1. What is the definition of Hive? What is the present version of Hive?

Ans-> Hive is a datawarehousing tool which runs on top of HDFS.It is used to carry out read,write and analysis of different tables.It is simple to use and uses a SQL kind of query language.

The HQL(Hive Query Language) which is similar to SQL internally converts the query to map reduce and executes the Hql query.

2. Is Hive suitable to be used for OLTP systems? Why?

Ans-> No Hive is not suitable for OLTP because hive doesnot allow frequent inserts or update.It is usually used as insert once query many times kind of scenario.Also Hive is specifically

designed to handle huge amount of data which can bes tored in distributed format and can use for Analytical processing i.e OLAP.

3. How is HIVE different from RDBMS? Does hive support ACID transactions. If not then give the proper reason.

Ans-> RDBMS	Hive
It is used to maintain database.	It is used to maintain data warehouse.
It uses SQL (Structured Query Language).	It uses HQL (Hive Query Language).
Schema is fixed in RDBMS.	Schema varies in it.
Normalized data is stored.	Normalized and de-normalized both type of data is stored.
It doesn’t support partitioning.	It supports automation partition.

older versions of Hive doesn’t support ACID transactions on tables.

Though in newer versions it supports by default ACID transactions are disabled and you need to enable it before start using it.

Below are the properties you need to enable ACID transactions.

SET hive.support.concurrency=true;

SET hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;

The follwoing are not required if you are using Hive 2.0

SET hive.enforce.bucketing=true;

SET hive.exec.dynamic.partition.mode=nostrict;

The following parameters are required for standalone hive metastore

SET hive.compactor.initiator.on=true;

SET hive.compactor.worker.threads=1

Besides this, you also need to create a Transactional table by using

TBLPROPERTIES ('transactional'='true') at the time of creating Managed table.

Managed Table should use ORC format.

4.Explain the hive architecture and the different components of a Hive

architecture?

> The major components of Hive and its interaction with the Hadoop is demonstrated in the figure below and all the components are described further:

• User Interface (UI) – As the name describes User interface provide an interface between user and hive. It enables user to submit queries and other operations to the system. Hive web UI, Hive command line, and Hive HD Insight (In windows server) are supported by the user interface.

• Hive Server – It is referred to as Apache Thrift Server. It accepts the request from different clients and provides it to Hive Driver.

• Driver – Queries of the user after the interface are received by the driver within the Hive. Concept of session handles is implemented by driver. Execution and Fetching of APIs modelled on JDBC/ODBC interfaces is provided by the user.

• Compiler – Queries are parses, semantic analysis on the different query blocks and query expression is done by the compiler. Execution plan with the help of the table in the database and partition metadata observed from the metastore are generated by the compiler eventually.

• Metastore – All the structured data or information of the different tables and partition in the warehouse containing attributes and attributes level information are stored in the metastore. Sequences or de-sequences necessary to read and write data and the corresponding HDFS files where the data is stored. Hive selects corresponding database servers to stock the schema or Metadata of databases, tables, attributes in a table, data types of databases, and HDFS mapping.

• Execution Engine – Execution of the execution plan made by the compiler is performed in the execution engine. The plan is a DAG of stages. The dependencies within the various stages of the plan is managed by execution engine as well as it executes these stages on the suitable system components.

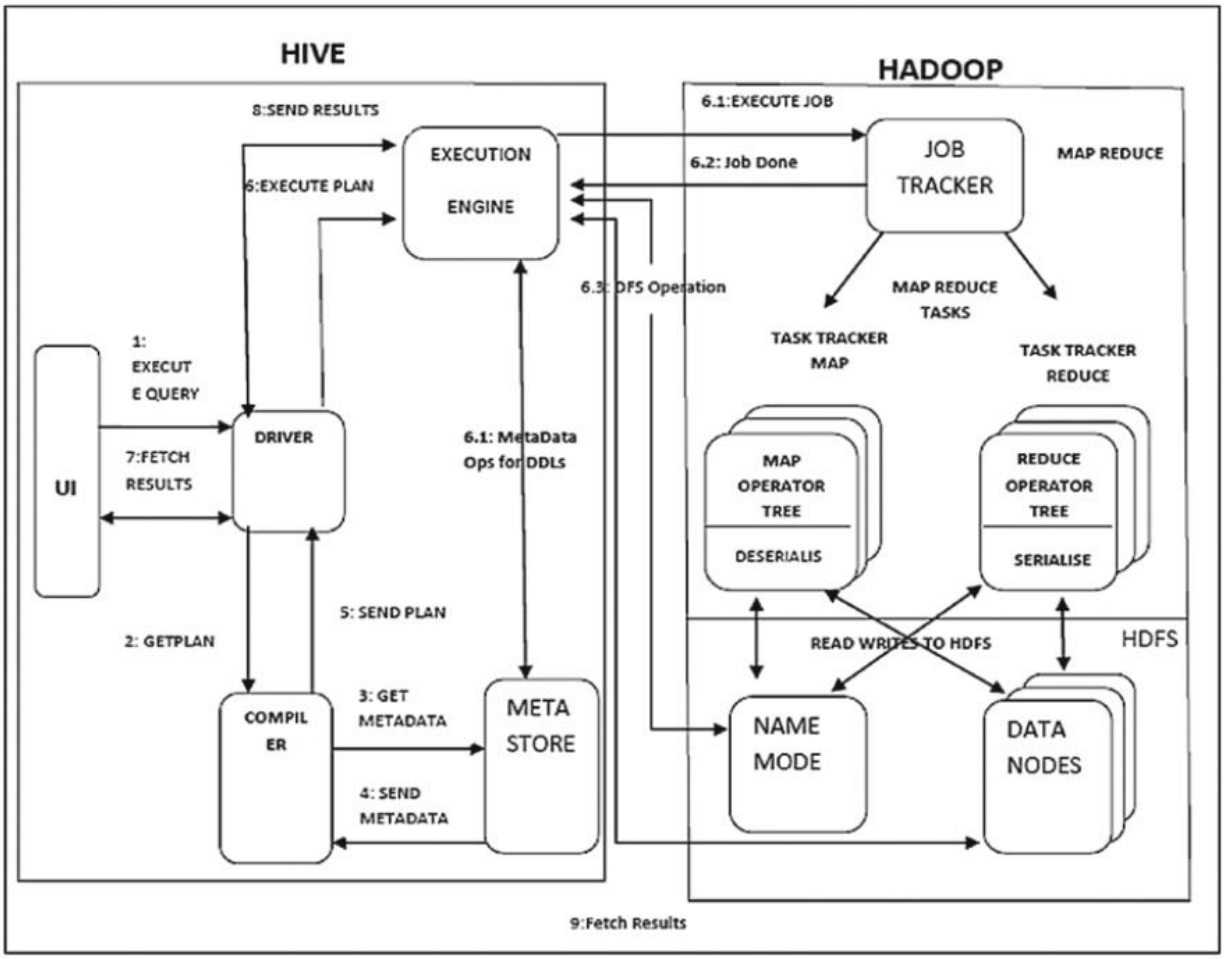


Diagram – Architecture of Hive that is built on the top of Hadoop In the above diagram along with architecture, job execution flow in Hive with Hadoop is demonstrated step by step.

