Scenario Based questions:

1.Will the reducer work or not if you use “Limit 1” in any HiveQL query?

Ans. If your query is a simple select query then no reducers are called.

If your query has something like aggregation along with group by or order by and lets say you are using MR as your execution engine then reducers will be called.

You can always use the explain keyword to see what map phases and reduce phase are involved in the query. That’s the best way to determine what goes on in each phase.

2. Suppose I have installed Apache Hive on top of my Hadoop cluster using default metastore configuration. Then, what will happen if we have multiple clients trying to access Hive at the same time?

Ans. The default metastore configuration allows only one Hive session to be opened at a time for accessing the metastore. Therefore, if multiple clients try to access the metastore at the same time, they will get an error. One has to use a standalone metastore, i.e. Local or remote metastore configuration in Apache Hive for allowing access to multiple clients concurrently.

3. Suppose, I create a table that contains details of all the transactions done by the customers: CREATE TABLE transaction\_details (cust\_id INT, amount FLOAT, month STRING, country STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ ;

Now, after inserting 50,000 records in this table, I want to know the total revenue generated for each month. But, Hive is taking too much time in processing this query. How will you solve this problem and list the steps that I will be taking in order to do so?

Ans. We can solve this problem of query latency by partitioning the table according to each month. So, for each month we will be scanning only the partitioned data instead of whole data sets.

As we know, we can’t partition an existing non-partitioned table directly. So, we will be taking following steps to solve the very problem:

1. Create a partitioned table, say partitioned\_transaction:

CREATE TABLE partitioned\_transaction (cust\_id INT, amount FLOAT, country STRING) PARTITIONED BY (month STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ‘,’ ;

2. Enable dynamic partitioning in Hive:

SET hive.exec.dynamic.partition = true;

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

3. Transfer the data from the non – partitioned table into the newly created partitioned table:

INSERT OVERWRITE TABLE partitioned\_transaction PARTITION (month) SELECT cust\_id, amount, country, month FROM transaction\_details;

Now, we can perform the query using each partition and therefore, decrease the query time.

4. How can you add a new partition for the month December in the above partitioned table?

Ans. For adding a new partition in the above table partitioned\_transaction, we will issue the command give below:

ALTER TABLE partitioned\_transaction ADD PARTITION (month=’Dec’) LOCATION ‘/partitioned\_transaction’;

5. I am inserting data into a table based on partitions dynamically. But, I received an error – FAILED ERROR IN SEMANTIC ANALYSIS: Dynamic partition strict mode requires at least one static partition column. How will you remove this error?

Ans. You cannot directly insert data(Hdfs File) into a Partitioned hive table. First you need to create a normal table, then you will insert that table data into a partitioned table.

set hive.exec.dynamic.partition.mode=strict means whenever you are populating a hive table it must have at least one static partition column.

set hive.exec.dynamic.partition.mode=nonstrict In this mode you don't need any static partition column.

6. Suppose, I have a CSV file – ‘sample.csv’ present in ‘/temp’ directory with the following entries:

id first\_name last\_name email gender ip\_address

How will you consume this CSV file into the Hive warehouse using built-in SerDe?

Ans. SerDe stands for serializer/deserializer. A SerDe allows us to convert the unstructured bytes into a record that we can process using Hive. SerDes are implemented using Java. Hive comes with several built-in SerDes and many other third-party SerDes are also available.

Hive provides a specific SerDe for working with CSV files. We can use this SerDe for the sample.csv by issuing following commands:

CREATE EXTERNAL TABLE sample

(id int, first\_name string,

last\_name string, email string,

gender string, ip\_address string)

ROW FORMAT SERDE ‘org.apache.hadoop.hive.serde2.OpenCSVSerde’

STORED AS TEXTFILE LOCATION ‘/temp’;

Now, we can perform any query on the table ‘sample’:

SELECT first\_name FROM sample WHERE gender = ‘male’;

7. Suppose, I have a lot of small CSV files present in the input directory in HDFS and I want to create a single Hive table corresponding to these files. The data in these files are in the format: {id, name, e-mail, country}. Now, as we know, Hadoop performance degrades when we use lots of small files.

