# Hive FAQ’s

**What is Hive?**

[Hive](http://hive.apache.org/) is a data warehouse system for Hadoop that facilitates easy data summarization, ad-hoc queries, and the analysis of large datasets stored in hadoop compatible file system..Hive was originally developed at Facebook. It’s now a [Hadoop](http://bigdataanalyticsnews.com/hadoop-interview-questions-setting-hadoop-cluster/) subproject with many contributors. Users need to concentrate only on the top level hive language rather than java map reduce programs. One of the main advantages of Hive is its SQLish nature. Thus it leverages the usability to a higher extend.A hive program will be automatically compiled into map-reduce jobs executed on Hadoop. In addition, HiveQL supports custom map-reduce scripts to be plugged into queries.

**Hive example**:

selecting the employee names whose salary more than 100 dollars from a hive table called tbl\_employee.

SELECT employee\_name FROM tbl\_employee WHERE salary > 100;

Users are excited to use Hive since it is very similar to SQL.

**What are the types of tables in Hive?**

There are two types of tables.

1. Managed tables.

2. External tables.

Only the drop table command differentiates managed and external tables. Otherwise, both type of tables are very similar.

**Does Hive support record level Insert, delete or update?**

[Hive](http://en.wikipedia.org/wiki/Apache_Hive) does not provide record-level update, insert, or delete. Henceforth, Hive does not provide transactions too. However, users can go with CASE statements and built in functions of Hive to satisfy the above DML operations. Thus, a complex update query in a RDBMS may need many lines of code in Hive.

**What kind of datawarehouse application is suitable for Hive?**

Hive is not a full database. The design constraints and limitations of Hadoop and HDFS impose limits on what Hive can do.

Hive is most suited for data warehouse applications, where

1) Relatively static data is analyzed,

2) Fast response times are not required, and

3) When the data is not changing rapidly.

Hive doesn’t provide crucial features required for OLTP, Online Transaction Processing. It’s closer to being an OLAP tool, Online Analytic Processing.So, Hive is best suited for data warehouse applications, where a large data set is maintained and mined for insights, reports, etc.

**How can the columns of a table in hive be written to a file?**

By using awk command in shell, the output from HiveQL (Describe) can be written to a file.

hive -S -e “describe table\_name;” | awk -F” ” ’{print 1}’ > ~/output.

**CONCAT function in Hive with Example?**

CONCAT function will concat the input strings. You can specify any number of strings separated by comma.

**Example**:

CONCAT (‘Hive’,’-’,’performs’,’-’,’good’,’-’,’in’,’-’,’Hadoop’);

**Output:**

Hive-performs-good-in-Hadoop

So, every time you delimit the strings by ‘-’. If it is common for all the strings, then Hive provides another command CONCAT\_WS. Here you have to specify the delimit operator first.

CONCAT\_WS (‘-’,’Hive’,’performs’,’good’,’in’,’Hadoop’);

Output: Hive-performs-good-in-Hadoop

**REPEAT function in Hive with example?**

REPEAT function will repeat the input string n times specified in the command.

**Example:**

REPEAT(‘Hadoop’,3);

**Output:**

HadoopHadoopHadoop.

Note: You can add a space with the input string also.

**TRIM function in Hive with example?**

TRIM function will remove the spaces associated with a string.

**Example:**  
TRIM(‘  Hadoop  ‘);

**Output:**  
Hadoop.

**Note:** If you want to remove only leading or trialing spaces then you can specify the below commands respectively.  
LTRIM(‘  Hadoop’);  
RTRIM(‘Hadoop  ‘);

**REVERSE function in Hive with example?**

REVERSE function will reverse the characters in a string.

**Example:**  
REVERSE(‘Hadoop’);

**Output:**  
poodaH

**LOWER or LCASE function in Hive with example?**

LOWER or LCASE function will convert the input string to lower case characters.

