Hive interview Questions

1. What is the definition of Hive? What is the present version of Hive?
Hive is a data warehouse system which is used to analyze structured data. It is built on the top of Hadoop.
present version of Hive:hive-4.0.0-alpha-1   https://dlcdn.apache.org/hive/

2. Is Hive suitable to be used for OLTP systems? Why?
NO. Hive is mainly used for batch processing i.e. OLAP and it is not used for OLTP because of the real-time operations of the database.

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

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.
Tables in rdms are sparse.	Table in hive are dense.
It doesn’t support partitioning.	It supports automation partition.
No partition method is used.	Sharding method is used for partition.

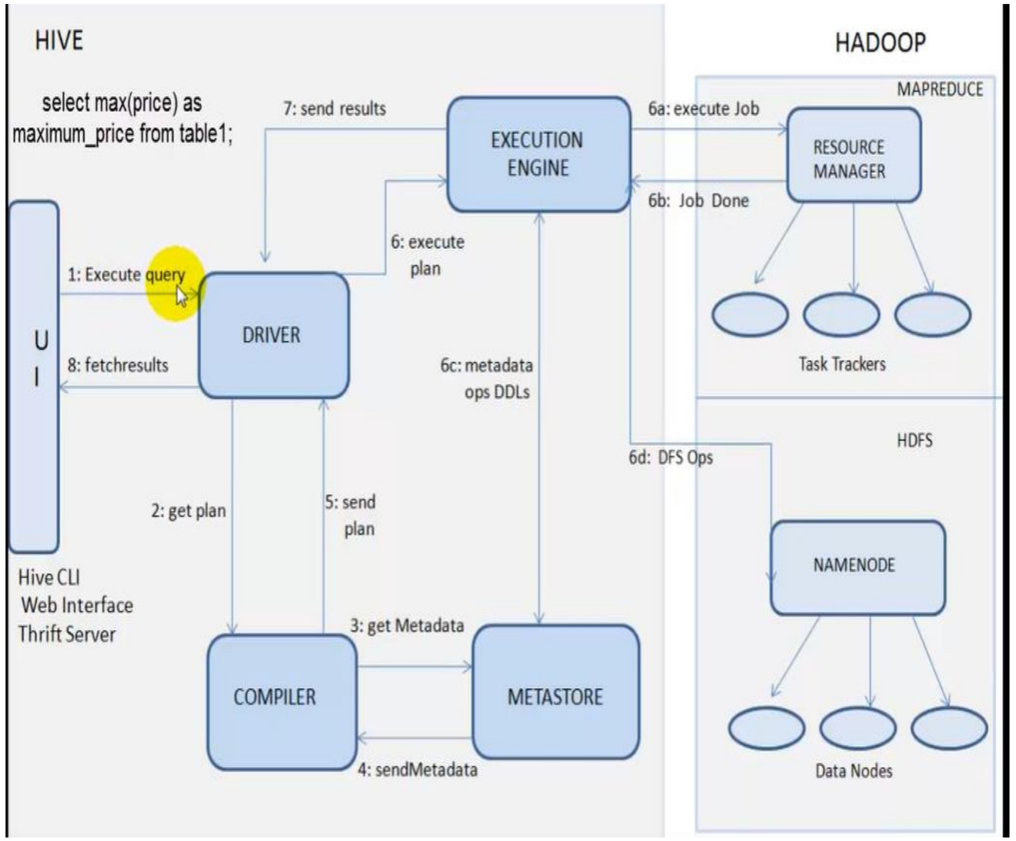
older versions of Hive doesn’t support ACID transactions on tables.
Though in newer versions it supports by default ACID transactions are disabled and you need to enable it before start using it.
Below are the properties you need to enable ACID transactions.

