis on large datasets using the power of distributed computation provided by spark .
- SparkSQLprovidedhighlevelAPIforworkingwithstructuredandsemi structured data in spark .it supports several data formats such as parquet ,avro,JSON and CSV .spark SQL also provides a dataframe API which is an abstraction on top of RDD

25) difference between RDD ,DF and DS ?

RDD (Resilient Distributed Dataset): RDD is the fundamental data structure in Spark. It is an immutable distributed collection of objects that can be processed in parallel. RDDs can store any type of objects, including user-defined classes, and support two types of operations: transformations and actions. Transformations create a new RDD from an existing one, while actions return a result to the driver program or write data to an external storage system. RDDs are suitable for low-level transformations and custom processing logic that require fine-grained control over data.

- **DataFrame**:DataFrameisadistributedcollectionofdataorganizedintonamed columns. It is conceptually equivalent to a table in a relational database or a data frame in R/Python. DataFrames are built on top of RDDs, but provide a higher-level API for structured data processing. DataFrames are immutable and support a rich set of operations, including filtering, grouping, aggregations, joins, and window functions. They also provide support for various data sources, such asCSV,Parquet,andJSON.DataFramesarebestsuitedforstructureddata processing where data is organized in rows and columns.
- **Dataset**:Datasetisastronglytypeddistributedcollectionofobjectsthatprovides the benefits of RDDs (strong typing, compile-time type safety) and DataFrames (optimized execution plans, query optimization). It is a recent addition to Spark and is available in Scala and Java (since Spark 1.6) and Python (since Spark 2.0). Datasets can be transformed using functional transformations, and Spark automatically generates efficient execution plans. Datasets can also be used with Spark's machine learning library, MLlib. Datasets are suitable for both structured and unstructured data processing and provide a high-level API for easy-to-use data manipulation.

In summary, RDDs are the fundamental data structure in Spark and provide low-level transformations and custom processing logic, DataFrames are best suited for structured data processing, and Datasets provide a strongly typed API for both structured and unstructured data processing with optimized execution plans.

26) How spark SQL is different from HQL and SQL?

- SparkSQLisamoduleinsparkthatprovidesaprogramminginterfacefor working with structured and semi-structured data using SQL syntax .it allow you to perform data analysis on large datasets using the power of distributed computation provided by spark .
- SparkSQLprovidedhighlevelAPIforworkingwithstructuredandsemi structured data in spark .it supports several data formats such as parquet

,avro,JSON and CSV .spark SQL also provides a dataframe API which is an abstraction on top of RDD

- HQL:isusedwithHive,whichisaDWHandSQLlikequerylanguagefor Hadoop .HQL is similar to SQL but is optimized for querying data stored in HDFS .HQL provides high level abstraction on top of MR .making it easy to work with large datasets .
- SQLisastandardlanguageusedforqueryingandmanipulatingdatain relational DBs

27) What is catalyst Optimizer ?

- CatalystoptimizerisaqueryoptimizerinsparkSQLmodulethatoptimizes and transform structured queries written in SQL ,DF or DS it leverages techniques from computer science and programming to genrate more efficient query plans that can be executed and distrubuted data processing plaforms
- Catalystoptimizerconsistof3maincomponents
  - ANALYSIS:analyzethequerytodeterminethemetatadata,suchas the schema of the data and the available functions and operators .it also validates the query to ensure that it is semantically correct.
  - LogicalOptimizations :itappliesrulestotransformthelogicalplan ,which represents the query in a tree like structure .it applies the set of algebraic transformations to optimize the logical plan and simplifying expressions.
  - Physicalplanning:thiscomponentgeneratesaphysicalplanwhich represents how the logical plan will be executed on the distributed computing system .it considers the available resources such as number of nodes and their HW configurations to generate optimized execution plan .it also leverages the cost based optimization techniques to generate a plan that minimizes the overall processing time and resource utilization
- Itisalsoextensiblemeansusercandefinetheirownrulesand optimization techniques to apply to their queries

28) What spark streaming ?

29)What is Dstream?

- KnownasDiscretizedstreamandisahighlevelabstractionninapache spark streaming that represents continuous stream of data .it allows developer to process real time streaming data from source such as kafka ,flume and HDFS in a scalable and fault tolerant manner
- ADstreamisrepresentedasasequenceofRDDwitheachRDDcontaining data from a specific time interval .the interval length is defined by the

batch interval parameter ,which is specified by the developer during the Dstream creation

- Dstreaminsparkstreamingprovidefaulttolerancebyreplicationdata across multiple nodes in cluster ,ensure that the data is not lost in case of node failure ,Dstream also support checkpointing which allows the spark streaming application to recover from failures and continue processing the stream from the point of failure.

30)What is checkpointing in spark streaming ?

- checkpointing insparkstreamingisamechanismthatallowsthesystem to receiver from failure and ensure data fault tolerance.
- InsparkstreamingDstreamisdividedintobatchesofdatawhichare processed in parallel using spark .checkpointing enables the spark streaming system to save the metadata of the streaming application's state to a fault tolerant storage system such as HDFS or any other reliable distributed file system .this metadata includes the configuration settings ,the RDD lineage and the job progress.
- Toenableityoucanusethe**checkpoint**()methodon**streamingContext** object

31)How to create microBatch and its benefits ?

- InsparkStreamingmicrobatchingisaprocessingmodelwherethe incoming data stream is divided into small batches or microbatches of fixed time duration ,typically ranging from 100ms to a few seconds ,each microbatch is processed independently as a spark RDD
-
- frompyspark.streamingimportStreamingContext
- ssc=StreamingContext(sparkContext,1)
- Benefits :
- Lowlatency:byprocessingdatainsmallbatchessparkstreamingcan achieve low latency
- Betterresourceutilizations :microbatchingallowssparkstreamingto better utilize cluster resources by scheduling the processing of small batches across the available nodes
- Faulttolerance:incaseoffailuresparkstreamingcanrecoverfromthe failure by replaying the lost microbatches
- Easeofuse:

32) What is windowing in spark streaming ?

- Windowingisatechniqueinsparkstreamingthatenablesusersto perform computations over a sliding window of data in a streaming

application .it allows user to group data into discrete chunks or windows and apply computation on these window as a whole.

- Insparkstreamingawindowisdefinedbytwoparametersthewindow duration and the slide duration .the window duration specifies the length of window while slide duration specifies the interval between two consecutive window .

33) Will you able to do datasets in python?

34)How to join two tables by using DF in spark?

- Usingjoin

```
From pyspark.sql import SparkSession

spar=sparksession.builder.appname("join two table").getorcreate()

df1=spark.read.csv("path",header=true,inferschema=true)

df2=spark.read.csv("path",header=true,inferschema=true)

join_df=df1.join(df2,df1[c1]==df2[c2])

join_df.show()
```

35)How to remove duplicate records in dataframe?

- DISTINCT
- DROPDUPLICATES
- WINDOWFUNCTIONWITHROWNUMBER
- GROUPBYWITHCOUNT

36)How to add columns in DF? =>withcolumn()

37) SQL basic concepts such as rank ROW ,Row number & dense rank ?

38)Query to find 2nd largest number in the table?

39)Query to find duplicate records in table ?

40)Difference between list and tuple?

- List=>mutable,tuple=>immutable
- Listarecommonlyusedfordynamicarrayswhereyouneedtiaddor remove items frequently ,

- Tuplesarecommonlyusedforfixedsizedcollectionswhereyoudon't need to modify the content
- Tuplesarefaster

41) Difference between def and lambda?

- Lambdafunctionareusewhereyouneedtoimplementthatcodeonce only for that specific point
- Defdefinesfunctionlambdaisanonymousfunction

42)What is aws EMR?

43) Explain about current working project?

44)How to write UDF in hive ?

45)File format row based vs column based?

- Rowbased;CSV,TSVandJSON
- Columnbased:parquet,ORC,ApacheArrow

46)ORC,AVRO,PARQUET?

47)What is RDD?

48)How to join two larger table in spark ?

49)Bucketed Tables?

50)How to read parquet file format in spark?

51)How to tune spark executor ,cores and executor memory?

Spark submit –executor-memory 4G –executor -cores 4 –num-executors 10

52) Default partition size in spark ?

128 mb

53) Is there any use of running spark program on a single machine?

- Useonlyfordevelopmentandtesting,smalldatastes,resourceconstrains ,educational purpose

54)How will you find how many resources are available in YARN?

55)Difference between cluster and Client mode?

56)Explain about dynamic allocation in spark?

- DynamicallocationisafeatureinApacheSparkthatallowsforautomatic adjustment of the number of executors in a Spark application based on the workload. It enables the application to scale up or down the number of executors based on the current workload, allowing for better resource utilization and reduced costs.
- Withdynamicallocation,Sparkautomaticallyacquiresnewexecutors when the workload increases and releases unused executors when the

workload decreases. This means that resources are allocated to the application only when needed, and the resources can be returned to the cluster when they are no longer needed.

- Therearethreemaincomponentofdynamicallocationisspark
  - Executorallocationmanager:itisresponsibleforwhentoaddor remove executor based on current workload
  - Shfulletracking:thiscomponenttracksthedatathatisbeing shuffled between the stages of the application and uses this information to determine when to add or remove executors
  - Executors:
- Dynamicallocationcanbeconfiguredinthesparkconfigurationfile.by default it is disabled ,but it can be enabled by setting the "**spark.dynamicAllocation.enabled" config property**

57) Difference between partition by and clustering by in hive ?

- PARTITIONBY:isawayofdividing atableintosmaller,moremanageable parts based on the values of one or more columns .when you partition a table the data is physically stored in separate directories based on partition column .partitioning makes it easy to query and manage large datasets because you can focus on a s
- mallersubsetsofthedata
- CREATETABLEMY_TABLE()PARTITIONBY(columnname)
- CLUSTERBY:clusteringisawayoforganizingdatawithinapartition based on the values of one or more columns .when you cluster a table ,data is sorted within each partition based on the clustering columns clustering can improve query performance because it reduces the amount of data that need to be scanned
- CREATETABLEMY_TABLE()CLUSTERDBY(COLUMNNAME)INTO32 BUCKETS

58)How to choose a partition column in hive ,tell me which column you shouldn't use partitioning and why?

- Chooseacolumnthathashighcardinality.thepartitioncolumnshould have many distinct values .this helps to ensure that each partition contains manageable amount of data if the partition has low cardinality then some partitions may be too large or too small which can lead to inefficient queries
- Chooseacolumnthatisfrequentlyusedinqueries.partitioningismost effective when partition column is frequently used in WHERE or JOIN conditions .if the partitioning column is rarely used in queries the partitioning may not improve the query performance

- Chooseacolumnthatisnotfrequentlyupdated:partitioningisbasedon columns values so if the column values changes frequently then the partitions may need to be updated frequently as well .this can be costly operation so best to choose the column with less frequent update
- Avoidpartitioningoncolumnswithlowselectivity:
- Avoidpartitioningoncolumnswithlongstringsorcomplexdatatypes: partitioning on such columns can be inefficient because the partition key may be too large to fit into memory
- Donotusecolumnwithlargenumberofuniquevalues(suchasprimary key)

59)How will you transfer data from unix system to HDFS ?