**Example:**  
LOWER(‘Hadoop’);  
LCASE(‘Hadoop’);

**Output:**  
hadoop

**Note:**  
If the characters are already in lower case then they will be preserved.

**UPPER or UCASE function in Hive with example?**

UPPER or UCASE function will convert the input string to upper case characters.

**Example:**  
UPPER(‘Hadoop’);  
UCASE(‘Hadoop’);

**Output:**  
HADOOP

**Note:**  
If the characters are already in upper case then they will be preserved.

**Double type in Hive – Important points?**

It is important to know about the double type in Hive. Double type in Hive will present the data differently unlike RDBMS.  
See the double type data below:  
24624.0  
32556.0  
3.99893E5  
4366.0

E5 represents 10^5 here. So, the value 3.99893E5 represents 399893. All the calculations will be accurately performed using double type. The maximum value for a IEEE 754 double is about 2.22E308.

It is crucial while exporting the double type data to any RDBMS since the type may be wrongly interpreted. So, it is advised to cast the double type into appropriate type before exporting.

**Rename a table in Hive – How to do it?**

Using ALTER command, we can rename a table in Hive.  
ALTER TABLE hive\_table\_name RENAME  TO new\_name;

There is another way to rename a table in Hive. Sometimes, ALTER may take more time if the underlying table has more partitions/functions. In that case, Import and export options can be utilized. Here you are saving the hive data into HDFS and importing back to new table like below.  
EXPORT TABLE tbl\_name TO ‘HDFS\_location';  
IMPORT TABLE new\_tbl\_name FROM ‘HDFS\_location';

If you prefer to just preserve the data, you can create a new table from old table like below.  
CREATE TABLE new\_tbl\_name AS SELECT \* FROM old\_tbl\_name;  
DROP TABLE old\_tbl\_name;

**How to change a column data type in Hive?**

ALTER TABLE table\_name CHANGE column\_name column\_name new\_datatype;  
Example: If you want to change the data type of ID column from integer to bigint in a table called employee.  
ALTER TABLE employee CHANGE id id BIGINT;

**Difference between order by and sort by in hive?**

SORT BY will sort the data within each reducer. You can use any number of reducers for SORT BY operation.  
ORDER BY will sort all of the data together, which has to pass through one reducer. Thus, ORDER BY in hive uses single reducer.  
ORDER BY guarantees total order in the output while SORT BY only guarantees ordering of the rows within a reducer. If there is more than one reducer, SORT BY may give partially ordered final results

**RLIKE in Hive?**

RLIKE (Right-Like) is a special function in Hive where if any substring of A matches with B then it evaluates to true. It also obeys Java regular expression pattern. Users don’t need to put % symbol for a simple match in RLIKE.

**Examples:**  
‘Express’ RLIKE ‘Exp’ –> True  
‘Express’ RLIKE ‘^E.\*’ –> True (Regular expression)

Moreover, RLIKE will come handy when the string has some spaces. Without using TRIM function, RLIKE satisfies the required scenario. Suppose if A has value ‘Express ‘ (2 spaces additionally) and B has value ‘Express’ RLIKE will work better without using TRIM.  
‘Express’ RLIKE ‘Express’ –> True

**Note:**  
RLIKE evaluates to NULL if A or B is NULL.

**Difference between external table and internal table in HIVE ?**

Hive has a relational database on the master node it uses to keep track of state. For instance, when you CREATE TABLE FOO(foo string) LOCATION ‘hdfs://tmp/';, this table schema is stored in the database. If you have a partitioned table, the partitions are stored in the database(this allows hive to use lists of partitions without going to the filesystem and finding them, etc). These sorts of things are the ‘metadata’.  
When you drop an internal table, it drops the data, and it also drops the metadata. When you drop an external table, it only drops the meta data. That means hive is ignorant of that data now. It does not touch the data itself.