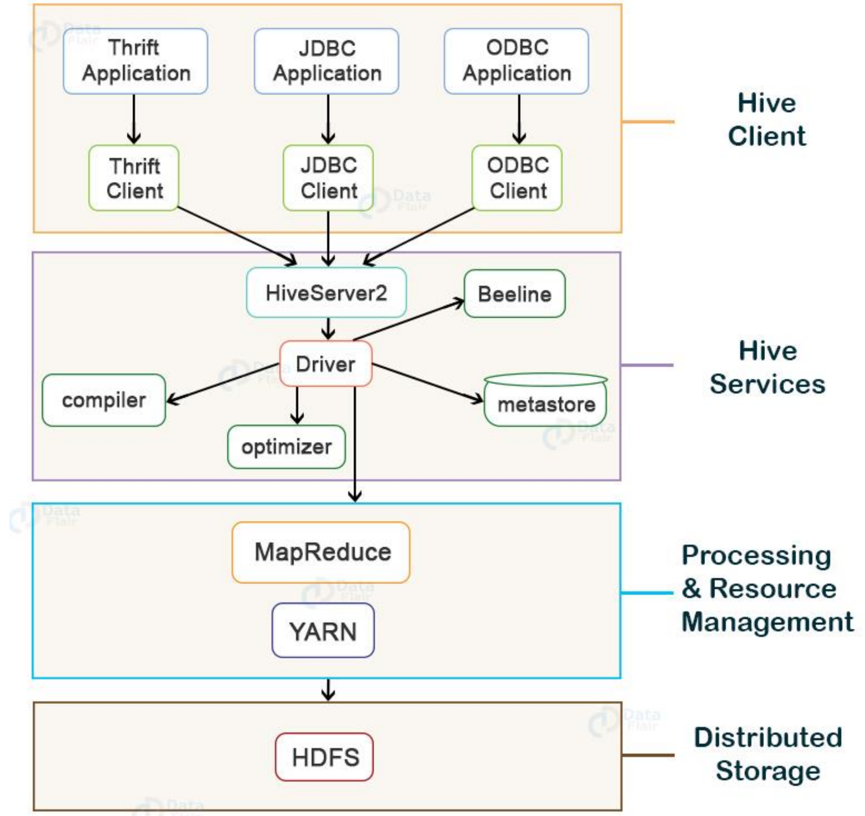
```
SET hive.support.concurrency=true;
SET hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
# The follwoing are not required if you are using Hive 2.0
SET hive.enforce.bucketing=true;
SET hive.exec.dynamic.partition.mode=nostrict;
# The following parameters are required for standalone 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?





Hive Architecture & Its Components

1. **Hive Client**
2. **Hive Services**
3. **Processing and Resource Management**
4. **Distributed Storage**

Hive supports applications written in any language like Python, Java, C++, Ruby, etc. using JDBC, ODBC, and Thrift drivers, for performing queries on the Hive. Hence, one can easily write a hive client application in any language of its own choice.
Hive clients are categorized into three types:

1. Thrift Clients

The Hive server is based on Apache Thrift so that it can serve the request from a thrift client.

2. JDBC client

Hive allows for the Java applications to connect to it using the JDBC driver. JDBC driver uses Thrift to communicate with the Hive Server.

3. ODBC client

Hive ODBC driver allows applications based on the ODBC protocol to connect to Hive. Similar to the JDBC driver, the ODBC driver uses Thrift to communicate with the Hive Server.
[ps2id id='Hive-Services' target=''/]Hive Service
To perform all queries, Hive provides various services like the Hive server2, Beeline, etc. The various services offered by Hive are:
1. Beeline
The Beeline is a command shell supported by HiveServer2, where the user can submit its queries and command to the system. It is a **JDBC** client that is based on **SQLLINE CLI** (pure Java-console-based utility for connecting with relational databases and executing SQL queries).

2. Hive Server 2

HiveServer2 is the successor of HiveServer1. HiveServer2 enables clients to execute queries against the Hive. It allows multiple clients to submit requests to Hive and retrieve the final results. It is basically designed to provide the best support for open API clients like JDBC and ODBC.
Note: Hive server1, also called a Thrift server, is built on Apache Thrift protocol to handle the cross-platform communication with Hive. It allows different client applications to submit requests to Hive and retrieve the final results.
It does not handle concurrent requests from more than one client due to which it was replaced by HiveServer2.

3. Hive Driver

The Hive driver receives the **HiveQL** statements submitted by the user through the command shell. It creates the session handles for the query and sends the query to the compiler.

4. Hive Compiler

Hive compiler parses the query. It performs semantic analysis and type-checking on the different query blocks and query expressions by using the metadata stored in metastore and generates an execution plan.
The execution plan created by the compiler is the **DAG(Directed Acyclic Graph)**, where each stage is a map/reduce job, operation on HDFS, a metadata operation.

5. Optimizer