• Step-1: Execute Query – Interface of the Hive such as Command Line or Web user interface delivers query to the driver to execute. In this, UI calls the execute interface to the driver such as ODBC or JDBC.

• Step-2: Get Plan – Driver designs a session handle for the query and transfer the query to the compiler to make execution plan. In other words, driver interacts with the compiler

• Step-3: Get Metadata – In this, the compiler transfers the metadata request to any database and the compiler gets the necessary metadata from the metastore.

• Step-4: Send Metadata – Metastore transfers metadata as an acknowledgment to the compiler.

• Step-5: Send Plan – Compiler communicating with driver with the execution plan made by the compiler to execute the query.

• Step-6: Execute Plan – Execute plan is sent to the execution engine by the driver.

• Execute Job

• Job Done

• Dfs operation (Metadata Operation)

• Step-7: Fetch Results – Fetching results from the driver to the user interface (UI).

• Step-8: Send Results – Result is transferred to the execution engine from the driver. Sending results to Execution engine. When the result is retrieved from data nodes to the execution engine, it returns the result to the driver and to user interface (UI).

5. Mention what Hive query processor does? And Mention what are the components of a Hive query processor?

Ans-> Hive Query processor takes the HQL query and does various operations that helps in executing the given HQL query.

The following are the main components of the Hive Query Processor:

- Parse and SemanticAnalysis (ql/parse) - This component contains the code for parsing SQL, converting it into Abstract Syntax Trees, converting the Abstract Syntax Trees into Operator Plans and finally converting the operator plans into a directed graph of tasks which are executed by Driver.java.
- Optimizer (ql/optimizer) - This component contains some simple rule based optimizations like pruning non referenced columns from table scans (column pruning) that the Hive Query Processor does while converting SQL to a series of map/reduce tasks.
- Plan Components (ql/plan) - This component contains the classes (which are called descriptors), that are used by the compiler (Parser, SemanticAnalysis and Optimizer) to pass the information to operator trees that is used by the execution code.
- MetaData Layer (ql/metadata) - This component is used by the query processor to interface with the MetaStore in order to retrieve information about tables, partitions and the columns of the table. This information is used by the compiler to compile SQL to a series of map/reduce tasks.
- Map/Reduce Execution Engine (ql/exec) - This component contains all the query operators and the framework that is used to invoke those operators from within the map/reduces tasks.
- Hadoop Record Readers, Input and Output Formatters for Hive (ql/io) - This component contains the record readers and the input, output formatters that Hive registers with a Hadoop Job.
- Sessions (ql/session) - A rudimentary session implementation for Hive.
- Type interfaces (ql/typeinfo) - This component provides all the type information for table columns that is retrieved from the MetaStore and the SerDes.
- Hive Function Framework (ql/udf) - Framework and implementation of Hive operators, Functions and Aggregate Functions. This component also contains the interfaces that a user can implement to create user defined functions.
- Tools (ql/tools) - Some simple tools provided by the query processing framework. Currently, this component contains the implementation of the lineage tool that can parse the query and show the source and destination tables of the query.

6. What are the three different modes in which we can operate Hive?

Ans->

1) Local Mode or Standalone Mode

Standalone mode is the default mode in which Hadoop run. Standalone mode is mainly used for debugging where you don't really use HDFS.
You can use input and output both as a local file system in standalone mode.

Standalone mode is usually the fastest Hadoop modes as it uses the local file system for all the input and output. Here is the summarized view of the standalone mode-

- Used for debugging purpose
- HDFS is not being used
- Uses local file system for input and output
- Default Hadoop Modes

2) Pseudo-distributed Mode

The pseudo-distribute mode is also known as a single-node cluster where both NameNode and DataNode will reside on the same machine.

In pseudo-distributed mode, all the Hadoop daemons will be running on a single node. Such configuration is mainly used while testing when we don't need to think about the resources and other users sharing the resource.

Here is the summarized view of pseudo distributed Mode-

- Single Node Hadoop deployment running on Hadoop is considered as pseudo distributed mode
- All the master & slave daemons will be running on the same node
- Mainly used for testing purpose
- Replication Factor will be ONE for Block

3) Fully-Distributed Mode (Multi-Node Cluster)

This is the production mode of Hadoop where multiple nodes will be running. Here data will be distributed across several nodes and processing will be done on each node.

Master and Slave services will be running on the separate nodes in fully-distributed Hadoop Mode.

- Production phase of Hadoop
- Separate nodes for master and slave daemons
- Data are used and distributed across multiple nodes

In the Hadoop development, each Hadoop Modes have its own benefits and drawbacks. Definitely fully distributed mode is the one for which Hadoop is mainly known for but again there is no point in engaging the resource while in testing or debugging phase. So standalone and pseudo-distributed Hadoop modes are also having their own significance.

7. Features and Limitations of Hive.

Ans->

Features:-

1)Hive is a stable batch-processing framework built on top of the Hadoop Distributed File system and can work as a data warehouse
2) Hive uses HIVE query language to query structure data which is easy to code. The 100 lines of java code we use to query a structure data can be minimized to 4 lines with HQL.
3) Hive is capable to process very large datasets of Petabytes in size
4) Since we store Hive data in HDFS so it's more fault tolerant
5) Hive supports users to access files from various locations such as Hbase,S3 or HDFS.

Limitations:-

1) Apache Hive doesn't support online transaction processing (OLTP) but Online Analytical Processing(OLAP) is supported
2) Hive does not support update and delete operation on tables
3) The latency in the apache hive query is very high
4) Hive is not used for real-time data querying since it takes a while to produce a result
5) Subqueries are not supported

8. How to create a Database in HIVE?

Ans-> Create Database is a statement used to create a database in Hive. A database in Hive is a namespace or a collection of tables.

The syntax for this statement is as follows:

CREATE DATABASE|SCHEMA [IF NOT EXISTS] Here, IF NOT EXISTS is an optional clause, which notifies the user that a database with the same name already exists.

We can use SCHEMA in place of DATABASE in this command.
The following query is executed to create a database named userdb:

hive> CREATE DATABASE [IF NOT EXISTS] userdb;
or hive> CREATE SCHEMA userdb;

The following query is used to verify a databases list:

hive> SHOW DATABASES;

9. How to create a table in HIVE?

Ans -> create table department_data
> (

> dept_id int,
> dept_name string,
> manager_id int,
> salary int)
> row format delimited
> fields terminated by ',';

10. What do you mean by describe and describe extended and describe formatted with respect to database and table?

Ans-> Describe- This will show table columns describe extended
Describe extended- This will show table columns, data types, and other details of the table. Other details will be displayed in single line. describe formatted
Descibe formatted- This will show table columns, data types, and other details of the table. Other details will be displayed into multiple lines.

