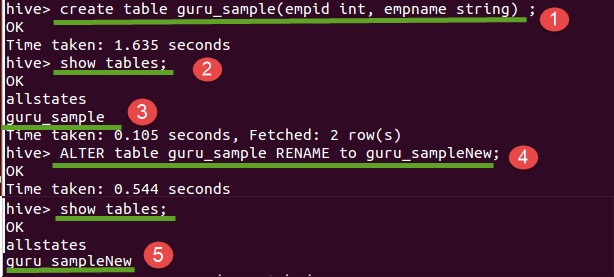
**HIVE Queries**



1. To create the internal table

Hive>CREATE TABLE guruhive\_internaltable (id INT,Name STRING);

Row format delimited

Fields terminated by '\t';

2. Load the data into internal table

Hive>LOAD DATA INPATH '/user/guru99hive/data.txt' INTO table guruhive\_internaltable;

3. Display the content of the table

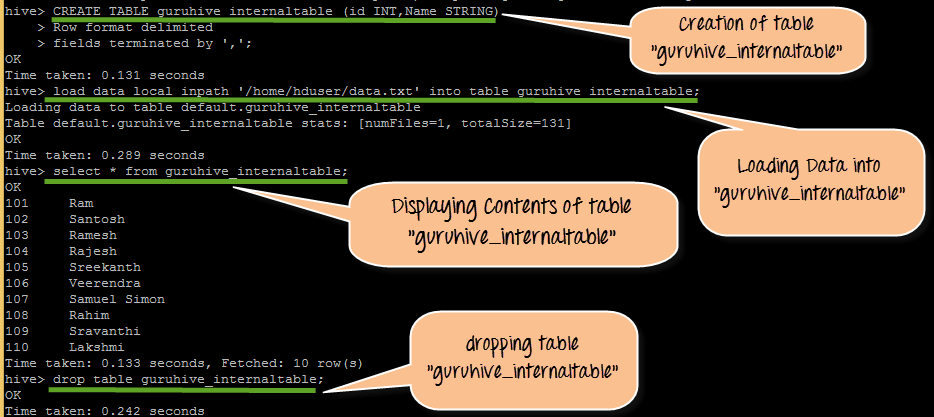
Hive>select \* from guruhive\_internaltable;

4. To drop the internal table

Hive>DROP TABLE guruhive\_internaltable;

If you dropped the guruhive\_internaltable, including its metadata and its data will be deleted from Hive.

From the following screenshot, we can observe the output



**Sample code Snippet for External Table**

1. Create External table

Hive>CREATE EXTERNAL TABLE guruhive\_external(id INT,Name STRING)

Row format delimited

Fields terminated by '\t'

LOCATION '/user/guru99hive/guruhive\_external;

2. If we are not specifying the location at the time of table creation, we can load the data manually

Hive>LOAD DATA INPATH '/user/guru99hive/data.txt' INTO TABLE guruhive\_external;

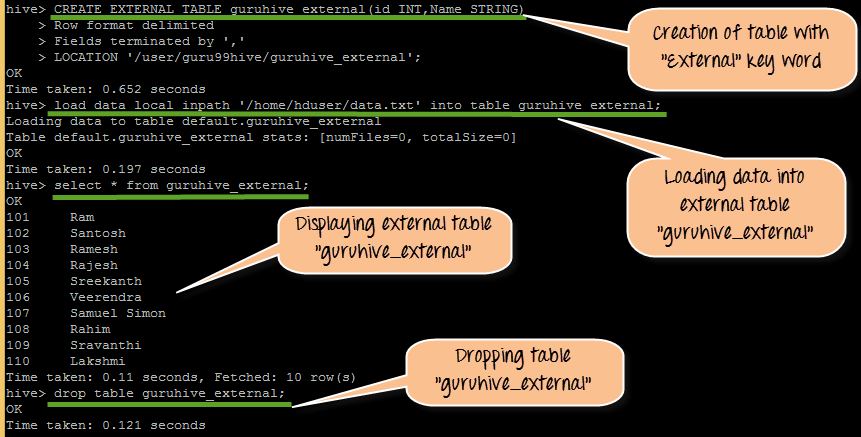
3. Display the content of the table

Hive>select \* from guruhive\_external;

4. To drop the internal table

Hive>DROP TABLE guruhive\_external;

From the following screenshot, we can observe the output



**Syntax:**

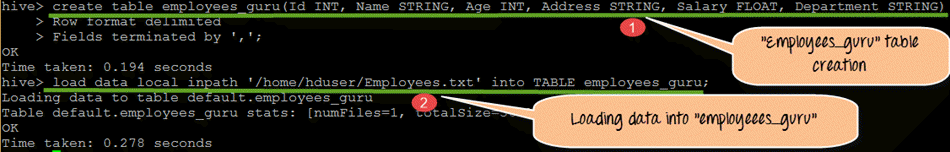
Create VIEW <VIEWNAME> AS SELECT

**Example:**

Hive>Create VIEW Sample\_ViewAS SELECT \* FROM employees WHERE salary>25000

In this example, we are creating view Sample\_View where it will display all the row values with salary field greater than 25000.