- Usinghadoopcli:youcanuseHADOOPCLItocopydatafromunixtohdfs
- Hadoopfs-putsrcdest
- Usinghadoopdistributedcopytool:DistCpisadistributeddatacopying tool that can transfer large amount of data between hadoop cluster or between hadoop and other storage system
- Hadoopdistcpsrcdest

60)What is repartition ? Is it possible to reduce the partition?

- RepartitioninginsparkreferstotheprocessofshufflingthedatainaDFto create a new set of partition with specified number .
- Repartitionshufflesthedatasoitsgenerallyrecommendedtoonly repartition when necessary
- Itisalsopossibletoreducepartitionsusingrepartition()andcoalesce(),
- Itisimportanttonotethatwhenreducingthenumberofpartitionswith coalesce() data may not be evenly distributed across the partitions which can be lead to performance issues therefore it is recommended to only use coalesce() when you are confident that the data is already well distributed across the partitions

61) How will you extract only different data from two different tables ?

62)What is the difference between SQL and NOSQL ?

- SQLDBsarewellsuitedforapplicationthatrequirestructureddata,strong consistency and ACID compliant
- NOSQLDBsareusedforapplicationthatrequireflexibilityscalabilityand ability to store and process large volumes of unstructured data

63)Who will run the spark job in your team ?

64)How will you find a particular text name in your hdfs?

- Hadoopfs-ls |grepfilename

65)Explain about the ingestion process ?

- Ingestionistheprocessofimportingdatafromexternalsourcesintoa system or application ,usually a data mgmt systems such as DBs or DWH
- Ingestioninvolves:
  - Dataextraction:pullingthedatafromvarioussources,suchasfiles ,APIs or DBs
  - Datatransformation:cleaningandformattingthedatatoensure consistency and compatibility with target system or application ,this may involve data mapping ,data validation and data conversion
  - Dataloading:thisinvolvedloadingthetransformeddataintotarget system
  - Datavalidation:

66)HBASE Basics ?

67)Explain about sort merge bucket join ?

- SortmergebucketjoinisatypeofoperationinhadoopMRthatisusedto **join large dataset that are too big to fit into the memory of a single node** .it is a variant of the standard MR sort merge join but it used the bucketing to improve the performance
- Inthisjoinmethodtheinputdatasetarefirstpartitionedandthensorted based on a common key .the partitioning is done based on the hash value of the key and each partition is stored in a separate bucket .the number of buckets is typically chosen to be a power of 2 for efficient distribution across the cluster
- Oncetheinputdatasetsarepartitionedandsortedthejoinoperationcan be performed by comparing the keys in each bucket and identifying the matching records .the join process is done in two MR phase
- MAPphase:inthisphasethemapperprocesseseachbucketofboth input dataset and emits a key value pair where key is the join key and the value is the data from the input record

68) Explain about tungsten ?

- tungsten isanoptimizationframeworkthatwasintroducedinspark1.5to improve the performance of spark application .it consist 3 key components
- MEMORYMGMT:tungsten leveragesoffheapmemoryforserialization and deserialization of data to avoid the overhead of garbage collection and improve performance

- BINARYPROCESSING:tungsten usesbinaryformatsforprocessingdata instead of using java objects ,which significantly reduces memory footprints and increases processing speed
- CODEGENERATION:tungsten generatesoptimizedcodeforsparkSQL queries during runtime using the catalyst optimizer .this reduces the overhead of interpreting code at runtime and improves the performance
- tungsten isdesignedtoworkwithbothbatchandstreamingworkloads and can be particularly effective for complex analytical workloads

69)How will you join two bigger tables ?

70)Explain about the outer left join ?

- Inotherwords,anouterleftjoinreturnsalltherowsfromthelefttable, and only the matching rows from the right table. If there are no matching rows in the right table, the result will still contain the rows from the left table, but with null values for the right table columns.

71)How to count the lines in a file by using linux command ?

- Wc-1example.txt

72)How to achieve map side join in hive ?

- Mapsidejoinisatechniqueinhivethatallowsjoiningtwolargetablesby performing the join operation on the map side (ie .before shuffling and reducing) this can significantly improves the performance of the join operation as it avoids the expensive shuffle and sort phases.
- Toachievemapsidejoinfollowingconditionsshouldbesatisfied
  - Boththablesshouldbebucketedonthejoinkeyorsortedonjoinkey
  - Boththableshouldhavethesamenumberofbuckets
- SELECT/*+MAPJOIN(table1) */*FROMtable1JOINtable2ON (table1.join_key=table2.join_key);

73)Will it got to the reducer if I am using the select command in hive ?

- ingeneral,theSELECTcommandinHiveisatransformationoperationand not a reduction operation. Therefore, it does not involve a shuffle and reduce phase, and all the data remains on the map side.
- However,theremaybesomecaseswheretheSELECTcommandcan trigger a shuffle and reduce phase. For example, if you use the DISTINCT keyword in your SELECT statement, Hive will need to perform a shuffle and reduce operation to eliminate duplicates. Similarly, if you use the GROUP BY or ORDER BY clauses in your SELECT statement, Hive will also need to perform a shuffle and reduce operation to group or sort the data.
- Ingeneral,though,ifyourSELECTstatementdoesnotinvolveanygrouping, aggregation, or sorting, then it should not trigger a shuffle and reduce operation, and all the data should remain on the map side.

74)How to validate the data once ingestion is done ?

- Validationthedataingestionisanimportantsteptoensurethatthe data is accurately loaded into the system
- DATAPROFILING:profilethedatausingtoolslikeapacheatlas,nifi or datafu to understand the data's characteristic such as data type ,length and distribution
- DATASAMPLING:randomlysamplesthedatafromsourceand target systems and compare it to ensure that the data is accurately transferred
- DATACHECKUSUMMING:generatethechecksumfromthesource and target data and compare them to ensure that the data is loaded accurately (CRC,MD5,SHA)
- DATAAGGREGATION:aggregatethedatabygroupingondifferent attributes and compare the results in the source and target systems
- DATAVISUALIZATION:visualizethedatausingtoolslikezeppelin tableau or PBI to identify anomalies and outliers
- DATAQUERYING:querythedatainthesourceandtargetsystemto ensure that the data is accurately loaded and transformed
- DATATESTING:createtestcasestovalidatethedatasuchasdata quality tests or data completeness tests and automate the testing process using tools like apache test kitchen ,or airflow

75)What is data profiling ?

- Dataprofilingistheprocessofanalyzingandexaminingdatafromvarious sources to gain insights into its structure, content, quality, and relationships. It involves collecting descriptive and statistical information about the data, such as the data type, format, range, frequency, and distribution.
- Dataprofilingcanbeusedtodiscoverdataqualityissues,suchasmissing values, inconsistent data types, duplicate records, or outliers, and to understand the data's patterns, dependencies, and relationships. It can also help in identifying potential data integration or transformation issues.
- Dataprofilingcanbeperformedusingvarioustoolsandtechniques,such as SQL queries, statistical analysis, data visualization, and machine learning algorithms. The output of data profiling is typically a report or a set of metrics that provides a summary of the data's characteristics and quality, and a set of recommendations for data improvement or remediation.

76) What is the use of split by command in sqoop ?

- The**splitby** optioninsqoopisusedtospecifythecolumntobeusedfor data splitting while importing data from RDB to HADOOP
- Whenimportingthedatausingsqoopthe**splitby**optionisusedtodivide the data into equal sized partitions based on the values in the specified column .this helps to distribute the data evenly across the mappers in hadoop cluster ,enabling parallel processing of the data and improving the overall performance of the data import process
- BydefaultsqoopusesthePKcolumnofthetablefordatasplitting ,however in some of the cases PK may not be evenly distributed or may not exists making it necessary to use a different column for splitting in such cases the **split by** option can be used

77)Difference between dataframe and datasets ?

- DFandDSbotharedistributedcollectionsofdataorganizedintonamed columns ,
- Typesafety:DSarestaticallytypedmenstypeofthedataineachcolumn

are known at compile time ,this allows for type checking and improved performance but also required more upfront work to define types.DF other hand are dynamically typed meaning types are inferred at runtime

- OOP:DSsupportsOOPconstructs,suchascaseclasseswhichcanbe used to define the schema of the DS this makes it easier to work with complex data types and enable compile time safety checks .other hand DF do not support these constructs and require more boilerplate code for working with the complex datatypes
- Serialization:DSusesmoreefficientserialization formatsthanDFswhich can lead to improved performance

78)Schema on read vs schema on write ?

- Thisaretwodifferentapproachestohandlebigdatasystems
- SCHEMAONWRITE:inanapproachinwhichdataisfirststructured according to a predefined schema and then stored in a structured format like database .this approach requires the data to be transformed into predefined schema before it can be loaded into DB .the schema on write is used in traditional DWH system and is well suited for use cases where the structure of the data is well known and predictable
- SCHEMAONREAD:isanapproachinwhichdataisstoredinitsraw ,unstructured form ,without being transformed into a pre-defined schema ,in this approach ,the data is read into a system and then structured and transformed into a schema as it is being queried .this approach allows for more flexibility in handling data that may have an unknown or changing structure .this approach is commonly used in big data systems ,such as

hadoop where unstructured data is stored in hdfs and process using tools like spark

79)Different type of partitions in hive ?

- Staticpartitions:inthistypeofpartitioningthepartitionsaredefined before loading the data into the tables .each partition is a subdirectory in the table directory and the partition values are included in the directory name
- Dynamicpartition:inthispartitioningtypehivecreatenewpartitions automatically while inserting data into the table ,dynamic partition is useful when the number of partition is large and it is not feasible to manually add each partition
- Listpartition:inthistypeofpartitioningdataisdividedonthematching values of one or more partition column .the values for each partition are defined in a list
- Rangepartition:inrangepartitiondataisdividedbasedontherangeof values in one or more partition columns .each partition is defined with a min and max values and all value within that range are included in the par tition

80)Find count based on age group ?

81)Write a code to find word like “error” in a log file by using pyspark ?

82)Difference between HADOOP and SPARK ?

83)How to find duplicates value in SQL ?

84)How to find second largest number in the table ?

85)Explain about CAP theorem ?

86)What are the file formats that you are using in your projects ?

87)What are the various hive optimization techniques you know ?

- partitioning
- Bucketing
- Indexing:hivesupportsindexingoncolumnsbyindexingcolumns,hive can skip reading data from unnecessary partitions or buckets .which speeds up query processing
- Vectorizations :isatechniquesthatprocessmultiplerowsofdataat once instead of processing one row at a time .this improves query performance by reducing the overhead of processing individual rows .
- Costbasedoptimizer : theCBOisanewoptimizationtechniquesthat was introduced in hive .14 .it uses statistics about the table and the query to optimize the execution plan
- Joinoptimization:MAPJOIN,BROADCASTJOINandSORT-MERGEJOIN

- Compression:hivesupportsseveralcompressionformatssuchas snappy,LZO,GZIP ,compression data can reduce the disk IO and network bandwidth used during query processing which can significantly improve query performance .
- Bucketmapjoin:
- Parallelexecution:hivesupportsparallelexecutionofqueriesusingTEZ or spark as execution engine

88)How MR will work ?