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

Ans-> We can use the property “TBLPROPERTIES("skip.header.line.count"="1");” while creating a table to skip header rows in Hive.

12. What is a hive operator? What are the different types of hive operators?

Ans->Apache Hive provides various Built-in operators for data operations to be implemented on the tables present inside Apache Hive warehouse. Hive operators are used for mathematical operations on operands. It returns specific value as per the logic applied.
Types of Hive Built-in Operators
Relational Operators
Arithmetic Operators
Logical Operators
String Operators
Operators on Complex Types

13. Explain about the Hive Built-In Functions?

Ans-> Hive supports various built in functions such as:-

Return Type	Signature	Description
BIGINT	round(double a)	It returns the rounded BIGINT value of the double.
BIGINT	floor(double a)	It returns the maximum BIGINT value that is equal or less than the double.
BIGINT	ceil(double a)	It returns the minimum BIGINT value that is equal or greater than the double.
double	rand(), rand(int seed)	It returns a random number that changes from row to row.
string	concat(string A, string B,...)	It returns the string resulting from concatenating B after A.
string	substr(string A, int start)	It returns the substring of A starting from start position till the end of string A.
string	substr(string A, int start, int length)	It returns the substring of A starting from start position with the given length.
string	upper(string A)	It returns the string resulting from converting all characters of A to upper case.
string	ucase(string A)	Same as above.
string	lower(string A)	t returns the string resulting from converting all characters of B to lower case.
string	lcase(string A)	Same as above.
string	trim(string A)	It returns the string resulting from trimming spaces from both ends of A.
string	ltrim(string A)	It returns the string resulting from trimming spaces from the beginning (left hand side) of A.
string	rtrim(string A)	rtrim(string A) It returns the string resulting from trimming spaces from the end (right hand side) of A.
string	regexp_replace(string A, string B, string C)	It returns the string resulting from replacing all substrings in B that match the Java regular expression syntax with C.
int	size(Map<K.V>)	It returns the number of elements in the map type.
int	size(Array<T>)	It returns the number of elements in the array type.
	value of <type> cast(<expr> as <type>)	It converts the results of the expression expr to <type> e.g. cast('1' as BIGINT) converts the string '1' to it integral r epresentation. A NULL is returned If the conversion does not succeed.
string	from_unixtime(int unixtime)	convert the number of seconds from Unix epoch (1970-01-01 00:00:00 UTC) to a string representing the timestamp of that moment in the current system time zone in the format of "1970-01-01 00:00:00"
string	to_date(string timestamp)	It returns the date part of a timestamp string: to_date("1970-01-01 00:00:00") = "1970-01-01"
int	year(string date)	It returns the year part of a date or a timestamp string: year("1970-01-01 00:00:00") = 1970, year("1970-01-01") = 1970
int	month(string date)	It returns the month part of a date or a timestamp string: month("1970-11-01 00:00:00") = 11, month("1970-11-01") = 11
int	day(string date)	It returns the day part of a date or a timestamp string: day("1970-11-01 00:00:00") = 1, day("1970-11-01") = 1
string	get_json_object(string json_string, string path)	It extracts json object from a json string based on json path specified, and returns json string of the extracted json object. It returns NULL if the input json string is invalid.

14. Write hive DDL and DML commands.

Ans -> Hive DDL commands:

1. CREATE
2. SHOW
3. DESCRIBE
4. USE
5. DROP
6. ALTER
7. TRUNCATE

Hive DML commands:

1. LOAD
2. SELECT
3. INSERT
4. DELETE
5. UPDATE
6. EXPORT
7. IMPORT

15.Explain about SORT BY, ORDER BY, DISTRIBUTE BY and CLUSTER BY in Hive.

Ans->

• ORDER BY x: guarantees global ordering, but does this by pushing all data through just one reducer. This is basically unacceptable for large datasets. You end up one sorted file as output.
• SORT BY x: orders data at each of N reducers, but each reducer can receive overlapping ranges of data. You end up with N or more sorted files with overlapping ranges.
• DISTRIBUTE BY x: ensures each of N reducers gets non-overlapping ranges of x, but doesn't sort the output of each reducer. You end up with N or more unsorted files with non-overlapping ranges.

• CLUSTER BY x: ensures each of N reducers gets non-overlapping ranges, then sorts by those ranges at the reducers. This gives you global ordering, and is the same as doing (DISTRIBUTE BY x and SORT BY x). You end up with N or more sorted files with non-overlapping ranges.

16.Difference between "Internal Table" and "External Table" and Mention when to choose “Internal Table” and “External Table” in Hive?

Ans->

Internal Table:-
In general, whenever we create a table inside a database in the Hive by default it is an Internal table also called the managed table. The reason Internal tables are managed because the Hive itself manages the metadata and data available inside the table. All the databases internal tables created in the Hive are by default stored at /user/hive/warehouse directory on our HDFS.When Internal tables(managed tables) are dropped all their metadata and table data got deleted permanently from our HDFS and can not be retrieved back. The Managed tables are not of any use when there is a requirement to use data available outside the Hive and also used by some other Hadoop utility on our HDFS(Hadoop Distributed File System) and the External table came into the picture.
Key points to remember about Internal tables:

- Hive takes the data file we load to the table to the /database-name/table-name inside our warehouse
- Internal table supports TRUNCATE command
- Internal tables also have ACID Support
- Internal tables also support query result caching means it can store the result of the already executed hive query for subsequent query
- Metadata and Table data both will be removed as soon as the table is dropped

External Table:-
External tables are an excellent way to manage data on the Hive since Hive does not have ownership of the data stored inside External tables. In case, if the user drops the External tables then only the metadata of tables will be removed and the data will be safe. The EXTERNAL keyword in the CREATE TABLE statement is used to create external tables in Hive. We also have to mention the location of our HDFS from where it takes the data. All the use cases where shareable data is available on HDFS so that Hive and other Hadoop components like Pig can also use the same data External tables are required. The metadata for External tables is managed by Hive but these tables take data from other locations on our HDFS.
Key points to remember about External tables

- Hive won’t take data to our warehouse
- The External table does not support the TRUNCATE command
- No support for ACID transaction property
- Doesn’t support query result caching
- Only metadata will be removed when the External table is dropped

17.Where does the data of a Hive table get stored?

Ans-> Hive stores data at the HDFS location /user/hive/warehouse folder if not specified a folder using the LOCATION clause while creating a table. Hive is a data warehouse database for Hadoop, all database and table data files are stored at HDFS location /user/hive/warehouse by default, you can also store the Hive data warehouse files either in a custom location on HDFS, S3, or any other Hadoop compatible file systems.