Here in this tutorial, we are going to create table “employees\_guru” with 6 columns.



From the above screen shot,

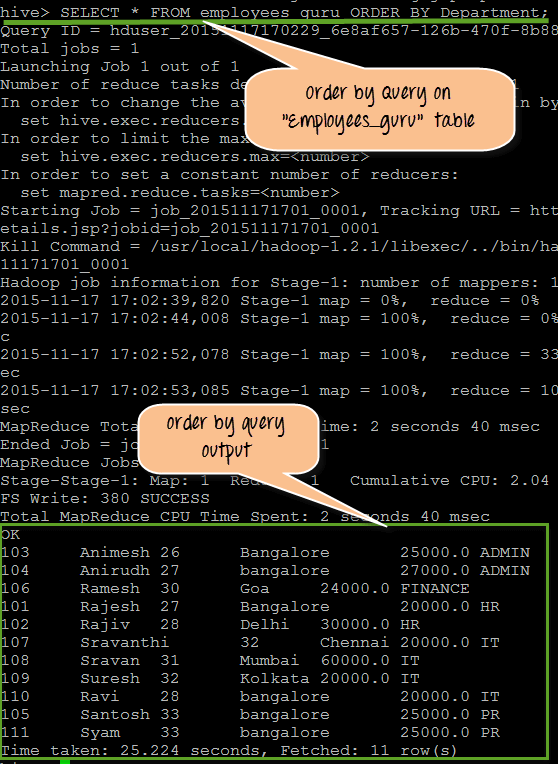
1. We are creating table “employees\_guru” with 6 column values such as Id, Name, Age, Address, Salary, Department, which belongs to the employees present in organization “guru.”
2. Here in this step we are loading data into employees\_guru table. The data that we are going to load will be placed under Employees.txt file

## Order by query:

The ORDER BY syntax in HiveQL is similar to the syntax of ORDER BY in SQL language.

Order by is the clause we use with “SELECT” statement in Hive queries, which helps sort data. Order by clause use columns on Hive tables for sorting particular column values mentioned with Order by. For whatever the column name we are defining the order by clause the query will selects and display results by ascending or descending order the particular column values.

If the mentioned order by field is a string, then it will display the result in lexicographical order. At the back end, it has to be passed on to a single reducer.



From the Above screen shot, we can observe the following

1. It is the query that performing on the “employees\_guru” table with the ORDER BY clause with Department as defined ORDER BY column name.”Department” is String so it will display results based on lexicographical order.
2. This is actual output for the query. If we observe it properly, we can see that it get results displayed based on Department column such as ADMIN, Finance and so on in orderQuery to be perform.

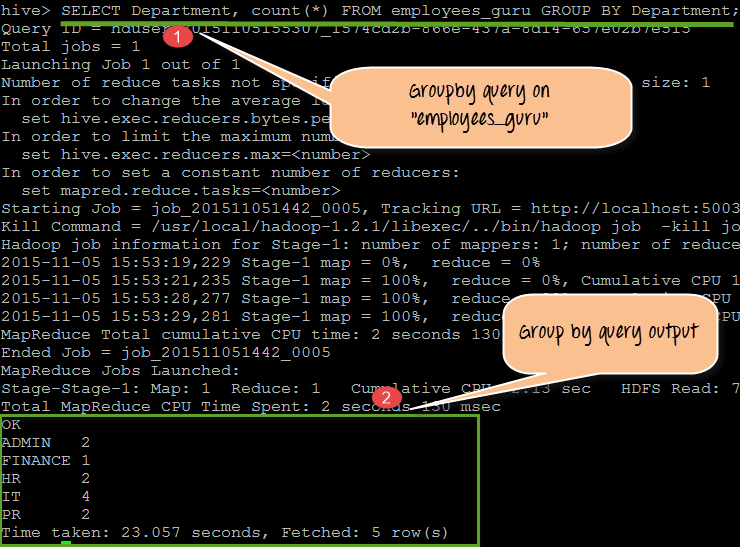
Query :

SELECT \* FROM employees\_guru ORDER BY Department;

## Group by query:

Group by clause use columns on Hive tables for grouping particular column values mentioned with the group by. For whatever the column name we are defining a “groupby” clause the query will selects and display results by grouping the particular column values.