**Bottom of Form**

**What is Hive Metastore?**

Hive metastore is a database that stores metadata about your Hive tables (eg. table name, column names and types, table location, storage handler being used, number of buckets in the table, sorting columns if any, partition columns if any, etc.). When you create a table, this metastore gets updated with the information related to the new table which gets queried when you issue queries on that table.

**Wherever Different(Drectory) I run hive query, it creates new metastore\_db, please explain the reason for it?**

Whenever you run the hive in embedded mode, it creates the local metastore. And before creating the metastore it looks whether metastore already exist or not. This property is defined in configuration file hive-site.xml. Property is “javax.jdo.option.ConnectionURL” with default value “jdbc:derby:;databaseName=metastore\_db;create=true”. So to change the behavior change the location to absolute path, so metastore will be used from that location.

**Is it possible to use same metastore by multiple users, in case of embedded hive?**

No, it is not possible to use metastore in sharing mode. It is recommended to use standalone “real” database like MySQL or PostGresSQL.

**Is multiline comment supported in Hive Script ?**

No.

**If you run hive as a server, what are the available mechanism for connecting it from application?**

There are following ways by which you can connect with the Hive Server:

**1. Thrift Client**: Using thrift you can call hive commands from a various programming languages e.g. C++, Java, PHP, Python and Ruby.

**2. JDBC Driver**: It supports the Type 4 (pure Java) JDBC Driver

**3. ODBC Driver:** It supports ODBC protocol.

**What is SerDe in Apache Hive ?**

A SerDe is a short name for a Serializer Deserializer. Hive uses SerDe (and FileFormat) to read and write data from tables. An important concept behind Hive is that it DOES NOT own the Hadoop File System (HDFS) format that data is stored in.

Users are able to write files to HDFS with whatever tools/mechanism takes their fancy("CREATE EXTERNAL TABLE" or "LOAD DATA INPATH," ) and use Hive to correctly "parse" that file format in a way that can be used by Hive. A SerDe is a powerful (and customizable) mechanism that Hive uses to "parse" data stored in HDFS to be used by Hive.

**Which classes are used by the Hive to Read and Write HDFS Files ?**

Following classes are used by Hive to read and write HDFS files •**TextInputFormat/HiveIgnoreKeyTextOutputFormat**: These 2 classes read/write data in plain text file format.

•**SequenceFileInputFormat/SequenceFileOutputFormat**: These 2 classes read/write data in hadoop SequenceFile format.

**Give examples of the SerDe classes whihc hive uses to Serializa and Deserilize data ?**

Hive currently use these SerDe classes to serialize and deserialize data: • MetadataTypedColumnsetSerDe: This SerDe is used to read/write delimited records like CSV, tab-separated control-A separated records (quote is not supported yet.)

**• ThriftSerDe:** This SerDe is used to read/write thrift serialized objects. The class file for the Thrift object must be loaded first.

**• DynamicSerDe:** This SerDe also read/write thrift serialized objects, but it understands thrift DDL so the schema of the object can be provided at runtime. Also it supports a lot of different protocols, including TBinaryProtocol, TJSONProtocol, TCTLSeparatedProtocol (which writes data in delimited records).

**What is ObjectInspector functionality ?**

Hive uses ObjectInspector to analyze the internal structure of the row object and also the structure of the individual columns. ObjectInspector provides a uniform way to access complex objects that can be stored in multiple formats in the memory, including: •Instance of a Java class (Thrift or native Java) •A standard Java object (we use java.util.List to represent Struct and Array, and use java.util.Map to represent Map) •A lazily-initialized object (For example, a Struct of string fields stored in a single Java string object with starting offset for each field) A complex object can be represented by a pair of ObjectInspector and Java Object. The ObjectInspector not only tells us the structure of the Object, but also gives us ways to access the internal fields inside the Object.

**What is the functionality of Query Processor in Apached Hive ?**

This component implements the processing framework for converting SQL to a graph of map/reduce jobs and the execution time framework to run those jobs in the order of dependencies.

