**APACHE HIVE**

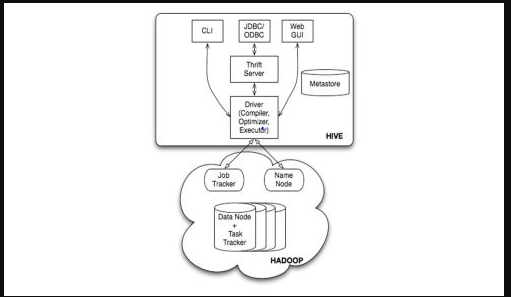
**HIVE Architecture and Its Component:**

**1. HIVE Introduction:**

Hive is a data warehousing infrastructure based on Apache Hadoop. Hadoop provides massive scale out and fault tolerance capabilities for data storage and processing on commodity hardware.

Hive is designed to enable easy data summarization, ad-hoc querying and analysis of large volumes of data. It provides SQL which enables users to do ad-hoc querying, summarization and data analysis easily. At the same time, Hive's SQL gives users multiple places to integrate their own functionality to do custom analysis, such as User Defined Functions (UDFs).

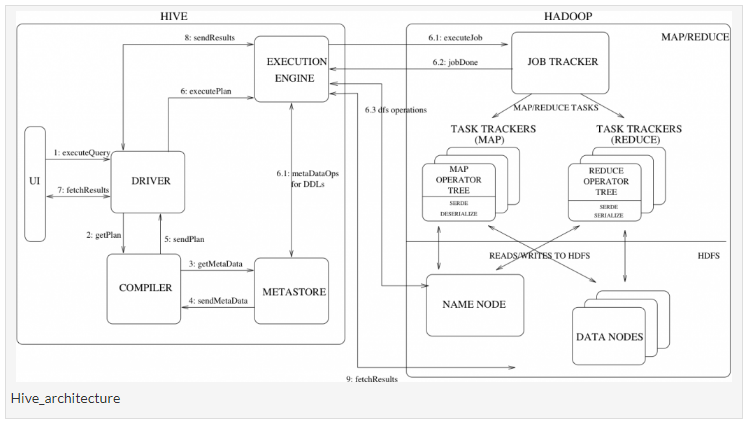
**2. HIVE Architecture:**

****

The above diagram shows the basic Hadoop Hive architecture. Primarily The diagram represents CLI (Command Line Interface),JDBC/ODBC and Web GUI (Web Graphical User Interface ).This represents when user comes with CLI(Hive Terminal) it directly connected to Hive Drivers,When User comes with JDBC/ODBC(JDBC Program) at that time by using API(Thrift Server) it connected to Hive driver and when the user comes with Web GUI(Ambari server) it directly connected to Hive Driver.

The hive driver receives the tasks(Queries) from user and send to Hadoop architecture.The Hadoop architecture uses name node,data node,job tracker and task tracker for receiving and dividing the work what Hive sends to Hadoop ([Mapreduce](http://www.hadooptpoint.com/hadoop-mapreduce/)) .

The below diagram represents clear internal Hadoop 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

**3. HIVE Components:**

**1. Hive Clients:**

Apache Hive supports different types of client applications for performing queries on the Hive. These clients can be categorized into three types:

* *Thrift Clients:* As Hive server is based on Apache Thrift, it can serve the request from all those programming language that supports Thrift.
* *JDBC Clients:* Hive allows Java applications to connect to it using the JDBC driver which is defined in the class org.apache.hadoop.hive.jdbc.HiveDriver.
* *ODBC Clients:* The Hive ODBC Driver allows applications that support the ODBC protocol to connect to Hive. (Like the JDBC driver, the ODBC driver uses Thrift to communicate with the Hive server.)

**2. Hive Services:**

Hive provides many services as shown in the image above. Let us have a look at each of them:

* **Hive CLI (Command Line Interface):**This is the default shell provided by the Hive where you can execute your Hive queries and commands directly.
* **Apache Hive Web Interfaces:**Apart from the command line interface, Hive also provides a web based GUI for executing Hive queries and commands.
* **Hive Server:**Hive server is built on Apache Thrift and therefore, is also referred as Thrift Server that allows different clients to submit requests to Hive and retrieve the final result.
* **Apache Hive Driver:**It is responsible for receiving the queries submitted through the CLI, the web UI, Thrift, ODBC or JDBC interfaces by a client. Then, the driver passes the query to the compiler where parsing, type checking and semantic analysis takes place with the help of schema present in the metastore. In the next step, an optimized logical plan is generated in the form of a DAG (Directed Acyclic Graph) of map-reduce tasks and HDFS tasks. Finally, the execution engine executes these tasks in the order of their dependencies, using Hadoop.
* **Metastore:**You can think metastore as a central repository for storing all the Hive metadata information. Hive metadata includes various types of information like structure of tables and the partitions along with the column, column type, serializer and deserializer which is required for Read/Write operation on the data present in HDFS. The metastore comprises of two fundamental units:
  + A service that provides metastore access to other Hive services.
  + Disk storage for the metadata which is separate from HDFS storage.