**HIVE INTERVIEW QUESTIONS**

1. **What is the definition of Hive? What is the present version of Hive?**

### Ans: Hive is a Data Warehousing service and not a typical database. [Apache hive](https://www.geeksforgeeks.org/apache-hive/) is a data warehousing tool built on top of [Hadoop](https://www.geeksforgeeks.org/hadoop-an-introduction/) and used for extracting meaningful information from data. Data warehousing is all about storing all kinds of data generated from different sources at the same location. The data is mostly available in 3 forms i.e. structured(SQL database), semi-structured(XML or JSON) and unstructured(music or video). To process the structured data available in the tabular format we use Hive on top of Hadoop. The Hive is so powerful that it can query Petabytes(PB) of data very efficiently.  “4.0.0-alpha-2” is the new version of the hive available in the market.

1. **Is Hive suitable to be used for OLTP systems? Why?**

Ans: With hive we can not perform the crud functions like insert, update at row level and therefore it’s not suitable for OLTP systems.

1. **How is HIVE different from RDBMS? Does hive support ACID**

**transactions. If not then give the proper reason.**

**Ans :** Relational databases, or RDBMS, is a database that stores data in a structured format with rows and columns, a structured form called “tables.” Hive, on the other hand, is a data warehousing system that offers data analysis and queries. Hive supports all ACID properties which enable us to use transactions, create transactional tables, and run queries like Insert, Update, and Delete on tables.  older versions of Hive doesn’t support ACID transactions on tables. Though in newer versions it supports by default ACID transactions are disabled and you need to enable it before start using it. Hive supports full ACID semantics at the row level so that one application can add rows while another reads from the same partition without interfering with each other.

**4. Explain the hive architecture and the different components of a Hive**

**architecture?**

**Ans: Hive architecture User Interface (UI) –**   
As the name describes User interface provide an interface between user and hive. It enables user to submit queries and other operations to the system. Hive web UI, Hive command line, and Hive HD Insight (In windows server) are supported by the user interface. 

* **Hive Server**– It is referred to as Apache Thrift Server. It accepts the request from different clients and provides it to Hive Driver.
* **Driver –**   
  Queries of the user after the interface are received by the driver within the Hive. Concept of session handles is implemented by driver. Execution and Fetching of APIs modelled on JDBC/ODBC interfaces is provided by the user.
* **Compiler –**   
  Queries are parses, semantic analysis on the different query blocks and query expression is done by the compiler. Execution plan with the help of the table in the database and partition metadata observed from the metastore are generated by the compiler eventually.
* **Metastore –**   
  All the structured data or information of the different tables and partition in the warehouse containing attributes and attributes level information are stored in the metastore. Sequences or de-sequences necessary to read and write data and the corresponding HDFS files where the data is stored. Hive selects corresponding database servers to stock the schema or Metadata of databases, tables, attributes in a table, data types of databases, and HDFS mapping.
* **Execution Engine –**   
  Execution of the execution plan made by the compiler is performed in the execution engine. The plan is a DAG of stages. The dependencies within the various stages of the plan is managed by the execution engine as well as it executes these stages on the suitable system components.

**5. Mention what Hive query processor does? And Mention what are the**

**components of a Hive query processor?**

Ans:Hive query processor convert graph of MapReduce jobs with the execution time framework.  So that the jobs can be executed in the order of dependencies.

* Parse and Semantic Analysis (ql/parse)
* Metadata Layer (ql/metadata)
* Type Interfaces (ql/typeinfo)
* Sessions (ql/session)
* Map/Reduce Execution Engine (ql/exec)
* Plan Components (ql/plan)
* Hive Function Framework (ql/udf)
* Tools (ql/tools)
* Optimizer (ql/optimizer)

**6. What are the three different modes in which we can operate Hive?**

**Ans:** Hive can operate in two modes depending on the size of data nodes in Hadoop.

These modes are,

* **Local mode**
* **Map reduce mode**

**When to use Local mode:**

* If the Hadoop installed under pseudo mode with having one data node we use Hive in this mode
* If the data size is smaller in term of limited to single local machine, we can use this mode
* Processing will be very fast on smaller data sets present in the local machine

**When to use Map reduce mode:**

* If Hadoop is having multiple data nodes and data is distributed across different node we use Hive in this mode
* It will perform on large amount of data sets and query going to execute in parallel way
* Processing of large data sets with better performance can be achieved through this mode

In Hive, we can set this property to mention which mode Hive can work? By default, it works on Map Reduce mode and for local mode you can have the following setting.

Hive to work in local mode set

**SET mapred.job.tracker=local;**

From the Hive version 0.7 it supports a mode to run map reduce jobs in local mode automatically.

**7. Features and Limitations of Hive.**

**Ans:**

|  |  |
| --- | --- |
| **Supported Computing Engine** | Hive supports MapReduce, Tez, and Spark computing engine. |
| **Framework** | Hive is a stable batch-processing framework built on top of the Hadoop Distributed File system and can work as a data warehouse. |
| **Easy To Code** | Hive uses HIVE query language to query structure data which is easy to code. The 100 lines of java code we use to query a structure data can be minimized to 4 lines with HQL. |
| **Declarative** | HQL is a declarative language like SQL means it is non-procedural. |
| **Structure Of Table** | The table, the structure is similar to the RDBMS. It also supports partitioning and bucketing. |
| **Supported data structures** | Partition, Bucket, and tables are the 3 data structures that hive supports. |
| **Supports ETL** | Apache hive supports ETL i.e. Extract Transform and Load. Before Hive python is used for ETL. |
| **Storage** | Hive supports users to access files from HDFS, Apache HBase, Amazon S3, etc. |
| **Capable** | Hive is capable to process very large datasets of Petabytes in size. |
| **Helps in processing unstructured data** | We can easily embed custom MapReduce code with Hive to process unstructured data. |
| **Drivers** | JDBC/ODBC drivers are also available in Hive. |
| **Fault Tolerance** | Since we store Hive data on HDFS so fault tolerance is provided by Hadoop. |
| **Area of uses** | We can use a hive for data mining, predictive modeling, and document indexing. |

| **Limitation** | **Explanation** |
| --- | --- |
| **Does not support OLTP** | Apache Hive doesn’t support online transaction processing (OLTP) but Online Analytical Processing(OLAP) is supported. |
| **Doesn’t support subqueries** | Subqueries are not supported. |
| **Latency** | The latency in the Apache hive query is very high. |
| **Only non-real or cold data is supported** | Hive is not used for real-time data querying since it takes a while to produce a result. |
| **Transaction processing is not supported** | HQL does not support the Transaction processing feature. |

**8. How to create a Database in HIVE?**

**Ans:** Create Database is a statement used to create a database in Hive. A database in Hive is a namespace or a collection of tables. The syntax for this statement is as follows:

CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database name>

Here, IF NOT EXISTS is an optional clause, which notifies the user that a database with the same name already exists. We can use SCHEMA in place of DATABASE in this command.

