**Hive interview Questions**

1. What is the definition of Hive? What is the present version of Hive? Ans. Apache Hive is a data warehouse software project built on top of Apache Hadoop for providing data query and analysis. Hive gives an SQL-like interface to query data stored in various databases and file systems that integrate with Hadoop.

2. Is Hive suitable to be used for OLTP systems? Why?

Ans. Is Hive suitable to be used for OLTP systems? Why? **No Hive does not provide insert and update at row level**. So it is not suitable for OLTP system.

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

Ans. Hive supports all ACID properties which enable us to use transactions, create transactional tables, and run queries like Insert, Update, and Delete on tables. Hive is a data warehouse database where the data is typically loaded from batch processing for analytical purposes and older versions of Hive doesn’t support ACID transactions on tables.

4. Explain the hive architecture and the different components of a Hive architecture?

Ans. This component diagram contains different units.

**User Interface:**Hive is a data warehouse infrastructure software that can create interaction between user and HDFS. The user interfaces that Hive supports are Hive Web UI, Hive command line, and Hive HD Insight (In Windows server).

**Meta Store:**Hive chooses respective database servers to store the schema or Metadata of tables, databases, columns in a table, their data types, and HDFS mapping.

**HiveQL Process Engine:**HiveQL is similar to SQL for querying on schema info on the Metastore. It is one of the replacements of traditional approach for MapReduce program. Instead of writing MapReduce program in Java, we can write a query for MapReduce job and process it.

**Execution Engine:**The conjunction part of HiveQL process Engine and MapReduce is Hive Execution Engine. Execution engine processes the query and generates results as same as MapReduce results. It uses the flavor of MapReduce.

**HDFS or HBASE:**Hadoop distributed file system or HBASE are the data storage techniques to store data into file system.

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

Ans. Following are the 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?

Ans. **Local mode**: In Hive local mode, Map Reduce jobs related to Hive run locally on a user machine. This is the default mode in which Hadoop uses local file system.

* **Distributed Mode**: In this mode, Hive as well as Hadoop is running in a fully distributed mode. NameNode, DataNode, JobTracker, TaskTracker etc run on different machines in this mode.
* **Pseudo-distributed Mode**: This is the mode used by developers to test the code before deploying to production. In this mode, all the daemons run on same virtual machine. With this mode, we can quickly write scripts and test on limited data sets

7. Features and Limitations of Hive.

Ans. The Various key-features of Apache Hive are:

1. Open-source: Apache Hive is an open-source tool. We can use it free of cost.

2. Query large datasets: Hive can query and manage huge datasets stored in Hadoop Distributed File System.

3. Multiple-users: Multiple users can query the data using Hive Query Language simultaneously.

4. Backward compatible: Apache Hive perfectly fits the low level interface requirement of Apache Hadoop.

5. Partitioning and Bucketing: Apache Hive supports partitioning and bucketing of data at the table level to improve performance.

6. File-formats: Hive provides support for various file formats such as textFile, ORC, Avro Files, SequenceFile, Parquet, RCFile, LZO Compression etc.

7. Hive Query Language: Hive uses Hive Query Language which is similar to SQL. We do not require any knowledge of programming languages to work with Hive. Only the knowledge of basic SQL query is enough to work with Hive.

8. Built-In function: Hive provides various Built-In functions.

9. User-Defined Functions: It also provides support for User-Defined Functions for the tasks like data cleansing and filtering. We can define UDFs according to our requirements

10. External Table: Apache Hive supports external tables. This allows us to process data without actually storing data in HDFS.

11. Fast: Hive is a fast, scalable, extensible tool and uses familiar concepts.

12. Warehouse: Apache Hive is a distributed data warehouse tool.

13. Table Structure: Table structure in Hive is similar to table structure in RDBMS.

14. ETL support: Hive supports ETL operations. Hive is an effective ETL tool.

15. Storage: Hive allows us to access files stored in HDFS and other similar data storage systems such as HBase.

16. OLAP: Hive is designed for OLAP (Online Analytical Processing).

17. Client application: Hive can support client applications written in PHP, Python, Java, C++ and Ruby.

18. Rule Based Optimizer: Hive has a rule based optimizer for optimizing logical plans.

19. Ad-hoc queries: Hive allows us to run Ad-hoc queries which are the loosely typed command or query whose value depends on some variable for the data analysis.