18) .Is it possible to change the default location of a managed table?

Ans-> The LOCATION keyword, we can change the default location of Managed tables while creating the managed table in hive. However, to do so, the user needs to specify the storage path of the managed table as the value to the LOCATION keyword, that will help to change the default location of a managed table

19.What is a metastore in Hive? What is the default database provided by Apache Hive for metastore?

Ans -> The Hive metastore is simply a relational database. It stores metadata related to the tables/schemas you create to easily query big data stored in HDFS. When you create a new Hive table, the information related to the schema (column names, data types) is stored in the Hive metastore relational database. Other information like input/output formats, partitions, HDFS locations are all stored in the metastore. Derby is the default database for the embedded metastore.

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

Ans-> Hive stores metadata information in the metastore using RDBMS instead of HDFS. The reason for choosing RDBMS is to achieve low latency as HDFS read/write operations are time consuming processes.

21.What is a partition in Hive? And Why do we perform partitioning in Hive?

Ans->Hive table partition is a way to split a large table into smaller logical tables based on one or more partition keys.
These smaller logical tables are not visible to users and users still access the data from just one table. Partition eliminates creating smaller tables, accessing, and managing them separately which results in Fast accessed to the data and Provides the ability to perform an operation on a smaller dataset.

22. What is the difference between dynamic partitioning and static partitioning?

Ans-> In static partitioning we need to specify the partition column value in each and every LOAD statement.
suppose we are having partition on column country for table t1(userid, name,occupation, country), so each time we need to provide country value
hive>LOAD DATA INPATH '/hdfspath of the file' INTO TABLE t1 PARTITION(country="US")

Dynamic partition allow us not to specify partition column value each time. the approach we follow is as below:
create a non-partitioned table t2 and insert data into it.
now create a table t1 partitioned on intended column(say country).
load data in t1 from t2 as below:

hive> INSERT INTO TABLE t2 PARTITION(country) SELECT * from T1;

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

Ans-> You can check whether a partition exist or not by following:-

Example:- show partitions {table} partition(country='india',state='MH');

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

Ans-> By using the ENABLE OFFLINE clause with ALTER TABLE statement.
Syntax:
ALTER TABLE t1 PARTITION (PARTITION_SPEC) ENABLE OFFLINE;
Example:
Hive> ALTER TABLE Sales PARTITION (country='England') ENABLE OFFLINE;

25.Why do we need buckets? How Hive distributes the rows into buckets?

Ans->

Bucketing in hive is useful when dealing with large datasets that may need to be segregated into clusters for more efficient management and to be able to perform join queries with other large datasets. The primary use case is in joining two large datasets involving resource constraints like memory limits.

Hive Bucketing a.k.a (Clustering) is a technique to split the data into more manageable files, (By specifying the number of buckets to create). The value of the bucketing column will be hashed by a user-defined number into buckets.

Bucketing can be created on just one column, you can also create bucketing on a partitioned table to further split the data to improve the query performance of the partitioned table.

26.In Hive, how can you enable buckets?

Ans-> set.hive.enforce.bucketing=true;

27. How does bucketing help in the faster execution of queries?

Ans-> Bucketing guarantee that all the data for the bucketing column remains together. E.g. if we bucket the employee table and use emp_id as the bucketing column, the value of this column will be hashed by a user-defined number of buckets (which must be optimized considering number of records). Records with the same emp_id will always be stored in the bucket. At the same time, one bucket may have many emp_id together having a more optimized chunk of data for mapreduce processing. bucketing is specially helpful, if you want to perform map-side join.

28.How to optimise Hive Performance? Explain in very detail.

Ans->

1 Avoid locking of tables

It is extremely important to make sure that the tables are being used in any Hive query as sources are not being used by another process. This can lead to locking of the table and our query can be stuck for an unknown time. We can use the parameters below for making sure that the tables are not being locked:

set hive.support.concurrency=false;

set hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DummyTxnManager;

set hive.bin.strict.locking.mode=false;

2 Use the Hive execution engine as TEZ

While executing Hive queries, TEZ execution engine is the preferred choice because it eliminates unnecessary disk access. It takes data from disk once, performs calculations, and produces output, thus saving us from multiple disk traversals. We can consider TEZ to be a much more flexible and powerful successor to the map-reduce framework. We can set the parameter below for using TEZ engine:

set hive.execution.engine=tez;

3 Use Hive Cost Based Optimizer (CBO)

Apache Hive provides a cost-based optimizer to improve performance. It generates efficient execution plans like how to order joins, which type of join to perform, the degree of parallelism etc. by examining the query cost. These decisions are collected by ANALYZE statements or the metastore itself, ultimately cutting down on query execution time and reducing resource utilization. We can set the parameter using :

set hive.cbo.enable=true;

4 Parallel execution at a Mapper & Reducer level

We can improve the performance of aggregations, filters, and joins of our hive queries by using vectorized query execution, which means scanning them in batches of 1024 rows at once instead of single row each time. We should explore the below parameters which will help to bring in more parallelism and which significantly improves query execution time:

set hive.vectorized.execution.enabled=true;

set hive.exec.parallel=true;

For example: Select a.*, b.* from (select * from table1) a Join (select * from table2) b On a.id=b.id ;

As we can see the two subqueries are independent so this might increase efficiency. One important thing to note is, parallel execution will increase cluster utilization. If the cluster utilization of a cluster is already very high, parallel execution will not help much.

5 Use STREAMTABLE option

When we are joining multiple tables, we can use STREAMTABLE option. By default, the right-most table gets streamed. For example: If we are joining 2 tables ‘huge_table’ join ‘small_table’, by default ‘small_table’ gets streamed as it is the rightmost table. In this case, ‘huge_table’ , being the bigger table,

will try to get buffered into memory and might cause java heap space issues or the job might run longer. In this case, what we can do is add /*+ STREAMTABLE('huge_table') */ and it will make 'huge_table' to be streamed rather than coming into memory. Hence, in this way, we can be free of remembering the order of joining tables.

6 Use Map Side JOIN Option

If one of the tables in the join is a small table and can be loaded into memory, we can force a MAPSIDE join like shown below: Select /*+ MAPJOIN(small_table) */ large_table.col1,large_table.col2 from large_table join small_table on large_table.col1 = small_table.col1;

A map side join can be performed within a mapper without using a Map/Reduce step. Also, We can let the execution engine take care of this by setting auto.convert.join as True.

set auto.convert.join = True ;

7 Avoid Calculated Fields in JOIN and WHERE clause

We should avoid using any calculated fields in JOIN and WHERE clauses as they take a long time to run the Hive query. We can use CTE(Create table expression) to handle those functionalities and can optimize our queries.