**9. How to create a table in HIVE?**

**Ans:** Create Table is a statement used to create a table in Hive. The syntax and example are as follows:

hive> CREATE TABLE IF NOT EXISTS employee (

id int,

name String,

salary String,

destination String)

COMMENT ‘Employee details’

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ‘\t’

LINES TERMINATED BY ‘\n’

STORED AS TEXTFILE;

**10. What do you mean by describe and describe extended and describe**

**formatted with respect to the database and table.**

Ans:Describe – It is used to get the column names and their data types.

Describe extended – it is used to show the columns, their data types and other details about the table in a single line.

Describe formatted – it is also used to show the table columns, their data types, and other details but in multiple lines.

**11. How to skip header rows from a table in Hive?**

Ans: While creating a table in the hive we can skip the header rows by adding this additional command**:**

ALTER TABLE tablename

SET TBLPROPERTIES ("skip.header.line.count"="1");

**12. What is a hive operator? What are the different types of hive operators?**

Ans: **Apache Hive** provides various Built-in operators for data operations to be implemented on the tables present inside the Apache Hive warehouse.

**Hive operators** are used for mathematical operations on operands. It returns the specific value as per the logic applied.

There are four types of operators in Hive:

* Relational Operators
* Arithmetic Operators
* Logical Operators
* Complex Operators

**13. Explain about the Hive Built-In Functions**

Ans: There are several types of Hive Built-in Functions such as Mathematical functions, Collection functions, Type conversion functions, Date functions, Conditional functions, and String functions.

Hive was designed to perform SQL like queries on the huge datasets stored in HDFS. It supports different types of data types, as well as operators that are not supported by different databases. For performing some specific mathematical and arithmetic operations, Hive provides some built-in functions. These built-in functions extract data from tables in hive and process the calculations. For using Hive built-in functions in our applications, we have to first check the application requirement. It is possible to call these hive built-in functions directly in our application.

**14. Write hive DDL and DML commands.**

Ans: Hive DDL (Data Definition Language) commands are the statements used for defining and changing the structure of a table or database in Hive. It is used to build or modify the tables and other objects in the database.

The several types of Hive DDL commands are:

1. CREATE
2. SHOW
3. DESCRIBE
4. USE
5. DROP
6. ALTER
7. TRUNCATE

Hive QL supports DDL, DML, and user-defined functions.

Hive DML (Data Manipulation Language) commands are used to insert, update, retrieve, and delete data from the Hive table once the table and database schema has been defined using Hive DDL commands.

The various Hive DML commands are:

1. [LOAD](https://data-flair.training/blogs/hive-dml-commands/#LOAD-command)
2. [SELECT](https://data-flair.training/blogs/hive-dml-commands/#SELECT-command)
3. [INSERT](https://data-flair.training/blogs/hive-dml-commands/#INSERT-command)
4. [DELETE](https://data-flair.training/blogs/hive-dml-commands/#DELETE-command)
5. [UPDATE](https://data-flair.training/blogs/hive-dml-commands/#UPDATE-command)
6. [EXPORT](https://data-flair.training/blogs/hive-dml-commands/#EXPORT-command)
7. [IMPORT](https://data-flair.training/blogs/hive-dml-commands/#IMPORT-command)

**15.Explain about SORT BY, ORDER BY, DISTRIBUTE BY and CLUSTER BY in Hive.**

Ans: The **SORT BY**and **ORDER BY** clauses are used to define the order of the output data. However, **DISTRIBUTE BY**and **CLUSTER BY**clauses are used to distribute the data to multiple reducers based on the key columns. We can use [Sort by or Order by or Distribute by or Cluster by](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+SortBy) clauses in a hive SELECT query to get the output data in the desired order.

* The SORT by clause **sorts the data per reducer**. As a result, if we have N number of reducers, **we will have N number of sorted files** in the output. These files can have **overlapping data ranges**. Also, the output data is not globally sorted because the hive sorts the rows before feeding them to reducers based on the key columns used in the SORT BY clause. The syntax of the SORT BY clause is as below:

**SELECT Col1, Col2,……Coln FROM TableName SORT BY Col1 <ASC | DESC>, Col2 <ASC | DESC>, …. Coln <ASC | DESC>**

**SELECT** SalesYear, Amount

**FROM** tbl\_Sales

SORT **BY** SalesYear;

* ORDER BY clause **orders the data globally**. Because it ensures the global ordering of the data, all the **data need to be passed from a single reducer only**. As a result, the order by clause **outputs one single file only**. Bringing all the data on one single reducer can become a performance killer, especially if our output dataset is significantly large. So, we should always avoid the ORDER BY clause in the hive queries. However, if we need to enforce a global ordering of the data, and the output dataset is not that big, we can use this hive clause to order the final dataset globally.

The syntax of the ORDER BY clause in hive is as below:

**SELECT Col1, Col2,……ColN FROM TableName ORDER BY Col1 <ASC | DESC>, Col2 <ASC | DESC>, …. ColN <ASC | DESC>**

* DISTRIBUTE BY clause is used to **distribute the input rows among reducers**. It ensures that all rows for the same key columns are going to the same reducer. So, if we need to partition the data on some key column, we can use the DISTRIBUTE BY clause in the hive queries. However, the DISTRIBUTE BY clause **does not sort the data either at the reducer level or globally**. Also, the same key values might not be placed next to each other in the output dataset.

As a result, the DISTRIBUTE BY clause may **output N number of unsorted files** where N is the number of reducers used in the query processing. But, the output files do not contain overlapping data ranges.

