03 Hive Interview Questions -KirkYagami👨‍💻🕵️

1. What is *Apache Hive* and what types of problems does it *solve*?

Apache Hive is a versatile data warehousing tool that provides SQL interface to query and manage structured data in distributed storage like Hadoop.

With many of its core functionalities similar to relational databases, Hive brings **hierarchical data processing and scalability** to the table. It's ideal for analytical queries over large datasets, forming a vital part of the Hadoop ecosystem.

Key Components of Hive

Hive Query Language (HiveQL)

HiveQL, a SQL-like scripting language, allows users to perform SQL queries on distributed datasets, abstracting the underlying MapReduce jobs.

Metastore

Hive stores metadata such as schema details, table info, and partition statistics in a centralized metadata repository called the Hive Metastore. This separation of metadata from data enables better data management and accessibility.

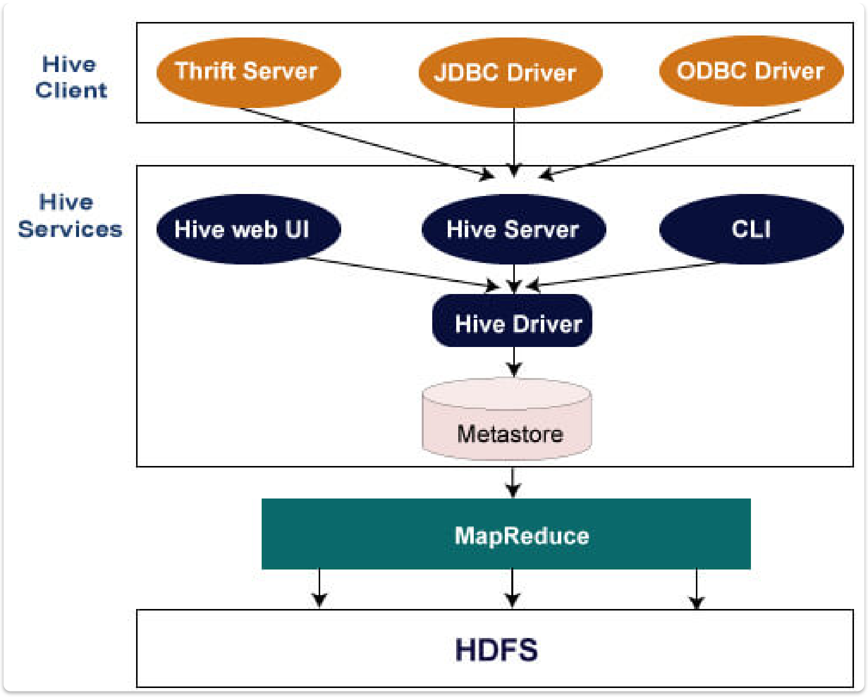
Hive Server

HiveServer is a service that works as an interface to Hive, allowing external clients to interact with Hive through JDBC, ODBC, or Thrift.

Optimizer

Hive Analyzer optimizes queries using techniques such as cost-based optimization, join reordering, predicate pushdown, and more to enhance query performance.

Hive Architecture



When you execute a Hive query, several components work together to process the tasks:

- User Interface: Initially, the user submits a SQL-like query through the Hive command line or a GUI tool such as Hue.
- Driver: The Driver consists of multiple components: Compiler, Query Planner, and Optimizer. The query goes through these elements, generating an execution plan.
- Metastore: Hive's Metastore contains schemas, partitions, information about serialized objects, and the location of Data in the Hadoop Distributed File System (HDFS).
- Hadoop: The Hadoop component is responsible for the actual storage and processing of data.
- Execution Engine: The Execution Engine carries out the TaskDeployment on Hadoop

Hive Use Cases

- Data Warehousing: Suitable for warehouse needs, providing relational database features with distributed data handling.
- ETL (Extract, Transform, Load): Performs ETL functions using familiar SQL constructs, enhancing usability for data engineers.
- Ad-Hoc Queries: Allows on-the-go data analysis, common in multi-department business environments.
- Metric Calculations: Ideal for tasks like KPI monitoring and report generation.

HiveQL and Hadoop

HiveQL abstracts the underlying MapReduce jobs and introduces a more user-friendly SQL-like methodology. When you run a Hive query, it is translated into a series of map/reduce tasks that get applied to the distributed data.

2. Can you specify the name of the table creator in Hive?

Yes, it is possible to specify the name of the table creator in Hive by using the optional "COMMENT" clause when creating the table.