For example: Original Query: select a.col1, b.col2 from table1 as a join table2 as b on (a.col1 +50 = b.col2);

Optimized query: with CTE as (select a.col1 + 50 as C1 FROM table1) select CTE.C1, b.col2 from CTE join table2 b on (CTE.C1 = b.col2);

8 Use SORT BY instead of ORDER BY

Hive supports both ORDER BY and SORT BY causes. ORDER BY works on a single reducer and it causes a performance bottleneck. But, SORT BY orders the data only within each reducer and performs a local ordering where each reducer's output will be sorted ensuring better performance.

9 Select columns which are needed

While we are using Hive, If we need only a few columns from a table, avoid using SELECT * FROM as it adds unnecessary time to the execution.

10 Suitable Input format Selection

Using appropriate file formats on the basis of data can significantly increase our query performance. Hive comes with columnar input formats like RCFile, ORC, etc. On comparing to Text, Sequence, and RC file formats, ORC shows better performance because Hive has a vectorized ORC reader which allows reducing the read operations in analytics queries by allowing each column to be accessed individually.

11 Limit (Filter) the data as early as possible

This is the fundamental principle for any tuning where we filter or drop records ahead in the process so that we can avoid dealing with long-running of queries and dealing with unnecessary results.

For example : a join b where a.col !=null can be written as (select * from a where a.col!=null) join b

12 Use Multi Query Inserts

Here we will have a common dataset (Superset) and then we will use that to insert into multiple tables based on specific conditions specified in the WHERE clause. This helps to load multiple tables in parallel when they have the common superset of data.

13 Modularize the code into logical pieces

This helps in terms of the maintenance of the code. Also helps in restarting the jobs in scenarios of failures. This can be used to achieve parallelism by running modules that are independent of each other thus saving time.

29. What is the use of Hcatalog?

Ans-> HCatalog is a tool that allows you to access Hive metastore tables within Pig, Spark SQL, and/or custom MapReduce applications. HCatalog has a REST interface and command line client that allows you to create tables or do other operations. You then write your applications to access the tables using HCatalog libraries.

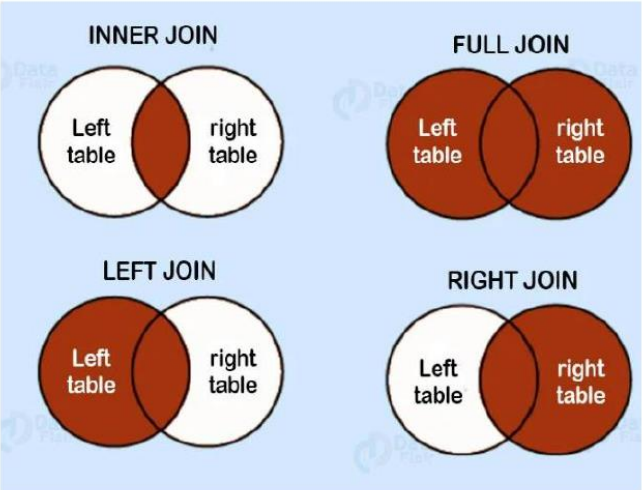
30. Explain about the different types of joins in Hive.

Ans->JOIN clause is used to combine and retrieve the records from multiple tables.. A JOIN condition is to be raised using the primary keys and foreign keys of the tables.

The following query executes JOIN on the CUSTOMER and ORDER tables, and retrieves the records:

```
join_table:

  table_reference JOIN table_factor [join_condition]
  | table_reference {LEFT|RIGHT|FULL} [OUTER] JOIN table_reference
  join_condition
  | table_reference LEFT SEMI JOIN table_reference join_condition
  | table_reference CROSS JOIN table_reference [join_condition]
```



31. Is it possible to create a Cartesian join between 2 tables, using Hive?

Ans-> Yes join_condition | table_reference [CROSS] JOIN table_reference join_condition;

32.Explain the SMB Join in Hive?

Ans-> Sort-Merge-Bucket (SMB) joins can be converted to SMB map joins as well. SMB joins are used wherever the tables are sorted and bucketed. The join boils down to just merging the already sorted tables, allowing this operation to be faster than an ordinary map-join. However, if the tables are partitioned, there could be a slow down as each mapper would need to get a very small chunk of a partition which has a single key.

The following configuration settings enable the conversion of an SMB to a map-join SMB:
set hive.auto.convert.sortmerge.join=true;
set hive.optimize.bucketmapjoin = true;
set hive.optimize.bucketmapjoin.sortedmerge = true;

33.What is the difference between order by and sort by which one we should use?

Ans-> Hive supports SORT BY which sorts the data per reducer. The difference between "order by" and "sort by" is that the former guarantees total order in the output while the latter only guarantees ordering of the rows within a reducer. If there are more than one reducer, "sort by" may give partially ordered final results.

Note: It may be confusing as to the difference between SORT BY alone of a single column and CLUSTER BY. The difference is that CLUSTER BY partitions by the field and SORT BY if there are multiple reducers partitions randomly in order to distribute data (and load) uniformly across the reducers. Basically, the data in each reducer will be sorted according to the order that the user specified. The following example shows :

SELECT key, value FROM src SORT BY key ASC, value DESC

34.What is the usefulness of the DISTRIBUTED BY clause in Hive?

An->DISTRIBUTE BY clause is used to distribute the input rows among reducers. It ensures that all rows for the same key columns are going to the same reducer. So, if we need to partition the data on some key column, we can use the DISTRIBUTE BY clause in the hive queries. However, the DISTRIBUTE BY clause does not sort the data either at the reducer level or globally. Also, the same key values might not be placed next to each other in the output dataset. As a result, the DISTRIBUTE BY clause may output N number of unsorted files where N is the number of reducers used in the query processing.

But, the output files do not contain overlapping data ranges. The syntax of the DISTRIBUTE BY clause in hive is as below:

SELECT Col1, Col2,……ColN FROM TableName DISTRIBUTE BY Col1, Col2, ….. ColN

SELECT SalesYear, Amount FROM tbl_Sales DISTRIBUTE BY SalesYear;

35.How does data transfer happen from HDFS to Hive?

Ans-> You really do not load data into Hive. Hive is not data store as well. Hive is query engine which you can use to process data in HDFS. So if you need to process/read data in HDFS using Hive you need to create table on top of it. If you know structure of data then it is simple create table statement along with

where that data resides in HDFS that is its location and you are good to go. You can use that data in Hive.

Just to add more, Hive keeps metadata about tables (like structure, data location etc) at separate database like MySQL and when hive needs to process data it just picks up location from table metadata and pulls that data from HDFS.

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

Ans->

The property of interest here is javax.jdo.option.ConnectionURL. The default value of this property is jdbc:derby;

databaseName=metastore_db;

create=true.

This value specifies that you will be using embedded derby as your Hive metastore and the location of the metastore is metastore_db. Also the metastore will be created if it doesn't already exist.