The syntax of the DISTRIBUTE BY clause in the hive is as below:

**SELECT Col1, Col2,……Coln FROM TableName DISTRIBUTE BY Col1, Col2, ….. Coln**

* CLUSTER BY clause **is a combination of DISTRIBUTE BY and SORT BY clauses together**. That means the output of the CLUSTER BY clause is equivalent to the output of **DISTRIBUTE BY + SORT BY** clauses. The CLUSTER BY clause distributes the data based on the key column and then sorts the output data by putting the **same key column values adjacent to each other**. So, the output of the CLUSTER BY clause is **sorted at the reducer level**. As a result, we can get **N number of sorted output files** where N is the number of reducers used in the query processing. Also, the CLUSTER by clause ensures that we are getting **non-overlapping data ranges** into the final outputs. However, if the query is processed by only one reducer the output will be equivalent to the output of the ORDER BY clause.

The syntax of the CLUSTER BY clause is as below:

**SELECT Col1, Col2,……ColN FROM TableName CLUSTER BY Col1, Col2, ….. ColN**

**16. Difference between "Internal Table" and "External Table" and Mention**

**when to choose “Internal Table” and “External Table” in Hive?**

Ans: The major difference between the internal and external table is how the data of the table is stored i.e whether it’s stored in warehouse directory inside hdfs or at hdfs location.

Also, if we drop an internal table then it’s metadata and original data both will be removed. While in external table only metadata gets deleted and the original data remains untouched.

In the internal table we are uploading and creating data then its going to warehouse and our table is formed but in the external table we already know the data i.e which data is present in hdfs file and which is supposed to be stored in the table, here we are not loading any data inside HDFS from our side.

**17.Where does the data of a Hive table get stored?**

Ans: Data of a hive table gets stored in the metastore; metastore is the central repository which stores the metadata information about the structure of tables, and partitions, including columns and datatypes.

**18.Is it possible to change the default location of a managed table?**

**Ans:** Yes, we can do it by using the clause – LOCATION ‘<hdfs\_path>’ we can change the default location of a managed table.

**19.What is a metastore in Hive? What is the default database provided by**

**Apache Hive for metastore?**

Ans: Metastore is the central repository that stores the metadata information about the structure of tables, partitions including columns, and data types. It also stores the information of the serializer and deserializer required for the read/write operation.

“Derby” Database is the default database provided by apache hive for metastore.

**20.Why does Hive not store metadata information in HDFS?**

Ans: Hive stores metadata information in the metastore using RDBMS instead of HDFS. The reason for choosing RDBMS is to achieve low latency as HDFS read/write operations are time consuming processes.

**21.What is a partition in Hive? And Why do we perform partitioning in**

**Hive?**

Ans: Hive organizes tables into partitions for grouping similar type of data together based on a column or partition key. Each Table can have one or more partition keys to identify a particular partition. Physically, a partition is nothing but a sub-directory in the table directory. Partitioning provides granularity in a Hive table and therefore, reduces the query latency by scanning only **relevant** partitioned data instead of the whole data set. For example, we can partition a transaction log of an e–commerce website based on months like Jan, February, etc. So, any analytics regarding a particular month, say Jan, will have to scan the Jan partition (sub–directory) only instead of the whole table data.

**22. What is the difference between dynamic partitioning and static**

**partitioning?**

Ans: Static Partitioning: In static partition, the value of the partition column is known to us as we ourself load the data in a specific partition. Also, static partitioning takes less time while loading the data. Plus static partitioning does not include any reducers in the process as the hive does not need to perform any operation for distinguishing the data.

Dynamic partitioning: In dynamic partitioning, the value of the partition column Is unknown to us, Here, the hive will identify the unique values from the partitioned column and therefore it takes a longer time while loading the data. Plus, Dynamic partitioning also includes reducers because the hive needs to calculate and perform some operations to separate the data.

**23. How do you check if a particular partition exists?**

Ans: We can check if a particular partition exists or not by simply writing a query:

Show PARTITIONS table\_name

PARTITION(partitioned\_column = ‘partition\_value’)

**24.How can you stop a partition form being queried?**

Ans: We can stop a partition from being queried by simply writing the query-

By using the ENABLE OFFLINE clause with ALTER TABLE statement

**25.Why do we need buckets? How Hive distributes the rows into buckets?**

Ans: Buckets in Hive are used in segregating Hive table data into multiple files or directories. They are used for efficient querying.

Here are the two main reasons for performing bucketing to a partition:

* Amap side joinrequires data belonging to a unique join key to be present in the same partition. However, what about those cases where our partition key differs from that of the join key? Therefore, we can perform a map side join by bucketing the table using the join key in such cases.
* Bucketing makes the sampling process more efficient and, thus, allows us to decrease the query time.

**Hive** usestheformula: hash\_function(bucketing\_column)modulo(num\_of\_buckets) to calculate the row’s bucket number. Here, hash\_function is based on the Data type of the column. The hash\_function is for integer data type: hash\_function (int\_type\_column)= value of int\_type\_column

**26. In Hive, how can you enable buckets?**

Ans: set.hive.enforce.bucketing=true;

Or partition = true;

**27.How does bucketing help in the faster execution of queries?**

Ans: It provides faster query response like partioning.In bucketing due to equal volumes of data in each partition, joins at Map side will be quicker.

**28. How to optimise Hive Performance? Explain in very detail.**

Ans: Best Hive Optimization techniques are: Execution Engine, Usage of Suitable File Format, Hive Partitioning, Bucketing in Hive, Vectorization in Hive, Cost-Based Optimization in Hive, and Hive Indexing.

### 1. Tez-Execution Engine in Hive

Tez Execution Engine – Hive Optimization Techniques, to increase the Hive performance of our hive query by using our execution engine as Tez. On defining Tez, it is a new application framework built on **Hadoop Yarn**.

That executes complex-directed acyclic graphs of general data processing tasks. However, we can consider it to be a much more flexible and powerful successor to the map-reduce framework.  
In addition, to write native YARN applications on Hadoop that bridges the spectrum of interactive and batch workloads Tez offers an API framework to developers. To be more specific,  working with petabytes of data over thousands of nodes allows those data access applications.