Optimizer performs the transformation operations on the execution plan and splits the task to improve efficiency and scalability.
6. Execution Engine
Execution engine, after the compilation and optimization steps, executes the execution plan created by the compiler in order of their dependencies using Hadoop.

7. Metastore

Metastore is a central repository that stores the metadata information about the structure of tables and partitions, including column and column type information.
It also stores information of serializer and deserializer, required for the read/write operation, and HDFS files where data is stored. This metastore is generally a relational database.
Metastore provides a Thrift interface for querying and manipulating Hive metadata.
We can configure metastore in any of the two modes:
•**Remote:** In remote mode, metastore is a Thrift service and is useful for non-Java applications.
•**Embedded:** In embedded mode, the client can directly interact with the metastore using JDBC.

8. HCatalog

HCatalog is the table and storage management layer for Hadoop. It enables users with different data processing tools such as Pig, MapReduce, etc. to easily read and write data on the grid.
It is built on the top of Hive metastore and exposes the tabular data of Hive metastore to other data processing tools.

9. WebHCat

WebHCat is the REST API for HCatalog. It is an HTTP interface to perform Hive metadata operations. It provides a service to the user for running Hadoop MapReduce (or YARN), Pig, Hive jobs.

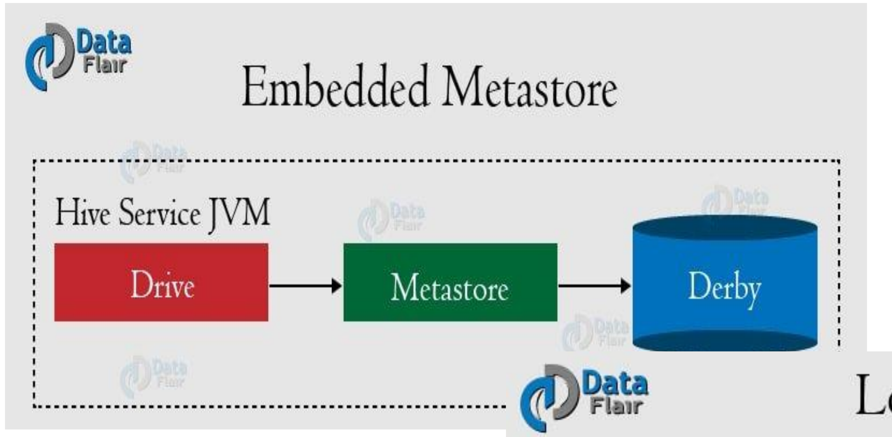
5. Mention what Hive query processor does? And Mention what are the components of a Hive query processor?

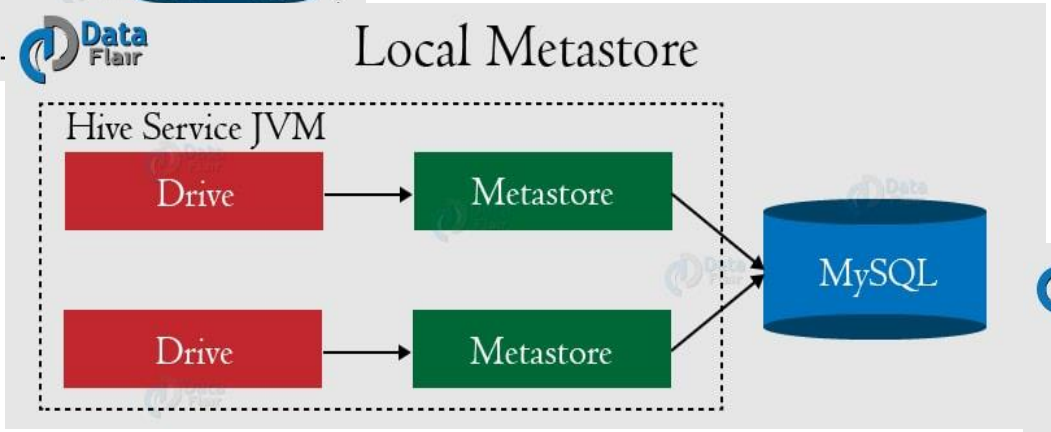
Hive query processor
It execute HQL

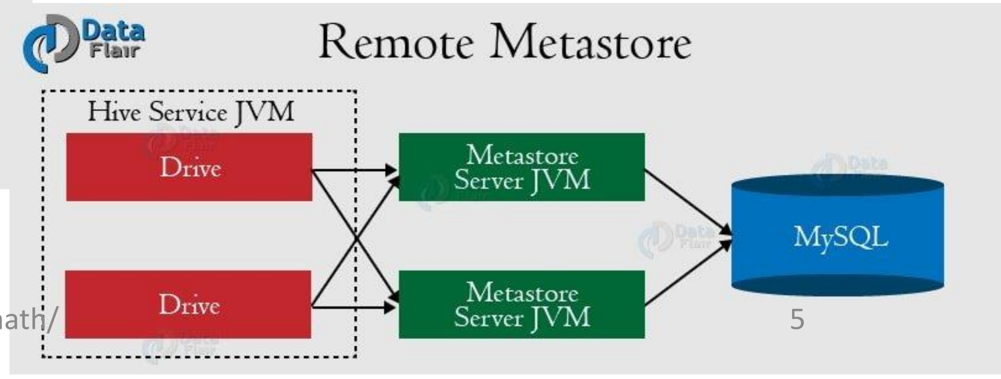
components of a Hive query processor
Parse and Semantic Analysis (ql/parse)
Metadata Layer (ql/metadata)
Type Interfaces (ql/typeinfo)
Sessions (ql/session)
Map/Reduce Execution Engine (ql/exec)
Plan Components (ql/plan)
Hive Function Framework (ql/udf)
Tools (ql/tools)
Optimizer (ql/optimizer)

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

Hive Metastore Modes
There are three modes for Hive Metastore deployment:
•Embedded Metastore
•Local Metastore
•Remote Metastore







7. Features and Limitations of Hive.

These are the following features of Hive:

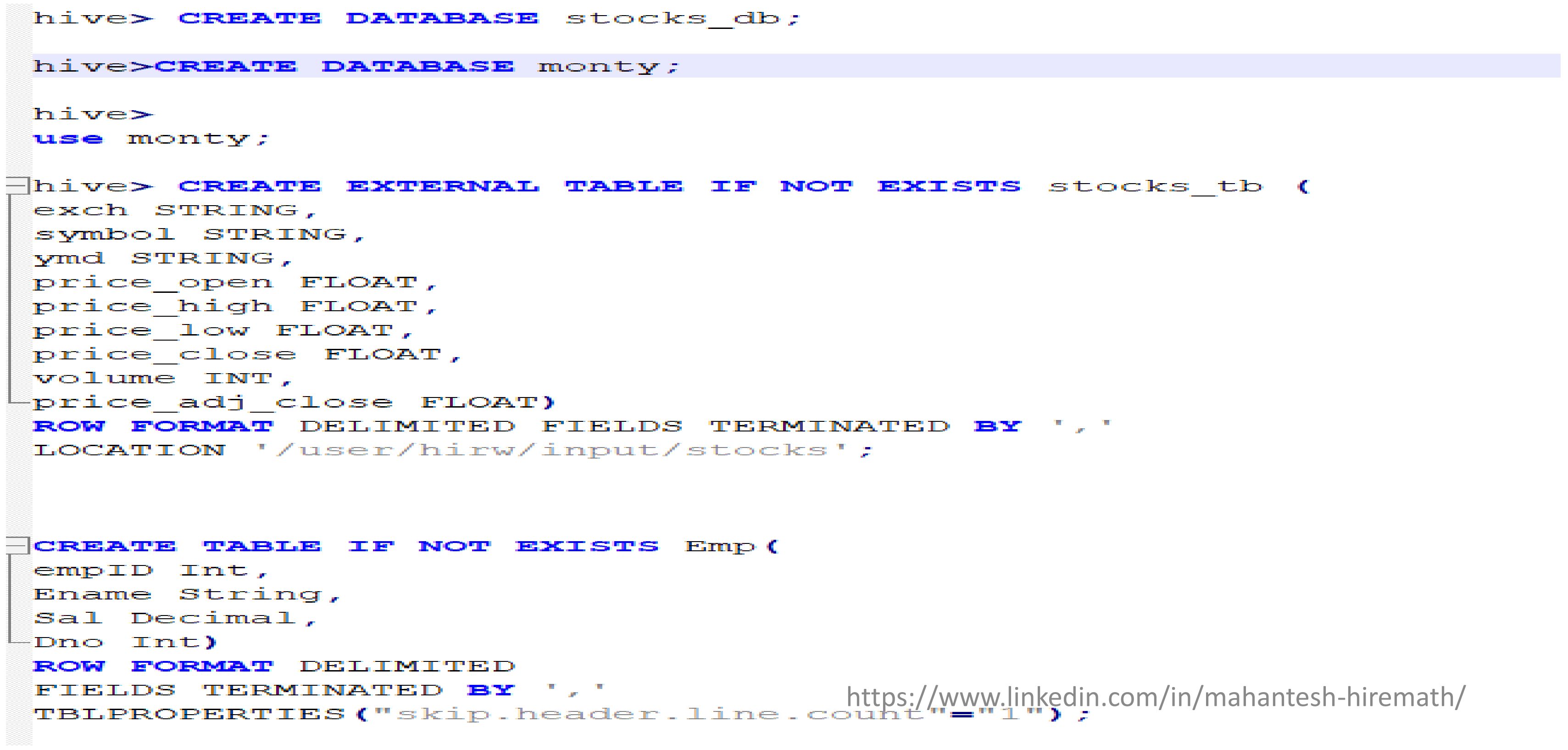
•Hive is fast and scalable.
•It provides SQL-like queries (i.e., HQL) that are implicitly transformed to MapReduce or Spark jobs.
•It is capable of analyzing large datasets stored in HDFS.
•It allows different storage types such as plain text, RCFile, and HBase.
•It uses indexing to accelerate queries.
•It can operate on compressed data stored in the Hadoop ecosystem.
•It supports user-defined functions (UDFs) where user can provide its functionality.

Limitations of Hive

•Hive is not capable of handling real-time data.
•It is not designed for online transaction processing.
•Hive queries contain high latency.

8. How to create a Database in HIVE?

9. How to create a table in HIVE?



