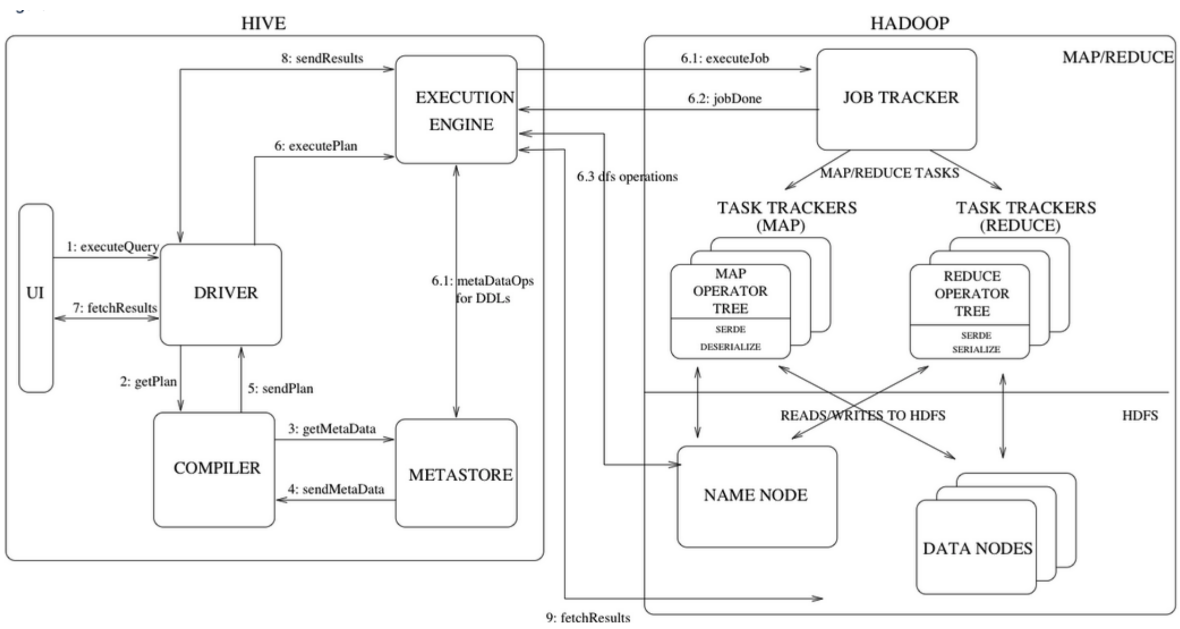
Hive Architecture

Hive Architecture



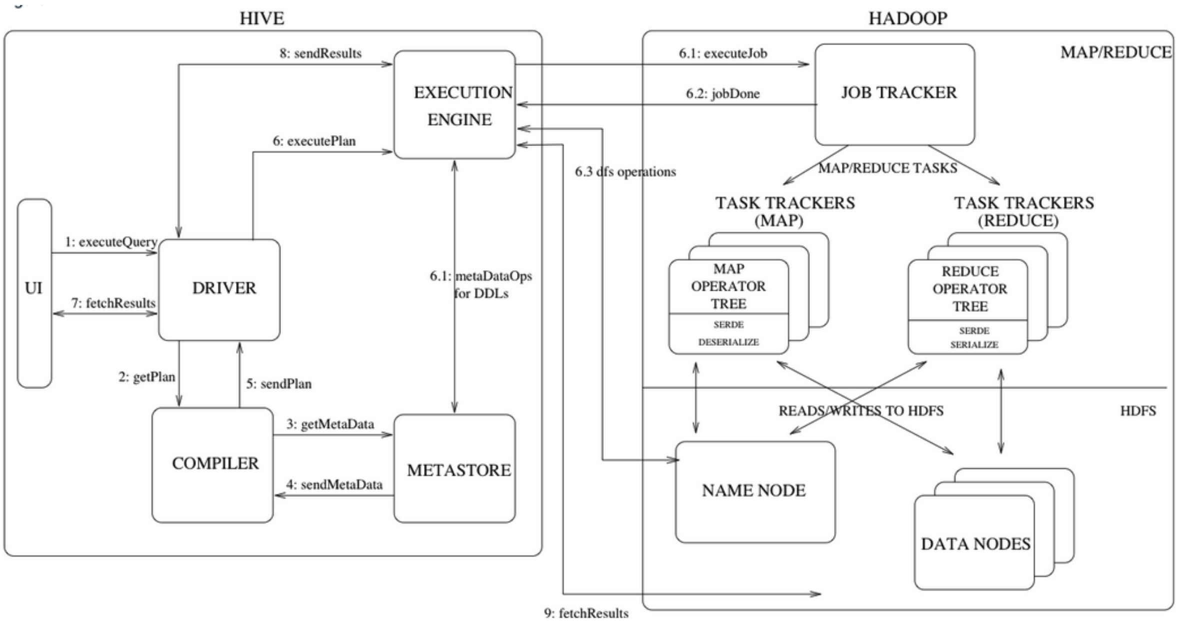
The Hive metadata store (or called metastore) can use either embedded, local, or remote databases.
Derby is the default database for the embedded metastore.