### 2. Usage of Suitable File Format in Hive

ORCFILE File Formate – if we use appropriate file format on the basis of data. It will drastically increase our query performance. Basically, for increasing query performance ORC file format is best suitable. Here, ORC refers to Optimized Row Columnar. That implies we can store data in an optimized way than the other file formats. To be more specific, ORC reduces the size of the original data up to 75%. Hence,  data processing speed also increases. On comparing to Text, Sequence, and RC file formats, ORC shows better performance.

### 3. Hive Partitioning

Hive Partition – Hive reads all the data in the directory Without partitioning. Further, it applies the query filters on it.  Since all data has to be read this is a slow as well as expensive. Also, users need to filter the data on specific column values frequently. Although, users need to understand the domain of the data on which they are doing analysis, to apply the partitioning in the Hive.  
Basically, by Partitioning all the entries for the various columns of the dataset are segregated and stored in their respective partition. Hence, While we write the query to fetch the values from the table, only the required partitions of the table are queried. Thus it reduces the time taken by the query to yield the result.

### 4. Bucketing in Hive

**Bucketing in Hive** – let’s suppose a scenario. At times, there is a huge dataset available. However, after partitioning on a particular field or fields, the partitioned file size doesn’t match with the actual expectation and remains huge. Still, we want to manage the partition results into different parts. Thus, to solve this issue of partitioning, Hive offers Bucketing concept. Basically,  that allows the user to divide table data sets into more manageable parts.  
Hence, to maintain parts that are more manageable we can use Bucketing. Through it, the user can set the size of the manageable parts or Buckets too.

### 5. Vectorization In Hive

Vectorization In Hive – to improve the performance of operations we use Vectorized query execution. Here operations refer to scans, aggregations, filters, and joins. It happens by performing them in batches of 1024 rows at once instead of single row each time.  
However, this feature is introduced in Hive 0.13. It significantly improves query execution time, and is easily enabled with two parameters settings:  
set hive.vectorized.execution = true  
set hive.vectorized.execution.enabled = true

### 6. Cost-Based Optimization in Hive (CBO)

Cost-Based Optimization in Hive – Before submitting for final execution Hive optimizes each Query’s logical and physical execution plan. Although, until now these optimizations are not based on the cost of the query.  
However, CBO, performs, further optimizations based on query cost in a recent addition to Hive. That results in potentially different decisions: how to order joins, which type of join to perform, the degree of parallelism and others.  
  
To use CBO, set the following parameters at the beginning of your query:

set hive.cbo.enable=true;  
set hive.compute.query.using.stats=true;  
set hive.stats.fetch.column.stats=true;  
set hive.stats.fetch.partition.stats=true;

Then, prepare the data for CBO by running Hive’s “analyze” command to collect various statistics on the tables for which we want to use CBO.

### 7. Hive Indexing

Hive Index – To increase your query performance indexing will definitely help. Basically, for the original table use of indexing will create a separate called index table which acts as a reference.  
As we know, there are many numbers of rows and columns, in a**Hive table**. Basically, it will take a large amount of time if we want to perform queries only on some columns without indexing. Because queries will be executed on all the columns present in the table. Moreover,  there is no need for the query to scan all the rows in the table when we perform a query on a table that has an index, it turned out as the major advantage of using indexing. Further, it checks the index first and then goes to the particular column and performs the operation.Hence, maintaining indexes will be easier for Hive query to look into the indexes first and then perform the needed operations within less amount of time

**29. What is the use of Hcatalog?**

Ans: For sharing Data structures with external systems, Hcatalog is a necessary tool. It offers access to the Hive metastore for reading and writing data in a Hive data warehouse.

**30. Explain about the different types of join in Hive.**

Ans: There are  4 different types of **joins in HiveQL** –

* **JOIN-**  It is very similar to Outer Join in SQL
* **FULL OUTER JOIN –** This join Combines the records of both the left and right outer tables. Basically, that fulfill the join condition.
* **LEFT OUTER JOIN-** Through this Join, All the rows from the left table are returned even if there are no matches in the right table.
* **RIGHT OUTER JOIN –** Here also, all the rows from the right table are returned even if there are no matches in the left table.

**31.Is it possible to create a Cartesian join between 2 tables, using Hive?**

Ans: No. As this kind of Join can not be implemented in mapreduce.

**32.Explain the SMB Join in Hive?**

Ans: SMB (Sort merge Bucket) is a join performed on bucket tables that have the same sorted, bucket, and join condition columns. It reads data from both bucket tables and performs common joins (map and reduce triggered) on the bucket tables. We need to enable the following properties to use SMB:

**> SET hive.input.format=> org.apache.hadoop.hive.ql.io.BucketizedHiveInputFormat;**

**> SET hive.auto.convert.sortmerge.join=true;**

**> SET hive.optimize.bucketmapjoin=true;**

**> SET hive.optimize.bucketmapjoin.sortedmerge=true;**

**> SET hive.auto.convert.sortmerge.join.noconditionaltask=true;**

**33.What is the difference between order by and sort by which one we should**

**use?**

Ans: Hive supports SORT BY which sorts the data per reducer. The difference between "order by" and "sort by" is that the former guarantees total order in the output while the latter only guarantees ordering of the rows within a reducer. If there are more than one reducer, "sort by" may give partially ordered final results.

Despite **ORDER BY** we should use SORT BY. Especially while we have to sort huge datasets. The reason is SORT BY clause sorts the data using multiple reducers. ORDER BY sorts all of the data together using a single reducer.

Hence, using ORDER BY will take a lot of time to execute a large number of inputs.

**34. What is the usefulness of the DISTRIBUTED BY clause in Hive?**

Ans: DISTRIBUTE BY clause is used to **distribute the input rows among reducers**. It ensures that all rows for the same key columns are going to the same reducer. So, if we need to partition the data on some key column, we can use the DISTRIBUTE BY clause in the hive queries. However, the DISTRIBUTE BY clause **does not sort the data either at the reducer level or globally**. Also, the same key values might not be placed next to each other in the output dataset.

As a result, the DISTRIBUTE BY clause may **output N number of unsorted files** where N is the number of reducers used in the query processing. But, the output files do not contain overlapping data ranges.

The syntax of the DISTRIBUTE BY clause in the hive is as below:

**SELECT Col1, Col2,……ColN FROM TableName DISTRIBUTE BY Col1, Col2, ….. ColN**

**35.How does data transfer happen from HDFS to Hive?**

Ans: You can import data from diverse data sources into HDFS, perform ETL processes, and then query the data in Apache Hive.