For example, in the below screen shot it’s going to display the total count of employees present in each department. Here we have “Department” as Group by value.



From the above screenshot, we will observe the following

1. It is the query that is performed on the “employees\_guru” table with the GROUP BY clause with Department as defined GROUP BY column name.
2. The output showing here is the department name, and the employees count in different departments. Here all the employees belong to the specific department is grouped by and displayed in the results. So the result is department name with the total number of employees present in each department.

Query:

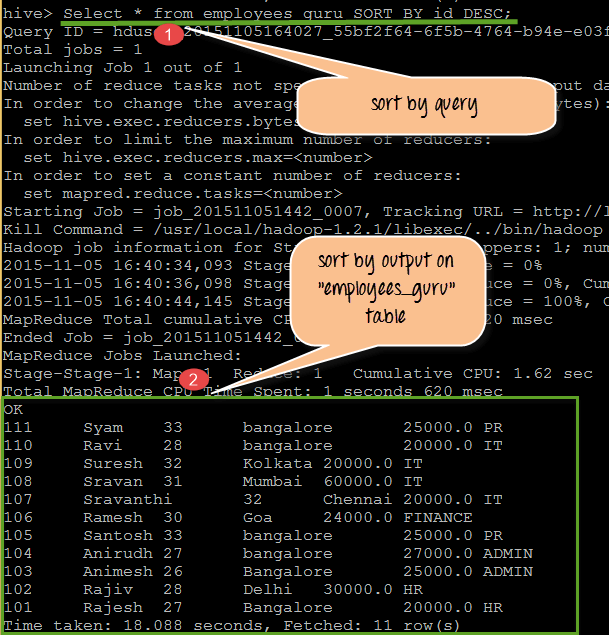
SELECT Department, count(\*) FROM employees\_guru GROUP BY Department;

## Sort by:

Sort by clause performs on column names of Hive tables to sort the output. We can mention DESC for sorting the order in descending order and mention ASC for Ascending order of the sort.

In this sort by it will sort the rows before feeding to the reducer. Always sort by depends on column types.

For instance, if column types are numeric it will sort in numeric order if the columns types are string it will sort in lexicographical order.



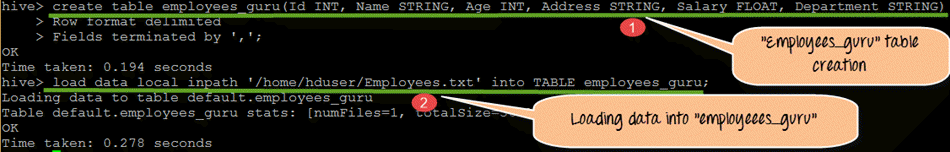
From the above screen shot we can observe the following:

1. It is the query that performing on the table “employees\_guru” with the SORT BY clause with “id” as define SORT BY column name. We used keyword DESC.
2. So the output displayed will be in descending order of “id”.

Query:

SELECT \* from employees\_guru SORT BY Id DESC;

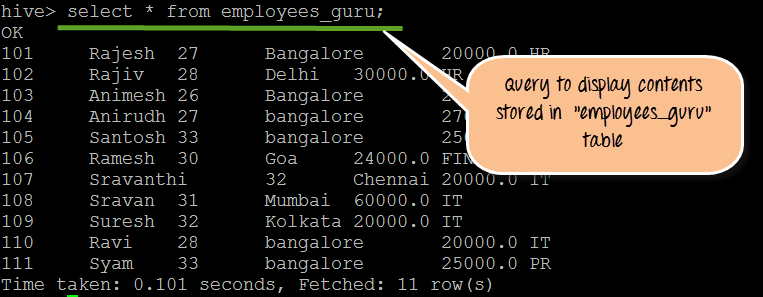
**Step 1)**In this step we are creating table “employees\_guru” with column names such as Id, Name, Age, Address, Salary and Department of the employees with data types.



From the above screenshot, we can observe the following,

1. Creation of table “employees\_guru”
2. Loading data from Employees.txt into table”employees\_guru”

**Step 2)**In this step we are displaying the contents stored in this table by using “Select” command. We can observe the table contents in the following screen shot.



Queries to be performed

1) Create table employees\_guru(Id INT, Name STRING, Age INT, Address STRING, Salary FLOAT, Department STRING)

> Row format delimited

> Fields terminated by ',';

2) load data local inpath '/home/hduser/Employees.txt' into TABLE employees\_guru;

3) select \* from employees\_guru;