```
hive> CREATE DATABASE stocks_db;

hive>CREATE DATABASE monty;

hive>
use monty;

hive> CREATE EXTERNAL TABLE IF NOT EXISTS stocks_tb (
exch STRING,
symbol STRING,
ymd STRING,
price_open FLOAT,
price_high FLOAT,
price_low FLOAT,
price_close FLOAT,
volume INT,
price_adj_close FLOAT)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/user/hirw/input/stocks';


CREATE TABLE IF NOT EXISTS Emp(
empID Int,
Ename String,
Sal Decimal,
Dno Int)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
TBLPROPERTIES("skip.header.line.count"="1");
```

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

Describe- This will show table columns
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 - 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?

```
TBLPROPERTIES("skip.header.line.count"="1");
ALTER TABLE tablename
SET TBLPROPERTIES ("skip.header.line.count"="1");
```

12. What is a hive operator? What are the different types of hive operators?
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
Built-In Functions
Hive supports the following built-in functions:

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.

DDL Command	Use With
CREATE	Database, Table
SHOW	Databases, Tables, Table Properties, Partitions, Functions, Index
DESCRIBE	Database, Table, view
USE	Database
DROP	Database, Table
ALTER	Database, Table
TRUNCATE	Table

```
CREATE (DATABASE|SCHEMA) [IF NOT EXISTS] database_name
[COMMENT database_comment]
[LOCATION hdfs_path]
[WITH DBPROPERTIES (property_name=property_value, ...)];
```

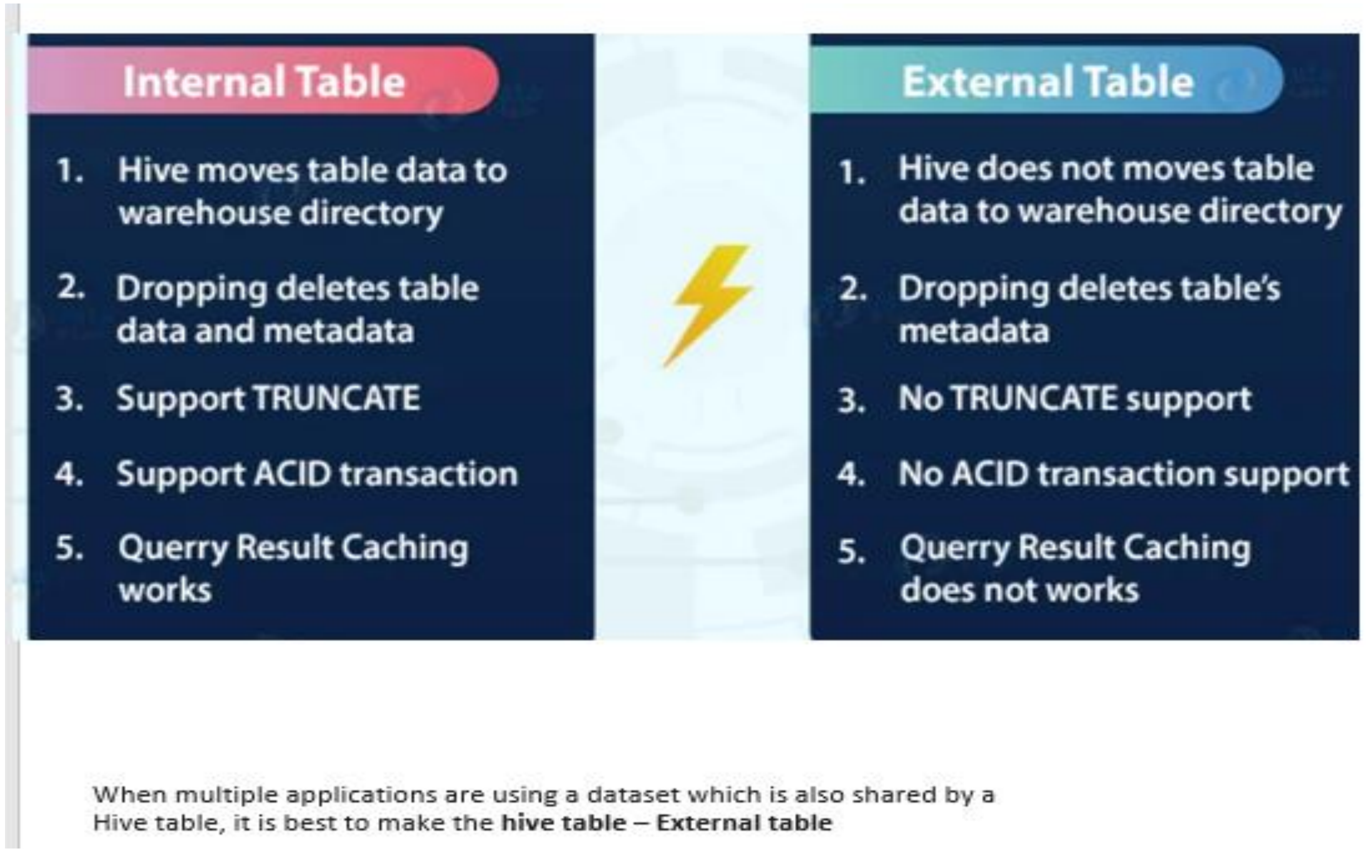
The various Hive DML commands are:

LOAD
SELECT
INSERT
DELETE
UPDATE
EXPORT
IMPORT

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

•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?



Internal Table	External Table
1. Hive moves table data to warehouse directory	1. Hive does not moves table data to warehouse directory
2. Dropping deletes table data and metadata	2. Dropping deletes table’s metadata
3. Support TRUNCATE	3. No TRUNCATE support
4. Support ACID transaction	4. No ACID transaction support
5. Querry Result Caching works	5. Querry Result Caching does not works

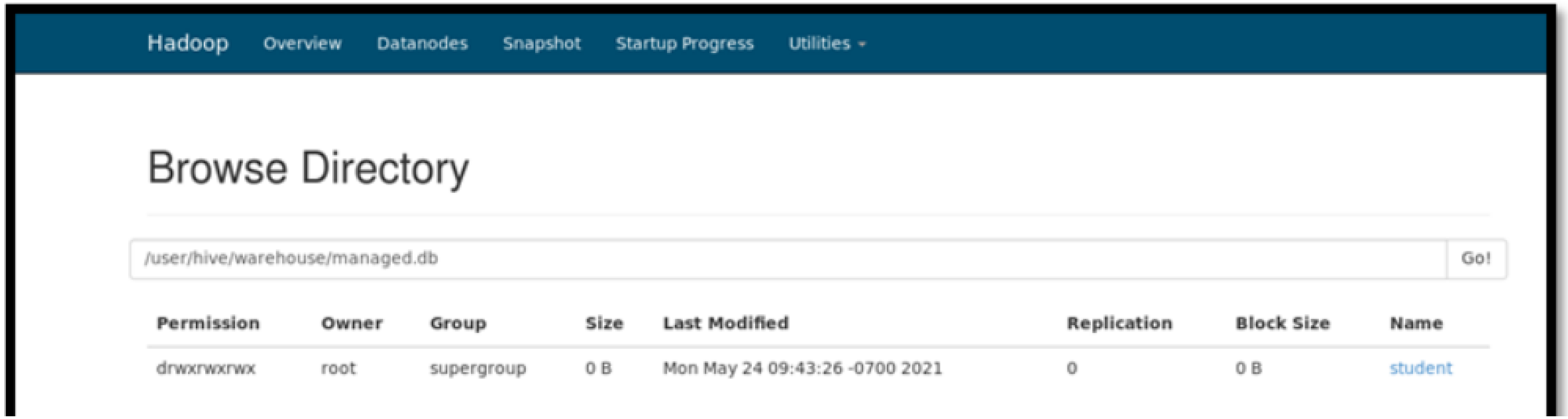
When multiple applications are using a dataset which is also shared by a Hive table, it is best to make the **hive table – External table**

We may use an internal table if:
1. Data is temporary and doesn’t affect businesses in real-time.
2. If we want the hive to manage the data and the tables.

We may use an external table if:
1. We want to use data outside HIVE for performing a different operations such as loading and merging.
2. The data is of production quality.

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

The data loaded in the hive database is stored at the HDFS path – **/user/hive/warehouse**. If the location is not specified, by default all metadata gets stored in this path.



Browse Directory

/user/hive/warehouse/managed.db

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxrwxrwx	root	supergroup	0 B	Mon May 24 09:43:26 -0700 2021	0	0 B	student

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

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?
metastore is database keeps all the information related to our databases, tables and relations as Metadata. When ever you want to know about database than in the Metastore we can easily find all the information.

Hive by default, metastore service runs in the same JVM as the Hive service. It uses embedded derby database stored on the local file system in this mode.
Thus both metastore service and hive service runs in the same JVM by using embedded Derby Database.

20. Why does Hive not store metadata information in HDFS?
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?
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.
Partition Table Advantages
Fast accessed to the data
Provides the ability to perform an operation on a smaller dataset

22. What is the difference between dynamic partitioning and static partitioning?
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 '/hdfs path of the file' INTO TABLE t1 PARTITION(country="US")
hive>LOAD DATA INPATH '/hdfs path of the file' INTO TABLE t1 PARTITION(country="UK")
```