You really do not load data into Hive. Hive is not data store as well. Hive is query engine which you can use to process data in HDFS. So if you need to process/read data in HDFS using Hive you need to create table on top of it. If you know structure of data then it is simple, create table statement along with where that data resides in HDFS that is its location and you are good to go. You can use that data in Hive.

Create a Hive table that corresponds to the data in HDFS.

Use the Hive Loader to load the data from HDFS into the Hive table.

The data is now available in Hive, and can be queried using SQL-like commands.The Hive Loader can handle various file formats, such as CSV, TSV, JSON, and ORC, and can also handle compressed files. The data transfer process can be automated using tools like Apache Oozie or Apache Airflow

**36.Wherever (Different Directory) I run the hive query, it creates a new**

**metastore\_db, please explain the reason for it?**

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

Also, we cannot use metastore in sharing mode. It is possible to use it in standalone “real” database. Such as MySQL or PostGresSQL.

**37.What will happen in case you have not issued the command: ‘SET**

**hive.enforce.bucketing=true;’ before bucketing a table in Hive?**

Ans: The command: ‘SET hive.enforce.bucketing=true;’ allows you to have the correct number of reducer while using ‘CLUSTER BY’ clause for bucketing a column. In case it’s not done, one may find the number of files generated in the table directory to be unequal to the number of buckets. As an alternative solution, one may also set the number of reducers equal to the number of buckets by using set mapred.reduce.task = num\_bucket.

**38.Can a table be renamed in Hive?**

Ans: We can rename the table name in the hive by using the **alter command.** This command allows you to change the table name as shown below:

ALTER TABLE name RENAME TO new\_name

**39.Write a query to insert a new column(new\_col INT) into a hive table at a**

**position before an existing column (x\_col)**

Ans: ALTER TABLE table\_name

CHANGE column new\_col INT

BEFORE x\_col

**40.What is serde operation in HIVE?**

Ans: Serde- Serializer and Deserializer. Hive uses the SerDe interface for IO. The interface handles both serialization and deserialization and also interpreting the results of serialization as individual fields for processing. A SerDe allows Hive to read in data from a table, and write it back out to HDFS in any custom format. It compresses the data set in bytes for better performance and then deserializes the data into an original form for reading it.

**41.Explain how Hive Deserializes and serialises the data?**

Ans: Hive allows the framework to read or write data in a particular format. These formats parse the structured or unstructured data bytes stored in HDFS in accordance with the schema definition of Hive tables. Hive provides a set of in-built SerDes and also allows the user to create custom SerDes based on their data definition.

**42.Write the name of the built-in serde in hive.**

Ans: Basically, to read and write **HDFS** files Hive uses these FileFormat classes currently:

* **TextInputFormat/HiveIgnoreKeyTextOutputFormat**

It read/write data in plain text file format.

* **SequenceFileInputFormat/SequenceFileOutputFormat**

It read/write data in **Hadoop** SequenceFile format.  
Moreover, to serialize and deserialize data Hive uses these Hive SerDe classes currently:

* **MetadataTypedColumnsetSerDe**

So, to read/write delimited records we use this Hive SerDe. Such as CSV, tab-separated control-A separated records (sorry, quote is not supported yet).

* **LazySimpleSerDe**

Also, to read the same data format as MetadataTypedColumnsetSerDe and TCTLSeparatedProtocol, we can use this Hive SerDe. Moreover, it creates Objects in a lazy way. Hence, that offers better performance.

Basically, with a specified encode charset starting in Hive 0.14.0, it supports read/write data.  
For example:  
  
**ALTER TABLE person SET SERDEPROPERTIES (‘serialization.encoding’=’GBK’)**  
Since, the configuration property hive.lazysimple.extended\_boolean\_literal is set to true (Hive 0.14.0 and later) LazySimpleSerDe can treat ‘T’, ‘t’, ‘F’, ‘f’, ‘1’, and ‘0’ as extended, legal boolean literals.

However, the default is false. Hence it means only ‘TRUE’ and ‘FALSE’ are treated as legal boolean literals.

* **Thrift SerDe in Hive**

To read/write Thrift serialized objects, we use this Hive SerDe. However, make sure, for the Thrift object the class file must be loaded first.

* **Dynamic SerDe in Hive**

**43.What is the need of custom Serde?**

Ans: Depending on the nature of data the user has, the inbuilt SerDe may not satisfy the format of the data. SO users need to write their own java code to satisfy their data format requirements.

**44.Can you write the name of a complex data type(collection data types) in**

**Hive?**

Ans: Yes, Hive supports the complex data types like array, Map, Struct, Union.

**45.Can hive queries be executed from script files? How?**

Ans: Yes, It is possible by using the source command.

For example:

Hive> source/path/to/file/query\_file.hql

**46.What are the default record and field delimiter used for hive text files?**

Ans: The default record delimiter is − \n And the filed delimiters are − \001,\002,\003

**47.How do you list all databases in Hive whose name starts with s?**

Ans:SHOW DATABASES LIKE ‘s.\*’

**48.What is the difference between LIKE and RLIKE operators in Hive?**

Ans: **LIKE** is an operator similar to LIKE in SQL. We use LIKE to search for string with similar text. i.e user\_name LIKE ‘%Smith’

**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.Hive provides RLIKE operator that can be used for searching advanced Regular Expressions in Java. **Ex:** user\_name RLIKE ‘.(Smith|Sam).’

This will return user\_name that has Smith or Sam in it.

**49. How to change the column data type in Hive?**

Ans: To change the column data type in hive we can use the query:

ALTER TABLE table\_name CHANGE column\_name column\_name new\_datatype;

**50.How will you convert the string ’51.2’ to a float value in the particular**

**column?**

Ans:Select cast(‘particular\_column’ AS FLOAT)

**51.What will be the result when you cast ‘abc’ (string) as INT?**

Ans: Hive will return NULL.

