NAAN MUDHALVAN

BIG DATA ANALYTICS

ASSESSMENT – V

Querying Data using Hive

Homework

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.

2. **Where does the data of a Hive table gets stored?**

   Hive data are stored in one of Hadoop compatible filesystem: S3, HDFS or other compatible filesystem.
   Hive metadata are stored in RDBMS like MySQL, see supported RDBMS.
   The location of Hive tables data in S3 or HDFS can be specified for both managed and external tables.
   The difference between managed and external tables is that DROP TABLE statement, in managed table, will drop the table and delete table's data. Whereas, for external table DROP TABLE will drop only the table and data will remain as is and can be used for creating other tables over it.

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.

4. **What is the difference between local and remote metastore?**

   Local Metastore:

   In local metastore configuration, the metastore service runs in the same JVM in which the Hive service is running and connects to a database running in a separate JVM, either on the same machine or on a remote machine.

   Remote Metastore:

In the remote metastore configuration, the metastore service runs on its own separate JVM and not in the Hive service JVM. Other processes communicate with the metastore server using Thrift Network APIs. You can have one or more metastore servers in this case to provide more availability.

5. What is the default database provided by Apache Hive for metastore?

By default, Hive provides an embedded Derby database instance backed by the local disk for the metastore. This is called the embedded metastore configuration.

6. What is the difference between external table and managed table?

Here is the key difference between an external table and managed table:

- In case of managed table, If one drops a managed table, the metadata information along with the table data is deleted from the Hive warehouse directory.
- On the contrary, in case of an external table, Hive just deletes the metadata information regarding the table and leaves the table data present in HDFS untouched.

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

Yes, it is possible to change the default location of a managed table. It can be achieved by using the clause – LOCATION '<hdfs_path>'.

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.

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.

10. What is dynamic partitioning and when is it used?

In dynamic partitioning values for partition columns are known in the runtime, i.e. It is known during loading of the data into a Hive table.

One may use dynamic partition in following two cases:

Loading data from an existing non-partitioned table to improve the sampling and therefore, decrease the query latency.

When one does not know all the values of the partitions before hand and therefore, finding these partition values manually from a huge data sets is a tedious task.

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:

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.