So, how will you solve this problem where we want to create a single Hive table for lots of small files without degrading the performance of the system?

Ans. One can use the SequenceFile format which will group these small files together to form a single sequence file. The steps that will be followed in doing so are as follows:

* Create a temporary table:

CREATE TABLE temp\_table (id INT, name STRING, email STRING, country STRING)ROW FORMAT FIELDS DELIMITED TERMINATED BY ‘,’ STORED AS TEXTFILE;

* Load the data into temp\_table:

LOAD DATA INPATH ‘/input’ INTO TABLE temp\_table;

* Create a table that will store data in SequenceFile format:

CREATE TABLE sample\_seqfile (id INT, name STRING, e-mail STRING, country STRING)

ROW FORMAT FIELDS DELIMITED TERMINATED BY ‘,’ STORED AS SEQUENCEFILE;

* Transfer the data from the temporary table into the sample\_seqfile table:

INSERT OVERWRITE TABLE sample SELECT \* FROM temp\_table;

Hence, a single SequenceFile is generated which contains the data present in all of the input files and therefore, the problem of having lots of small files is finally eliminated.

LOAD DATA LOCAL INPATH ‘Home/country/state/’

OVERWRITE INTO TABLE address;

8. The following statement failed to execute. What can be the cause?

The local inpath should contain a file and not a directory.

9. Is it possible to add 100 nodes when we already have 100 nodes in Hive? If yes, how?

Yes, we can add the nodes by following the below steps:

Step 1: Take a new system; create a new username and password

Step 2: Install SSH and with the master node setup SSH connections

Step 3: Add ssh public\_rsa id key to the authorized keys file

Step 4: Add the new DataNode hostname, IP address, and other details in /etc/hosts slaves file:

192.168.1.102 slave3.in slave3

Step 5: Start the DataNode on a new node

Step 6: Login to the new node like suhadoop or:

ssh -X hadoop@192.168.1.103

Step 7: Start HDFS of the newly added slave node by using the following command:

./bin/hadoop-daemon.sh start data node

Step 8: Check the output of the jps command on the new node

Hive Practical questions:

Hive Join operations

Create a table named CUSTOMERS(ID | NAME | AGE | ADDRESS | SALARY)

create table CUSTOMERS(

ID int,

NAME string,

AGE int,

ADDRESS varchar,

SALARY decimal

);

Create a Second table ORDER(OID | DATE | CUSTOMER\_ID | AMOUNT)

create table ORDER(

OID big int,

DATE int,

CUSTOMER\_ID int,

AMOUNT int

);

Now perform different joins operations on top of these tables

(Inner JOIN, LEFT OUTER JOIN ,RIGHT OUTER JOIN ,FULL OUTER JOIN)

SELECT CUSTOMERS.ID, NAME, AGE, ADDRESS,SALARY, OID, DATE, AMOUNT

FROM CUSTOMERS

JOIN ORDER

ON CUSTOMERS.ID = ORDER.CUSTOMER\_ID;

SELECT CUSTOMERS.ID, NAME, AGE, SALARY, OID, DATE, AMOUNT

FROM CUSTOMERS

LEFT JOIN ORDER

ON CUSTOMERS.ID = ORDER.CUSTOMER\_ID;

SELECT CUSTOMERS.ID, NAME, AGE, SALARY, ID, ORDERDATE, AMOUNT

FROM CUSTOMERS

RIGHT JOIN ORDER

ON CUSTOMERS.ID = ORDER.CUSTOMER\_ID;

SELECT CUSTOMERS.ID, NAME, AGE, SALARY, OID, DATE, AMOUNT

FROM CUSTOMERS

FULL OUTER JOIN ORDER

ON CUSTOMERS.ID = ORDER.CUSTOMER\_ID;