**52.What does the following query do?**

**a. INSERT OVERWRITE TABLE employees**

**b. PARTITION (country, state)**

**c. SELECT ..., se.cnty, se.st**

**d. FROM staged\_employees se;**

Ans:

* 1. INSERT OVERWRITE Table employees is used to replace any existing data in the table or partition and insert with the new rows. When working with the partition you can also specify to overwrite only when the partition exists using the IF NOT EXISTS option.
  2. PARTITION(country, state), This the dynamic partitioning partitions of a given table with their partition keys i.e. column names from the table schema.
  3. Select … it’s basically the creation of the schema for the table with different column names.
  4. From staged\_employees se; it’s basically the name of the table from where we are doing the extraction of the columns and forming our resultant schema.

**53. Write a query where you can overwrite data in a new table from the**

**existing table.**

Ans: INSERT OVER WRITE TABLE employees PAITION(Country, state)

SELECT ----- Se cnty, se-st From staged- employees se;

**54.What is the maximum size of a string data type supported by Hive?**

**Explain how Hive supports binary formats.**

Ans: The maximum size of a string data type supported by Hive is 2 GB. Hive supports the text file format by default, and it also supports the binary format sequence files, ORC files, Avro data files, and Parquet files.

* **Sequence file**: It is a splittable, compressible, and row-oriented file with a general binary format.
* **ORC file**: Optimized row columnar (ORC) format file is a record-columnar and column-oriented storage file. It divides the table in row split. Each split stores the value of the first row in the first column and follows subsequently.
* **Avro data file**: It is the same as a sequence file that is splittable, compressible, and row-oriented but without the support of schema evolution and multilingual binding.
* **Parquet file:** In Parquet format, along with storing rows of data adjacent to one another, we can also store column values adjacent to each other such that both horizontally and vertically datasets are partitioned

**55. What File Formats and Applications Does Hive Support?**

Ans: Text File

* Sequence File
* RC File
* AVRO File
* ORC File
* Parquet File

#### Hive Text File Format

**Hive Text file format** is a default storage format. You can use the text format to interchange the data with other client application. The text file format is very common most of the applications. Data is stored in lines, with each line being a record. Each lines are terminated by a newline character (\n).

The text format is simple plane file format. You can use the compression (BZIP2) on the text file to reduce the storage spaces.

Create a TEXT file by add storage option as **‘STORED AS TEXTFILE’** at the end of a Hive CREATE TABLE command.

**Hive Text File Format Examples**

Below is the Hive CREATE TABLE command with storage format specification:

Create table textfile\_table

(column\_specs)

stored as textfile;

#### Hive Sequence File Format

**Sequence files** are Hadoop flat files which stores values in binary key-value pairs. The sequence files are in binary format and these files are able to split. The main advantages of using sequence file is to merge two or more files into one file.

Create a sequence file by add storage option as **‘STORED AS SEQUENCEFILE’** at the end of a Hive CREATE TABLE command.

**Hive Sequence File Format Example**

Below is the Hive CREATE TABLE command with storage format specification:

Create table sequencefile\_table

(column\_specs)

stored as sequencefile;

#### Hive RC File Format

**RCFile** is row columnar file format. This is another form of Hive file format which offers high row level compression rates. If you have requirement to perform multiple rows at a time then you can use RCFile format.

The RCFile are very much similar to the sequence file format. This file format also stores the data as key-value pairs.

Create RCFile by specifying **‘STORED AS RCFILE’** option at the end of a CREATE TABLE Command:

**Hive RC File Format Example**

Below is the Hive CREATE TABLE command with storage format specification:

Create table RCfile\_table

(column\_specs)

stored as rcfile;

#### Hive AVRO File Format

**AVRO** is open source project that provides data serialization and data exchange services for Hadoop. You can exchange data between Hadoop ecosystem and program written in any programming languages. Avro is one of the popular file format in Big Data Hadoop based applications.

Create AVRO file by specifying **‘STORED AS AVRO’** option at the end of a CREATE TABLE Command.

**Hive AVRO File Format Example**

Below is the Hive CREATE TABLE command with storage format specification:

Create table avro\_table

(column\_specs)

stored as avro;

#### Hive ORC File Format

The **ORC file** stands for Optimized Row Columnar file format. The ORC file format provides a highly efficient way to store data in Hive table. This file system was actually designed to overcome limitations of the other Hive file formats. The Use of ORC files improves performance when Hive is reading, writing, and processing data from large tables.

Create ORC file by specifying **‘STORED AS ORC’** option at the end of a CREATE TABLE Command.

**Hive ORC File Format Examples**

Below is the Hive CREATE TABLE command with storage format specification:

Create table orc\_table

(column\_specs)

stored as orc;

#### Hive Parquet File Format

**Parquet** is a column-oriented binary file format. The parquet is highly efficient for the types of large-scale queries. Parquet is especially good for queries scanning particular columns within a particular table. The Parquet table uses compression Snappy, gzip; currently Snappy by default.

Create Parquet file by specifying **‘STORED AS PARQUET’** option at the end of a CREATE TABLE Command.

**Hive Parquet File Format Example**

Below is the Hive CREATE TABLE command with storage format specification:

Create table parquet\_table

(column\_specs)

stored as parquet;

**56.How do ORC format tables help Hive to enhance its performance?**

Ans: We can easily store the Hive Data with the ORC (Optimized Row Column) format in a compressed form, which helps to streamline several limitations.

**57.How can Hive avoid mapreduce while processing the query?**

Ans: We can avoid mapreduce while processing the query by setting hive.exec.mode.local.auto = true.

**58.What is view and indexing in hive?**

Ans: Views are generated based on user requirements. We can save any result set data as a view. The usage of view in Hive is same as that of the view in SQL. It is a standard RDBMS concept. We can execute all DML operations on a view. Like create, drop etc.

An Index is nothing but a pointer on a particular column of a table. Creating an index means creating a pointer on a particular column of a table. **Hive**indexing is a query optimization technique to reduce the time needed to access a column or a set of columns within a Hive database. Index can be dropped and created.

**59.Can the name of a view be the same as the name of a hive table?**

Ans: No. The name of a view must be unique whne compared to all other tables and views present in the same database.

**60.What types of costs are associated in creating indexes on hive tables?**

Ans: Indexes occupy space and there is a processing cost in arranging the values of the column on which the index is created.

**61.Give the command to see the indexes on a table.**

Ans: SHOW INDEX ON table\_name

This will list all the indexes created on any of the columns in the table table\_name.

