1. Explain Hive Architecture in Brief.

Architecture of Hive

The following component diagram depicts the architecture of Hive:



This component diagram contains different units. The following table describes each unit:

|  |  |
| --- | --- |
| Unit Name | Operation |
| 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. |

Working of Hive

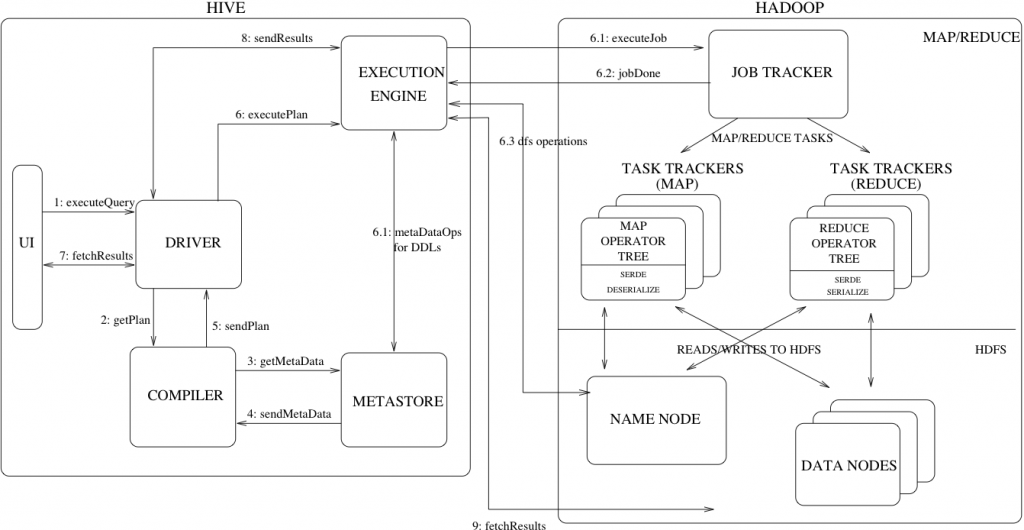
The following diagram depicts the workflow between Hive and Hadoop.



The following table defines how Hive interacts with Hadoop framework:

|  |  |
| --- | --- |
| Step No. | Operation |
| 1 | Execute Query  The Hive interface such as Command Line or Web UI sends query to Driver (any database driver such as JDBC, ODBC, etc.) to execute. |
| 2 | Get Plan  The driver takes the help of query compiler that parses the query to check the syntax and query plan or the requirement of query. |
| 3 | Get Metadata  The compiler sends metadata request to Metastore (any database). |
| 4 | Send Metadata  Metastore sends metadata as a response to the compiler. |
| 5 | Send Plan  The compiler checks the requirement and resends the plan to the driver. Up to here, the parsing and compiling of a query is complete. |
| 6 | Execute Plan  The driver sends the execute plan to the execution engine. |
| 7 | Execute Job  Internally, the process of execution job is a MapReduce job. The execution engine sends the job to JobTracker, which is in Name node and it assigns this job to TaskTracker, which is in Data node. Here, the query executes MapReduce job. |
| 7.1 | Metadata Ops  Meanwhile in execution, the execution engine can execute metadata operations with Metastore. |
| 8 | Fetch Result  The execution engine receives the results from Data nodes. |
| 9 | Send Results  The execution engine sends those resultant values to the driver.  The driver sends the results to Hive Interfaces. |

The below diagram represents clear internal Hadoop Hive Architecture

[](http://www.hadooptpoint.com/wp-content/uploads/2015/01/Hive_architecture.png)

Hive Architecture

The above diagram shows how a typical query flows through the system

**Step 1 :-**The UI calls the execute interface to the Driver

**Step 2 :-**The Driver creates a session handle for the query and sends the query to the compiler to generate an execution plan

**Step 3&4 :-**The compiler needs the metadata so send a request for getMetaData and receives the sendMetaData request from MetaStore.

**Step 5 :-**This metadata is used to typecheck the expressions in the query tree as well as to prune partitions based on query predicates. The plan generated by the compiler  is a DAG of stages with each stage being either a map/reduce job, a metadata operation or an operation on HDFS. For map/reduce stages, the plan contains map operator trees (operator trees that are executed on the mappers) and a reduce operator tree (for operations that need reducers).

**Step 6 :-** The execution engine submits these stages to appropriate components (steps 6, 6.1, 6.2 and 6.3). In each task (mapper/reducer) the deserializer associated with the table or intermediate outputs is used to read the rows from HDFS files and these are passed through the associated operator tree.Once the output generate  it is written to a temporary HDFS file though the serializer. The temporary files are used to provide the to subsequent map/reduce stages of the plan.For DML operations the final temporary file is moved to the table’s location

**Step 7&8&9 :-**For queries, the contents of the temporary file are read by the execution engine directly from HDFS as part of the fetch call from the Driver

1. Explain Hive Components in Brief.

**UI :-**UI means User Interface, The user interface for users to submit queries and other operations to the system.

**Driver :-**The Driver is used for receives the quires from UI .This component implements the notion of session handles and provides execute and fetch APIs modeled on JDBC/ODBC interfaces.

**Compiler :-**The component that parses the query, does semantic analysis on the different query blocks and query expressions and eventually generates an execution plan with the help of the table and partition metadata looked up from the metastore.

**MetaStore :-**The component that stores all the structure information of the various tables and partitions in the warehouse including column and column type information, the serializers and deserializers necessary to read and write data and the corresponding HDFS files where the data is stored.

**Execution Engine :-**The component which executes the execution plan created by the compiler. The plan is a DAG of stages. The execution engine manages the dependencies between these different stages of the plan and executes these stages on the appropriate system components.