20. Data Visualization: Hive can be used for Data Visualization. Integrating Hive with Apache Tez will provide the real time processing capabilities.

Let us now study some limitations of Apache Hive.

### Limitations

1. Hive is not designed for the OLTP (Online transaction processing). We can use it for OLAP.

2. It does not offer real-time queries.

3. It provides limited subquery support.

4. Latency of Hive is generally very high.

8. How to create a Database in HIVE?

Ans. hive> CREATE SCHEMA userdb;

9. How to create a table in HIVE?

Ans. CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db\_name.] table\_name

[(col\_name data\_type [COMMENT col\_comment], ...)]

[COMMENT table\_comment]

[ROW FORMAT row\_format]

[STORED AS file\_format]

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

Ans. 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?

Ans. create external table employee (id int, name string)

lines terminated by '\n'

location '/user/hirw/employees’

tblproperties ("skip.header.line.count"="1");

12.What is a hive operator? What are the different types of hive operators?

Ans. 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

Ans. 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) | It 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 representation. 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.

Ans. Hive DML (Data Manipulation Language) commands are used to insert, update, retrieve, and delete data from the Hive table once the table and database schema has been defined using Hive DDL commands.

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.

Ans.SORT BY:Hive uses the columns in SORT BY to sort the rows before feeding the rows to a reducer. The sort order will be dependent on the column types. If the column is of numeric type, then the sort order is also in numeric order. If the column is of string type, then the sort order will be lexicographical order.

### ORDER BY:In Hive, ORDER BY guarantees total ordering of data, but for that, it has to be passed on to a single reducer, which is normally performance-intensive and therefore in strict mode, hive makes it compulsory to use LIMIT with ORDER BY so that reducer doesn’t get overburdened.

### DISTRIBUTE BY:Hive uses the columns in Distribute By to distribute the rows among reducers. All rows with the same Distribute By columns will go to the same reducer.

It ensures each of N reducers gets non-overlapping ranges of the column, but doesn’t sort the output of each reducer. You end up with N or more unsorted files with non-overlapping ranges.

**Cluster By:** is a short-cut for both Distribute By and Sort By.

CLUSTER BY x ensures each of N reducers gets non-overlapping ranges, then sorts by those ranges at the reducers.

16.Difference between "Internal Table" and "External Table" and Mention when to choose “Internal Table” and “External Table” in Hive?

Ans. For External Tables, Hive stores the data in the LOCATION specified during creation of the table(generally not in warehouse directory). If the external table is dropped, then the table metadata is deleted but not the data. For Internal tables, Hive stores data into its warehouse directory

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

Ans. 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.

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

Ans. Yes, you can do it by using the clause – LOCATION '<hdfs\_path>' we can 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?

Ans. Hive Metastore is a component in Hive that stores the catalog of the system that contains the metadata about Hive create columns, Hive table creation, and partitions. Metadata is mostly stored in the traditional form of RDBMS. The Apache Hives make use of the Derby databases to store the metadata.

Derby is the default database for the embedded metastore. Derby embeded JDBC driver class. Unit test data goes in here on your local filesystem. If you want to run Derby as a network server so the metastore can be accessed from multiple nodes, see Hive Using Derby in Server Mode.

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

Ans. 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?

Ans. The partitioning in Hive means dividing the table into some parts based on the values of a particular column like date, course, city or country. The advantage of partitioning is that since the data is stored in slices, the query response time becomes faster.

22.What is the difference between dynamic partitioning and static partitioning?

Ans. While the system is up and running, static partitions are continuously present. Dynamic partitions are created and activated only as needed. Also, the space which dynamic partitions occupy during job processing is released when processing is finished.

23.How do you check if a particular partition exists?

Ans. SHOW PARTITIONS table\_name

PARTITION(partitioned\_column=’partition\_value’)

24.How can you stop a partition form being queried?

Ans. By using the ENABLE OFFLINE clause with ALTER TABLE statement.

25.Why do we need buckets? How Hive distributes the rows into buckets?

Ans. The bucketing in Hive is a data organizing technique. It is similar to partitioning in Hive with an added functionality that it divides large datasets into more manageable parts known as buckets. So, we can use bucketing in Hive when the implementation of partitioning becomes difficult.