**62. Explain the process to access subdirectories recursively in Hive queries.**

Ans: By using the below commands, we can access subdirectories recursively in Hive:

hive> Set mapred.input.dir.recursive=true;

hive> Set hive.mapred.supports.subdirectories=true;

Hive tables can be pointed to the higher level directory, and this is suitable for the directory structure

/data/country/state/city/

**63.If you run a select \* query in Hive, why doesn't it run MapReduce?**

**Ans:**

To understand the reason, first we need to know what map and reduce phases mean:-

1. **Map:** Basically a filter which filters and organizes data in sorted order. For e.g. It will filter col1\_name, col2\_name from a row in the second query. However in 1st query you are reading every column, no filtering is required. Hence no Map phase
2. **Reduce**: Reduce is just summary operation data across the rows. for e.g. sum of a coloumn! In both the queries you don't need any summary data. Hence no reducer.

The hive.fetch.task.conversion property of Hive lowers the latency of MapReduce overhead, and in effect when executing queries such as SELECT, FILTER, LIMIT, etc. it skips the MapReduce function.

**64.What are the uses of Hive Explode?**

**Ans:** Hadoop Developers consider an array as their input and convert it into a separate table row. To convert complicated data types into desired table formats, Hive uses Explode.

**65. What is the available mechanism for connecting applications when we**

**run Hive as a server?**

**Ans: Thrift Client**: Using Thrift, we can call Hive commands from various programming languages, such as C++, PHP, Java, Python, and Ruby.

[**JDBC Driver**](https://intellipaat.com/blog/java-jdbc/): JDBC Driver enables accessing data with JDBC support, by translating calls from an application into SQL and passing the SQL queries to the Hive engine.

**ODBC Driver**: It implements the ODBC API standard for the Hive DBMS, enabling ODBC-compliant applications to interact seamlessly with Hive.

**66.Can the default location of a managed table be changed in Hive?**

Ans: Yes, you can do it by using the clause – LOCATION '<hdfs\_path>' we can change the default location of a managed table

67.What is the Hive ObjectInspector function?

Ans: Hive ObjectInspector is a group of flexible APIs to inspect value in different data representation, and developers can extend those API as needed, so technically, object inspector supports arbitrary data type in java. public interface PrimitiveObjectInspector .

**68.What is UDF in Hive?**

**Ans: User Defined Functions**, also known as UDF, allow you to create custom functions to process records or groups of records. Hive comes with [a comprehensive library of functions](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+UDF). There are however some omissions, and some specific cases for which UDFs are the solution.

**69.Write a query to extract data from hdfs to hive.**

**Ans:** This is the most common way to move data into Hive when the ORC file format is required as the target data format. Then Hive can be used to perform a fast parallel and distributed conversion of your data into ORC.

[**https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.4.3/bk\_dataintegration/content/moving\_data\_from\_hdfs\_to\_hive\_external\_table\_method.html**](https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.4.3/bk_dataintegration/content/moving_data_from_hdfs_to_hive_external_table_method.html)

**70.What is TextInputFormat and SequenceFileInputFormat in hive.**

**Ans: TextInputFormat** is one of the file formats of [Hadoop](https://data-flair.training/blogs/hadoop-introduction-tutorial-quick-guide/). As the name suggest,it is used to read lines of text files.  
Basically it helps in generating [key-value pairs](https://data-flair.training/blogs/key-value-pairs-hadoop-mapreduce/) from the text. Firstly text files are broken into lines with the help of line feed(moving one line forward) or carraige return(moving cursor to the begging of the line) to check end of line, this is called as splits.  
After splits are created, key-value pairs are generated with the help of TextInputFormat. In [MapReduce](https://data-flair.training/blogs/hadoop-mapreduce-introduction-tutorial-comprehensive-guide/) data elements are always structured as Key-Value pair.  
So, TextInputFormat helps to generate key and value pair,

key- It is the position in the file  
Value- complete actual line of text

for eg :Text file: humpty dumpty set on wall  
humpty dumpty had a great fall

key-value pair will be like

[KEY=0].[VALUE=humpty dumpty set on wall]

[KEY=26].[VALUE=humpty dumpty had a great fall]

After generation of key-value pair it is passed to Map function to produce intermediate output. then this intermediate output is passed to Reduce function to produce final output.

[Hadoop](https://data-flair.training/blogs/hadoop-introduction-tutorial-quick-guide/) Sequence files are one of the Apache Hadoop specific file formats which stores data in serialized key-value pair. Serialized in the sense: Stream of bytes.

In general, Apache Hadoop supports text files which are quite commonly used for storing the data, besides the text files it also supports binary files and one of these binary formats are called Sequence files. Hadoop Sequence file is a flat file structure which consists of serialized/binary key-value pairs. This is the same format in which the data is stored internally during the processing of the [MapReduce](https://data-flair.training/blogs/hadoop-mapreduce-introduction-tutorial-comprehensive-guide/) tasks.

Hadoop SequenceFile is used in MapReduce as input/Output formats. By default [Mapper](https://data-flair.training/blogs/mapper-in-hadoop-mapreduce/) output is stored on local file system which is in Mapper node. Outputs of Maps are stored using SequenceFile. Internally Hadoop uses SequenceFile format for the Mapper which is stored in the local file system.

**What is the purpose of sequence file?**

1) To enable/store/process binary data

2) The other objective of using SequenceFile is to pack many small files into a single large SequenceFile for the MapReduce computation since the design of Hadoop prefer large files. Sequence file also work well as containers for smaller files. [HDFS](https://data-flair.training/blogs/comprehensive-hdfs-guide-introduction-architecture-data-read-write-tutorial/) and MapReduce are optimized for large files, so packing small files into a sequencefile makes storing and processing the smaller files more efficient.

**71.How can you prevent a large job from running for a long time in a hive?**

**Ans:** This can be achieved by setting the MapReduce jobs to execute in strict mode set hive.mapred.mode=strict;

The strict mode ensures that the queries on partitioned tables cannot execute without defining a WHERE clause.

