**Explain Hive Architecture in Brief.**

Hive is an ETL and Data warehousing tool developed on top of Hadoop Distributed File System (HDFS). Hive makes job easy for performing operations like

Data encapsulation

Ad-hoc queries

Analysis of huge datasets

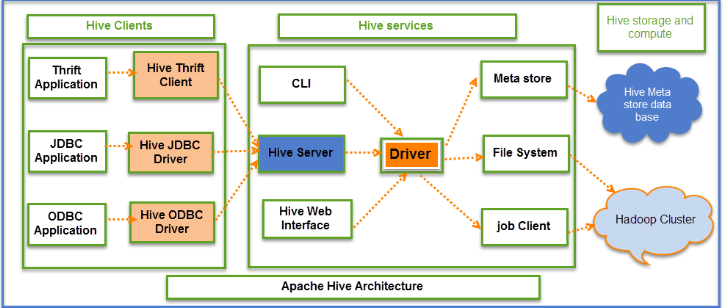
The below diagram describes the Architecture of Hive

Hive Clients – Apache Hive supports all application written in languages like C++, Java, Python etc. using JDBC, Thrift and ODBC drivers. Thus, one can easily write Hive client application written in a language of their choice.

Hive Services – Hive provides various services like web Interface, CLI etc. to perform queries.

Processing framework and Resource Management – Hive internally uses Hadoop MapReduce framework to execute the queries.

Distributed Storage – As seen below that Hive is built on the top of Hadoop, so it uses the underlying HDFS for the distributed storage.



**Explain Hive Components in Brief.**

**Command Line Interface**: By default, it is the way to access Hive queries and commands

**Hive Server**: It runs Hive as a server exposing a thrift service, which enables access from a range of clients written in different languages.

**Hive Web Interface**: Hive Hive provides web-based GUI for executing Hive queries and commands

**Shell**: A shell is the command line interface which allows interactive queries similar to MySQL shell connected to the database. It also supports web and JDBC clients. Driver, compiler and execution engine take the hiveql scripts and run in Hadoop environment.

**Driver**: The driver is 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 compiler parses the query does semantic analysis on the different query blocks and query expressions. It eventually generates an execution plan with the help of the table and partition metadata looked up from the metastoremetastore.

**Execution Engine**: This component executes the execution plan created by the compiler. The plan has a DAG of stages. The engine manages the dependencies between these different stages of the plan. It executes these stages on the associated system components.

**Metastore**: This component stores all the structure information of the various table and partitions in the warehouse. It includes column and column type information, the serializers and deserializers (also called SerDe) necessary to read and write data and the corresponding hdfs files where the data is stored.