An example is as follows:

```
CREATE TABLE my_table (
  col1 string,
  col2 int
)
COMMENT 'Created by Victor'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

Here are the questions starting from number 3, with detailed answers and code examples:

3) What are the different types of tables available in Hive?

In Hive, there are two primary types of tables:

1. **Managed Table**:
   - **Description**: In a managed table, both the data and the schema are managed by Hive. When you drop a managed table, both the table schema and the data are deleted. Hive takes full control over the data storage.
   - **Example**:

     ```
     CREATE TABLE managed_table (
       id INT,
       name STRING
     );
     ```

   - **Data Location**: Typically stored under `hdfs://namenode_server/user/hive/warehouse`.
2. **External Table**:
   - **Description**: In an external table, only the schema is managed by Hive. The actual data is managed outside of Hive, and it can be in HDFS or another

storage system. Dropping an external table removes only the table schema but leaves the data intact.
- **Example**:

```
CREATE EXTERNAL TABLE external_table (
  id INT,
  name STRING
)
LOCATION 'hdfs://namenode_server/user/hive/external_data';
```

- **Data Location**: Can be in any location specified by the `LOCATION` clause.

4) Can a table be renamed in Hive?

Yes, a table can be renamed in Hive using the `ALTER TABLE` command with the `RENAME TO` clause.

- **Example**:

```
ALTER TABLE old_table_name RENAME TO new_table_name;
```

5) Is Hive suitable to be used for OLTP systems? Why?

No, Hive is not suitable for OLTP (Online Transaction Processing) systems. Hive is designed for OLAP (Online Analytical Processing) workloads, which are typically read-heavy and involve complex queries and aggregations over large datasets.

- **Reasons**:
  - **Lack of Support for Row-Level Operations**: Hive does not support efficient row-level updates or deletes.
  - **Batch Processing**: Hive is optimized for batch processing and large-scale data analysis rather than real-time transactions.
  - **High Latency**: Hive queries often involve high latency due to the data processing and MapReduce framework.

6) Can we change the data type of a column in a Hive table?

Yes, you can change the data type of a column in a Hive table using the `ALTER TABLE ... REPLACE COLUMNS` command. This command allows you to replace the column definitions, including data types.

- Example:

```
ALTER TABLE table_name REPLACE COLUMNS (
  column1 INT,
  column2 STRING,
  new_column INT
);
```

7) What is a metastore in Hive?

A metastore in Hive is a relational database that stores metadata about Hive tables, partitions, databases, and other schema information. It keeps track of the structure of Hive tables and the locations of the data.

- Components:
  - Database Tables: Stores information about tables, columns, partitions, etc.
  - Configurations: Maintains configurations and connections to the underlying data storage.
  - Usage: Helps in querying and managing schema and metadata information efficiently.

8) Why do we need Hive?

Hive is used to:

- Provide SQL-like Interface: It allows users to query and analyze data using SQL-like queries, making it easier for those familiar with SQL.
- Manage Large Datasets: Hive is optimized for handling large-scale data stored in Hadoop.
- Data Warehousing: It offers data warehousing capabilities, including partitioning and indexing, for efficient querying and analysis.
- Integration: It integrates with Hadoop, enabling users to process and analyze data using the Hadoop ecosystem.

9) What is the default location where Hive stores table data?

By default, Hive stores table data in the following location in HDFS:

- **Location**:

  ```
  hdfs://namenode_server/user/hive/warehouse
  ```

This is the default directory where Hive places table data files unless a different location is specified in the `LOCATION` clause of a table definition.

10) Is there a date data type in Hive?

Yes, Hive supports the `TIMESTAMP` data type for storing date and time information. It stores dates in the `java.sql.Timestamp` format, which includes both date and time.

- **Example**:

  ```
  CREATE TABLE example_table (
    id INT,
    event_time TIMESTAMP
  );
  ```

11) What are collection data types in Hive?

Hive supports three collection data types:

1. **ARRAY**:
   - **Description**: A collection of elements of the same type.
   - **Example**:

     ```
     CREATE TABLE array_table (
       id INT,
       names ARRAY<STRING>
     );
     ```

2. **MAP**:
   - **Description**: A collection of key-value pairs where keys and values can be of different types.
   - **Example**:

     ```
     CREATE TABLE map_table (
       id INT,
     ```

```
  properties MAP<STRING, STRING>
);
```

3. **STRUCT**:
   - **Description**: A collection of fields, where each field can be of a different type.
   - **Example**:

   ```
   CREATE TABLE struct_table (
     id INT,
     person STRUCT<name:STRING, age:INT>
   );
   ```

12) Can Hive queries be executed from script files? How?

Yes, Hive queries can be executed from script files using the `source` command in the Hive CLI.

- **Example**:

```
Hive> source /path/to/file/file_with_query.hql;
```

This command executes all the HiveQL statements contained in the specified script file.

13) What is the importance of the .hiverc file?

The `.hiverc` file is used to set up the Hive environment by specifying commands that should be executed whenever the Hive CLI starts. It is commonly used for setting up session parameters, such as enabling strict mode or setting default database contexts.

- **Example**: Setting strict mode