**72.When do we use explode in Hive?**

**https://www.projectpro.io/recipes/explain-use-of-explode-and-lateral-view-hive#:~:text=Lateral%20View%20Explode%20is%20another,value%20pair%20from%20the%20map.**

**Ans:** Explode is a built-in Hive function that takes an array or a map as input and returns a new row for each element in the array or key-value pair in the map. In otherwords, it "explodes" the input into multiple rows. Let's look at an example:

SELECT explode(array(1, 2, 3));

This query will return three rows, each containing a single integer value:

1

2

3

Similarly, if we have a map with key-value pairs, we can use explode to split it into separate rows:

SELECT explode(map('a', 1, 'b', 2, 'c', 3));

This query will return three rows, each containing a single key-value pair:

a 1

b 2

c 3

**73. Can Hive process any type of data formats? Why? Explain in very detail**

**Ans:** [**https://www.linkedin.com/pulse/file-formats-apache-hive-prateek-kumar/**](https://www.linkedin.com/pulse/file-formats-apache-hive-prateek-kumar/)

Apache Hive is an open source data warehouse software that facilitates querying and managing of large datasets residing in distributed storage. Hive provides a language called HiveQL which allows users to query and is similar to SQL.

Like SQL, HiveQL handles structured data only. By default, Hive has derby database to store the data in it. We can configure Hive with MySQL database. As mentioned HiveQL can handle only structured data. Data is eventually stored in files. There are some specific file formats which Hive can handle such as:

• TEXTFILE

• SEQUENCEFILE

• RCFILE

• ORCFILE

**74.Whenever we run a Hive query, a new metastore\_db is created. Why?**

**Ans:** A local metastore is created when we run Hive in an embedded mode. Before creating, it checks whether the metastore exists or not, and this metastore property is defined in the configuration file, hive-site.xml. The property is:

javax.jdo.option.ConnectionURL

with the default value:

jdbc:derby:;databaseName=metastore\_db;create=true

Therefore, we have to change the behavior of the location to an absolute path so that from that location the metastore can be used.

**75. Can we change the data type of a column in a hive table? Write a**

**complete query.**

Ans: Yes,ALTER TABLE table\_name CHANGE old\_col\_name new\_col\_name new\_data\_type

**76.While loading data into a hive table using the LOAD DATA clause, how**

**do you specify it is a hdfs file and not a local file?**

Ans: We can specify using the syntax i.e for using local file we use- <file:///file_path> While for hdfs file we can simply write the complete path starting with the root directory.

**77. What is the precedence order in Hive configuration?**

Ans:In Hive we can use following precedence order to set the configurable properties.

* Hive SET command has the highest priority
* -hiveconf option from Hive Command Line
* hive-site.xml file
* hive-default.xml file
* hadoop-site.xml file
* hadoop-default.xml file

**78.Which interface is used for accessing the Hive metastore?**

Ans: With WebHCat, applications can make HTTP requests to access the Hive metastore (HCatalog DDL) or to create and queue Hive queries and commands, Pig jobs, and MapReduce or YARN jobs (either standard or streaming). WebHCat was formerly named Templeton.

Features of HWI

**Schema Browsing**

An alternative to running 'show tables' or 'show extended tables' from the CLI is to use the web-based schema browser. The Hive metadata is presented in a hierarchical manner allowing you to start at the database level and click to get information about tables including the SerDe, column names, and column types.

**Detached Query Execution**

A power user issuing multiple Hive queries simultaneously would have multiple CLI windows open. The Hive Web Interface manages the session on the web server, not from inside the CLI window. This allows a user to start multiple queries and return to the web interface later to check the status.

**No Local Installation**

Any user with a web browser can work with Hive. This has the usual web interface benefits. In particular, a user wishing to interact with Hadoop or Hive requires access to many ports. A remote or VPN user would only require access to the Hive Web Interface running by default on 0.0.0.0 tcp/9999.

**79.Is it possible to compress json in the Hive external table ?**

Ans: Yes, we need to gzip our files and put them as is (\*.gz) into the table location. Also one thing to note is that we need to uncompress before selecting it like json. As we can't use both serde (json and gzip).

**80.What is the difference between local and remote megastores?**

**Ans:** Hive is the data-warehousing framework, so hive does not prefer single session. To overcome this limitation of Embedded Metastore, for **Local Metastore** was introduced. This mode allows us to have many Hive sessions i.e. many users can use the metastore at the same time.

metastore service still runs in the same process as the Hive. But it connects to a database running in a separate process, either on the same machine or on a remote machine.

Before starting Apache Hive client, add the JDBC / ODBC driver libraries to the Hive lib folder**.**

In this mode, metastore runs on its own separate JVM, not in the Hive service JVM. If other processes want to communicate with the metastore server they can communicate using Thrift Network APIs.

We can also have one more metastore servers in this case to provide more availability. This also brings better manageability/security because the database tier can be completely firewalled off. And the clients no longer need share database credentials with each Hiver user to access the metastore database.

**81.What is the purpose of archiving tables in Hive?**

Ans: The Hadoop archive property in the hive is to reduce the number of files present in the particular partition section**.** This recipe helps you archive file partitions for file count reduction in Hive The in-built support in Hive to convert files in existing partitions to Hadoop Archive (HAR) is one approach to reducing the number of files in sections, as the number of files in the filesystem directly affects the memory consumption in the Namenode. A partition can be archived using the “ARCHIVE” keyword in the “alter table” command. Once the command is issued, a MapReduce job will perform the archiving. Unlike Hive queries, there is no output on the CLI to indicate the process. The syntax for archiving is given below.

Syntax: Alter table table\_name ARCHIVE partition(“any\_partition”)

**82. What is DBPROPERTY in Hive?**

Ans. By default, the Hive database will be created inside the default warehouse directory i.e **/user/hive/warehouse**. But if we want we can store the database in some other HDFS location as well but mentioning the same in the location field. The DB properties are nothing but mentioning the details about the database created by the user. Suppose the name of the user, the type of the database and the tables it has, the date on which the database is created etc. This makes the other user easy the recognize the database and use it according to the requirement.

You can set key-value pairs in the DBPROPERTIES associated with a database using the ALTER DATABASE command. No other metadata about the database can be changed, including its name and directory location:

hive> ALTER DATABASE financials SET DBPROPERTIES ('edited-by' = 'Joe Dba');

There is no way to delete or “unset” a DBPROPERTY.

**83. Differentiate between local mode and MapReduce mode in Hive?**

Ans: The main difference between local mode and MapReduce mode in the hive is the way they execute. In Local mode, the pig script runs on a single machine without the need for a Hadoop cluster or hdfs. Local mode is used for development purposes to see how the script would behave in an actual environment.