Choosing Which Query Engine to Use

With different query engines available to you, and with different options available in those engines, you'll have to decide which to use for a particular task. Here are some guidelines to help you decide.

First, if you're considering whether you should be using a big data system or to a relational database system, remember that relational databases are optimized to store relatively small amounts of data, to provide immediate query results, and to allow for in-place modification of data. Big data engines such as Hive and Impala, on the other hand, are better optimized for large amounts of read-only data. They provide excellent scalability at low cost. So if you're working with smaller amounts of data (no more than a terabyte or two) and you need in-place modification of data, you probably don't need or even want to use a big data system.

If you do need a big data system, Impala is typically faster than Hive and is good for interactive and ad-hoc queries when you're exploring data.

However, Impala lacks some of the features that Hive provides. For example, a very long-running query that experiences failures (due to computers in the cluster failing, for example) will fail in Impala, but Hive has fault tolerance and likely will still complete the query. This makes Hive a good choice for batch processing and ETL jobs using SQL. Hive also offers extensible record formats and file formats; Impala is more limited in the accepted file formats.

Hive developers have found ways to improve Hive's query performance in recent versions. Hive works by creating jobs that run in a different engine (originally MapReduce, which can be rather slow) and the underlying engine can be changed. Rather than MapReduce, you may be able to use Apache Spark or Apache Tez, both of which are faster than MapReduce. Newer versions of Hive also support an architecture called LLAP (Live Long And Process) which caches metadata similarly to Impala, reducing query latency. You may want to test some typical queries against your own tables to see if one of these works better for you than Impala for interactive and ad-hoc queries.

There are a few other considerations when deciding how to complete tasks in big data systems. While both Hive and Impala can insert individual records into a table, this is not a recommended way to populate a table, because it tends to create small files that are inefficient to process. Recent versions of Hive do include some limited support for updating and deleting records, but full transactions (including COMMIT and ROLLBACK) are not yet implemented. Hive and Impala do not support stored procedures, as relational databases do. Relational database engines also typically have extensive support for indexing, while Hive has only limited support for indexing, and Impala does not support it. However, if you need the speed for searching massive datasets that indexing provides, other tools such as Cloudera Search, which uses Apache Solr, can be used instead.

You might also be curious about Apache Spark, which is another powerful large-scale data processing engine. It provides APIs for writing custom data processing code, but working directly with it requires programming skill. If you don't already have skills with Spark, Hive and Impala are typically a better choice for data analysis and data processing tasks.