89)How many types of tables have in hive ?

- Managed
- external

90)How to drop table in HBASE ?

- Openthehbaseshellbytypinghbaseshellinterminal
- Disablethetableusing**disable"tablename"**
- Dropthetableusing**drop""tablename""**
- Makesuretotakeabackupofthetableoritsdatabeforedroppingitasit

will permanently delete the table and its data.

91)Difference between batch and realtime processing ?

92)How can apache spark can be used alongside HADOOP ?

93)Data explode and lateral view in hive ?

- Explode:isafunctionusedtotransformacolumnthatcontainsanarray or map data type into multiple rows.it is used to flatten the data and create multiple rows from single row
- explode(col1),col1=(1,2,3)
- 1
- 2
- 3
- Lateralview:isusedtoexplodemultiplearraysormapsatthesametime or to unnest data structure .it is often used in combination with the explode

94)Combiner.shuffling sorting in map reduce ?

- COMBINER:isaminireducerthatperformsalocalaggregationsofthe intermediate map output .it helps reduce the amount of data transfer between the map and reduce stages .the combiner is executed on the output of the map tasks before it is transferred to the reduce task
- Shuffling : istheprocessoftransferringdatafromthemaptasktoreduce tasks .it involves copying the intermediate k-v pairs generated by the map tasks to the nodes where the reduce task are running .during the shuffle

phase the data is partitioned based on the key ,and each partition is sent to the appropriate reduce tasks

- Sorting : istheprocessofsortingtheintermediatekey-valuepairs generated by map phase .it is done to group the values for each key in a contiguous blocks so that they can be processed by the reduce tasks in a sorted order .sorting is performed in both map and reduce phase

95)Difference betweenreducebykeyandgroupbykey?

- Reducebykey:operationperformsthereduceoperationonthevaluesof each key in a partition and then shuffle the results across the partitions .finally the same reduce operation is applied on the shuffled outputs of all the partitions .this result in a combined output for each key
- Groupbykey: operationgroupsallthevaluesassociatedwithagivenkey and then shuffles the results .this operation results in a new RDD that contains paris of key and their corresponding group values .this operation can be computationally expensive
- Reducebykeycombinesvaluesofthesamekeylocallyoneachpartition and the shuffle the results across the partitions ,whereas groupbykey shuffles all the data over the network which can be computationally expensive therefore reduceykey is often preferred over groupbykey

96)What is serde in hive ?

- Serdestandforserializationanddeserializationinhive.itisasoftware framework used to read data from and write data to HDFS or any other supported system .serDes used to transform data from structured or semistructured format such as CSV,TSV,JSON,AVRO,ORC into hadoop compatible serialized format and vice versa

97)Why datasets preferred compared to DF

- StronglyTyped:Datasetsarestronglytyped,meaningthattheyprovide compile-time type safety, while DataFrames are weakly typed. This allows developers to catch type errors at compile time instead of runtime, improving the code's robustness.
- Performance:DatasetsofferbetterperformancethanDataFrames because they use the more efficient and optimized Spark SQL execution engine. This is because Datasets are strongly typed, which allows Spark to perform more aggressive optimizations.
- FunctionalProgrammingSupport:Datasetssupportfunctional programming constructs such as map, filter, and reduce, making it easier to express complex transformations on data. DataFrames, on the other hand, rely on more imperative SQL-like constructs such as select and where.

- Encoders:DatasetsuseEncoderstoconvertbetweenJVMobjectsand Spark's internal binary format. Encoders provide significant performance benefits by reducing the overhead of object serialization and deserialization. DataFrames do not have this feature.
- Compatibility:DatasetsarecompatiblewithbothJavaandScala,whereas DataFrames are only compatible with Scala and Python.

Overall, Datasets offer better type safety, performance, functional programming support, and compatibility compared to DataFrames, which makes them preferred for complex analytics and machine learning applications in Apache Spark. However, DataFrames are still a popular choice for simple ETL tasks and SQL-like operations due to their ease of use and simplicity.

98)How do you check file size in hadoop ?

Hadoop fs –du -h file.txt

99)What is the vectorization and why it is used ?

- Vectorizationistheprocessofperformingasetofoperationsonalarge set of data using a single instruction or a small set of instructions. It is used to speed up the processing of large datasets by reducing the number of instructions that need to be executed.
- InthecontextofbigdataprocessingframeworkslikeApacheSpark, vectorization refers to the use of optimized code to perform operations on data in batches, rather than row by row. This allows for more efficient use of CPU and memory resources, resulting in faster processing times and better performance.
- **Vectorizationisusedinvariousdataprocessingtasks,suchaslinear algebra operations, machine learning algorithms, and image and signal processing.** It is especially useful in data processing applications that involve large datasets, as it allows for efficient processing of large amounts of data.
- Forexample,inmachinelearningalgorithms,vectorizationcanbeusedto perform computations on large matrices of data. This can significantly improve the training and inference times of machine learning models, as it allows for efficient processing of large amounts of data in parallel.
- Overall,vectorizationisanimportanttechniqueforimprovingthe performance and scalability of data processing applications, especially in the context of big data processing frameworks like Apache Spark.

100) Complex data types in hive ?

- Array
- Map
- Struct
- Union: isacollectionofdataofdifferentdatatypes.hiveunioncanbe defined using the **UNIONTYPE** keyword and accessing using the union indexing operator **[ ]**
-

101) Comparison of file formats in hive ?

- InHive,thereareseveralfileformatsavailableforstoringdata.Eachfile format has its own advantages and disadvantages, and the choice of file format depends on the specific use case and requirements. Here are some commonly used file formats in Hive and their comparisons:

1. Text: Text file format is the most basic and widely used file format in Hive. Text files are human-readable, and can be compressed using standard compression algorithms like gzip or bzip2. However, text files have poor performance and are not suitable for large datasets.
2. SequenceFile:SequenceFileisabinaryfileformatoptimizedforHadoop. It stores data in a binary format, which allows for efficient serialization and deserialization of complex data structures. SequenceFiles can be compressed using standard compression algorithms and support block-level compression, which can reduce storage space and improve performance.
3. ORC:OptimizedRowColumnar(ORC)isacolumnarfileformatdesigned for Hadoop. ORC files store data in a highly compressed and efficient columnar format, which allows for fast query performance and reduced storage space.
4. ORCfilesalsosupportpredicatepushdownandotheradvancedfeatures for efficient processing.
5. Parquet: Parquet is a columnar file format similar to ORC, but with some differences in implementation. Parquet files are highly compressed and optimized for query performance. Parquet files support complex data types and schema evolution, which makes them suitable for large, evolving datasets.

Overall, the choice of file format in Hive depends on the specific use case and requirements. Text files are suitable for small datasets or for debugging purposes, while binary file formats like SequenceFile, ORC, and Parquet are suitable for large, production datasets. Columnar file formats like ORC and

Parquet are more efficient for analytic queries, while text and binary file formats are more suitable for simple queries or data interchange.

102) sampling in hive ?

- Samplinginhiveisatechniqueusedtoselectasubsetofdatafromalarger dataset for analysis or testing purposes .hive supports several sampling methods that can be used to select a subset of data from a table or a partition
- **Randomsampling :**SELECT*FROMmytableTABLESAMPLE(10PERSENT);it will select 10% of random data from source
- **Stratifiedsampling :**stratifiedsamplingisasamplingmethodusedtoensure that the selected sample is representative of the entire dataset by dividing the data into strata or group based on a specified column expression

```
SELECT * FROM mytable TABLE SAMPLE(10 PERCENT)

BUCKET 5

OUT OF 100 ON category
```

- **Systematicsampling :** thissamplingselecteverynthrowfromthedataset

```
SELECT * FROM mytable

TABLE SAMPLE (10 PERCENT)

STRATIFY ON id

ORDER BY id;
```

- SamplinginHivecanbeusefulfortesting,analyzingorexploringlargedatasets before performing full-scale operations, and can also improve query performance by reducing the amount of data to be processed. However, it is important to note that sampling introduces some degree of sampling error, and the results of the analysis may not be perfectly representative of the entire dataset.

103) what are the different type of xml files in hadoop?

- ConfigurationXMLfiles :usedtostoreconfigurationsettingsforHadoop services and components .this files contains key-value pairs that defines the conf properties
- Dataxmlfiles :usedtostoredatainstructuredformatusingXMLtagsand attributes .these files can be processed using hadoop like MR tools or hive to ETL into Hadoop cluster

104) what are techniques you used to tune your spark application?

- Partitioning
- Memorymgmt:itincludestuningthememoryallocationforsparksjvmand other components such as the driver and executors .configuring the right amount of memory for each component can help to reduce garbage collections overhead and improve performance
- Serialization
- Caching ;
- Shuffleoptimization:usingthetechniqueslikebucketing,sortingandusing custom partitioning
- Hardwareoptimization:Hardwareoptimizationinvolvestuningthehardware resources, such as CPU, memory, and disk, to improve the performance of a Spark application. Techniques such as using SSDs for storage and optimizing network bandwidth can improve performance.

105) difference between broadcast and accumulators ?

- Broadcast:Broadcastvariablesareread-onlyvariablesthatarecachedoneach worker node for efficient access during task execution. These variables are useful when a large amount of data needs to be shared across tasks, such as lookup tables, and are typically used to reduce data transfer overhead. Broadcast variables are created using the broadcast() method and are shared across all the tasks in a Spark job.
- Accumulators:Accumulatorsarewrite-onlyvariablesthatareusedtoaccumulate values across multiple tasks in a distributed environment. These variables are used to implement counters, sums, and other aggregation functions that need to be shared across multiple stages of a Spark job. Accumulators can only be updated in a task and are read-only on the driver. Accumulators are created using the accumulator() method.

In summary, broadcast variables are used to share data across all tasks, while accumulators are used to accumulate values across tasks. Broadcast variables are read-only, while accumulators are write-only.

106) what is broadcast join ?

107) what is difference between sparkcontext and sparksession ?

- Thesearetwocomponentsofsparkthatprovidesentrypointtointeractwith spark
- **Sparkcontext:** istheentrypointtothesparkcorefunctionalityandrepresents the connection to a spark cluster .it is the legacy entry point to spark and used to create RDD ,accumulators ,broadcast variable
- **Sparksession** :isintroducedinspark2.xasanenhancedandunifiedentrypoint to interact with spark . it is higher level API that combines the functionality of "**sparkcontext**","**sqlcontext**","**hivecontext**"and "**streaming context**" . spark session provides API to create DF,DS and Spark SQL operations
- SparkcontextislowlevelAPI,sparksessionishighlevelAPI
- SparkcontextisusedtocreateRDDandperforminglowleveloperations
- SparksessionisusedtocreateDF,DS,SSQLoperations

108) type of transformation

109) what are operations of dataframes?

110) difference between partitioning and bucketing ?

111) how do we handle incremental data ?
Handling incremental data refers to the process of incorporating new data into an existing data set over time. In Apache Spark, there are several ways to handle incremental data, depending on the use case and data format. Here are some common approaches:

1. Append data: In this approach, new data is added to an existing data set. This can be done using the union operation to concatenate two DataFrames or by simply appending new data to an existing file or directory.