Note that the location of the metastore (metastore_db) is a relative path. Therefore, it gets created where you launch Hive from. If you update this property (in your hive-site.xml) to be, say an absolute path to a location, the metastore will be used from that location.

37.What will happen in case you have not issued the command: 'SET hive.enforce.bucketing=true;' before bucketing a table in Hive?

Ans->

The command: *'SET hive.enforce.bucketing=true;'* allows one to have the correct number of reducer while using 'CLUSTER BY' clause for bucketing a column. In case it's not done, one may find the number of files that will be generated in the table directory to be not equal to the number of buckets. As an alternative, one may also set the number of reducer equal to the number of buckets by using *set mapred.reduce.task = num_bucket*.

38.Can a table be renamed in Hive?

Ans->

You can rename the table name in the hive. You need to use the alter command. This command allows you to change the table name as shown below.

ALTER TABLE name RENAME TO new_name;

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

Ans->

The following query is used for inserting a new column:

ALTER TABLE h_table

CHANGE COLUMN new_col INT

BEFORE x_col

40.What is serde operation in HIVE?

Ans-> SerDe means Serializer and Deserializer. Hive uses SerDe and FileFormat to read and write table rows. Main use of SerDe interface is for IO operations. A SerDe allows hive to read the data from the table and write it back to the HDFS in any custom format. If we have unstructured data, then we use RegEx SerDe which will instruct hive how to handle that record.We can also write our own Custom SerDe in any format. Let us see the definition of Serializer and Deserailizer.

Deserailizer

Deserializer is conversion of string or binary data into java object when we any submit any query.

Serailizer

Serializer converts java object into string or binary object. It is used when writing the data such as insert-selectstatements.

41. Explain how Hive Deserializes and serialises the data?

Ans->

Serialization — Process of converting an object in memory into bytes that can be stored in a file or transmitted over a network.

Deserialization — Process of converting the bytes back into an object in memory.

Java understands objects and hence object is a deserialized state of data. When you use the same concept, Hive understands “columns” and hence if given a “row” of data, the task of converting that data into columns is the Deserialization part of Hive SerDe. In short

“A select statement creates deserialized data(columns) that is understood by Hive. An insert statement creates serialized data(files) that can be stored into an external storage like HDFS”.

42.Write the name of the built-in serde in hive.

Ans->

.Hive currently uses these FileFormat classes to read and write HDFS files:

TextInputFormat/HiveIgnoreKeyTextOutputFormat: These 2 classes read/write data in plain text file format.

SequenceFileInputFormat/SequenceFileOutputFormat: These 2 classes read/write data in Hadoop SequenceFile format.

43.What is the need of custom Serde?

Ans->

In most cases, users want to write a Deserializer instead of a SerDe, because users just want to read their own data format instead of writing to it.For example, the RegexDeserializer will deserialize the data using the configuration parameter 'regex', and possibly a list of column names.

44.Can you write the name of a complex data type(collection data types) in Hive?

Ans->In addition to primitive data types, Hive also supports a few complex data types: Struct, MAP , and Array . Complex data types are also known as collection data types.

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

Ans-> Running Hive query through Unix Shell script is by creating a file e.g abc.hql and write your query within that file and call that file using -f parameter in hive.

e.g. “hive -f “/datalake/user/abc.hql””

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

Ans->The default record delimiter is − \n And the field delimiters are − \001,\002,\003.

47.How do you list all databases in Hive whose name starts with s?

Ans-> SHOW (DATABASES|SCHEMAS) [LIKE identifier_with_wildcards];

Example:- SHOW DATABASES LIKE 's%';

48.What is the difference between LIKE and RLIKE operators in Hive?

Ans-> LIKE is an operator similar to LIKE in SQL. We use LIKE to search for string with similar text. RLIKE (Right-Like) is a special function in Hive where if any substring of A matches with B then it evaluates to true. It also obeys Java regular expression pattern.

Hive provides RLIKE operator that can be used for searching advanced Regular Expressions in Java.

E.g. user_name RLIKE ‘.(Smith|Sam).’

This will return user_name that has Smith or Sam in it.

49. How to change the column data type in Hive?

Ans-> ALTER TABLE table_name CHANGE old_column_name new_column_name new_datatype;

Example:- ALTER TABLE tableA CHANGE salary salary BIGINT;

50. How will you convert the string ’51.2’ to a float value in the particular column?
Ans-> select cast (’51.2’ as float)

51.What will be the result when you cast ‘abc’ (string) as INT?
Ans-> It prints NULL.

52 .What does the following query do? a. INSERT OVERWRITE TABLE employees b. PARTITION (country, state) c. SELECT ..., se.cnty, se.st d. FROM staged_employees se;

Ans->Due to the use of the OVERWRITE keyword, all data in the original same partition in the target table is overwritten. If there is no partition in the target table, the entire table will be overwritten.

If the OVERWRITE keyword is deleted or replaced with INTO, hive will append instead of replacing the data in the original partition or table. This feature is only supported after Hive v0.8.0.

53.Write a query where you can overwrite data in a new table from the existing table.

Ans-> INSERT OVERWRITE TABLE B SELECT A.value FROM A;

54. What is the maximum size of a string data type supported by Hive? Explain how Hive supports binary formats.

Ans-> The maximum size of a string data type supported by Hive is 2 GB. Hive supports the text file format by default, and it also supports the binary format sequence files, ORC files, Avro data files, and Parquet files.
Sequence file: Sequence files are Hadoop flat files which stores values in binary key-value pairs. The sequence files are in binary format and these files are able to split. The main advantages of using sequence file is to merge two or more files into one file.
Create a sequence file by add storage option as ‘STORED AS SEQUENCEFILE’ at the end of a Hive CREATE TABLE command.

55. What File Formats and Applications Does Hive Support?

Ans->Hive supports the text file format by default, and it also supports the binary format sequence files, ORC files, Avro data files, and Parquet files.

Sequence file: It is a splittable, compressible, and row-oriented file with a general binary format.

ORC file: Optimized row columnar (ORC) format file is a record-columnar and column-oriented storage file. It divides the table in row split. Each split stores the value of the first row in the first column and follows subsequently.

Avro data file: It is the same as a sequence file that is splittable, compressible, and row-oriented but without the support of schema evolution and multilingual binding.

Parquet file: In Parquet format, along with storing rows of data adjacent to one another, we can also store column values adjacent to each other such that both horizontally and vertically datasets are partitioned.

56.How do ORC format tables help Hive to enhance its performance?

Ans-> Using appropriate file formats on the basis of data can significantly increase our query performance. Hive comes with columnar input formats like RCFile, ORC, etc. On comparing to Text, Sequence, and RC file formats, ORC shows better performance because Hive has a vectorized ORC reader which allows reducing the read operations in analytics queries by allowing each column to be accessed individually. Stored as columns and compressed, which leads to smaller disk reads. The columnar format is also ideal for vectorization optimizations in Tez. ORC has a built-in index, min/max values, and other aggregates that cause entire stripes to be skipped during reads.

