211720104069

S.Jeevasakthi

#Question 1: What is a metastore in Hive?

Metastore is the central repository of Apache Hive metadata.
It stores metadata for Hive tables (like their schema and location) And
partitions in a relational database. It provides client access to
This information by using metastore service API.

#Question 2: Where does the data of a Hive table gets stored?

Hive stores its database and table metadata in a metastore,
Which is a database or file backed store that enables easy data abstraction and discovery.

#Question 3: Why Hive does 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.

#Question 4: What is the difference between local and remote metastore?

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.

#Question 5: What is the default database provided by Apache Hive for metastore?

In 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.

#Question 6: What is the difference between external table and managed table?

Managed tables are Hive owned tables where the entire lifecycle of the tables'

Data are managed and controlled by Hive.

External tables are tables where Hive has loose coupling with the data.

#Question 7: Is it possible to change the default location of a managed table?

Yes, you can do it by using the clause – LOCATION '<hdfs_path>'

We can change the default location of a managed table.

#Question 8: What is a partition in Hive?

Hive organizes tables into partitions for grouping similar type of data together based on a column or partition key.

Each Table can have one or more partition keys to identify a particular partition.

Physically, a partition is nothing but a sub-directory in the table directory.

#Question 9: Why do we perform partitioning in Hive?

Partitioning provides granularity in a Hive table and therefore,

Reduces the query latency by scanning only relevant partitioned data instead of the whole data set.

For example, we can partition a transaction log of an e – commerce website

Based on month like Jan, February, etc. So, any analytics regarding a particular month, say Jan, Will have to scan the Jan partition (sub – directory) only instead of the whole table data.

#Question 10: What is dynamic partitioning and when is it used?

Dynamic partitioning is the strategic approach to load the data from the non-partitioned table Where the single insert to the partition table is called a dynamic partition.

#Question 11: Suppose, you create a table that contains details of all the transactions done

By the customers of year 2022: CREATE TABLE transaction_details (cust_id

INT, amount FLOAT, month STRING, country STRING) ROW FORMAT

DELIMITED FIELDS TERMINATED BY ‘,’ ;

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

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:

Create a partitioned table, say partitioned_transaction:

1. 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.