dynamic partition allow us not to specify partition column value each time. the approach we follows 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;
```

make sure that partitioned column is always the last one in non partitioned table(as we are having country column in t2)

23. How do you check if a particular partition exists?
we can check whether a particular partition exists or not:
SHOW PARTITIONS table_name
PARTITION(partitioned_column='partition_value')

24. How can you stop a partition form being queried?
By using the ENABLE OFFLINE clause with ALTER TABLE atatement.
Syntax:
ALTER TABLE t1 PARTITION (PARTITION_SPEC) ENABLE OFFLINE;
Example:
Hive> ALTER TABLE TownsList_Dynamic PARTITION (country='England') ENABLE OFFLINE;
Now, let's issue the SELECT statement.
Hive> select * from TownsList_Dynamic where country='England'

25. Why do we need buckets? How Hive distributes the rows into buckets?
Bucketing is a method to evenly distributed the data across many files. Create multiple buckets and then place each record into one of the buckets based on some logic mostly some hashing algorithm.
Bucketing feature of Hive can be used to distribute/organize the table/partition data into multiple files such that similar records are present in the same file.
While creating a Hive table, a user needs to give the columns to be used for bucketing and the number of buckets to store the data into.
Which records go to which bucket are decided by the Hash value of columns used for bucketing.

26. In Hive, how can you enable buckets?
set.hive.enforce.bucketing=true;

27. How does bucketing help in the faster execution of queries?
In bucketing, the partitions can be subdivided into buckets based on the hash function of a column.
It gives extra structure to the data which can be used for more efficient queries.

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

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.coll, b.col2 from table1 as a join table2 as b on (a.coll +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.

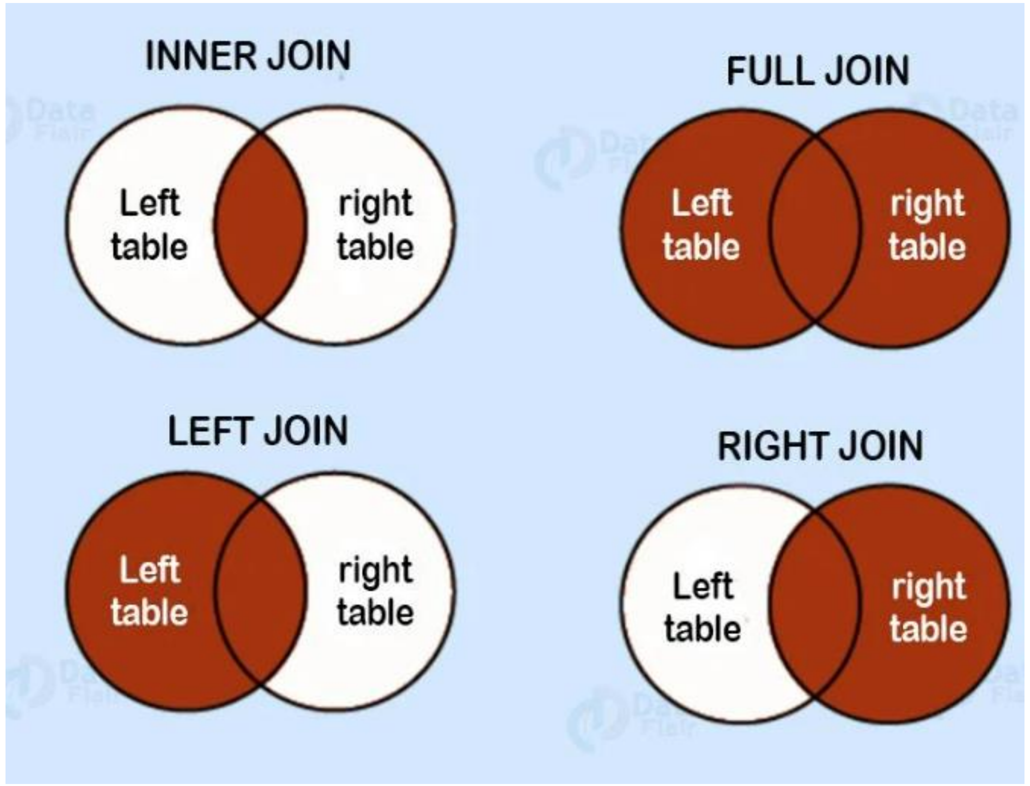
10 Suitable Input format Selection
Using appropriate file formats on the basis of data can significantly increase our query performance. Hive comes with columnar input formats like RCFile, ORC, etc. On comparing to Text, Sequence, and RC file formats, ORC shows better performance because Hive has a vectorized 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 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.

29. What is the use of Hcatalog?
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 join in Hive.
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]
```