BUILD A DATA PIPELINE WITH HIVE

Download a data from the given location -

https://archive.ics.uci.edu/ml/machine-learning-databases/00360/

1. Create a hive table as per given schema in your dataset

Ans create Table Airflow1(

Date timestamp,

Time timestamp,

CO(GT) decimal,

PT08.S1(CO) int,

NMHC(GT),

C6H6(GT) float,

T08.S2(NMHC) int,

NOx(GT) decimal,

PT08.S3(NOx) int,

NO2(GT) decimal,

PT08.S4(NO2),

PT08.S5(O3) int,

T float,

RH float,

AH float

);

STORED AS Airflow location '/storage/Airflow1.csv’;

2. try to place a data into table location

Ans. hadoop fs -put filepath/Airflow.csv /user

hadoop fs -cp user/Airflow.csv /storage

3. Perform a select operation .

Ans. select \* from Airflow1;

4. Fetch the result of the select operation in your local as a csv file .

Ans. insert overwrite local directory '/home/carter/staging/Airflow.csv'

row format delimited fields terminated by ',' select \* fromAirflow1;

5. Perform group by operation .

Ans. SELECT Date, count(\*) FROM Airflow1 GROUP BY date;

7. Perform filter operation at least 5 kinds of filter examples .

Ans.

SELECT \* FROM Airflow WHERE your\_date\_column BETWEEN '10/03/2004' AND '15/03/2004';

select \* from Airflow1 where RH in (60.0,76.4,69.7);

select \* from Airflow1 where RH>11.2;

select \* from Airflow1 where Time<’13:00:00’;

select \* from Airflow1 where C6H6(GT)<10.8;

8. show and example of regex operation

Ans.select name,regexp\_extract(email\_id,’@(.\*)’,1) as email\_id from emp\_info;

OK

Name email\_id

Martin gmail.com

James gmail.com

Richard yahoo.com

Time taken: 0.07 seconds,Fetched:3 row(s)

9. alter table operation

Ans. ALTER table Airflow1 Drop COLUMN NOx(GT);

10 . drop table operation

Ans. Drop TABLE IF EXISTS Airflow1;

12 . order by operation .

Ans. select \* from Airflow1 ORDER BY NOx(GT);

13 . where clause operations you have to perform .

Ans. select \* from Airflow1 where RH>11.2;

14 . sorting operation you have to perform .

Ans. select \* from Airflow sort by RH desc;

15 . distinct operation you have to perform .

Ans. Select distinct CO(GT), PT08.S3(NOx) from Airflow1;

16 . like an operation you have to perform .

Ans. SELECT \* FROM Airflow1

WHERE Date LIKE 'e%';

17 . union operation you have to perform .

Ans. SELECT CO(GT),PT08.S1(CO) FROM Airflow1

UNION

SELECT CO(GT),PT08.S1(CO) FROM Airflow2;

18 . table view operation you have to perform .

Ans. Create VIEW Airflow\_1 AS SELECT \* from Airflow1 where RH>58.4

hive operation with python

Create a python application that connects to the Hive database for extracting data, creating sub tables for data processing, drops temporary tables.fetch rows to python itself into a list of tuples and mimic the join or filter operations

**To install you'll need these libraries:**

pip install sasl

pip install thrift

pip install thrift-sasl

pip install PyHive

**After installation, you can connect to Hive like this:**

from pyhive import hive

conn = hive.Connection(host="YOUR\_HIVE\_HOST", port=PORT, username="YOU")

**use the connection to make a Pandas dataframe:**

import pandas as pd

df = pd.read\_sql("SELECT \* FROM Employee", conn)

**creating sub tables for data processing**

df[[‘Empid’,'Age','NAME',’SALARY’,’Experience’,’No’,’Address’,’Rating’]]

**drops temporary column**

df.drop(labels=3)

df.drop('’Rating’', axis=1)

**Data to Tuple**

subset = df[[‘Empid’,'Age','NAME',’SALARY’]]

tuples = [tuple(x) for x in subset.to\_numpy()]

**Cover Data To List**

df.to\_records().tolist()

(or)

**list(df.to\_records())**

**Join**

res = df.copy(deep=True)

df1 = df.join(res,on=’EmpId’,how='inner')

**Regex**

**# columns which have letters 'a' or 'A' in its name.**

df.filter(regex ='[aA]')