**Explain Hive Architecture in Brief**



There are 5 major blocks in the hive

**UI –** User interface is the one which interacts with the user and helps the end user to write the query.

There are two types of hive UI

Command line and web interface but the command line is the most used ui.

Hive is written in HQL(Hive Query Language) which is very similar to that of the SQL query

**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-** After the Driver the control is transferred to the compiler and it performs the sematic analysis on the query blocks and query expressions and then generates the execution plan with the help of the table and partition metadata looked up from the metastore.

**Metastore-** It is 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 –** Execution engine takes plan from the compiler and then executes the stages one by one in order and also it maintains the interdependences of the different stages with each other. It is the one which deals with HDFS for the data to be processed and also stores data in the HDFS

**HIVE ARCHITECTURE**

**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