1. **When Hive is best suited and when is it not?**

Hive is a data warehousing tool based on Hadoop. As we know Hadoop provides massive scale out on distributed infrastructure with high degree of fault tolerance for data storage and processing. Hadoop uses Map Reduce algorithm to process huge amount of data with minimal cost as it does not require high end machines to process such amount of data. Hive processor converts most of its queries into a Map Reduce job which runs on Hadoop cluster. Hive is designed for easy and effective data aggregation, ad-hoc querying and analysis of huge volumes of data.

Even though Hive gives SQL dialect it does not give SQL like latency as it ultimately runs Map Reduce programs underneath. As we all know, Map Reduce framework is built for batch processing jobs it has highlatency, even the fastest hive query would take several minutes to get executed on relatively smaller set of data in few megabytes. We cannot simply compare the performance of traditional SQL systems like Oracle, MySQL or SQL Server as these systems are meant to do something and Hive is meant to do else. Hive aims to provide acceptable (but not optimal) latency for interactive querying over small data sets for sample queries. Like we said earlier, hive is not an OLTP (Online transaction Processing) application and not meant to be connected with systems which needs interactive processing. It is meant to be used to process batch jobs on huge data which is immutable. A good example of such kind of data would be Web logs, Application Logs, calldata records (CDR) etc.

1. **When should one use Hive over MapReduce?**

**Hive** - An application that creates map-reduce and Tez jobs based on a SQL like language called HQL.

If there already exists pre-defined library of Java Mappers or Reducers for a job then it is a wise decision to **use** Hadoop **MapReduce** instead of Pig and **Hive**. If the hadoop developers require good amount of testability when combining lots of large data sets then they should **use MapReduce** instead of Pig and **Hive**.

1. **What is Hive metastore?**

The **Hive metastore** service stores the metadata for **Hive** tables and partitions in a relational database, and provides clients (including **Hive**) access to this information via the **metastore** service API.

1. **How can Hive improve performance with orc file format tables?** 
   1. ORC File Format Full Form is Optimized Row Columnar File Format.ORC File format provides very efficient way to store relational data then RC file,By using ORC File format we can reduce the size of original data up to 75%.Comparing to Text,Sequence,Rc file formats ORC is better .
   2. Using ORC files improves performance when Hive is reading, writing, and processing data.Comparing to Text,Sequence and Rc.RC and ORC shows better performance than Text and Sequence File formats.
2. **What is thrift server and client, jdbc and odbc driver importance in hive?**

The Hive ODBC Driver is a software library that implements the Open Database Connectivity (ODBC) API standard for the Hive database management system, enabling ODBC compliant applications to interact seamlessly (ideally) with Hive through a standard interface. This driver will NOT be built as a part of the typical Hive build process and will need to be compiled and built separately according to the instructions below.

**Apache Thrift** – a scalable cross-language software framework that enables the Hive ODBC driver (specifically the Hive client) to communicate with the Hive Server. See this link for the details on [Thrift Installation](http://wiki.apache.org/thrift/ThriftInstallation). The Hive ODBC driver was developed with Thrift trunk version r790732, but the latest revision should also be fine. Make sure you note the Thrift install path during the Thrift build process as this information will be needed during the Hive client build process. The Thrift install path will be referred to as THRIFT\_HOME.

**Hive Server** – a service through which clients may remotely issue Hive commands and requests. The Hive ODBC driver depends on Hive Server to perform the core set of database interactions. Hive Server is built as part of the Hive build process. More information regarding Hive Server usage can be found

Thrift is a cross language RPC framework which generate code and cobines a software stack finally execute the Thrift code in remote server. Thrift compiler acts as interpreter between server and client. Thrift server allows a remove client to submit request to Hive, using different programming languages like Python, Ruby and scala.

JDBC driver: A JDBC driver is a software component enabling a Java application to interact with a database.

ODBC driver: ODBC accomplishes DBMS independence by using an *ODBC driver* as a translation layer between the application and the DBMS.

1. **What is the importance of partition in hive?**

To analyze a particular set of data, not required to load entire data, desired data partition is a good approach. To achieve this goal, Hive allows to partition the data based on particular column. Static partition and Dynamic partition, both can optimize the Hive performance. For Instant, required a particular year information, partition based on year.

1. **What is the use of bucketing in hive?**

Tables or partitions are sub-divided into buckets, to provide extra structure to the data that may be used for more efficient querying. Bucketing works based on the value of hash function of some column of a table.

To process many chunks of files, to analyze vast amount of data, sometime burst the process and time. Bucketing is a sampling concept to analyze the data, by using hashing algorithm.

1. **What is the difference between static partitioning and dynamic partitioning in hive?**

To prune data during query, partition can minimize the query time. The partition is created when the data is inserted into table. Static partition can insert individual rows where as Dynamic partition can process entire table based on a particular column. At least one static partition is must to create any (static, dynamic) partition. If you are partitioning a large datasets, doing sort of a ETL flow Dynamic partition partition recommendable.