2. Overwritedata:Inthisapproach,newdatareplacestheexistingdata.Thiscanbe done by simply overwriting an existing file or directory, or by using the overwrite option when writing a new DataFrame.
3. DeltaLake:DeltaLakeisanopen-sourcestoragelayerthatbringsACID transactions to Apache Spark and big data workloads. Delta Lake provides support for handling incremental data in a scalable and efficient way, with features such as versioning, time travel, and conflict detection.
4. Kafkaintegration:ApacheKafkaisadistributedstreamingplatformthatcanbe used for ingesting and processing real-time data. Spark provides an integration with Kafka through the Structured Streaming API, which allows for the processing of streaming data in real-time.
5. Change Data Capture (CDC): CDC is a technique for capturing changes made to a database and propagating those changes to other systems. In Apache Spark, CDC can be implemented using tools such as Apache NiFi or Apache Flume, which can capture changes made to a database and send them to Spark for processing.

These are just some of the approaches to handling incremental data in Apache Spark.

The choice of approach will depend on factors such as data volume, frequency of updates, and performance requirements.

112)explain the YARN architecture ?

Apache Hadoop YARN (Yet Another Resource Negotiator) is a resource management framework in Hadoop that separates the resource management and job scheduling functions. YARN provides a central platform for managing and scheduling resources in a Hadoop cluster.

The architecture of YARN is based on the following components:

1. ResourceManager(RM):TheRMisthecentralauthoritythatmanagesthe allocation of resources in the cluster. It receives resource requests from clients and schedules applications on the cluster.
2. NodeManager(NM):TheNMrunsoneachnodeintheclusterandisresponsible for managing resources such as CPU, memory, and disk. It communicates with the RM to request resources and report the status of the resources it manages.
3. ApplicationMaster(AM):TheAMisaper-applicationprocessthatisresponsible for managing the application’s resources and scheduling its tasks. It

communicates with the RM to request resources for the application and with the NMs to launch and monitor the application's tasks.

4. Container:Acontainerisaresourceallocationthatincludes CPU,memory,and disk. It is created by the RM and allocated to an AM, which then launches a task inside the container.

The YARN architecture follows a hierarchical design where the RM manages the overall resources in the cluster, while the NMs manage the resources on individual nodes. The AMs manage the resources required by individual applications and allocate containers to run the application's tasks.

The use of YARN allows for efficient allocation and management of resources in the Hadoopcluster,enablingmultipleapplicationstorunconcurrentlyandshareresources.

●

113) what is incremental sqoop? How you can do it ?
Incremental Sqoop is a technique used to import only the new or updated records from a source database into a Hadoop ecosystem, rather than importing the entire data every time. This approach reduces the time and resources required for importing data, making the process more efficient and faster.

There are two types of incremental Sqoop imports:

1. Appendmode:Thismodeimportsonlythenewrecordsaddedtothesource database after the last import. The import command appends the new data to the existing data in the target Hadoop ecosystem.
2. Lastmodifiedmode:Thismodeimportstherecordsthathavebeenupdatedor modified in the source database since the last import. The import command compares the last modified timestamp of the records in the source database with the timestamp of the previously imported records, and imports only the modified records.

To perform incremental Sqoop, you need to use the --incremental option with either append or lastmodified mode, depending on your requirements.

For example, to perform an incremental import in append mode, you can use the following command:

```
sqoop import --connect jdbc:mysql://localhost/db_name --username user_name
    --password password --table table_name --incremental append --check-column
    column_name --last-value last_imported_value --target-dir
    /path/to/target/directory
```

In this command, --incremental specifies that the import is incremental, --check-column specifies the column to use for checking for new records, and --last-value specifies the value of the last imported record. The --target-dir option specifies the directory in which the data will be stored in Hadoop.

Similarly, you can use --incremental lastmodified mode to perform an incremental import based on the last modified timestamp of the records.

-

114) why we use HBASE and how it store the data ?

- HbaseisacolumnorienteddistributedNOSQLDBthatisdesignedtohandle large amount of structured data .it is a part of hadoop ecosystem and runs on top of HDFS
- HBASEisusedforstoringandmanaginglarge,sparsedatasetsthatare distributed across a cluster of commodity servers .it is often used in scenarios where you need random ,realtime R/W access to the large amount of data ,such as OLTP or real time analytics application
- HBASEstoresdataintables,whicharecomposedofrowsandcolumns.each rows has a unique row key and each column can contain multiple versions of the data each with its own timestamp .hbase tables re sorted by row key and data can be accessed by row key or scanning over a range of row keys
- HBaseisdesignedtoprovidefast,randomreadandwriteaccesstolarge amounts of data, even in the face of hardware failures. It achieves this by storing multiple copies of the data across a cluster of servers, and by using a distributed architecture that allows data to be processed in parallel. HBase also supports automatic sharding and rebalancing of data across the cluster, making it easy to scale out as your data grows.
-

115) sqoop

116) HBASE

117) HIVE

118)SPARK

119) OPTIMIZATION TECHNIQUES IN SPARK

120)Differecne between managed table and external table ,why we create both table ?what is the location ?,where the data get stored ?

121) how many types of data we can sqoop ?

Sqoop supports a wide range of data sources, including:

1. RelationaldatabasessuchasMySQL,Oracle,SQLServer,PostgreSQL,etc.
2. MainframesourcessuchasIBMDB2andVSAM.
3. Cloud-baseddatasourcessuchasAmazonS3andAzureBlobStorage.
4. NoSQLdatabasessuchasHBaseandMongoDB.

122)sort bykey clusterby key,orderbykey ,groupby key,distributedby key with examples ?

123)what is boundary query in sqoop ?

- Boundaryqueryisafeatureofsqoopthatallowsyoutospecifytherangeofdata to be imported from a database based on a specific column range value .it is particularly useful when you have a large table with millions of rows and you only want to import a specific range of rows

Sqoop import –connect “ ” –username “ ” –password “ “

–table mytable –boundry-query “select min(id) ,max(id) from mytable”

–split by id –target -dir

124)sqoop eval command why we use ?

- hesqoop evalcommandisusedtoexecuteaSQLstatementagainstadatabase and display the results without importing or exporting any data. It can be useful for checking the connectivity to a database, verifying that the SQL statement produces the expected output, and testing complex queries before running them through a full import or export job. The sqoop eval command is especially useful for troubleshooting issues related to connectivity or authorization to a database, and for validating the correctness of SQL statements.

125)how we can decide number of bucketing ?

- Sizeofthedata:Thesizeofthedataisoneoftheimportantfactorsindeciding the number of buckets. If the size of the data is large, then we can increase the number of buckets to improve performance.
- Availableresources:TheavailableresourceslikememoryandCPUplayacrucial role in deciding the number of buckets. If we have more resources available, then we can increase the number of buckets to improve performance.
- Querypatterns:QuerypatternsalsoplayaroleindecidingthenumberofbucketsIf the queries are typically focused on a subset of columns or a particular range of values, then we can use a higher number of buckets.
- Skewnessofdata:Ifthedataisskewed,thenweneedtoconsiderahigher number of buckets to balance the data distribution.
- Hardwarespecifications:Hardwarespecificationssuchasdiskspeed,network bandwidth, and processor speed also affect the number of buckets. If the hardware is high-end, we can use a higher number of buckets.

126) is it possible to do bucketing and partitioning on the same column ?

- Yes,itispossibletodobucketingandpartitioningonthesamecolumninHive. However, it is generally not recommended to do so because it can cause unnecessary overhead and might not provide any additional benefits.
- Bucketingismainlyusedforsampling,whilepartitioningisusedfordividingdata into logical units for easy retrieval and processing. If you partition and bucket on the same column, it would essentially create subdirectories within each partition, and each subdirectory would contain a number of buckets.
- ThiscanleadtoalotofsmallfilesanddirectoriesWhichcanimpactquery performance and slow down the processing. Therefore, it is generally recommended to either partition or bucket the data based on your use case, rather than doing both on the same column.

127) major issue faced in spark development ?

- Performanceissues:Sparkjobscanbecomesloworunresponsivedueto inefficient code, resource allocation, or data skew.
- Memorymanagement:Sparkisamemory-intensiveapplicationandit'simportant to optimize memory usage in order to avoid out of memory errors.
- Dataformatcompatibility:Sparksupportsawidevarietyofdataformats,butnot all formats are compatible with all data sources. This can cause issues when trying to load or process data.
- Debugging:DebuggingSparkjobscanbechallenging,aserrorscanoccurin different parts of the job, and the logs can be difficult to interpret.
- Resourcecontention:WhenrunningmultipleSparkjobsonacluster,resource contention can become an issue, with jobs competing for CPU, memory, and other resources.
- Dataskew:Whendataisnotevenlydistributedacrosspartitions,sometasks may take longer than others to complete, leading to performance issues and slower job completion times.

128)how to load data in hive table ?

- Createtable
- Preparedata
- Loaddata: LOADDATAINPATH“path”

Or we can do same through directly sqoop

129) map vs map partition ?

- Botharetransformation
- Themaptransformationappliesagivenfunctiontoeachelementoftheand returns a new RDD consisting of transformed elements .the function is applied to each element of RDD independently ,so it can’t use information from other element in the RDD
- Mappartitionstransformationappliesagivenfunctiontoeachpartitionof and returns a new RDD consisting of the transformed partitions
- Thefunctiontakeiteratorthatcontainsthetransformedelementwithinthat partition .this allows the function to use information from other elements within the same partition ,which can lead to performance gain
- So,themaindifferencebetweenmapandmapPartitionsisthegranularityat which the function operates. While map applies the function to each individual element, mapPartitions applies the function to entire partitions of the RDD at

once. This can make mapPartitions more efficient in certain cases, especially when the function has high overhead, such as when it involves I/O operations or establishing database connections. However, it also means that mapPartitions requires more memory than map, since it must process larger amounts of data at once.

130) if we have some header information in a file how to read from it and how to convert it to a dataset or dataframe

131) how to increase the performance of sqoop ?

- Increasethenumberofmapper:bydefaultsqoopusesonlyonemapperto import data from DBs .increasing the number of mappers can speed up the import process. This can be done using **-m** option
- Usethe"**--direct**"option:the"--direct"optionenablesdirectdatatransfer between the DB and hadoop ,bypassing the traditional import process .this can significantly improve import performance
- Usethe""**--fetch-size""option:**thefetchsizeoptioncontrolsthenumberof rows fetched in each database read operation ,increasing this value can reduce the number of round trips to the DB ,improving performance .
- Usecompression:
- Use"**--hive-import**"option:ifyouareimportingdatatohiveusingthe –hive-import option can improve performance by bypassing the intermediate HDFS storage step
- Use"**--batch**":–batchoptionenablesbatchingofrowsduringimportreducing the number of round trips to the database

132) while sqooping some data loss,how to handle such things ?