26.In Hive, how can you enable buckets?

Ans. The command set hive. enforce. bucketing = true; allows the correct number of reducers and the cluster by column to be automatically selected based on the table. Otherwise, you would need to set the number of reducers to be the same as the number of buckets as in set mapred.

27.How does bucketing help in the faster execution of queries?

Ans.Bucketing in hive is the concept of breaking data down into ranges, which are known as buckets, to give extra structure to the data so it may be used for more efficient queries. The range for a bucket is determined by the hash value of one or more columns in the dataset (or Hive metastore table).

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

Ans.Hive performance tuning refers to the collective processes and steps designed to improve and accelerate the performance of your Hive environments. When queries are not optimized, simple statements take longer to execute, resulting in performance lags and downtime.

1. 1 Avoid locking of tables. ...
2. 2 Use the Hive execution engine as TEZ. ...
3. 3 Use Hive Cost Based Optimizer (CBO) ...
4. 4 Parallel execution at a Mapper & Reducer level. ...
5. 5 Use STREAMTABLE option. ...
6. 6 Use Map Side JOIN Option. ...
7. 7 Avoid Calculated Fields in JOIN and WHERE clause.

29. What is the use of Hcatalog?

Ans. 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.

Ans. JOIN is a clause that is used for combining specific fields from two tables by using values common to each one. It is used to combine records from two or more tables in the database.

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

Ans. Cross join, also known as Cartesian product, is a way of joining multiple tables in which all the rows or tuples from one table are paired with the rows and tuples from another table.

32.Explain the SMB Join in Hive?

Ans. SMB is a join performed on bucket tables that have the same sorted, bucket, and join condition columns. It reads data from both bucket tables and performs common joins (map and reduce triggered) on the bucket tables. We need to enable the following properties to use SMB: > SET hive.

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

Ans. 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.

34.What is the usefulness of the DISTRIBUTED BY clause in Hive?

Ans. DISTRIBUTE BY clause is used to distribute the input rows among reducers. It ensures that all rows for the same key columns are going to the same reducer. So, if we need to partition the data on some key column, we can use the DISTRIBUTE BY clause in the hive queries.

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

Ans. **Moving data from HDFS to Apache Hive**

1. Ingest the data. You create a single Sqoop import command that imports data from diverse data sources, such as a relational database, into HDFS.
2. Convert the data to ORC format. ...
3. Incrementally update the imported data.

36.Wherever (Different Directory) I run the hive query, it creates a 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.

37.What will happen in case you have not issued the command: ‘SET hive.enforce.bucketing=true;’ before bucketing a table in Hive?

Ans. The command: 'SET hive. enforce. bucketing=true;' allows one to have the correct number of reducer while using 'CLUSTER BY' clause for bucketing a column. In case it's not done, one may find the number of files that will be generated in the table directory to be not equal to the number of buckets.

38.Can a table be renamed in Hive?

Ans. 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)

Ans. ALTER TABLE table\_name

CHANGE COLUMN new\_col INT

BEFORE x\_col

40.What is serde operation in HIVE?

Ans. SerDe is short for Serializer/Deserializer. Hive uses the SerDe interface for IO. The interface handles both serialization and deserialization and also interpreting the results of serialization as individual fields for processing.

41.Explain how Hive Deserializes and serialises the data?

Ans. The interface handles both serialization and deserialization and also interpreting the results of serialization as individual fields for processing. A SerDe allows Hive to read in data from a table, and write it back out to HDFS in any custom format.

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

Ans. SerDe is short for Serializer/Deserializer. Hive uses the SerDe interface for IO. The interface handles both serialization and deserialization and also interpreting the results of serialization as individual fields for processing.

43.What is the need of custom Serde?

Ans. Main use of SerDe interface is for IO operations. A SerDe allows hive to read the data from the table and write it back to the HDFS in any custom format. If we have unstructured data, then we use RegEx SerDe which will instruct hive how to handle that record. We can also write our own Custom SerDe in any format.

44.Can you write the name of a complex data type(collection data types) in Hive?

Ans. There are three complex types in hive, arrays: It is an ordered collection of elements. The elements in the array must be of the same type. map: It is an unordered collection of key-value pairs.

45.Can hive queries be executed from script files? How?

Ans. It is possible by using the source command.

