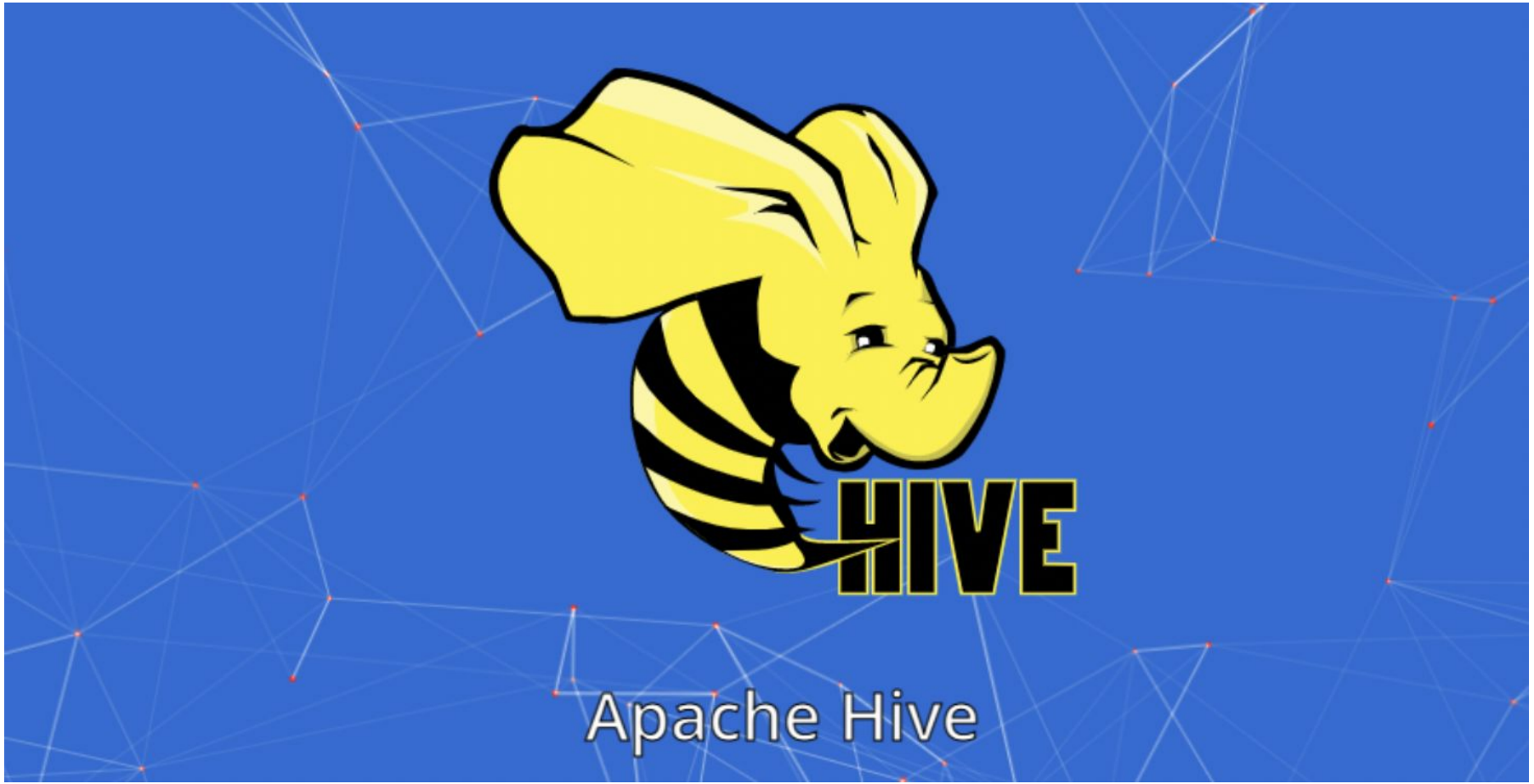

HIVE
Apache Hive

What is Hive?

GROW DATA SKILLS

Apache Hive is an open source **data warehouse** system built on top of Hadoop Haused for querying and analyzing large datasets stored in Hadoop files.