- Whenwesqoopdatafromasourcetotarget,thereisapossibilitythatsomedata may lost due to various reasons such as network failure ,disk failure or any other reasons to handle such situation we can follow these approaches.
- Incrementalimports :sqoopprovidetheoptionofperformingincremental imports where we can specify a boundary for rows to be imported by specifying a boundary .we can ensure that only new or updated rows are imported and the old rows are not imported
- Validation:wecanvalidatethedataimportedbycomparingitwiththesouce data ,we can use tools like apache nifi or nifi registry for validation .we can also validate the data using SQL or python script

- Backup:wecantakebackupofthedatabeforeimportingincaseofdatalosswe can restore the data from the backup
- Retrymechanism:
- Monitoring :Wecanmonitorthesqoopimportprocessusingtoolssuchas Apache Ambari, Cloudera Manager, or Hortonworks Data Platform. We can monitor the import status, import rate, and other important metrics to detect any issues early.

133) what happens when sqoop fails in between the large data import job ?

If Sqoop fails in between a large data import job, then the data transfer process will stop, and it will not import any more data. In such cases, we need to handle the failures and resume the import process from where it left off. There are different ways to handle this scenario:

1. Usethe--incrementaloption:Sqoopprovidesthe--incrementaloptionto import data incrementally. This option is useful when the data is already imported once, and we want to import only the new data. By specifying the --incremental option with the appropriate mode, we can tell Sqoop to import only the new or updated data since the last import.
2. Usethe--last-valueoption:The--last-valueoptionisusedinconjunction with the --incremental option. It specifies the maximum value of the column that was imported in the previous import. This option is useful when we want to import the new data that is added to the database after the last import.
3. Use the --append option: The --append option tells Sqoop to append the data to an existing table instead of overwriting it. This option is useful when we want to resume the import process from where it left off.
4. Usetheretrymechanism:Wecanusetheretrymechanismtohandlethefailures during the import process. We can specify the number of retries and the interval between retries to resume the import process from where it left off. We can use shell scripts, Oozie, or other job schedulers to automate this process.

134) components of hadoop and their services ?

135)what are the important configuration files in hadoop ?

- core-site.xml:ThisfilecontainscoreHadoopconfigurationpropertiessuchas the default filesystem name, Hadoop cluster name, and I/O settings.
- hdfs-site.xml:ThisfilecontainsHadoopDistributedFileSystem(HDFS) configuration properties such as the number of replicas to be created for each block, the data directory, and HDFS permissions.
- mapred-site.xml:ThisfilecontainsApacheMapReduceconfigurationproperties such as the job tracker and task tracker settings, the maximum number of concurrent tasks to be run, and the map and reduce task memory settings.
- yarn-site.xml:ThisfilecontainsYetAnotherResourceNegotiator(YARN) configuration properties such as the resource manager settings, the node manager settings, and the maximum number of application attempts to be made.
- hadoop-env.sh:Thisfilecontainsenvironmentvariablesandsettingsthatare used by various Hadoop components.
- log4j.properties:Thisfilecontainssettingsforlogginganddebugging messages generated by Hadoop components.

These configuration files are typically located in the Hadoop installation directory under the etc/hadoop subdirectory. By modifying these configuration files, you can customize the behavior of Hadoop components to better suit your specific requirements.

136)fault tolerance in case of data node ?

- Replication:hadoopstoresmultiplereplicasofeachdatablockondifferentdata nodes to provide fault tolerance .the default is 3 .which means that each data block is stored on 3 different data nodes . if the data node fails the block stored on that node are automatically replicated to other data node to maintain the desired replication factor
- Heartbeatmechanism: eachdatanodesendsaperiodicheartbeatmessageto the name node to indicate that it is still alive and functioning properly . if the name node does not receive the heartbeat message form the data node for a specific periodic period of time .it marks the data node as failed and initiates the process of replicating its data blocks on the other data node.
- Blockscanning andverification :hadoopperiodicallyscanstheconsistencyof data blocks on different data nodes .if inconsistencies are detected ,the data blocks are either repaired or deleted and replaced with correct copies.
- Rebalancing :hadoopperiodicallyrebalancesthedatablocksacrossthedata nodes to ensure that each data node has approximately the same amount of data .this helps prevent individual data nodes from becoming overloaded and potentially failing .

137)fault tolerance in case of name node ?

- Checkpointing :thenamenodecreatesperiodiccheckpointsofthefilesystem metadata and stores them in directory called the **secondary name node** .in the event of a namenode failure the secondary namenode can be used to restore the file system metadata and bring up a new name node
- Journaling:thenamenoderecordsallthechangestothefilesystemmetadata in a **write ahead log called editlog.** The editlog is stored on the local file system of the namenode and is periodically synchronized with remote storage location for durability in the event of a name node failure .the most recent version of the editlog can be used to restore a file system metadata and bring up a new name node
- HA:hadoopprovidesHAfeaturethatallowsforautomaticfailoverbetweentwo or more namenode in a cluster .with HA enables .one namenode acts as the active node while the other are in standby mode . if the active namenode fails .one of the standby nodes is automatically promoted to become the active node .ensuring the file system remain available

138)what is rack awareness ?

- Itisafeatureinhadoopthathelpsoptimizetheperformanceandreliabilityof data processing in a hadoop cluster by taking into account the physical topology of the network on which the cluster deployed
- Inahadoopcluster,nodesaretypicallyorganizedintorackswhicharephysical groupings of machines within a data center .by default hadoop assumes the all nodes are equally likely to fail and therefore assign data blocks randomly to nodes in the cluster .however this can result in data blocks being stored on multiple nodes within the same rack. Which can create spof if entire rack fails
- Rackawarenesshelpstomitigatethisriskbyensurethatdataisdistributed across racks as well as nodes .with rack awareness enabled .hadoop will attempt to store replicas of a data blocks on a node in different rack thereby reducing the risk of data loss in the event of a rack failure.
- Rackawarenesscanalsohelpstoimproveperformancebyminimizingnetwork traffic between node in different racks

139) problems with having a lot of small files in HFS ?

- Namenodememoryoverhead:eachfileanddirectoryinhdfsisrepresentedas an inode in the name node memory .having a large number of small files can

quickly fill up the namenode's memory and cause it to run out of heap space .leading to performance degradation and even cluster failure

- Namenodescalability: sincethenamenodemanagesthemetadataforallthe files in the cluster .having large number of small files can reduce the namenode scalability as the number of file and directories increase the namenode performance can suffer leading to slower processing and increased latency
- Datalocality:oneofthekeybenefitofHDFSisitsabilitytostoredataina distributed manner across the cluster .however when there are a lot of small files the data is spread across many blocks .which can reduce data locality and increase network traffic between nodes
- Mapreducejobperformance; MRjobcanbeslowwhenprocessingsmallfiles because each file require its own map task which can be slow to start and consume a lot of overhead

140) how you can overcome small file problem ?

- sequenceFile: hadoopprovidesabuiltinsequencefileformatthatallowsyouto store multiple small files into a single large file .which can be split into multiple blocks .this can help to reduce the overhead associated with managing small files
- CombineFileInputFormat:itallowsyoutogroupsmallfiletogetherintolarger splits to reduce the number of input splits of MR jobs
- HADOOPARCHIVERS(HAR) :itisanotherbuiltinfeatureofhadoopthatallows you to pack large number of small files into a single archive file that can be stored on hdfs
- Flume:isadistributedreliableandavailableservicethatcanefficientlycollect aggregate and move large amount log data from different sources to a centralized data store like HDFS .flume can be used to collect small files and write them to HDFS as larger file
- ApacheHudi:ApacheHudiisadatamanagementframeworkthatcanbeusedto efficiently ingest and manage large amounts of data into Hadoop's distributed file system (HDFS). Hudi can help to manage small files by creating partitioned and compacted datasets.
- UseaDifferentStorageSystem:Iftheprimaryusecaseinvolveshandlingalarge number of small files, then it may be worthwhile to consider using a different storage system that is better suited for handling small files, such as Amazon S3 or Google Cloud Storage.

141) What is a block scanner in HDFS ?

- InHDFS,ablockscannerisabackgroundprocessthatverifiestheintegrityofthe data stored in HDFS. It scans the data blocks in the file system and checks for any potential data corruptions or errors. The block scanner is responsible for identifying and reporting any such errors so that they can be fixed.
- Theblockscannerworksbyperiodicallyscanningtheblocksofdatastoredon each data node. It verifies the checksum of the data blocks and compares them with the original checksums generated when the blocks were written to HDFS. If the checksums do not match, the block scanner identifies the corrupted blocks and alerts the NameNode. The NameNode can then take corrective measures such as replicating the data block to a new location, or triggering a repair process to fix the corrupted blocks.
- Overall,theblockscannerplaysanimportantroleinensuringthereliabilityand data integrity of HDFS.

142) what do you mean by HA of name node and how it is achieved ?

143) significance of counters in MR ?

- CounteraretypeofmechanisminMRusedfortrackingandreportingthe progress of a job .they are used to monitor the execution of a MR job and provide feedback on its status .such as the number of records processed or the number of errors encountered
- Counterscandefinedandupdatedwithinthemapperandreducerfunctionsofa MR job and their values can be aggregated across multiple nodes in the cluster they are especially useful for debugging and performance tuning as they allow developers to identify and address issues with their code .such as unbalanced workload distribution or inefficient processing
- ThereareseveraltypesofcountersinMapReduce,includingbuilt-incounterslike the number of input records or output records, as well as custom counters that can be defined by the developer to track specific metrics or events during the execution of the job. Overall, counters provide a powerful tool for monitoring and optimizing the performance of MapReduce jobs.

143) why the output of Map task are spilled to local disk and not in HDFS ?

The output of the map task is spilled to the local disk instead of HDFS for efficiency reasons. This is because HDFS is designed for large file operations, and writing small

amounts of data to HDFS can be relatively slow. In addition, writing to local disk allows for faster access and processing of intermediate data within a single node, rather than having to transfer it over the network to HDFS and then back again for further processing. By spilling data to local disk, it allows the map task to free up memory and continue processing more data without being limited by the amount of available memory on the node. Once the map task is completed, the spilled data can then be read back into memory and merged together before being passed on to the reduce task for further processing.

144)define speculative execution ?

Speculative execution is a feature in hadoop MR where redundant copies of a task are executed on different nodes in the cluster to increase job performance and ensure timely completion .when a task is running slowly on a particular node .speculative execution enables another copy of the same task to start running on a different node and then the first task to complete is used while the other copy is terminated .this helps delayed in job completion due to slow or failed nodes and ensure that job completes in a timely manner ,however it may lead to extra resource consumption in the cluster so it needs to be used with caution

145)is it legal to set the number of reducer tasks to zero ?

- SettingthenumberofreducertaskstozeroislegalinHadoop.Whenthenumber of reducer tasks is set to zero, the output of the map task is stored in the HDFS and no reduce task is executed. This is useful when we want to sort the data or perform some other operation that only requires the map phase. However, it should be noted that setting the number of reducer tasks to zero may not always be the most efficient way to process the data, as it may lead to larger intermediate data that needs to be sorted and spilled to disk. It is recommended to experiment with different numbers of reducers to find the optimal value for a given job.

146)where does the data of hive table gets stored ?

- ThedataofaHivetableisstoredintheHadoopDistributedFileSystem(HDFS) or in any other storage system that is compatible with Hadoop. By default, Hive

tables are stored as files in HDFS, but it is possible to configure Hive to use other storage systems such as Amazon S3, Microsoft Azure Data Lake Storage, or Apache HBase.