For example −

Hive> source /path/to/file/file\_with\_query.hql

46.What are the default record and field delimiter used for hive text files?

Ans. 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?

Ans. To list out the databases in Hive warehouse, enter the command 'show databases'. The database creates in a default location of the Hive warehouse. In Cloudera, Hive database store in a /user/hive/warehouse.

48.What is the difference between LIKE and RLIKE operators in Hive?

Ans. One significant difference between LIKE / RLIKE and the full-text search predicates is that the former act on exact fields while the latter also work on analyzed fields.

49.How to change the column data type in Hive?

Ans. By using this command below one can change the column data type: 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?

Ans. convert a string '51.2' to a float value in Python by using the float() function:

String=’51.

51.What will be the result when you cast ‘abc’ (string) as INT?

Ans. This method returns the string as a primitive type int. If the string does not contain a valid integer then it will throw a NumberFormatException.

52.What does the following query do?

Ans. a. INSERT OVERWRITE TABLE employees

b. PARTITION (country, state)

c. SELECT ..., se.cnty, se.st

d. FROM staged\_employees se;

Ans. INSERT OVERWRITE TABLE employees: The INSERT OVERWRITE clause specifies that the existing data in the employees table will be completely overwritten by the result of the SELECT statement.

PARTITION (country, state): The PARTITION clause is used to partition the data in the employees table based on the values of country and state. Partitioning the data makes it more organized and efficient to retrieve and process.

SELECT ..., se.cnty, se.st: The SELECT statement retrieves data from the staged\_employees table (se) and selects some columns, including cnty and st, which are used to populate the country and state partitions in the employees table. The dots (...) represent other columns that may also be selected in the query.

FROM staged\_employees se: The FROM clause specifies the source of the data that is being inserted into the employees table. In this case, the data is being retrieved from the staged\_employees table (se), which is aliased as staged\_employees for the purpose of the query.

In summary, this query inserts data into the employees table by overwriting the existing data, partitions the data based on the values of country and state, and retrieves the data from the staged\_employees table.

53.Write a query where you can overwrite data in a new table from the existing table.

Ans. CREATE TABLE NEW\_TABLE\_NAME AS

SELECT [ column1, column2...columnN ]

FROM EXISTING\_TABLE\_NAME

[ WHERE ]

54.What is the maximum size of a string data type supported by Hive? Explain how Hive supports binary formats.

Ans. he 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?

Ans. Hive facilitates managing large data sets supporting multiple data formats, including comma-separated value (. csv) TextFile, RCFile, ORC, and Parquet. The PXF Hive connector reads data stored in a Hive table.

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

Ans. 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. The ORC format improves query performance also by the way it stores data in a file.

57.How can Hive avoid mapreduce while processing the query?

Ans.You can make Hive avoid MapReduce to return query results by setting the hive. exec. mode. local.

58.What is view and indexing in hive?

Ans. Basically, Apache Hive View is similar to Hive tables, that are generated on the basis of requirements.

Apache Hive View on External Table

* As a Hive view, we can save any result set data.
* Well, we can say its usage is as same as the use of views in SQL.
* Although, we can perform all type of DML operations on Hive views.

On defining indexing in Hive we can say these are pointers to particular column name of a table.

* However, the user has to manually define the Hive index
* Basically, we are creating the pointer to particular column name of the table, wherever we are creating Hive index.
* By using the Hive index value created on the column name, any Changes made to the column present in tables are stored.

59.Can the name of a view be the same as the name of a hive table?

Ans. The name of a view must be unique, and it cannot be the same as any table or database or view's name.

60.What types of costs are associated in creating indexes on hive tables?

Ans. Basically, 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.

Ans.To see the index for a specific table use SHOW INDEX: SHOW INDEX FROM yourtable; To see indexes for all tables within a specific schema you can use the STATISTICS table from INFORMATION\_SCHEMA: SELECT DISTINCT TABLE\_NAME, INDEX\_NAME FROM INFORMATION\_SCHEMA.

62. Explain the process to access subdirectories recursively in Hive queries.

Ans. 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;

63.If you run a select \* query in Hive, why doesn't it run MapReduce?