**Difference between Hive and Pig**

Hive was designed to appeal to a community comfortable with SQL. Its philosophy was that we don't need yet another scripting language. Hive supports map and reduce transform scripts in the language of the user's choice (which can be embedded within SQL clauses). It is widely used in Facebook by analysts comfortable with SQL as well as by data miners programming in Python. SQL compatibility efforts in Pig have been abandoned AFAIK - so the difference between the two projects is very clear.

Supporting SQL syntax also means that it's possible to integrate with existing BI tools like Microstrategy. Hive has an ODBC/JDBC driver (that's a work in progress) that should allow this to happen in the near future. It's also beginning to add support for indexes which should allow support for drill-down queries common in such environments.

Finally--this is not pertinent to the question directly--Hive is a framework for performing analytic queries. While its dominant use is to query flat files, there's no reason why it cannot query other stores. Currently Hive can be used to query data stored in Hbase (which is a key-value store like those found in the guts of most RDBMSes), and the HadoopDB project has used Hive to query a federated RDBMS tier.

Pig: a dataflow language and environment for exploring very large datasets.

Pig tends to create a flow of data: small steps where in each you do some processing

Hive: a distributed data warehouse

Hive gives you SQL-like language to operate on your data, so transformation from RDBMS is much easier (Pig can be easier for someone who had not earlier experience with SQL)

Hive, which is RDBMS based, needs the data to be first imported (or loaded) and after that it can be worked upon. So if you were using Hive on streaming data, you would have to keep filling buckets (or files) and use hive on each filled bucket, while using other buckets to keep storing the newly arriving data.

It is also worth noting, that for Hive you can nice interface to work with this data (Beeswax for HUE, or Hive web interface), and it also gives you metastore for information about your data (schema, etc) which is useful as a central information about your data.

I use both Hive and Pig, for different queries (I use that one where I can write query faster/easier, I do it this way mostly ad-hoc queries) - they can use the same data as an input. But currently I'm doing much of my work through Beeswax

## PIG and HIVE:

**Type of flow:**  
PIG is a procedural data-flow language. A procedural language is executing step-by-step approach defined by the programmers. You can control the optimization of every step.   
HIVE looks like SQL language. Thus, it becomes declarative language. You can specify what should be done rather how should be done. Optimization is difficult in HIVE since HIVE depends on its own optimizer.  
  
**Ease of use:**  
PIG requires some additional time to learn since the syntax is new and different.  
HIVE is very special since it almost looks like SQL. Developers seeing HIVE commands are excited to use it.  
  
**Nature of usage:**  
PIG is recommended for Programmers and software developers. The main reason is its efficiency in computing. When your query becomes complex with most of joins and filters, then PIG is strongly recommended.  
Hive is mostly in Analytics area. Ofcourse, it rules the analytics in Hadoop. (Datawarehouse solution). While generating the reports, people prefer to code in HIVE than PIG. If your query has minimum of joins and filters you can go ahead with HIVE. On the other hand, if the query has lot of joins, HIVE may degrade the performance.  
  
**Type of Data:**  
PIG handles both structured and unstructured data efficiently.  
Hive handles structured data very efficiently.  
  
**Intermediate layers:**  
PIG represent data in term of variables. Whenever, you want to store an intermediate result, then it is easy to store that in a variable and you can refer it later.  
HIVE represent data in terms of tables. It is difficult to store the intermediate result in HIVE. (You have to create a table and insert the values from another table). Thus, when a complex query comes into picture, HIVE code may exceed hundred of lines.  
  
**Debugging facility:**  
PIG code can be debugged in Local.  
Debugging HIVE code in local is complex and time consuming.