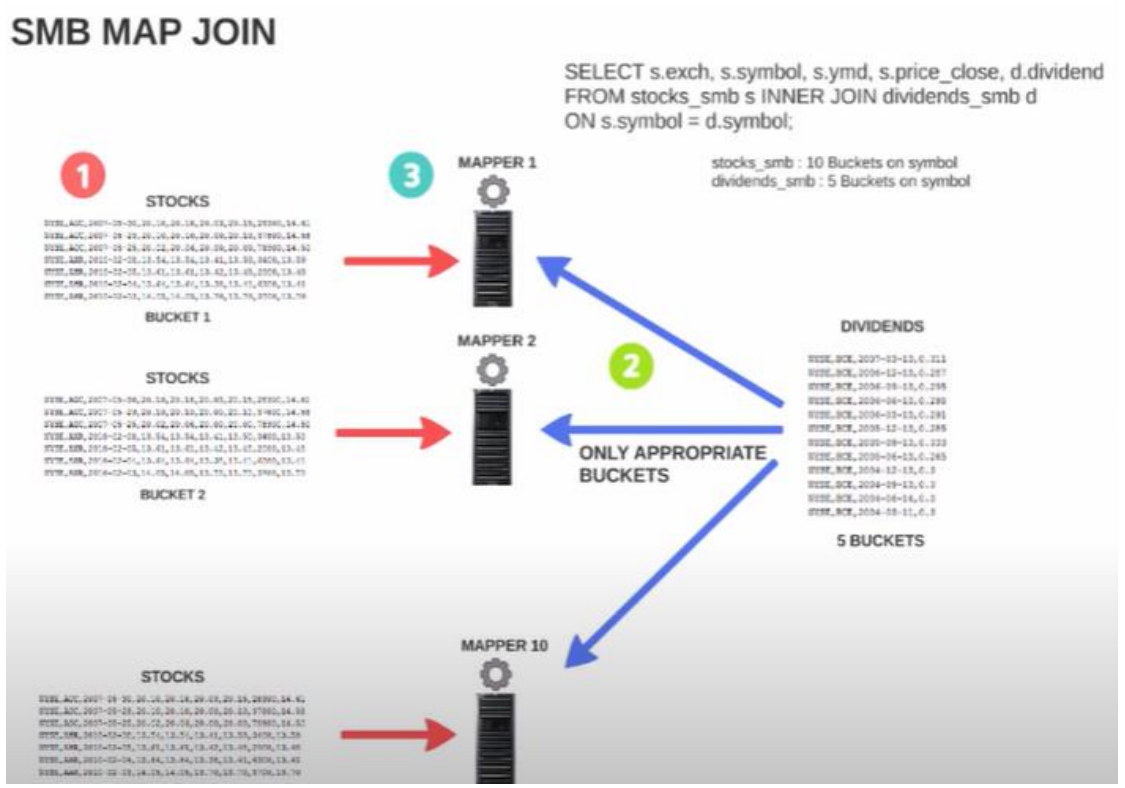
31. Is it possible to create a Cartesian join between 2 tables, using Hive?
Yes

```
join_condition
  | table_reference [CROSS] JOIN table_reference join_condition
```

32. Explain the SMB Join in Hive?

Basically, in Mapper, only Join is done. Moreover, all the buckets are joined with each other at the mapper which are corresponding.

In Hive, while each mapper reads a bucket from the first table and the corresponding bucket from the second table, in SMB join. Basically, then we perform a merge sort join feature. Moreover, we mainly use it when there is no limit on file or partition or table join.
Also, when the tables are large we can use Hive Sort Merge Bucket join. However, using the join columns, all join the columns are bucketed and sorted in SMB.
Although, make sure in SMB join all tables should have the same number of buckets.



```
SELECT s.exch, s.symbol, s.ymd, s.price_close, d.dividend
FROM stocks_smb s INNER JOIN dividends_smb d
ON s.symbol = d.symbol;
```

stocks_smb : 10 Buckets on symbol
dividends_smb : 5 Buckets on symbol

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

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

To query data in HDFS in Hive, you apply a schema to the data and then store data in ORC format. Incrementally update the imported data.
Updating imported tables involves importing incremental changes made to the original table using Sqoop and then merging changes with the tables imported into Hive.

36. Wherever (Different Directory) I run the hive query, it creates a new metastore_db, please explain the reason for it?
Basically, it creates the local metastore, while we run the hive in embedded mode.
Also, it looks whether metastore already exist or not before creating the metastore.

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?
we need to set the property hive.enforce.bucketing = true,
so that Hive knows to create the number of buckets declared in the table definition to populate the bucketed table.
38. Can a table be renamed in Hive?
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)
ALTER TABLE table_name
CHANGE COLUMN new_col INT
BEFORE x_col
40. What is serde operation in HIVE?
41. Explain how Hive Deserializes and serialises the data?
SerDe is a Library which is built-in to the Hadoop API
Hive uses Files systems like HDFS or any other storage (FTP) to store data, data here is in the form of tables (which has rows and columns).
SerDe - Serializer, Deserializer instructs hive on how to process a record (Row). Hive enables semi-structured (XML, Email, etc) or unstructured records (Audio, Video, etc)
to be processed also. For Example If you have 1000 GB worth of RSS Feeds (RSS XMLs).
You can ingest those to a location in HDFS. You would need to write a custom SerDe based on your XML structure
so that Hive knows how to load XML files to Hive tables or other way aroun

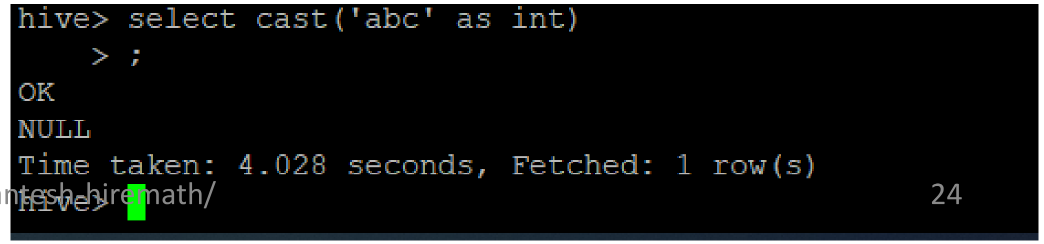
42. Write the name of the built-in serde in hive.