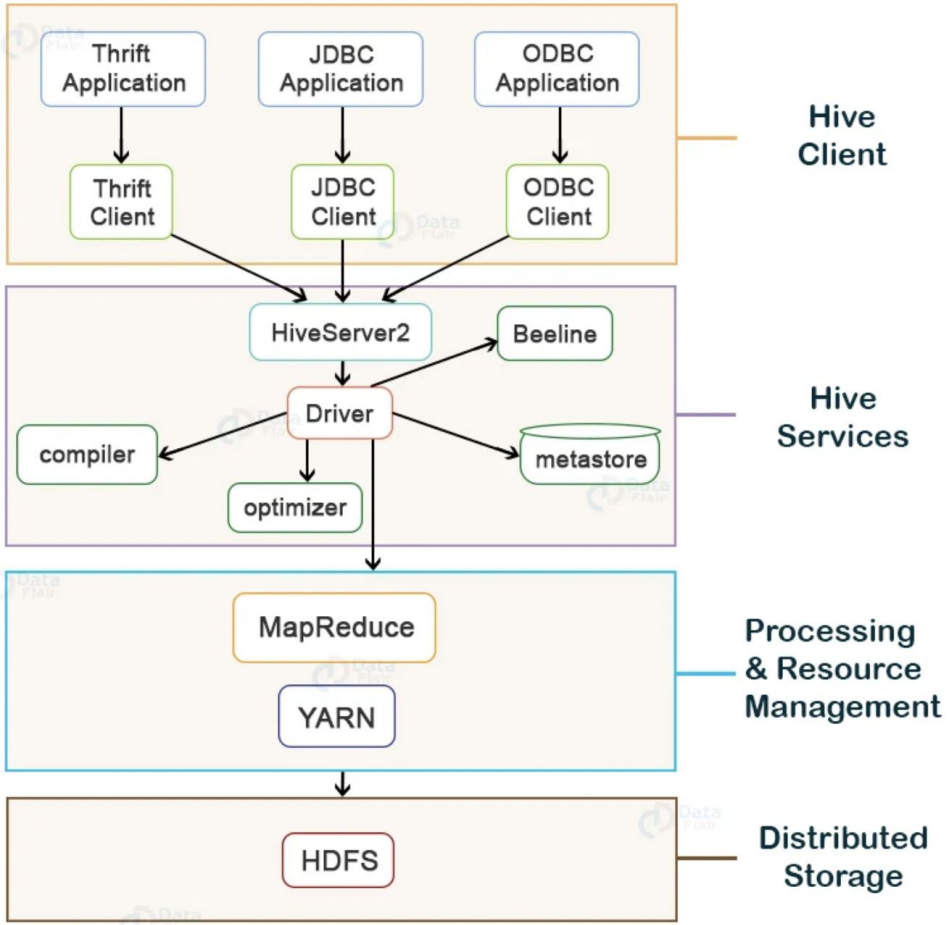
Initially, you have to write complex Map-Reduce jobs, but now with the help of the Hive, you just need to submit merely SQL queries. Hive is mainly targeted towards users who are **comfortable with SQL**.

Hive use language called **HiveQL (HQL)**, which is similar to SQL. HiveQL automatically translates **SQL-like queries into MapReduce jobs.**

Hive abstracts the complexity of Hadoop. The main thing to notice is that there is **no need to learn java** for Hive.

The Hive generally runs on your workstation and converts your SQL query into a series of jobs for execution on a Hadoop cluster. Apache Hive organizes data into tables. This provides a means for attaching the structure to data stored in HDFS.

Hive Architecture



Hive Architecture & Component

GROW DATA SKILLS

The major components of Apache Hive are:

- **Hive Client**
  - **Thrift Client**: The Hive server is based on Apache Thrift so that it can serve the request from a thrift client.
  - **JDBC Client:** Hive allows for the Java applications to connect to it using the JDBC driver. JDBC driver uses Thrift to communicate with the Hive Server.
  - **ODBC Client**: Hive ODBC driver allows applications based on the ODBC protocol to connect to Hive. Similar to the JDBC driver, the ODBC driver uses Thrift to communicate with the Hive Server.
- **Hive Services**
  - **Beeline**: The Beeline is a command shell supported by HiveServer2, where the user can submit its queries and command to the system. It is a JDBC client that is based on SQLLINE CLI (pure Java-console-based utility for connecting with relational databases and executing SQL queries).