57. How can Hive avoid mapreduce while processing the query?

Ans-> When you perform a "select * from ", Hive fetches the whole data from file as a FetchTask rather than a mapreduce task which just dumps the data as it is without doing anything on it. However, while using "select from ", Hive requires a map-reduce job since it needs to extract the 'column' from each row by parsing it from the file it loads.

58.What is view and indexing in hive?

Ans->A view allows a query to be saved and treated like a table. It is a logical construct, as it does not store data like a table.When a query references a view, the information in its definition is combined with the rest of the query by Hive's query planner. Logically, you can imagine that Hive executes the view and then uses the results in the rest of the query.

For example, consider the following query with a nested subquery:

FROM (

SELECT * FROM people JOIN cart ON (cart.people_id=people.id) WHERE firstname='john'

) a SELECT a.lastname WHERE a.id=3;

It is common for Hive queries to have many levels of nesting. In the following example, the nested portion of the query is turned into a view:

CREATE VIEW shorter_join AS SELECT * FROM people JOIN cart ON (cart.people_id=people.id) WHERE firstname='john';

Now the view is used like any other table. In this query we added a WHERE clause to the SELECT statement. This exactly emulates the original query:

SELECT lastname FROM shorter_join WHERE id=3;

An Index is nothing but a pointer on a particular column of a table. Creating an index means creating a pointer on a particular column of a table.

However, the user has to manually define the Hive index
Basically, we are creating the pointer to particular column name of the table, wherever we are creating Hive index.By using the Hive index value created on the column name, any Changes made to the column present in tables are stored.

Syntax:-

Create INDEX < INDEX_NAME> ON TABLE < TABLE_NAME(column names)>

hive> CREATE INDEX index_salary ON TABLE employee(salary)
AS 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler';

59.Can the name of a view be the same as the name of a hive table?

Ans-> The name of a view must be unique, and it cannot be the same as any table or database or view's name.

60.What types of costs are associated in creating indexes on hive tables?

Ans-> Indexes occupies space and there is a processing cost in arranging the values of the column on which index is created.

61.Give the command to see the indexes on a table.

Ans-> SHOW INDEX ON table_name

62. Explain the process to access subdirectories recursively in Hive queries.

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

Shell
hive> Set mapred.input.dir.recursive=true;
hive> Set hive.mapred.supports.subdirectories=true;

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

63. If you run a select * query in Hive, why doesn't it run MapReduce?

Ans-> Query you are reading every column, no filtering is required. Hence no Map phase

select * FROM TABLE –Q1

select col1,col2 FROM TABLE –Q2

To understand the reason, first we need to know what map and reduce phases mean:

Map: Basically a filter which filters and organizes data in sorted order.

For e.g. It will filter col1_name, col2_name from a row in the second query. For query in which you are reading every column, no filtering is required. Hence no Map phase.

Reduce: Reduce is just summary operation data across the rows. for e.g. sum of a column! In both the queries you don't need any summary data. Hence no reducer.

64.What are the uses of Hive Explode?

Ans->Explode is a User Defined Table generating Function(UDTF) in Hive. It takes an array (or a map) as an input and outputs the elements of the array (or a map) as separate rows. UDTFs can be used in the SELECT expression list and as a part of LATERAL VIEW.LATERAL VIEW statement is used with UDTF such as explode(). It creates a virtual table by applying the UDTF to each row of the base table and then joins resulting output rows to the input row.

Ex:- select explode(invested_stocks) as stock_name from investment_report;

65. What is the available mechanism for connecting applications when we run Hive as a server?

Ans->Thrift Client: Using Thrift, we can call Hive commands from various programming languages, such as C++, PHP, Java, Python, and Ruby.

JDBC Driver: JDBC Driver enables accessing data with JDBC support, by translating calls from an application into SQL and passing the SQL queries to the Hive engine.

ODBC Driver: It implements the ODBC API standard for the Hive DBMS, enabling ODBC-compliant applications to interact seamlessly with Hive.

66.Can the default location of a managed table be changed in Hive?

Ans->Yes, we can change the default location of Managed tables using the LOCATION keyword while creating the managed table.The user has to specify the storage path of the managed table as the value to the LOCATION keyword.

For ex -

CREATE TABLE if not exists demo

(id int, firstname STRING,lastname STRING, country string)

LOCATION '/user/demo';

67.What is the Hive ObjectInspector function?

Ans-> Hive uses ObjectInspector to analyze the internal structure of the row object and also the structure of the individual columns. ObjectInspector provides a uniform way to access complex objects that can be stored in multiple formats in the memory, including: Instance of a Java class (Thrift or native Java).

68.What is UDF in Hive?

Ans->UDF stands for User Defined Functions. Hive is a powerful tool that allows us to provision sql queries on top of stored data for basic querying and/or analysis. And on top of an already rich set of built-in functions, it allows us to extend its functionality by writing custom functions of our own. Types of UDFs available in Hive:
1. UDF (User Defined Function): Operates on a single row and generates a single output.
2. UDAF (User Defined Aggregate Function): Operates on a group of rows and generates a single output. Generally, used with an accompanying 'group by' clause.
3. UDTF (User Defined Tabular Function): Operates on a single row and generates multiple rows as an output.

69.Write a query to extract data from hdfs to hive.

Ans->Example :- load data inpath 'javachain/student.txt' into table student;

70.What is TextInputFormat and SequenceFileInputFormat in hive.

Ans-> TEXTFILE format is a famous input/output format used in Hadoop. In Hive if we define a table as TEXTFILE it can load data of from CSV (Comma Separated Values), delimited by Tabs, Spaces, and JSON data. This means fields in each record should be separated by comma or space or tab or it may be JSON(JavaScript Object Notation) data.

By default, if we use TEXTFILE format then each line is considered as a record.

At the end, we need to specify the type of file format. If we do not specify anything it will consider the file format as TEXTFILE format.The TEXTFILE input and TEXTFILE output format are present in the Hadoop package as shown below:

org.apache.hadoop.mapred.TextInputFormat

org.apache.hadoop.mapred.TextOutputFormat

We know that Hadoop's performance is drawn out when we work with a small number of files with big size rather than a large number of files with small size. If the size of a file is smaller than the typical block size in Hadoop, we consider it as a small file. Due to this, a number of metadata increases which will become an overhead to the NameNode. To solve this problem sequence files are introduced in Hadoop. Sequence files act as a container to store the small files.

Sequence files are flat files consisting of binary key-value pairs. When Hive converts queries to MapReduce jobs, it decides on the appropriate key-value pairs to be used for a given record. Sequence files are in the binary format which can be split and the main use of these files is to club two or more smaller files and make them as a one sequence file.

In Hive we can create a sequence file by specifying STORED AS SEQUENCEFILE in the end of a CREATE TABLE statement.

