**Practical 7A: Querying, Sorting, Aggregating data using HiveQL**

**What is HIVE:**

Hive is a data warehouse system which is used to analyse structured data. It is built on the top of Hadoop. It was developed by Facebook.

Hive provides the functionality of reading, writing, and managing large datasets residing in distributed storage. It runs SQL like queries called HQL (Hive query language) which gets internally converted to MapReduce jobs.

Using Hive, we can skip the requirement of the traditional approach of writing complex MapReduce programs. Hive supports Data Definition Language (DDL), Data Manipulation Language (DML), and User Defined Functions (UDF).

**Steps: Querying, Sorting, Aggregating data using HiveQL**

1.Open the Cloudera.



2.Open the terminal, Now we use hive command to enter the hive shell prompt and in hive shell we could execute all of the hive commands.



3. Now we will see the databases which are already existing using below command.

**Show databases;**

****

4.If we want to drop the database with the entire data (rows) then we will use below command. Here we don’t have any existing database rather than default so if for example I have ‘office’ as a database then will drop the database along with the data using command as,

**drop database office cascade;**

5. Creating a database name ‘RJC’ using below command.

**Create database RJC;**

****

6. So if we want to see whether this RJC database is created or not we will use below

command,

**show databases;**

****

7.Now we want to check whether we have any tables inside this rjc database or not. So first we will move to this database rjc using below command,

**Use rjc;**

Now we have moved inside this rjc database. Now we will check out which are the tables available using below command,

**show tables;**

8. Creating a database “rjc”.

**create database rjc;**

Now let’s move to this database using below command so now all the work we will do or perform it should be done within this rjc database.

**use rjc;**

****

9. Now we will create a table inside this rjc database named as employee using below

command,

**create table employee(ID int, name string, salary float, age int)**

After this we will not put semicolon , When we will be loading the data from some existing csv file or maybe some other text files so we have to mention that how that data has to be loaded here. We are simply creating the schema of the table with some certain fields or attributes along with their datatypes and then I’m mentioning

**➢ row format delimited**

**➢ fields terminated by ‘,’;**

****

row format delimited means , every record is present in one row andfields terminated by ‘,’ and fields are terminated by comma. So as soon as it encountersone comma so that means that one is the value of some field and after comma it isencountering abc so that abc is the value of some another field. By default it is a “tab” character that means fields are separated by “tab”.

10. So now we will see the schema of the table using below command,

**describe employee;**

It will give different fields of table employee along with their respective datatypes.



11. By default the internal table would store in the warehouse directory of hive. Whereas the external tables are available in the hdfs. And if we drop the internal table so then the table data and the metadata associated with that table will be deleted from the hdfs. Whereas when we drop the external table then only the metadata associated with that table will be deleted whereas the table data will be untouched by hive as it would be residing in the hdfs and it would be outside the warehouse directory of the hive.

So Now we will check how the table which we will be created is internal table or external table using below command,

**describe formatted employee;**

****

**By default hive createsInternal table or Managed Table.**

12. Now we will create the external table using below command,

**create external table emloyee2 (ID int, name string, salary float, age int)**

**➢ row format delimited**

**➢ fields terminated by ‘,’**

**➢ stored as textfile;**

****

13. Checking the schema of the table using below command,

**describe employee2;**

****

16. Creating a new external table named as employee3 in the specific location using below command,

**create external table emloyee3 (ID int, name string, salary float, age int)**

**➢ row format delimited**

**➢ fields terminated by ‘,’**

**➢ location '/user/cloudera/vj';**

It will first create 'vj' directory inside the /user/cloudera and then inside 'vj' the employee3 table get stored.



17. To see the schema of the employee3 table we use below command,

**describe employee3;**

****

19. Now move to terminal and listing out all the tables using below command;

**show tables;**

****

**20. ALTER COMMANDS**

Now we are changing the name of the **employee3** table to **emptable** using below command,



21. Now we will check whether the name of the **employee3** table changes to **emptable** or not using below command,

**show tables;**

****

22. First we will see the fields of emptable then we will add new column as **surname** in emptable using below command**,**

**describe emptable;**

**Alter table emptable add columns (surname string);**

**describe emptable;**

****

23.Now we will change field name of the emptable to first\_name using alter command,

**Alter table emptable change name first\_name string;**

**describe emptable;**

****

**Loading the data in the table**

24. Before loading the data in the table we will first create the csv file. Now open the new terminal ,

using ls command list out all the directories --> change the directory to document directory -->

use ls command to list all the files present inside the document folder or directory



25. Now creating new file as Student.csv using below command,

gedit Student.csv

As soon as we hit enter it will create a Student.csv file as it was not existing earlier.

26. Creating a new database as rjcstudent.

create database rjcstudent;

show databases;

Using rjcstudent database.

use rjcstudent;

28. Creating new table student inside rjcstudent database.

create table student (ID int, Name string, Age int)

➢ partitioned by(Course string)

➢ row format delimited

➢ fields terminated by ‘,’;



29. To see the structure or schema of the table,

describe student;



30. Loading data in the student table from Student.csv file which we have created in

document directory. Here we are partitioning based on course = ‘Hadoop’.

load data local inpath ‘/home/cloudera/Documents/Student.csv’ into table student

➢ partitioned by(Course string)

➢ row format delimited



It is partitioning based on Hadoop.

select \* from student;



31. Now we similarly partition for course = ML and course = Python.

32. Now go to browser refresh the page and select database as rjcstudent and click in

preview student table.





Drop: to the drop the entire table we can use drop table command.

Syntax: DROP table tablename;

****

****













Creating a new table as employee.

create table employee(ID int, Name string, Dept string, yoj int, salary float,

Country string)

➢ row format delimited

➢ fields terminated by ‘,’;

➢ tblproperties(“skip.header.line.count” =”1”);



Loading the data into employee table from employee2.csv file which we have created

and it is present in /home/cloudera/Documents directory.

load data local inpath ‘/home/cloudera/Documents/employee2.csv’ into table

empgroup

Displaying the table using below command,

select \* from employee;



















Now creating employee.csv file.

gedit employee.csv

Creating new database for performing querying operations.

Create database hiveql;

Using database hiveql and creating table employee inside the hiveql datanase.

create table employee(ID int, Name string, Department string, YOJ int, Salary float)

➢ row format delimited

➢ fields terminated by ‘,’;

➢ tblproperties(“skip.header.line.count” =”1”);



Loading the data into employee table from employee.csv file which we have created

earlier and it is present in /home/cloudera/Documents directory.

load data local inpath ‘/home/cloudera/Documents/employee.csv’ into table

employee

Displaying the table using below command,

select \* from employee;



Groupby clause

Now we display the total sum of salary of employees country wise using below

command,

select country, sum(salary) from empgroup group by country;



Groupby clause along with the having clause

Taking the total sum of salary countrywise using groupby clause and from that selecting

or displaying those country whose total sum of salary>50000 using having clause.

select country, sum(salary) from empgroup group by country having

sum(salary)>50000;



Instead of order by if we have sort by,

Select \* from employee sort by salary desc;

Now we can see the similar result as we got from order by and sort by so what is the

difference between the two is that it depends on number of reducers in order by we got

number of reducers is 1 and by using sort by here is also we got number of reducers is 1

so the difference between the two is that Order by will guarantee the total order in the

output whereas sort by will only guarantee the ordering of the rows within the reducer.

Order by gives us completely sorted result whereas sort by give us partially sorted result.