Remote databases are preferred. If meta data is stored in default database, reinstalling hive means losing those information. Remote databases will not have this problem. Also if the Derby database is the metastore, only one connection with the Hive engine is possible. But with external database as metastore, any number of connections with Hive engine is possible

- UI – The user interface for users to submit queries and other operations to the system.
- Driver – The component which receives the queries.

  - Handles sessions
- Compiler – The component that parses the query,
  - does semantic analysis,
  - generates an execution plan with the help of metadata from metastore.
- Metastore – The component that stores metadata
  - all the structure information of the various tables and partitions
  - columns and column type information,
  - the serializers and deserializers necessary to read and write data
  - corresponding HDFS files where the data is stored.
- Execution Engine – The component which executes the execution plan created by the compiler.
  - The execution plan is a DAG of stages.

Typical Query Flow in Hive



1. Query is fired through UI. UI calls excuteQuery to Driver
2. Driver creates and handle a session for the query. get execution plan from Compiler
3. Compiler creates execution plan based on meta data. getMetaData from MetaStore
4. MetaData is send from MetaStore to Compiler.
    a. Meta data is use to typecheck the expressions in the query tree
    b. also to prune partitions based on query predicates.
5. Execution plan (DAG) generated by compiler is send to the Driver
6. The execution engine submits the job to Hadoop
    a. executeJob and update meta data to meta store
    b. jobDone by Hadoop
    c. dfs operations are performed
7. Once the output is generated, it is written to a temporary HDFS file though the serializer. contents of the temporary file are read by the execution engine directly from HDFS as part of the fetch call
    a. sendResults
    b. fetchResults from Hadoop

Meta Store

To store data abstraction and to utilize for data discovery

- The data abstraction information is given during table creation and reused every time the table is referenced.
- data discovery, enables users to discover and explore relevant and specific data in the warehouse.

Compiler

- Parser – Transform a query string to a parse tree representation
- Semantic Analyser – Transform the parse tree to an internal query representation, which is still block based and not an operator tree. As part of this step, the column names are verified and expansions like * are performed. Type-checking and any implicit type conversions are also performed at this stage. If the table under consideration is a partitioned table, which is the common scenario, all the expressions for that table are collected so that they can be later used to prune the partitions which are not needed. If the query has specified sampling, that is also collected to be used later on.
- Logical Plan Generator – Convert the internal query representation to a logical plan, which consists of a tree of operators.
- Query Plan Generator – Convert the logical plan to a series of map-reduce tasks.

Optimizer

Design - Apache Hive - Apache Software Foundation

This page contains details about the Hive design and architecture. A brief technical report about Hive is available at hive.pdf.

https://cwiki.apache.org/confluence/display/hive/design#Design-HiveArchitecture

Managed (Internal) Table Vs External Table

Managed Table

Hive manages data. ie; the data, its properties and data layout will and can only be changed via Hive command. Undefined behaviour can be expected if we change data without hive.

If a managed table or partition is dropped, the data and metadata associated with that table or partition are deleted.

If the PURGE option is not specified, the data is moved to a trash folder for a defined duration.

- ARCHIVE/UNARCHIVE/TRUNCATE/MERGE/CONCATENATE only work for managed tables
- DROP deletes data for managed tables while it only deletes metadata for external ones
- ACID/Transactional only works for managed tables
- Query Results Caching only works for managed tables
- Only the RELY constraint is allowed on external tables
- Some Materialized View features only work on managed tables

Use managed tables when Hive should manage the lifecycle of the table, or when generating temporary tables.

External tables

An external table describes the metadata / schema on external files.

External tables can access data stored in sources such as Azure Storage Volumes (ASV) or remote HDFS locations.

If the structure or partitioning of an external table is changed, an MSCK REPAIR TABLE table_name statement can be used to refresh metadata information.

Use external tables when files are already present or in remote locations, and the files should remain even if the table is dropped.

Managed vs. External Tables - Apache Hive - Apache Software Foundation

This document lists some of the differences between the two but the fundamental difference is that Hive assumes that it owns the data for managed tables. That means that the data, its properties and data layout will and can only be changed via Hive command. The data still lives in a normal file system and

https://cwiki.apache.org/confluence/display/Hive/Managed+vs.+External+Tables