**5 MAJOR COMPONENTS OF HIVE :**

There are 5 major components of hive.They are

1 .UI

2 .Driver

3 .Compiler

4. MetaStore

5. Execution Engine

**U I(User Interface):**

It is the interface used by user for interacting with hive and execute querries in hive

**Driver**

As the name suggests Driver acts like it is responsible for overall flow and execution of a querry right from querry submission.It performs the following function

1.It receives the queries

2.It implements the notion of session handles and provides execute and

fetch APIs modelled on JDBC/ODBC interface

3.The Driver creates a session handle for the query and sends the query to the compiler to generate an execution plan

**Compiler:**

It perform the function of parsing the querry and check if the querry is semantically correct

after checking for syntax it sends a request to metastore for checking the presence of data and if data is present it creates a execution plan

**MetaStore**

It is the component which acts like a storage

It contains all the metadata of all tables in the form of RDBMS

When you create a Hive table, the table definition (column names, data types, comments, etc.) are stored in the Hive Metastore.

By default it uses a derby database which is helpful in single mode but it does not support multi user function which is highly required when we work in cluster so we can configure to any RDBMS like mysql DB

**Execution Engine**

It is 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.

It also decides whether a map reduce job is required or map only job

or just a select(\*)(a cat like command e=which does not require MR)

**Hive Architecture:**

The following steps explains the execution of a querry in hive

Step 1 :- The UI as soon as a querry is written a call is made to driver by UI

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 after getting the querry checks whether the querry isn semantically correct and in order to create a execution plan it needs the metadata of the table mentioned in querry to check whether a table is present or not so it 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