Ans. Map acts like a filter - like “where” on rows and like “select <field list>” on columnsReducer acts like an aggregator- it kinda does the job of a “group by”When you enter Select \* from table - there is no where, field list of a group by, so there is no need for a mapper to run or a reducer to run. It then becomes a simple “Load the table and display results”

64.What are the uses of Hive Explode?

Ans. The explode function explodes an array to multiple rows. Returns a row-set with a single column (col), one row for each element from the array.

65. What is the available mechanism for connecting applications when we run Hive as a server?

Ans. Thrift Client: Using Thrift, we can call Hive commands from various programming languages, such as C++, PHP, Java, Python, and Ruby.

66.Can the default location of a managed table be changed in Hive?

Ans. Yes, you can do it by using the clause – LOCATION '<hdfs\_path>' we can change the default location of a managed table.

67.What is the Hive ObjectInspector function?

Ans. A key concept when working with Generic UDF and UDAF is the ObjectInspector. In generic UDFs, all objects are passed around using the Object type. Hive is structured this way so that all code handling records and cells is generic, and to avoid the costs of instantiating and deserializing objects when it's not needed.

68.What is UDF in Hive?

Ans. Hive Functions: **User Defined Functions** (UDFs) in hive are used to plug in our own logic in terms of code into hive when we are not able to get the desired result from hive's built in functions. We can invoke the UDFs from hive query.

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

Ans. CREATE TABLE <hive\_table\_name>

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

AS

SELECT\*FROM <hdfs\_file\_path>;

70.What is TextInputFormat and SequenceFileInputFormat in hive.

Ans. TextInputFormat and SequenceFileInputFormat are two input formats used in Apache Hive to read data from HDFS.

**TextInputFormat:** This input format is used to read plain text files and treat each line as a record in Hive. It is the default input format in Hive and is used when reading delimited text files, such as CSV or tab-delimited files.

**SequenceFileInputFormat:** This input format is used to read data stored in Hadoop's SequenceFile format, which is a binary file format for storing key-value pairs. This format is used when data needs to be stored in a compact, binary form, and is optimized for reading and writing data in a MapReduce context.

In summary, TextInputFormat is suitable for reading plain text files, while SequenceFileInputFormat is more suitable for reading binary data stored in a key-value format.

71.How can you prevent a large job from running for a long time in a hive?

Ans. There are several ways to prevent a large job from running for a long time in Hive:

1.Indexing: Indexing the data can help speed up query execution time, as Hive can use the index to skip over irrelevant data.

2.Partitioning: Partitioning the data into smaller chunks based on a column can help reduce the amount of data that needs to be processed, thereby reducing the running time.

3.Bucketing: Bucketing the data into smaller chunks based on the value of a column can help Hive distribute the data evenly across multiple reducers, reducing the overall running time.

4.MapReduce Settings: You can adjust the settings of your MapReduce job, such as increasing the number of reducers or increasing the amount of memory available to the job, to help it run faster.

5.Materialized Views: Creating materialized views of frequently used data sets can help speed up query execution time, as the view data can be pre-aggregated and stored, reducing the amount of data that needs to be processed at runtime.

6.Query Optimization: Writing efficient Hive queries that use appropriate functions and algorithms can also help reduce running time.

7.Cluster Configuration: Increasing the number of nodes in the cluster, or using a more powerful type of node, can help reduce the running time of large jobs.

72.When do we use explode in Hive?

Ans. The explode function explodes an array to multiple rows. Returns a row-set with a single column (col), one row for each element from the array.

73.Can Hive process any type of data formats? Why? Explain in very detail

Ans. Hive can process a variety of data formats, including plain text, binary, and serialized formats. Some of the most common data formats that Hive can process include:

1.Delimited text files (CSV, tab-delimited, etc.): Hive can process plain text files where fields are separated by a delimiter, such as a comma or a tab. These files are usually human-readable and easy to generate, making them a popular choice for data storage and transfer.

2.Sequence files: Hive can process Hadoop's SequenceFile format, which is a binary format for storing key-value pairs. Sequence files are compact and optimized for reading and writing data in a MapReduce context, making them a popular choice for storing intermediate data in big data workflows.

3.RCFile: RCFile is a row-columnar format that is optimized for reading and writing large amounts of data in Hive. RCFile stores data in a column-wise fashion, which can lead to more efficient compression and faster query execution times compared to row-based formats like text and sequence files.