There are three types of sequence files:

Uncompressed key/value records.

Record compressed key/value records - only 'values' are compressed here

Block compressed key/value records - both keys and values are collected in 'blocks' separately and compressed. The size of the 'block' is configurable.

Hive has its own SEQUENCEFILE reader and SEQUENCEFILE writer libraries for reading and writing through sequence files.

Hive uses the SEQUENCEFILE input and output formats from the following packages:

org.apache.hadoop.mapred.SequenceFileInputFormat

org.apache.hadoop.hive.ql.io.HiveSequenceFileOutputFormat

71.How can you prevent a large job from running for a long time in a hive?

Ans-> This can be achieved by setting the MapReduce jobs to execute in strict mode set hive.mapred.mode=strict; The strict mode ensures that the queries on partitioned tables cannot execute without defining a WHERE clause.

72.When do we use explode in Hive?

Ans-> Explode is a User Defined Table generating Function(UDTF) in Hive. It takes an array (or a map) as an input and outputs the elements of the array (or a map) as separate rows. UDTFs can be used in the SELECT expression list and as a part of LATERAL VIEW.LATERAL VIEW statement is used with UDTF such as explode(). It creates a virtual table by applying the UDTF to each row of the base table and then joins resulting output rows to the input row.

73.Can Hive process any type of data formats? Why? Explain in very detail.

Ans->HiveQL handles structured data only. By default, Hive has derby database to store the data in it. We can configure Hive with MySQL database. As mentioned, HiveQL can handle only structured data. Data is eventually stored in files.

Hive supports several file formats:

Text File

SequenceFile

RCFile

Avro Files

ORC Files

Parquet

Custom INPUTFORMAT and OUTPUTFORMAT

The hive.default.fileformat configuration parameter determines the format to use if it is not specified in a CREATE TABLE or ALTER TABLE statement.

Text file is the parameter's default value.

Hive Text File Format

Hive Text file format is a default storage format. You can use the text format to interchange the data with other client application. The text file format is very common most of the applications. Data is stored in lines, with each line being a record. Each lines are terminated by a newline character (\n). The text format is simple plane file format. You can use the compression (BZIP2) on the text file to reduce the storage spaces.

Hive Sequence File Format

Sequence files are Hadoop flat files which stores values in binary key-value pairs. The sequence files are in binary format and these files are able to split. The main advantages of using sequence file is to merge two or more files into one file.

Hive RC File Format

RCFile is row columnar file format. This is another form of Hive file format which offers high row level compression rates. If you have requirement to perform multiple rows at a time then you can use RCFile format. The RCFile are very much similar to the sequence file format. This file format also stores the data as key-value pairs.

Hive AVRO File Format

AVRO is open source project that provides data serialization and data exchange services for Hadoop. You can exchange data between Hadoop ecosystem and program written in any programming languages. Avro is one of the popular file format in Big Data Hadoop based applications.

Hive ORC File Format

The ORC file stands for Optimized Row Columnar file format. The ORC file format provides a highly efficient way to store data in Hive table. This file system was actually designed to overcome limitations of the other Hive file formats. The Use of ORC files improves performance when Hive is reading, writing, and processing data from large tables.

Hive Parquet File Format

Parquet is a column-oriented binary file format. The parquet is highly efficient for the types of largescale queries. Parquet is especially good for queries scanning particular columns within a particular table. The Parquet table uses compression Snappy, gzip; currently Snappy by default.

74.Whenever we run a Hive query, a new metastore_db is created. Why?

Ans-> Whenever you run the hive in embedded mode, it creates the local metastore. And before creating the metastore it looks whether metastore already exist or not. This property is defined in configuration file hive-site.xml. Property is “javax.jdo.option.ConnectionURL” with default value “jdbc:derby:;databaseName=metastore_db;create=true”. So to change the behavior change the location to absolute path, so metastore will be used from that location.

75.Can we change the data type of a column in a hive table? Write a complete query.

Ans-> ALTER TABLE table_name CHANGE column_name column_name new_datatype;

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

Ans-> Hive provides us the functionality to load pre-created table entities either from our local file system or from HDFS. The LOAD DATA statement is used to load data into the hive table.

Syntax: LOAD DATA [LOCAL] INPATH '' [OVERWRITE] INTO TABLE ;

Note: The LOCAL Switch specifies that the data we are loading is available in our Local File System. If the LOCAL switch is not used, the hive will consider the location as an HDFS path location. The OVERWRITE switch allows us to overwrite the table data.

77.What isthe precedence order in Hive configuration?

Ans-> In Hive we can use following precedence order to set the configurable properties.

Hive SET command has the highest priority

-hiveconf option from Hive Command Line

hive-site.xml file

hive-default.xml file

hadoop-site.xml file

hadoop-default.xml file

78. Which interface is used for accessing the Hive metastore?

Ans-> WebHCat API web interface can be used for Hive commands. It is a REST API that allows applications to make HTTP requests to access the Hive metastore (HCatalog DDL). It also enables users to create and queue Hive queries and commands.

79. Is it possible to compress json in the Hive external table ?

Ans->Just gzip your files and put them as is (*.gz) into the table location.

80. What is the difference between local and remote metastores?

Ans->Local Metastore: Here metastore service still runs in the same JVM as Hive but it connects to a database running in a separate process either on same machine or on a remote machine. Remote Metastore: Metastore runs in its own separate JVM not on hive service JVM.

81.What is the purpose of archiving tablesin Hive?

Ans-> Due to the design of HDFS, the number of files in the filesystem directly affects the memory consumption in the namenode. While normally not a problem for small clusters, memory usage may hit the limits of accessible memory on a single machine when there are >50-100 million files. In such situations, it is advantageous to have as few files as possible. The use of Hadoop Archives is one approach to reducing the number of files in partitions. Hive has built-in support to convert files in existing partitions to a Hadoop Archive (HAR) so that a partition that may once have consisted of 100's of files can occupy just ~3 files (depending on settings). However, the trade-off is that queries may be slower due to the additional overhead in reading from the HAR.

82.What is DBPROPERTY in Hive?

Ans->DBPROPERTIES – Optional but used to specify any properties of database in the form of (key, value) separated pairs.

CREATE DATABASE IF NOT EXISTS test_db COMMENT "Test Database created for tutorial" WITH DBPROPERTIES('Date' = '2014-12-03', 'Creator' = 'Bala G', 'Email' = 'bala@somewhere.com');

83. Differentiate between local mode and MapReduce mode in Hive.

Ans-> MapReduce mode: In MapReduce mode, Hive script is executed on Hadoop cluster. The Hive scripts are converted into MapReduce jobs and then executed on Hadoop cluster (hdfs)

Local mode: In this mode, Hive script runs on a Single machine without the need of Hadoop cluster or hdfs. Local mode is used for development purpose to see how the script would behave in an actual environment