Also, make sure that that org.apache.hadoop.hive.serde is the deprecated old Hive SerDe library.
Hence, look at org.apache.hadoop.hive.serde2 for the latest version.

43. What is the need of custom Serde?
Despite Hive SerDe users want to write a Deserializer in most cases. It is because users just want to read their own data format instead of writing to it
By using the configuration parameter ‘regex’, the RegexDeserializer will deserialize the data, and possibly a list of column names (see serde2.MetadataTypedColumnsetSerDe).
Some important points about Writing Hive SerDe:

Basically, Hive SerDe, not the DDL, defines the table schema. Since some of the SerDe in Hive are implementations use the DDL for configuration.
However, the SerDe can also override that.
Moreover, Column types can be arbitrarily nested arrays, maps, and structures.
However, with CASE/IF or when using complex or nested types the callback design of ObjectInspector allows lazy deserialization.

44. Can you write the name of a complex data type(collection data types) in Hive?
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?
Hive> source /path/to/file/file_with_query.hql
46. What are the default record and field delimiter used for hive text files?
The default record delimiter is - \n
And the filed delimiters are - \001,\002,\003 What do you mean by schema on read?
The schema is validated with the data when reading the data and not enforced when writing data.
47. How do you list all databases in Hive whose name starts with s?
SHOW (DATABASES|SCHEMAS) [LIKE identifier_with_wildcards];
SHOW DATABASES LIKE 's%';
48. What is the difference between LIKE and RLIKE operators in Hive?
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
49. How to change the column data type in Hive?
ALTER TABLE table_name CHANGE column_name column_name new_datatype;
50. How will you convert the string ’51.2’ to a float value in the particular column?
select cast (’51.2’ as float)
51. What will be the result when you cast ‘abc’ (string) as INT?
NULL



```
hive> select cast('abc' as int)
    > ;
OK
NULL
Time taken: 4.028 seconds, Fetched: 1 row(s)
hive>
```

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;

appends the records into employees table(from staged_employees) partition (country, state) of the Hive partitioned table.
53. Write a query where you can overwrite data in a new table from the existing table.
1.Create a similar table , say tabB , with same structure.

create table tabB like tableA;
2.Then you could apply your filter and insert into this new table.

INSERT OVERWRITE TABLE tabB SELECT a.Age FROM TableA

54. What is the maximum size of a string data type supported by Hive? Explain how Hive supports binary formats.
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: It is a splittable, compressible, and row-oriented file with a general binary format.

55. What File Formats and Applications Does Hive Support?

Hive and Impala tables in HDFS can be created using text files. Sequence files, Avro data files, and Parquet file formats.
Data serialization is a way of representing data in memory as a series of bytes.
Avro is an efficient data serialization framework and is widely supported throughout Hadoop and its ecosystem

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

Using the ORC format leads to a reduction in the size of the data stored, as this file format has high compression ratios.
As the data size is reduced, the time to read and write the data is also reduced.

57. How can Hive avoid mapreduce while processing the query?
As per my knowledge all cases we can't avoid mapreduce
We can use hive query optimization techniques
Partitioning
Bucketing
Using Tez as Execution Engine
Using Compression
Using ORC Format
Join Optimizations
Cost-based Optimizer

58. What is view and indexing in hive?

The goal of Hive indexing is to improve the speed of query lookup on certain columns of a table.
Without an index, queries with predicates like 'WHERE tab1. col1 = 10' load the entire table or partition and process all the rows;

Views are generated based on user requirements. You can save any result set data as a view. The usage of view in Hive is same as that of the view in SQL. It is a standard RDBMS concept. We can execute all DML operations on a view.

Creating a View
You can create a view at the time of executing a SELECT statement. The syntax is as follows:

```
CREATE VIEW [IF NOT EXISTS] view_name [(column_name [COMMENT column_comment], ...) ]
[COMMENT table_comment]
AS SELECT ...
```

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.

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

58. What is view and indexing in hive?

The goal of Hive indexing is to improve the speed of query lookup on certain columns of a table.
Without an index, queries with predicates like 'WHERE tab1. col1 = 10' load the entire table or partition and process all the rows;

Views are generated based on user requirements. You can save any result set data as a view. The usage of view in Hive is same as that of the view in SQL. It is a standard RDBMS concept. We can execute all DML operations on a view.

Creating a View
You can create a view at the time of executing a SELECT statement. The syntax is as follows:

```
CREATE VIEW [IF NOT EXISTS] view_name [(column_name [COMMENT column_comment], ...) ]
[COMMENT table_comment]
AS SELECT ...
```

