**Hive Architecture**

Hive sits on top of the Hadoop Distributed File System (HDFS) and MapReduce systems.

****

In the case of MapReduce, both the Hadoop 1 and Hadoop 2 components. With Hadoop 1, Hive queries are converted to MapReduce code and executed using the MapReduce v1 infrastructure, like the JobTracker and TaskTracker.

With Hadoop 2, YARN has decoupled resource management and scheduling from the MapReduce framework. Hive queries can still be converted to MapReduce code and executed, now with MapReduce v2 and the YARN infrastructure.

There is a new framework under development called Apache Tez, which is designed to improve Hive performance for batch-style queries and support smaller interactive (also known as *real-time*) queries. At the time of writing, the Apache Tez project is still in incubation, and doesn’t yet have a production-ready release.

If it helps you visualize how all the pieces fit together, think of the HDFS and MapReduce systems as being parts of the Apache Hadoop operating system, with Hive — as well as other components, such as HBase — as higher-level functions or applications.

Moving up the diagram, you find the Hive Driver, which compiles, optimizes, and executes the HiveQL. The Hive Driver may choose to execute HiveQL statements and commands locally or spawn a MapReduce job, depending on the task at hand. The Hive Driver stores table metadata in the metastore and its database.

By default, Hive includes the Apache Derby RDBMS configured with the metastore in what’s called embedded mode. Embedded mode means that the Hive Driver, the metastore, and Apache Derby are all running in one Java Virtual Machine (JVM).

This configuration is fine for learning purposes, but embedded mode can support only a single Hive session, so it normally isn’t used in multi-user production environments. Two other modes exist local and remote, which can be better support multiple Hive sessions in production environments. Also, you can configure any RDBMS that’s compliant with the Java Database Connectivity (JDBC) Application Programming Interface (API) suite.

The key to application support is the Hive Thrift Server, which enables a rich set of clients to access the Hive subsystem. The open source [SQuirreL SQL client](http://squirrel-sql.sourceforge.net/) is included as an example. The main point is that any JDBC-compliant application can access Hive via the bundled JDBC driver.

To continue with the Hive architecture drawing, note that Hive includes a Command Line Interface (CLI), where you can use a Linux terminal window to issue queries and administrative commands directly to the Hive Driver. If a graphical approach is more your speed, there’s also a handy web interface so that you can access your Hive-managed tables and data via your favorite browser.