Q1. Why does Hive not store metadata information in HDFS?

Ans. Using RDBMS instead of HDFS, Hive stores metadata information in the metastore. Basically, to achieve low latency we use RDBMS. Because **HDFS read/write operations** are time-consuming processes.

Q2. When should we use SORT BY instead of ORDER BY?

Ans. Despite **ORDER BY** we should use SORT BY. Especially while we have to sort huge datasets. The reason is SORT BY clause sorts the data using multiple reducers. ORDER BY sorts all of the data together using a single reducer.

Hence, using ORDER BY will take a lot of time to execute a large number of inputs.

Q3. How Hive distributes the rows into buckets?

Ans. By using the formula: hash_function (bucketing_column) modulo (num_of_buckets) Hive determines the bucket number for a row. Basically, hash_function depends on the column data type. Although, hash_function for integer data type will be:

hash_function (int_type_column)= value of int_type_column

Q4. Wherever (Different Directory) I run the hive query, it creates new metastore_db, please explain the reason for it?

Ans. 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. Hence, in configuration file hive-site.xml. Property is “javax.jdo.option.ConnectionURL” with default value “jdbc:derby:;databaseName=metastore_db;create=true” this property is defined. Hence, to change the behavior change the location to the absolute path, thus metastore will be used from that location.

Q5. How can you configure remote metastore mode in Hive?

Ans. Basically, hive-site.xml file has to be configured with the below property, to configure metastore in Hive –

hive.metastore.uris

thrift: //node1 (or IP Address):9083

IP address and port of the metastore host

Q6. What is ObjectInspector functionality?

Ans. To analyze the structure of individual columns and the internal structure of the row objects we use ObjectInspector. Basically, it provides access to complex objects which can be stored in multiple formats in Hive.

Q7. What are the different components of a Hive architecture?

Ans. The different components of the Hive are:

- **User Interface**: This calls the execute interface to the driver and creates a session for the query. Then, it sends the query to the compiler to generate an execution plan for it
- **Metastore**: This stores the metadata information and sends it to the compiler for the execution of a query
- **Compiler**: This generates the execution plan. It has a DAG of stages, where each stage is either a metadata operation, a map, or reduces a job or operation on HDFS
- **Execution Engine**: This acts as a bridge between the Hive and Hadoop to process the query. Execution Engine communicates bidirectionally with Metastore to perform operations, such as create or drop tables.

Q8. What is a partition in Hive and why is partitioning required in Hive

Ans. Partition is a process for grouping similar types of data together based on columns or partition keys. Each table can have one or more partition keys to identify a particular partition.

Partitioning provides granularity in a Hive table. It reduces the query latency by scanning only relevant partitioned data instead of the entire data set. We can partition the transaction data for a bank based on month — January, February, etc. Any operation regarding a particular month, say February, will only have to scan the February partition, rather than the entire table data.

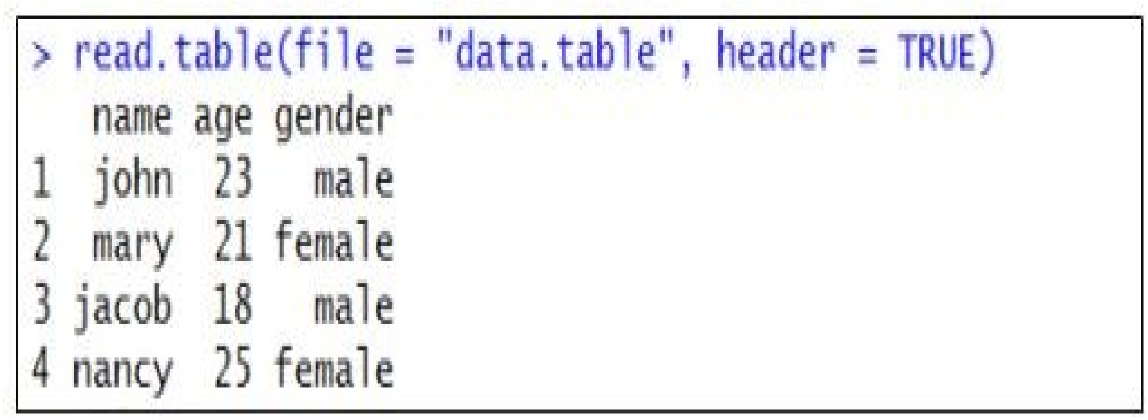
Q9. **Why does Hive not store metadata information in HDFS?**

Ans. We know that the Hive's data is stored in HDFS. However, the metadata is either stored locally or it is stored in RDBMS. The metadata is not stored in HDFS, because HDFS read/write operations are time-consuming. As such, Hive stores metadata information in the metastore using RDBMS instead of HDFS. This allows us to achieve low latency and is faster.

Q10. **Suppose there are several small CSV files present in /user/input directory in HDFS and you want to create a single Hive table from these files. The data in these files have the following fields: {registration_no, name, email, address}. What will be your approach to solve this, and**

where will you create a single Hive table for multiple smaller files without degrading the performance of the system?

Ans. Using SequenceFile format and grouping these small files together to form a single sequence file can solve this problem. Below are the steps:



```
> read.table(file = "data.table", header = TRUE)
   name age gender
1  john  23   male
2  mary  21 female
3 jacob  18   male
4 nancy  25 female
```

Q11. Write a query to insert a new column(new_col INT) into a hive table (h_table) at a position before an existing column (x_col).

Ans. The following query will insert a new column:

ALTER TABLE h_table

CHANGE COLUMN new_col INT

BEFORE x_col