**Versatality:**  
Writing user defined functions (UDF) is easy with PIG.  
UDF in HIVE is complex.  
  
**Maintenance:**  
PIG requires a bit more maintenance than HIVE.  
Maintenance in HIVE is very easy.  
  
**Persistent:**  
In PIG, you may not retain the values of variables. Every time you have to rerun the PIG code to get the value from a variable.  
In HIVE, the tables (external) will remain with life even you quit the session. This is because the external tables will still point to HDFS file.  
  
**Development time:**  
PIG development may require more time than HIVE. But it is purely based on the familiarity with PIG code.  
Being a SQLish language, the development time will be very less.  
  
**Compatibility:**  
RDBMS to PIG compatibility is slightly complex since the code syntax of PIG is entirely different.  
Most of the SQL statements that you executed in your RDBMS already can be put in directly in HIVE. It will work. Only few things need to be modified compared to RDBMS SQL.  
  
**Handling BIG data:**  
PIG efficiently handles more amount of data.  
HIVE sometimes leads to memory overflow or deceptive performance. However, several parameters are there to adjust and address the issue.   
  
**Giants with these Giants:**  
PIG - used by Yahoo!, Twitter, LinkedIn  
HIVE - used by Facebook.

|  |  |  |
| --- | --- | --- |
| **Type of flow:** | PIG is a procedural data-flow language. A procedural language is executing step-by-step approach defined by the programmers. You can control the optimization of every step. | HIVE looks like SQL language. Thus, it becomes declarative language. You can specify what should be done rather how should be done. Optimization is difficult in HIVE since HIVE depends on its own optimizer. |
| **Ease of use:** | PIG requires some additional time to learn since the syntax is new and different. | HIVE is very special since it almost looks like SQL. Developers seeing HIVE commands are excited to use it. |
| **Nature of usage:** | PIG is recommended for Programmers and software developers. The main reason is its efficiency in computing. When your query becomes complex with most of joins and filters, then PIG is strongly recommended. | Hive is mostly in Analytics area. Ofcourse, it rules the analytics in Hadoop. (Datawarehouse solution). While generating the reports, people prefer to code in HIVE than PIG. If your query has minimum of joins and filters you can go ahead with HIVE. On the other hand, if the query has lot of joins, HIVE may degrade the performance. |
| **Type of Data:** | PIG handles both structured and unstructured data efficiently. | Hive handles structured data very efficiently. |
| **Intermediate layers:** | PIG represent data in term of variables. Whenever, you want to store an intermediate result, then it is easy to store that in a variable and you can refer it later. | HIVE represent data in terms of tables. It is difficult to store the intermediate result in HIVE. (You have to create a table and insert the values from another table). Thus, when a complex query comes into picture, HIVE code may exceed hundred of lines. |
| **Debugging facility:** | PIG code can be debugged in Local. | Debugging HIVE code in local is complex and time consuming. |
| **Versatality:** | Writing user defined functions (UDF) is easy with PIG. | UDF in HIVE is complex. |
| **Maintenance:** | PIG requires a bit more maintenance than HIVE. | Maintenance in HIVE is very easy. |
| **Persistent:** | In PIG, you may not retain the values of variables. Every time you have to rerun the PIG code to get the value from a variable. | In HIVE, the tables (external) will remain with life even you quit the session. This is because the external tables will still point to HDFS file. |
| **Development time:** | PIG development may require more time than HIVE. But it is purely based on the familiarity with PIG code. | Being a SQLish language, the development time will be very less. |
| **Compatibility:** | RDBMS to PIG compatibility is slightly complex since the code syntax of PIG is entirely different. | Most of the SQL statements that you executed in your RDBMS already can be put in directly in HIVE. It will work. Only few things need to be modified compared to RDBMS SQL. |
| **Handling BIG data:** | PIG efficiently handles more amount of data. | HIVE sometimes leads to memory overflow or deceptive performance. However, several parameters are there to adjust and address the issue. |
| **Giants with these Giants:** | PIG - used by Yahoo!, Twitter, LinkedIn | HIVE - used by Facebook. |