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.

```
hive> CREATE INDEX inedx_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?
No, we cannot use same name for a table and view in Hive. So we have to select a unique name for a view in Hive.

60. What types of costs are associated in creating indexes on hive tables?
There is a processing cost in arranging the values of the column on which index is created since Indexes occupies
61. Give the command to see the indexes on a table.
SHOW INDEX ON table_name
62. Explain the process to access subdirectories recursively in Hive queries.
We can use following commands in Hive to recursively access sub-directories:
hive> Set mapred.input.dir.recursive=true;
hive> Set hive.mapred.supports.subdirectories=true;
Once above options are set to true, Hive will recursively access sub-directories of a directory in MapReduce.

63. If you run a select * query in Hive, why doesn't it run MapReduce?
query you are reading every column, no filtering is required. Hence no Map phase
select * FROM TABLE –Q1 And select col1,col2 FROM TABLE –Q12
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.
query 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 coloumn! In both the queries you don't need any summary data. Hence no reducer.

so, 1st query as no map-reduce, 2nd query has only mappers but no reduces.

64. What are the uses of Hive Explode?
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.

Explode function syntax
select explode(<ARRAY>) from <table_name>;
(or)
select explode (<MAP>) from <table_name>;

It will return n number of rows where n is the size of the array/map. This function represent each element of array/map as a row.

65. What is the available mechanism for connecting applications when we run Hive as a server?
There are following ways by which you can connect with the Hive Server:
1. Thrift Client: Using thrift you can call hive commands from a various programming languages e.g. C++, Java, PHP, Python and Ruby.
2. JDBC Driver : It supports the Type 4 (pure Java) JDBC Driver
3. ODBC Driver: It supports ODBC protocol.

66. Can the default location of a managed table be changed in Hive?
using the LOCATION keyword, we can change the default location of Managed tables while creating the managed table in Hive.
67. What is the Hive ObjectInspector function?
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?
User Defined Functions, also known as UDF, allow you to create custom functions to process records or groups of records. Hive comes with a comprehensive library of functions
There are however some omissions, and some specific cases for which UDFs are the solution
Basically, we can use two different interfaces for writing Apache Hive User Defined Functions.

Simple API
Complex API

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

. Move the data to HDFS:

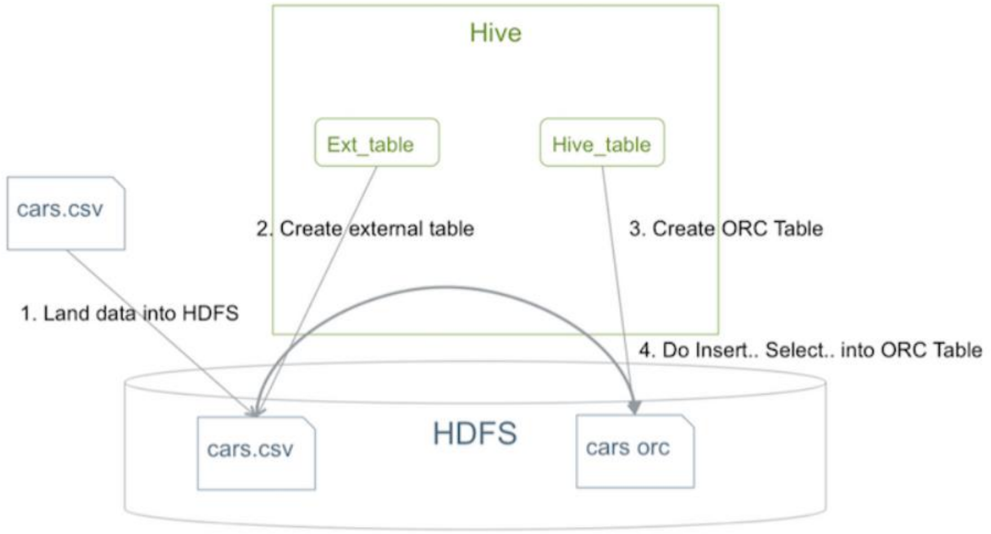
```
[<username>@cn105-10 ~]$ hdfs dfs -copyFromLocal cars.csv /user/<username>/visdata
[<username>@cn105-10 ~]$ hdfs dfs -ls /user/<username>/visdata
Found 1 items
-rwxrwxrwx   3 <username> hdfs      22100 2015-08-12 16:16 /user/<username>/visdata/cars.csv
```

```
CREATE EXTERNAL TABLE IF NOT EXISTS Cars( Name STRING, Miles_per_Gallon INT, Cylinders
INT, Displacement INT, Horsepower INT, Weight_in_lbs INT, Acceleration DECIMAL, Year
DATE, Origin CHAR(1)) COMMENT 'Data about cars from a public database' ROW FORMAT
DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE location
```

```
CREATE TABLE IF NOT EXISTS mycars( Name STRING, Miles_per_Gallon INT, Cylinders INT,
Displacement INT, Horsepower INT, Weight_in_lbs INT, Acceleration DECIMAL, Year DATE,
Origin CHAR(1)) COMMENT 'Data about cars from a public database' ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',' STORED AS ORC;
```

```
INSERT OVERWRITE TABLE mycars SELECT * FROM cars;
```

```
hive> select * from mycars limit 3;
```



70. What is TextInputFormat and SequenceFileInputFormat in hive.

TEXTFILE

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.

We can create a TEXTFILE format in Hive as follows:

create table table_name (schema of the table) row format delimited fields terminated by ',' | stored as TEXTFILE.

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.

In Hive we can create a sequence file format as follows:

create table table_name (schema of the table) row format delimited fileds terminated by ',' | stored as SEQUENCEFILE

71. How can you prevent a large job from running for a long time in a hive?
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?
We need to use Explode in Hive to convert complex data types into desired table formats.
explode UDTF basically emits all the elements in an array into multiple rows.

73. Can Hive process any type of data formats? Why? Explain in very detail
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. There are some specific file formats which Hive can handle such as:

Following are the Apache Hive different file formats:

Text File
Sequence File
RC File
AVRO File
ORC File
Parquet File

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.

Create a TEXT file by add storage option as 'STORED AS TEXTFILE' at the end of a Hive CREATE TABLE command.

Hive Text File Format Examples

Below is the Hive CREATE TABLE command with storage format specification:

```
Create table textfile_table
(column_specs)
stored as textfile;
```

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.

Create a sequence file by add storage option as 'STORED AS SEQUENCEFILE' at the end of a Hive CREATE TABLE command.

Hive Sequence File Format Example

Below is the Hive CREATE TABLE command with storage format specification:

```
Create table sequencefile_table
(column_specs)
stored as sequencefile;
```

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.

Create RCFile by specifying 'STORED AS RCFILE' option at the end of a CREATE TABLE Command:

Hive RC File Format Example

Below is the Hive CREATE TABLE command with storage format specification:

```
Create table RCfile_table
(column_specs)
stored as rcfile;
```

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.

Create AVRO file by specifying 'STORED AS AVRO' option at the end of a CREATE TABLE Command.

Hive AVRO File Format Example

Below is the Hive CREATE TABLE command with storage format specification:

```
Create table avro_table
(column_specs)
stored as avro;
```

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.

Create ORC file by specifying 'STORED AS ORC' option at the end of a CREATE TABLE Command.

Hive ORC File Format Examples

Below is the Hive CREATE TABLE command with storage format specification:

```
Create table orc_table
(column_specs)
stored as orc;
```

Hive Parquet File Format
Parquet is a column-oriented binary file format. The parquet is highly efficient for the types of large-scale 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.

Create Parquet file by specifying 'STORED AS PARQUET' option at the end of a CREATE TABLE Command.

Hive Parquet File Format Example

Below is the Hive CREATE TABLE command with storage format specification:

```
Create table parquet_table
(column_specs)
stored as parquet;
```

74. Whenever we run a Hive query, a new metastore_db is created. Why?
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.
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 ?
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 '<The table data location>' [OVERWRITE] INTO TABLE <table_name>;
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 is the precedence order in Hive configuration?
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?
WebHCat API web interface

79. Is it possible to compress json in the Hive external table ?
Just gzip your files and put them as is (*.gz) into the table location

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

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).

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

In MapReduce mode,
Pig script is executed on Hadoop cluster. The Pig scripts are converted into MapReduce jobs and then executed on Hadoop cluster (hdfs)

Local mode:

In this mode, Pig 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.