- When a Hive table is created, it is mapped to a directory in HDFS. The data files associated with the table are stored in this directory. Each file represents a partition of the table, and the files are stored in a subdirectory that corresponds to the partition. Hive also maintains metadata about the table, such as the schema and the location of the data files, in a metadata store such as Apache Derby, MySQL, or PostgreSQL.
- /user/hive/warehouse

147)why hdfs is not used by hive metastore for storage ?

- HiveMetastoredoesnotuseHDFSforstoragebecauseHDFSisdesignedfor large-scale distributed data storage and processing, whereas the metastore needs to store metadata about Hive tables, partitions, and other database objects in a structured way that can be queried efficiently. Using HDFS for metastore storage would result in unnecessary complexity and reduced performance, as the metastore requires frequent read and write operations, and HDFS is optimized for large-scale sequential I/O operations rather than random I/O operations. Instead, Hive Metastore typically uses an RDBMS (Relational Database Management System) such as MySQL or PostgreSQL for storing metadata.

149)when should we use order by and sort by ?

- Bothareusedtosortthedatabasedononeormorecolumns
- **Orderby**isusedtosorttheentiredatasetbasedonthespecifiedcolumns.itis used when you want to sort the data in a single reducer
- **Sortby**isusedwhentosortthedataonlywithineachreducer.itisusedwhen you want to sort the data within each reducer and you do not need globally sorted output
- Orderbycanbeveryexpensiveintermsofperformanceespeciallyifyouhavea large dataset ,since it requires shuffling all the data to a single reducer .if possible use sort by instead of order by

150)what do you mean by data locality ?

151)installation modes in hadoop ?

- Localorstandalone:runsonsinglemachine
- Pseudo-distributedmode :singlemachinebuteachhadoopdaemonrunsina separate java process
- Fullydistributedmode:

152) What is the role of a combiner in hadoop ?

- ThecombinerisanoptionaloptimizationfunctioninHadoopthatisusedto perform local aggregation of intermediate key-value pairs on a MapReduce node before they are sent to the reducer. The main role of the combiner is to reduce the amount of data that is transferred over the network between the MapReduce nodes, which can significantly reduce the processing time and network bandwidth consumption.
- Thecombinerfunctionreceivestheintermediatekey-valuepairsoutputtedbythe mapper function and performs a local aggregation operation. The output of the combiner function is a set of key-value pairs, which are then passed to the reducer for further processing. The combiner function is typically the same as the reducer function, although it is not guaranteed to execute on every node or in any specific order.
- UsingacombinerfunctioncanimprovetheperformanceofMapReducejobsby reducing the amount of data shuffled across the network, which can help to speed up the processing of large data sets. However, not all MapReduce jobs are suitable for a combiner, as some types of data processing may require all intermediate key-value pairs to be processed by the reducer.

153) what is the role of partitioner in hadoop ?

- InHadoop,thePartitionerisresponsibleforassigningtheoutputofamapperto a reducer. It takes the key of the intermediate key-value pair generated by the mapper as input and returns the index of the reducer that will handle that key.
- TheroleofthePartitioneristoensurethatallthekeyswiththesamehashvalue go to the same reducer. This helps to ensure that all the values associated with a particular key are processed together by the same reducer, which can greatly reduce the amount of data that needs to be shuffled across the network.
- Bydefault,Hadoopusesthehashvalueofthekeytodeterminethepartition,but users can provide their own partitioner implementation by implementing the Partitioner interface. This allows users to customize the partitioning scheme based on the specific characteristics of their data and application.

154) when should we use HBASE?

- HBASEisadistributednon-relationalDBthatcanhandlelargeamountof structured data .it is built on top of Hadoop and provides low latency access
- Largeamountsofstructureddataneedtobestoredandprocessed
- FastrandomR/Waccesstodataisrequired
- DataneedstobedistributedacrossaclusterofcommodityHWforfault tolerance and scalability
- Highwritethroughputisrequired
- Complexqueriesarenotneeded,ashbaseonlysupportsk-vlookupsandrange scans

155) difference between RDBMS and NoSQL

156) what is column family ?

- InApacheHBase,acolumnfamilyisalogicalgroupingofcolumnsthatsharethe same characteristics and access control. It is a collection of related data stored together, consisting of one or more columns. Each column family has a name and every column in HBase must belong to a column family. Column families are defined when creating an HBase table and cannot be modified later.
- Forexample,ifwehaveatabletostoreemployeedata,wemightdefinetwo column families - one for the employee information like name, age, salary, and another for the employee's address information like street, city, state. The benefit of grouping related columns in a column family is that they can be efficiently accessed and updated together. Additionally, in HBase, column families are stored together physically on disk, allowing for more efficient disk I/O operations

157) difference between hive and hbase ?

- HiveisadatawarehousingsystemforstructureddataWhileHBaseisaNoSQL database for unstructured or semi-structured data.
- HiveisoptimizedforRead-heavyworkloads,whileHBaseisoptimizedfor write-heavy workloads.
- HiveprovidesaSQL-likeinterfaceforqueryingdata,whileHBaseprovidesa programmatic interface (using Java APIs) for accessing data.
- HivesupportsbatchprocessingoflargedatasetsWhileHBaseprovidesreal-time access to data.
- HivestoresdatainatabularformatWhileHBasestoresdatainacolumnar format.

158) how do reducers communicate with each other ?

- Reducerscommunicatewitheachotherthroughtheprocesscalled“shuffled”.in this process the output of the map task is partitioned andsorted and then transferred to the reducers .each reducer receives one or more partitions of the data ,and the data is sorted by key within each partition
- ThecommunicationbetweenreducersismanagedbytheJobTracker,which assigns tasks to nodes and tracks their progress. Once all the reducers have completed their tasks, the JobTracker collects the results and presents them to the user.

159) what do you know about sequencefileinputformat ?

- InHadoop,SequenceFileInputFormatisafileinputformatthatisusedtoread sequence files. Sequence files are binary files that store key-value pairs, which can be of different types. The SequenceFileInputFormat reads these files and creates a key-value pair as an input record for each record in the file.
- TheSequenceFileInputFormatisusedinMapReduceprogramstoreaddata from sequence files. It provides a simple API for reading key-value pairs from sequence files. When a MapReduce job is run with the SequenceFileInputFormat, the RecordReader created by the input format reads each record from the sequence file and creates a key-value pair that is passed as input to the mapper function.
- TheSequenceFileInputFormatiswidelyusedinHadoopapplicationstostore and process large amounts of data efficiently. It is particularly useful when the data is in a binary format and needs to be processed as key-value pairs.

160) what is block report ?

- InHDFS,blockreportisaregularreportsentbytheDNtoNNwhichcontainsa list of all the blocks currently stored on the DN .the purpose of block report is to inform the NN about the current state of the DN’s block so that the NN can keep track of the blocks and their locations.block report are sent periodically or on-demand basis by the DN .once the NN receives a block report ,it updates its block maps and location info and uses this info to efficiently schedule the data processing tasks

161) what is a distributed cache?

- In Hadoop, the Distributed Cache is a mechanism for caching files and archives required by the MapReduce job, so they are available to all the nodes in the cluster. It is a facility

provided by Hadoop to cache files (text, archives, jars, etc.) needed by applications. The distributed cache is used to make these files available on every node in the Hadoop cluster.

- The Distributed Cache copies the necessary files from the client machine to the Hadoop cluster, and then the files are made available to the mapper or reducer tasks. This helps to avoid the need to send the files over the network for every task and hence reduces network traffic. The Distributed Cache is managed by the JobTracker and accessed by the TaskTrackers. It can be used for sharing data, configuration, or executable files.

162) what factors we should consider before defining the block size ?

There are several factors to consider when defining the block size in Hadoop:

1. Sizeofdata:Thesizeofthedatabeingstoredcanimpacttheblocksize.Smaller blocks are more suitable for smaller files while larger blocks are better for larger files.
2. Disk size: The block size should be chosen based on the size of the disks on the nodes in the cluster. Larger disks can handle larger block sizes.
3. Networkbandwidth:Thenetworkbandwidthavailablebetweennodescanalso be a factor in selecting the block size. Smaller block sizes can reduce the impact of network latency and congestion.
4. Memory:Theblocksizeshouldalsobechosenbasedontheamountofmemory available on the nodes in the cluster. Larger blocks may require more memory to process.
5. Performance:Theblocksizecanhaveanimpactontheperformanceofthe cluster. In general, larger block sizes can improve performance by reducing the overhead of managing small files.

Overall, the block size should be chosen based on a balance between these factors to ensure optimal performance and storage efficiency for the cluster.

163)what is the purpose of Record reader ?

- The recordreader is a hadoop inputformat API class responsible for reading input data and producing a sequence of k-v pairs .it reads data from inputsplit,which represents a logical unit of work in hadoop.the record reader reads the data of the split and then convert it into k-v pairs that are passed to the mapper for processing

164)NORMALIZATION VS DENORMALIZATION ?

- Normalizationanddenormalizationaretwodifferenttechniquesusedin database design to organize and structure data.
- Normalizationistheprocessofstructuringadatabaseinsuchawaythatit minimizes data redundancy and avoids data inconsistencies. It involves breaking down a large table into smaller, more specific tables and creating relationships between them. The goal of normalization is to reduce data redundancy and improve data integrity. Normalization also makes it easier to modify the database schema without affecting the entire database.
- Denormalization,ontheotherhand,istheprocessofintentionallyadding redundant data to a database in order to improve data retrieval performance. It involves combining multiple tables into a single table or duplicating data in multiple tables. The goal of denormalization is to improve query performance by reducing the number of joins required to retrieve data.
- Insummary,normalizationisusedtoreducedataredundancyandimprovedata integrity, while denormalization is used to improve query performance. The choice between normalization and denormalization depends on the specific needs and requirements of the application or system

165) what are the 4 dimensions in HBASE?

- Row key : this dimension identifies a unique row in a table and used to access data stored in a row .row key are sorted lexicographically and can be used for range scans
- Column family : this dimension groups related data together and provides a means to apply the same attributes to a group of columns .all column with a column family share the same prefix.
- Column qualifier : this dimensions represents an individual cell within a row and identifies a specific column within a column family .column qualifiers are appended to the end of the column family prefix
- Time stamp : this dimension represents the version of a cell .each cell can have multiple versions ,each with its own time stamp .the most recent version is the default version that is returned when a query is performed

166) what is partition,shuffle and sort phase ?

167) how can you optimize the MR jobs ?

- Input/output formats: Choosing the appropriate input/output format can improve the performance of a job. For example, using SequenceFileInputFormat or AvroInputFormat can improve the performance of jobs that read large numbers of small files.
- Combiners: Combiners are a type of local reducer that can be used to aggregate the intermediate key-value pairs before sending them over the network. This reduces the amount of data that needs to be transmitted, and can improve the performance of the job.