4.ORCFile: ORCFile is a highly optimized file format for storing large amounts of data in Hive. ORCFile uses a combination of compression, encoding, and column pruning to reduce the amount of data that needs to be processed during query execution, resulting in faster query times compared to other formats.

5.Parquet: Parquet is a columnar storage format that is optimized for use in big data processing and analytical workflows. Parquet is designed to work well with complex data structures, making it a good choice for storing and processing large amounts of semi-structured or structured data in Hive.

74.Whenever we run a Hive query, a new metastore\_db is created. Why?

Ans. Therefore, we have to change the behavior of the location to an absolute path so that from that location the metastore can be used.

75.Can we change the data type of a column in a hive table? Write a complete query.

Ans. Yes, you can change the data type of a column in a Hive table. Here's a complete example query to change the data type of a column in a Hive table:

ALTER TABLE <table\_name> CHANGE COLUMN <column\_name> <column\_name> <new\_data\_type>;

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 ?

Ans. While loading data into a Hive table using the LOAD DATA clause, you specify that the file is in HDFS and not a local file by including the HDFS file path in the LOAD DATA statement.

Here's an example of how you can load data into a Hive table from an HDFS file using the LOAD DATA clause:

LOAD DATA INPATH 'hdfs://<hdfs\_path>/<file\_name>' INTO TABLE <table\_name>;

Note: You need to replace <hdfs\_path> with the HDFS path of your data file and <file\_name> with the name of your data file, and <table\_name> with the name of the Hive table into which you want to load the data.

77.What is the precedence order in Hive configuration?

Ans. In Hive, the precedence order of configuration settings determines which value takes precedence when there are multiple sources of configuration information. The following is the precedence order of Hive configuration in descending order:

Command-line options: Command-line options specified when starting the Hive client or a Hive service such as HiveServer2 take the highest precedence and override any other configuration settings.

Hive configuration file: Hive configuration file (hive-site.xml) contains settings specific to your Hive installation. Settings in the Hive configuration file take precedence over defaults in the Hive code.

Environmental variables: Environmental variables can be used to set configuration options in Hive. They take precedence over settings in the Hive configuration file but have lower precedence than command-line options.

Defaults: Finally, Hive uses default values for configuration settings that are not specified in any of the higher precedence sources. These default values are built into the Hive code and can be overridden by specifying values in higher precedence sources.

It's important to keep in mind that when multiple sources of configuration information are present, the settings in the highest precedence source will override settings in lower precedence sources. This allows you to customize your Hive configuration to meet your specific requirements.

78.Which interface is used for accessing the Hive metastore?

Ans. WebHCat API web interface can be used for Hive commands. It is a REST API that allows applications to make HTTP requests to access the Hive metastore (HCatalog DDL).

79.Is it possible to compress json in the Hive external table?

Ans. Yes, it is possible to compress JSON data in a Hive external table. Hive supports several compression formats, including GZIP, Snappy, LZO, and BZIP2, among others.

To store compressed JSON data in a Hive external table, you need to create the table with the appropriate storage format and compress the data before loading it into the table. For example, to create a Hive external table for compressed JSON data stored in GZIP format, you can use the following query:

CREATE EXTERNAL TABLE <table\_name> (

<column\_name\_1> <data\_type\_1>,

<column\_name\_2> <data\_type\_2>,

...

)

ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'

STORED AS TEXTFILE

LOCATION '<hdfs\_path>'

TBLPROPERTIES ('compression.type'='GZIP');

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

Ans. 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?

Ans. can use Hadoop archiving to reduce the number of hdfs files in the Hive table partition. Hive has built in functions to convert Hive table partition into Hadoop Archive (HAR). HAR does not compress the files, it is analogous to the Linux tar command.

82.What is DBPROPERTY in Hive?

Ans. The DB properties are nothing but mentioning the details about the database created by the user. Suppose the name of the user, the type of the database and the tables it has, the date on which the database is created etc. This makes the other user easy the recognize the database and use it according to the requirement.

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

Ans. Local mode is actually a local simulation of MapReduce in Hadoop'sLocalJobRunner class. MapReduce mode (also known as Hadoop mode): Pig is executed on the Hadoop cluster. In this case, the Pig Script gets converted into a series of MapReduce jobs that are then run on the Hadoop cluster.