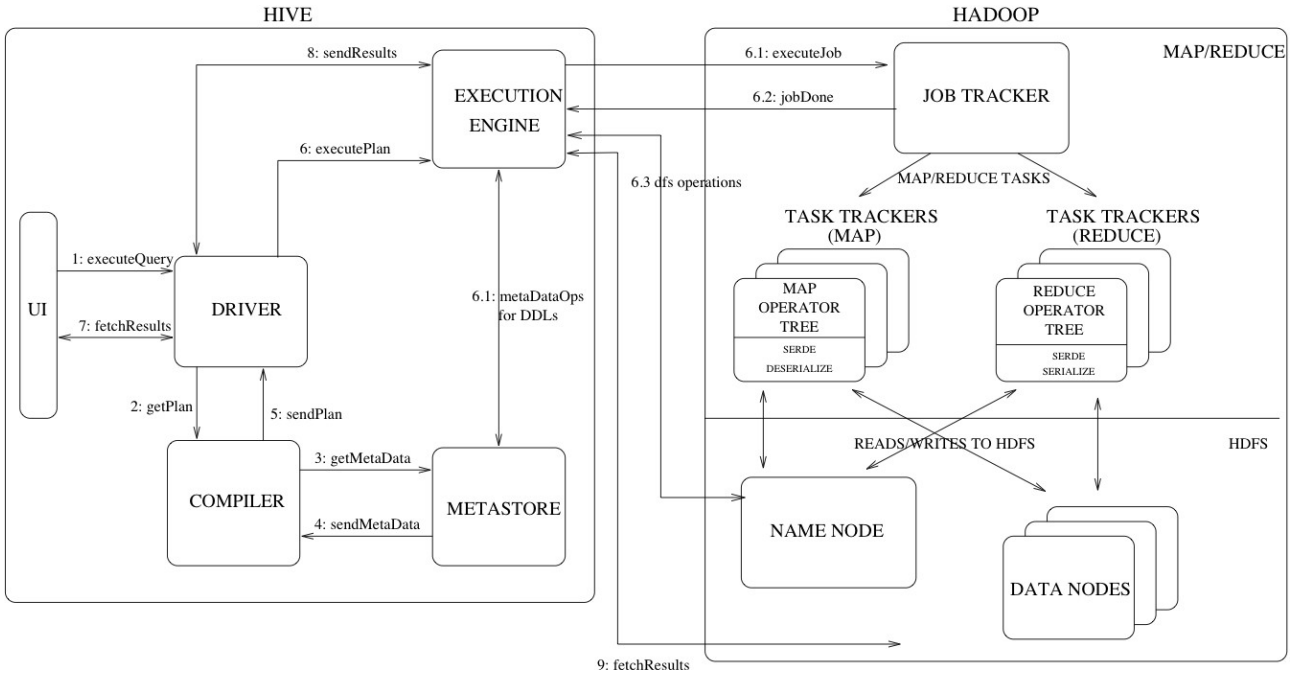
Bigdata Assignment 2.9

- **Explain Hive Architecture in Brief.**

Ans -



- UI – The user interface for users to submit queries and other operations to the system. As of 2011 the system had a command line interface and a web based GUI was being developed.
- Driver – The component which receives the queries. 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.

Explain Hive Components in Brief.

Ans -

Apache Hive is an open source data warehouse system built on top of Hadoop. It is used for querying and analyzing large datasets stored in Hadoop files. The Apache Hive components are -

- **Metastore –** It stores metadata for each of the tables like their schema and location. Hive also includes the partition metadata. This helps the driver to track the progress of various data sets distributed over the cluster. It stores the data in a traditional RDBMS format. Backup server regularly replicates the data which it can retrieve in case of data loss.
- **Driver –** It acts like a controller which receives the HiveQL statements. The driver starts the execution of statement by creating sessions. It monitors the life cycle and progress of the execution and stores the necessary metadata generated during the execution of a HiveQL statement. It also acts as a collection point of data or query result obtained after the Reduce operation.
- **Compiler –** It performs the compilation of the HiveQL query. This converts the query to an execution plan. The plan contains the tasks which contains steps needed to be performed by the MapReduce to get the output as translated by the query. The compiler in Hive converts the query to an **Abstract Syntax Tree (AST)**. First, it check for compatibility and compile time errors, then converts the AST to a **Directed Acyclic Graph (DAG).**

- **Optimizer –** It performs various transformations on the execution plan to provide optimized DAG. It aggregates the transformations together, such as converting a pipeline of joins to a single join, for better performance. The optimizer can also split the tasks, such as applying a transformation on data before a reduce operation, to provide better performance.
- **Executor –** Once compilation and optimization complete, the executor executes the tasks. Executor takes care of pipelining the tasks.
- **CLI, UI, and Thrift Server –** CLI (command-line interface) provide a user interface for an external user to interact with Hive. Thrift server in Hive allows external clients to interact with Hive over a network, similar to the JDBC or ODBC protocols.