```
set hive.cli.print.header=true;
```

14) What do you mean by schema on read?

Schema on read means that the schema is applied to the data only when it is read, rather than when it is written. This allows for more flexibility in how data is stored and queried, as the schema validation occurs during query execution rather than during data ingestion.

- **Example**: When querying a JSON file with a flexible schema, Hive interprets the structure of the data during query time rather than requiring a fixed schema at data load.

15) How do you list all databases whose names start with 'p'?

You can list all databases whose names start with 'p' using the `SHOW DATABASES` command with a pattern.

- **Example**:

```
SHOW DATABASES LIKE 'p%';
```

16) What does the "USE" command in Hive do?

The `USE` command sets the current database context for subsequent Hive queries. All subsequent queries will be executed within the specified database.

- **Example**:

```
USE my_database;
```

17) Explain DBPROPERTY in Hive?

`DBPROPERTY` is a way to define and view properties associated with a Hive database. These properties can include information such as comments or custom configurations for the database.

- **Example**:

```
CREATE DATABASE my_database
COMMENT 'This is my database';

SHOW DATABASES LIKE 'my_database';
```

18) Write a query to insert a new column (new_col INT) into a Hive table (htab) at a position before an existing column (x_col).

You cannot directly specify the position of a new column using standard SQL commands in Hive. However, you can use the `CHANGE COLUMN` clause to modify the existing columns. To effectively achieve column reordering, you might need to recreate the table.

- **Example**:

  ```
  ALTER TABLE htab
  CHANGE COLUMN x_col x_col STRING,
  ADD COLUMNS (new_col INT);
  ```

19) While loading data into a Hive table using the LOAD DATA clause, how do you specify it is an HDFS file and not a local file?

To load data from HDFS, you simply omit the `LOCAL` keyword in the `LOAD DATA` statement.

- **Example**:

  ```
  LOAD DATA INPATH '/hdfs/path/to/file' INTO TABLE table_name;
  ```

20) If you omit the OVERWRITE clause while creating a Hive table, what happens to files that are new and files that already exist?

If you omit the `OVERWRITE` clause, new files are added to the target directory, and existing files are not removed or overwritten. Existing files remain in the directory if their names do not match the new files.

- **Example**: Without `OVERWRITE`, files in `/hdfs/path/to/directory` remain, and only new files are appended.

Certainly!

21) What is a Table Generating Function in Hive? (continued)

A table-generating function in Hive takes a column of data and expands it into multiple rows or columns. This is useful for dealing with complex data types such as arrays or maps.

- **Example**:

```sql
-- Suppose we have a table with an array column
CREATE TABLE array_table (
  id INT,
  names ARRAY<STRING>
);

-- Sample data
INSERT INTO array_table VALUES (1, ARRAY('Alice', 'Bob', 'Charlie'));

-- Using the explode function to expand the array into multiple rows
SELECT id, name
FROM array_table
LATERAL VIEW explode(names) exploded_names AS name;
```

22) What is the difference between LIKE and RLIKE operators in Hive?

The `LIKE` and `RLIKE` operators are used for pattern matching in Hive, but they differ in their functionality:

1. **LIKE**:
   - **Description**: Uses SQL-style pattern matching with simple wildcard characters (`%` for zero or more characters, `_` for a single character).
   - **Example**:

     ```sql
     SELECT * FROM table_name
     WHERE street_name LIKE '%Chi%';
     ```

   - **Usage**: Suitable for basic pattern matching.
2. **RLIKE**:
   - **Description**: Uses Java regular expressions for pattern matching, allowing for more complex and powerful pattern matching.
   - **Example**:

     ```sql
     SELECT * FROM table_name
     WHERE street_name RLIKE '.*(Chi|Oho).*';
     ```

   - **Usage**: Allows advanced pattern matching with regular expressions.

23) How will you convert the string '51.2' to a float value in the price column?

You can convert a string to a float using the `CAST` function in Hive. This is useful for type conversions in your queries.

- **Example**:

  ```
  SELECT CAST('51.2' AS FLOAT) AS price_float;
  ```

24) Can the name of a view be the same as the name of a Hive table?

No, the name of a view must be unique within a database and cannot be the same as any existing table or other view. Hive enforces unique names for views and tables to avoid confusion and potential conflicts.

25) Can we LOAD data into a view?

No, you cannot use the `LOAD DATA` statement to load data into a view. Views are virtual tables that represent the result of a query; they do not store data themselves. Data must be loaded into underlying tables, not views.

26) What types of costs are associated with creating an index on Hive tables?

Creating an index on Hive tables involves the following costs:

1. **Storage Cost**:
   - **Description**: Indexes occupy additional storage space to store index data.
   - **Impact**: This increases the storage requirements for the table.
2. **Processing Cost**:
   - **Description**: There is overhead associated with maintaining and updating the index as data is modified.
   - **Impact**: Inserting, updating, or deleting data may become slower due to index maintenance.

27) Give the command to see the indexes on a table.

To view the indexes on a table in Hive, use the `SHOW INDEX` command.

- **Example**:

  ```
  SHOW INDEX ON table_name;
  ```

This command lists all the indexes created on the specified table, including information about the columns indexed and index type.