- Partitioning: Partitioning the data can help to balance the load across the reducers and improve the overall performance of the job.
- Caching: Caching frequently accessed data in memory can improve the performance of the job. For example, if a job needs to perform a lookup on a large table, it may be faster to cache the table in memory rather than doing a disk read for each lookup.
- Tuning JVM settings: Tuning the Java Virtual Machine (JVM) settings can help to improve the performance of MapReduce jobs. For example, increasing the heap size can reduce the number of garbage collections and improve the overall performance.
- Compression: Compressing the input/output data can help to reduce the amount of disk I/O and improve the performance of the job.
- Speculative execution: Enabling speculative execution can help to improve the overall performance of the job by running multiple copies of the same task in parallel and choosing the fastest one.
- Task parallelism: Splitting the input data into smaller chunks and processing them in parallel can improve the performance of the job. This can be achieved using techniques such as map-side joins or multi-stage map-reduce.
- Avoiding unnecessary sorting: Sorting the data can be an expensive operation. If the output of the mapper is already sorted, it may be unnecessary to sort it again in the reducer.
- Avoiding unnecessary data shuffling: Data shuffling can also be an expensive operation. If the data does not need to be shuffled, it may be possible to avoid the shuffle step altogether.

168) what are the advantages of combiner ?

- Reduced network traffic: Combiners are used to aggregate intermediate values generated by the Mapper before they are sent to the Reducer. This reduces the amount of data that needs to be transferred over the network, which in turn reduces network congestion and speeds up the processing time.
- Reduced disk I/O: By aggregating the intermediate results in memory, the combiner reduces the amount of data that needs to be written to disk. This not only speeds up the processing time but also reduces the load on the Hadoop cluster.
- Improved performance: Combiners can significantly improve the performance of MapReduce jobs by reducing the amount of data that needs to be processed by the Reducer. This can result in faster processing times and reduced resource utilization.
- Better resource utilization: By reducing the amount of data that needs to be transferred over the network and written to disk, combiners can help improve the overall resource utilization of the Hadoop cluster, allowing it to handle more jobs concurrently.

169) can we always use a combiner ?

- No,wecannotalwaysuseacombiner.Althoughacombinercanimprovethe performance of MapReduce jobs, it is not always safe to use. The combiner function must satisfy certain conditions to be used safely:
- Thecombinerfunctionmustbecommutativeandassociative,whichmeansit can be applied in any order and still produce the same result.

- The input and output types of the combiner function must match the input and output types of the reducer function. The combiner function must not change the
- number of output records from the mapper. If any of these conditions are not met, the use of a combiner can lead to incorrect results or even job failures. Therefore,
- it is important to carefully test and validate the use of combiners in MapReduce jobs.

170) different schedulers in YARN?

- YARN provides several schedulers to manage resource allocation and scheduling of jobs on a hadoop cluster .
- FIFO scheduler : this is a simple scheduler that assigns resource to job in the order in which they are submitted .it ha no notion of priorities and it not suitable for large clusters .
- Capacity Scheduler : this scheduler allows sharing of a large cluster across multiple organization or users .it supports the concept of queues ,where each queue can have its own set of policies for resource allocation ,scheduling and preemption
- Fair scheduler : this scheduler is designed to allocate resources fairly among a set of jobs. It uses a concept of a fair shares to allocate resources to jobs ,so that each job gets its fair share of resources over time .
- Delay scheduler : this scheduler allows user to submit jobs that are scheduled to run at a specific time in the future .it is useful for batch jobs that do not need to run immediately

- Capacity scheduler with preemption : this is an extension of the capacity scheduler that supports preemption of resources from jobs that are not using them efficiently and allocating them to jobs that need them more .

171) hive metastore vs warehouse ?

- InthecontextofApacheHive,themetastoreandwarehousearetwoimportant components:
- HiveMetastore:TheHivemetastoreisacentralizedrepositorythatstores metadata information about Hive tables, such as schema, data types, column names, partitioning schemes, storage location, and more. The metastore provides a schema or table abstraction layer to Hive clients and query engines.
- HiveWarehouse:TheHivewarehouseisaHadoopDistributedFileSystem (HDFS) directory or any other supported storage system where Hive tables are physically stored as files. The warehouse contains the actual data of Hive tables and is the location where Hive queries are executed.
- Insummary,themetastorestoresthemetadatainformationaboutHivetables, while the warehouse stores the actual data of the tables.

172) hive vs beeline ?

- HiveandBeelinearebothtoolsusedtointeractwithHive,adatawarehousing solution built on top of Hadoop. Hive provides a SQL-like language, called HiveQL, to query data stored in Hadoop, while Beeline is a command-line interface to execute HiveQL queries. Here are some differences between Hive and Beeline:
- Userinterface:Hivehasawebinterfaceandacommand-lineinterface,while Beeline is only a command-line interface.

- Performance: Beeline is typically faster than Hive because it bypasses the Hive service and connects directly to the HiveServer2 service.
- Security: Beeline supports Kerberos authentication, while Hive requires additional configuration to enable Kerberos authentication.
- Features: Hive provides more features and functionality than Beeline, such as data modeling, partitioning, and more.
- Overall, Hive and Beeline serve different purposes and can be used together to interact with data stored in Hadoop.

173) temporary table in hive ?

- InHive,atemporarytableisatablethatisnotstoredpermanentlyondisk.It exists only for the duration of a Hive session and is dropped automatically when the session is closed.
- Temporarytablesareusefulwhenyouwanttoperformsomeintermediate processing or analysis in Hive and do not want to create a permanent table. They can also be used to stage data before inserting it into a permanent table.

174) what are the different ways to insert data into hive ?

- INSERT INTO SELECT: This method allows us to select data from one or more tables and
- `INSERT INTO TABLE target_table SELECT column1, column2, ... FROM source_table WHERE condition`
- INSERT INTO VALUES: This method allows us to insert values into a Hive table directly. The syntax is as follows:
- `INSERT INTO TABLE target_table VALUES (value1, value2, ...), (value1, value2, ...), ...;`

- INSERT OVERWRITE: This method allows us to overwrite the existing data in a Hive table with new data. The syntax is as follows:
- `INSERT OVERWRITE TABLE target_table SELECT column1, column2, ... FROM source_table WHERE condition;`
- LOAD DATA: This method allows us to load data from an external file into a Hive table. The syntax is as follows:
- `LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE target_table [PARTITION (partition_column=value)];`
- FROM: This method allows us to create a Hive table from the data in an external file. The syntax is as follows:
- `CREATE TABLE target_table [PARTITIONED BY (partition_column)] AS SELECT column1, column2, ... FROM source_table WHERE condition;`

175) what is UDF ,UDAF,UDTF in hive ?

- InHive,UDFstandsforUser-DefinedFunction,UDAFstandsforUser-Defined Aggregate Function, and UDTF stands for User-Defined Table-Generating Function.
- AUDFisafunctionthatcanbewritteninaprogramminglanguagesuchasJava or Python, and can be used in Hive queries to perform custom operations on data. UDFs can take in one or more input parameters, and return a single output value.
- AUDAFisafunctionthatperformsaggregationonasetofinputvalues,and returns a single output value. Examples of UDAFs include SUM, AVG, MAX, and MIN. A UDAF can be used in Hive queries to perform custom aggregation operations.
- AUDTFisafunctionthattakesinoneormoreinputvalues,andreturnsatableas output. UDTFs can be used in Hive queries to generate tables based on custom

logic. For example, a UDTF could take in a list of strings, and return a table with each string split into separate columns.

- All of these functions can be defined by the user and added to Hive using the ADD JAR command, allowing for customization and flexibility in data processing.

176) What is the purpose of view in hive ?

- InHive,aviewisavirtualtablethatdoesnotstoredatabutinsteadreferences data from other tables or views. Views are created by defining a query that specifies the data to include and exclude, and can be used to simplify complex queries, provide security by limiting access to sensitive data, and reduce redundancy by eliminating the need for repeated queries.
- Whenauserqueriesaview,thequeryisexecutedagainsttheunderlyingtables or views that the view references, and the results are returned as if they were coming from a regular table. The underlying data remains unchanged, so if the data in the tables or views changes, the data returned by the view also changes accordingly.
- Insummary,thepurposeofaviewinHiveistoprovideasimplified,secure,and efficient way to query data from one or more tables or views without having to repeat the same query logic over and over again.

177) how MR is different in spark than the traditional hadoop ?

In Spark, MapReduce is implemented differently than traditional Hadoop in several ways:

1. In-memoryprocessing:Sparkprocessesdatainmemory,whereastraditional Hadoop MapReduce reads data from disk, processes it, and writes it back to disk, which can be slower.

2. ResilientDistributedDatasets(RDDs):SparkintroducestheconceptofRDDs, which are immutable distributed collections of objects that can be processed in parallel. RDDs can be cached in memory and reused across multiple operations, which can speed up processing.
3. DAG-basedexecution:SparkusesaDirectedAcyclicGraph(DAG)tooptimizethe execution of tasks. The DAG breaks down a Spark job into smaller tasks that can be executed in parallel and optimizes their execution based on their dependencies.
4. Interactive queries: Spark allows for interactive queries and data exploration, whereas traditional Hadoop MapReduce is optimized for batch processing.
5. Built-inlibraries:Sparkcomeswithbuilt-inlibrariesformachinelearning,graph processing, and stream processing, which can be used to process data without the need to write custom MapReduce jobs.

Overall, Spark offers a faster and more flexible way to process large-scale data compared to traditional Hadoop MapReduce.

178) what is spark config ? can you configure cpu cores in sparkcontext ?

- Sparkconfigisawaytoconfigurevariousparametersforasparkapplication, such as the amount of memory to use, the number of CPU cores to allocate, etc.
- Yes,wecanconfigurethenumberofCPUcorestouseforaSparkapplicationby setting the spark.executor.cores property in the SparkConf object. We can create a SparkConf object and set the desired properties, and then pass it to the SparkContext object while creating it. Here's an example:
- 
-

- ```
  conf = SparkConf().setAppName("myApp").setMaster("local[*]").set("spark.executor.cores", "4")
  ```
- ```
  sc = SparkContext(conf=conf)
  ```

179) let's say you have a csv file of 10 columns ,the 10th column is the address column which have a comma separated values ,how would you replace the comma separated value with spaces after loading file into RDD ?

180) suppose we have a file that we do not want to be partitioned after loading in spark ? how would you achieve it ?

- If we have a file that we do not want to be partitioned after loading in Spark, we can use the coalesce() function to reduce the number of partitions to 1. This will ensure that the file is not partitioned and is processed as a single unit.

181) i load the 1GB file into memory and then apply the filter operation and then saved the file into HDFS ? how many files will be created in HDFS ?

- It depends on the number of partitions that the RDD had before saving to HDFS. By default, Spark creates one partition for each block of the input file. The block size is typically 128MB or 256MB in Hadoop. If the 1GB file had 4 blocks of 256MB each, and Spark had not repartitioned the RDD before saving, then there would be 4 files created in HDFS, one for each partition. However, if you had
- repartitioned the RDD before saving and specified only one partition using repartition(1) or coalesce(1), then only one file would be created in HDFS.

182) I am getting memory out of exception using group by key and want to resolve it but do not want to apply another transformation how would you do that ?