Hive Architecture & Component

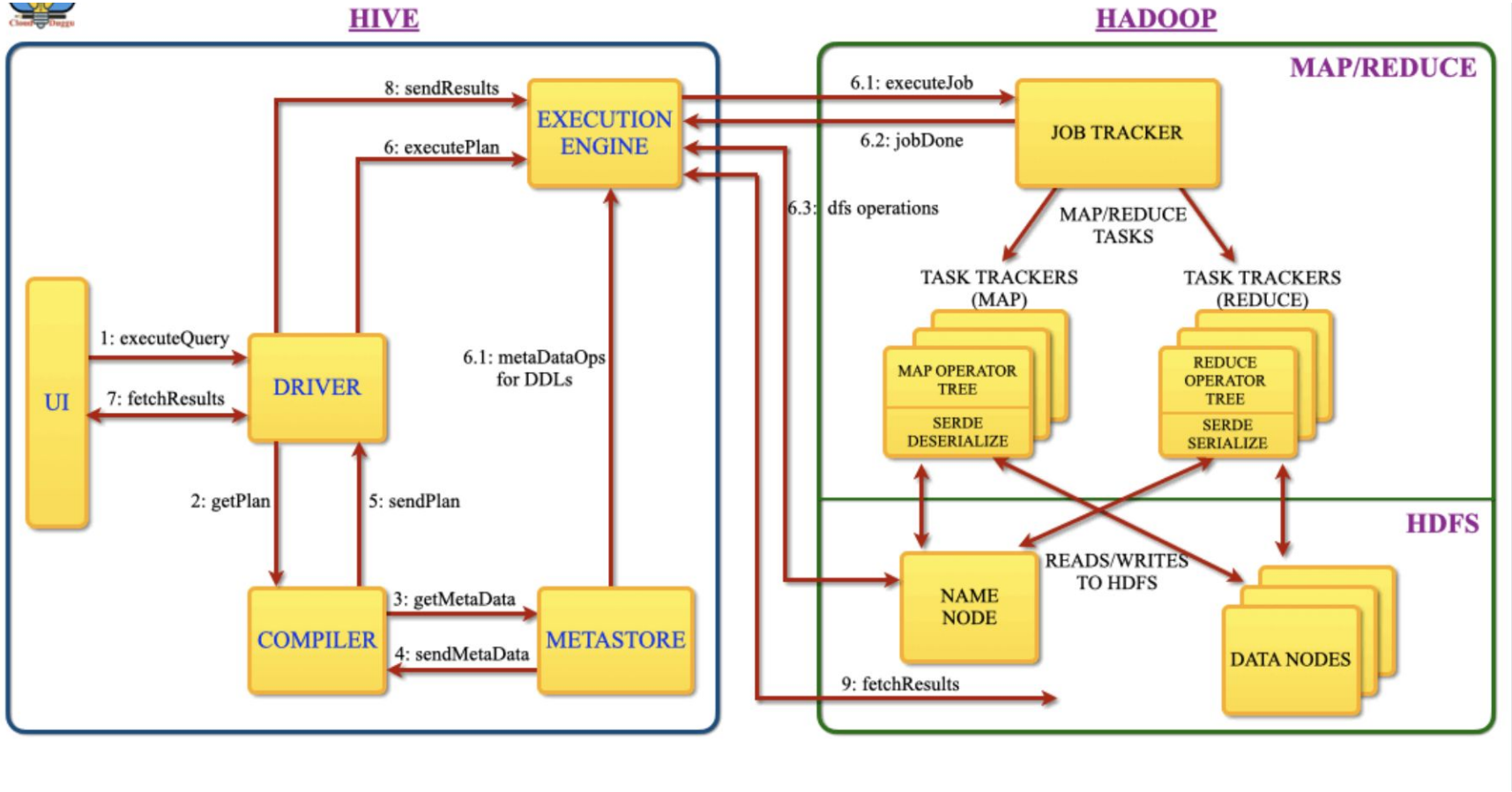
- **Hive Server 2 :** HiveServer2 is the successor of HiveServer1. HiveServer2 enables clients to execute queries against the Hive. It allows multiple clients to submit requests to Hive and retrieve the final results. It is basically designed to provide the best support for open API clients like JDBC and ODBC. **Note:** Hive server1, also called a Thrift server, is built on Apache Thrift protocol to handle the cross-platform communication with Hive. It allows different client applications to submit requests to Hive and retrieve the final results. It does not handle concurrent requests from more than one client due to which it was replaced by HiveServer2.
- **Hive Driver:** The Hive driver receives the HiveQL statements submitted by the user through the command shell. It creates the session handles for the query and sends the query to the compiler.
- **Hive Compiler:** Hive compiler parses the query. It performs semantic analysis and type-checking on the different query blocks and query expressions by using the metadata stored in metastore and generates an execution plan. The execution plan created by the compiler is the DAG(Directed Acyclic Graph), where each stage is a map/reduce job, operation on HDFS, a metadata operation.
- **Optimizer:** Optimizer performs the transformation operations on the execution plan and splits the task to improve efficiency and scalability.
- **Execution Engine:** Execution engine, after the compilation and optimization steps, executes the execution plan created by the compiler in order of their dependencies using Hadoop.

Hive Architecture & Component

- **Metastore:** Metastore is a central repository that stores the metadata information about the structure of tables and partitions, including column and column type information. It also stores information of serializer and deserializer, required for the read/write operation, and HDFS files where data is stored. This metastore is generally a relational database. Metastore provides a Thrift interface for querying and manipulating Hive metadata.

- We can configure metastore in any of the two modes:
  - **Remote**: In remote mode, metastore is a Thrift service and is useful for non-Java applications.
  - **Embedded**: In embedded mode, the client can directly interact with the metastore using JDBC.

Hive Query Flow



Hive Query Flow

GROW DATA SKILLS

- Step. 1- The user will submit a request to the driver.
- Step. 2- The Driver creates a session handler for the query and sends the query to the compiler to generate an execution plan.
- Step. 3- The compiler checks the required metadata information from the metastore.
- Step. 4- This metadata is used to type check the expressions in the query tree as well as to prune partitions based on query predicates.
- Step. 5- 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.
- Step. 6- The execution engine process the steps. Each stage of the task has a mapper and reducer that is used to read the rows from Hadoop HDFS and written on a temporary HDFS file. Now that temp file is used to provide the required data to map/reduce stages in the plan.
- Step. 7- The User will make a fetch call to the driver.
- Step. 8- The driver will interact with the execution engine to fetch the result.
- Step. 9- The execution engine will take the result and send it to the driver program and then the drive will send data to the user.