- If you are getting a memory out of exception while performing a groupBy operation in Spark and do not want to apply another transformation, one possible solution is to use reduceByKey instead of groupBy.
- The reduceByKey transformation combines the values of each key using an associative and commutative function, which allows for parallel execution and can significantly reduce the amount of data that needs to be shuffled across the network. In contrast, the groupBy operation groups all the values for each key into a single sequence, which can cause memory issues if the number of values for a given key is large.

183)I want to create nosql table on top of data frame how would you do that ?

We will use save method of DF API

```
From pyspar.sql import sparksession

spark=sparksession.builder.appname("save as nosql").getorcreate()

df=spark.read.csv("path")

df.write.format("org.apache.spark.sql.execution.datasources.hbase").option("table","table_name").option("zkrul","zookeeper_quorm").save()
```

In this example, we first read a CSV file into a DataFrame. Then, we use the write method to save the DataFrame as a NoSQL table. We specify the format as org.apache.spark.sql.execution.datasources.hbase, which is the format for HBase

tables. We also specify the name of the table (table_name) and the URL of the ZooKeeper quorum (zookeeper_quorum). These options may vary depending on the NoSQL database we are using.

184) what are different modes in spark ?

- Local mode: In this mode, Spark runs on a single machine, and the input data is read from the local file system. This mode is mainly used for testing and development purposes.
- Standalone mode: In this mode, Spark runs on a cluster of machines, and it has its own built-in cluster manager. In this mode, the Spark driver program runs on one of the machines in the cluster, and the worker nodes execute tasks assigned by the driver program.
- Cluster mode: In this mode, Spark runs on a cluster of machines, and it uses an external cluster manager like Apache Mesos, Hadoop YARN, or Kubernetes. In this mode, the Spark driver program runs on one of the machines in the cluster, and the worker nodes execute tasks assigned by the driver program.

185) how would you replace null values in DF with zero ?

- df.fillna(0)

186) I have to rename a column in DF to some other name ,how would you do it ?

- df.withcolumnrenamed(“oldname”,”newname”)

187) ways to create DF?

- From RDD to DF=> rdd.toDF(“c1”,”c2”..)
- From a sequence to case classes using the **createDataFrame()**
- From csv,json,parquet file using read

188)tell me syntax to create RDD from a file ?

RDD=SC.textFile(“path”)

189)if you are reading form the CSV or text File will you follow the same approach or different in spark ?

190) lets say that you are reading form a csv file and creating one DF which has two column id and name ,can you select the Row which starts with “S”

- df_s=df.filter(col(“name”).star tswith(“s”))

191)how do you decide the number of executors ? how much memory will you allocate to the driver in this case ?

The number of executors and memory allocation for the driver in a Spark application depends on several factors such as the available resources, the size of the dataset, the complexity of the transformations and actions, and the workload.

To decide the number of executors, we can consider the following factors:

1. Sizeofthedata:Ifthedataislarge,wemayneedmoreexecutorstoprocessit efficiently.
2. Available resources: We need to take into account the number of nodes in the cluster, the amount of memory and cores available on each node, and any other applications running on the cluster.
3. Complexityoftransformationsandactions:Ifthetransformationsandactions are complex, we may need more executors to handle the workload.
4. Networkbandwidth:Ifthereisahighnetworkbandwidth,wemaybeableto process more data per executor.

To determine the memory allocation for the driver, we need to consider the following factors:

1. Sizeofthedriverprogram:Ifthedriverprogramislarge,wemayneedmore memory.
2. Complexityoftheprogram:Iftheprogramiscomplex,wemayneedmore memory.
3. Amountofdatareturnedtothedriver:Iftheamountofdatareturnedtothedriver is large, we may need more memory.

Typically, we allocate 1-2 GB of memory to the driver and the remaining memory to the executors. We can also use dynamic allocation to allocate resources based on the workload.

192)lets say that there is one function where in you pass the date and you have to check the format of date as “mm-dd-yyyy” also if the user pass invalid date ,that has to be detected to ?

193)lets say that you have an array of elements and you need to remove duplicates from it ? how will you do it ? also you need to identify duplicates?

194) if you have a 10 GB data table and 100GB data table which table need to be broad casted?

195) what is a diamond problem in spark and how will you resolve it ?

- The diamond problem in Spark arises when there are multiple operations that depend on the same parent DataFrame or RDD. In such a scenario, the parent RDD or DataFrame is processed multiple times, which can lead to performance issues. To resolve the diamond problem, we can use caching or persisting to
- store the parent RDD or DataFrame in memory or on disk, depending on the size of the data. This will ensure that the parent data is processed only once and the subsequent operations can directly use the cached data. Another approach is to use the checkpoint() method, which writes the RDD to disk and removes it from
- memory. This can help in reducing the memory footprint and avoid recomputation of the parent RDD.

- In general, it is a good practice to avoid creating multiple dependencies on the same parent RDD or DataFrame to avoid the diamond problem.
-

196) what is the cluster size you had ?

197) what type of instance are used in your cluster ?

198) what is used for bigdata platform in your company (aws,azure,GCP) ?

199) what else can be the most performant for data import export other than sqoop ?

- Apart from Sqoop, some other tools that can be used for data import/export are:
- Flume: Flume is a distributed, reliable, and available system for efficiently collecting, aggregating, and moving large amounts of log data from many different sources to a centralized data store. It is often used for streaming data sources such as log files, social media feeds, and other continuous streams of data.
- Kafka: Kafka is a distributed streaming platform that is often used as a real-time data pipeline. It can be used for a variety of use cases such as data ingestion, real-time analytics, and event-driven architectures.
- NiFi: NiFi is an open-source data integration tool that provides a web-based user interface to design and manage data flows. It supports the integration of various data sources and can handle large volumes of data.
- Apache Storm: Apache Storm is a distributed real-time processing system that can be used for processing large volumes of data in real-time. It is often used for stream processing and can handle complex event processing.

- The choice of the tool depends on the specific use case and the requirements of the organization.

200)why it is needed to use HDFS for structured data ?

HDFS (Hadoop Distributed File System) is a distributed file system designed to store and manage large volumes of data across a cluster of commodity hardware. HDFS is commonly used for storing structured data because of its ability to handle large files and its fault-tolerance and scalability features.

Here are some reasons why HDFS is beneficial for structured data:

1. Scalability:HDFSisdesignedtoscalehorizontally,whichmeansthatitcan handle large volumes of data without degrading performance. This makes it a good choice for storing structured data that can grow in size over time.
2. Fault-tolerance: HDFS uses replication to ensure that data is stored redundantly across multiple nodes in the cluster. This means that if a node fails, the data can still be accessed from another node in the cluster, making it a reliable option for storing important structured data.
3. Cost-effective: HDFS is designed to run on commodity hardware, which means that it can be a cost-effective option for storing structured data compared to proprietary storage solutions.
4. IntegrationwithHadoopecosystem:HDFSispartoftheHadoopecosystem, which means that it can integrate easily with other Hadoop tools like MapReduce, Spark, and Hive. This makes it easier to perform analytics on structured data stored in HDFS.

Overall, HDFS is a good choice for storing structured data because of its scalability, fault-tolerance, cost-effectiveness, and integration with the Hadoop ecosystem.

201)which process you follow ELT OR ETL ?

202) DB related commands

203) delta lake datalake data ware house

204) Data brick architecture

205) what is uber mode in spark?

- The Uber mode is a feature in Apache Spark's YARN client mode, which allows Spark to dynamically adjust the number of executor instances based on the workload.
- In a typical Spark application running on YARN, the number of executor instances is fixed when the application is submitted. This means that if the workload varies significantly over time, the cluster may be underutilized or overutilized, which can lead to inefficient resource allocation and longer processing times.
- The Uber mode solves this problem by allowing Spark to dynamically add or remove executor instances based on the workload. This can result in better resource utilization, faster processing times, and lower costs.
- To use the Uber mode in Spark, you need to set the "spark.yarn.am.extraLibraryPath" configuration property to the directory containing the "uber-spark.jar" file. You can also set other configuration properties such as "spark.dynamicAllocation.enabled" to enable dynamic allocation of executor instances.

- It's important to note that the Uber mode is only available in Spark's YARN client mode and is not available in other deployment modes such as standalone or Mesos. Additionally, the Uber mode may not be suitable for all applications, and it's important to test and validate its effectiveness before using it in production.

206) where will you see production spark application log?

- In a production Spark application, the logs are typically generated on the worker nodes and are stored on the local file system of each worker node. These logs contain valuable information about the application's execution, such as error messages, performance metrics, and debugging information.
- By default, the log files for Spark applications are stored in the "logs" directory within the Spark home directory on each worker node. The log files are organized by role (driver or executor) and by application ID, which is assigned by the cluster manager.
- For example, in a standalone Spark cluster, the log files for the driver and each executor are stored in separate subdirectories within the "logs" directory, with names such as "driver-<application ID>" and "executor-<application ID>-<worker node hostname>.log".
- In addition to the local logs on each worker node, Spark also provides a web-based user interface called the Spark Web UI, which allows you to monitor the progress and performance of Spark applications in real-time. The Web UI displays detailed information about the application's execution, including log messages, task metrics, and stage details.
- To access the Spark Web UI, you can navigate to the URL of the Spark application, which is typically provided by the cluster manager or can be obtained

by running the "spark-submit" command with the "--verbose" option. From the Web UI, you can also download the log files for the application or specific stages or tasks

```
http://<driver-node-hostname>:4040
```

In this example, <driver-node-hostname> is the hostname or IP address of the node running the Spark driver program. The port number "4040" is the default port used by the Spark Web UI.

If the Spark application is running on a cluster, you may need to use a different hostname or IP address, depending on the network configuration of your cluster. Additionally, the port number may be different if it has been customized in the Spark configuration.

To find the correct URL for the Spark Web UI, you can check the output of the "spark-submit" command or consult the documentation for your specific cluster manager or deployment environment. You can also look for the "SparkUI" link in the output of the Spark driver program, which is printed to the console or log file when the program starts.

207) what is log4j file in spark?

-
- In Spark applications, Log4j is a logging framework used to manage and configure the logging output. It provides a flexible and configurable way to output log messages of various levels, such as INFO, DEBUG, WARN, and ERROR, to various destinations, such as console, files, and network sockets.

- The Log4j configuration file is typically named "log4j.properties" or "log4j.xml" and is located in the "conf" directory of the Spark installation or in the application's classpath.
- In a Spark application, you can use Log4j to log messages from the driver program and from the executor tasks. You can also configure Log4j to use different appenders and layouts, which define the format and destination of the log messages.

209) get idea about zookeper?

210)HDFS architecture ?

211) what are design pattern in spark

212) what is diamond problem in scala?

213)SPARK STAGES TASK JOBS

214)

MANAGERIAL ROUND QUESTION :

1)what is your cluster size

40 node cluster

2) how much data you deal with on daily basis

5-7GB

3) what is your role in your big data project ?

4) are you using on premise setup or its in cloud

5)which big data distribution are you using

- **Cloudera**
- **Hor tonworks**
- **MapR**

6) what's the most challenging thing that you faced in your big data project ?

7) whats the configuration of each node in the cluster ?

16 cpu

64 GBRAM

8) did you faced any performance challenges ,how did you optimize that ?

9)lets say we have 100gb data which we are doing a sqoop .how much time it will take?