Big Data And Hadoop

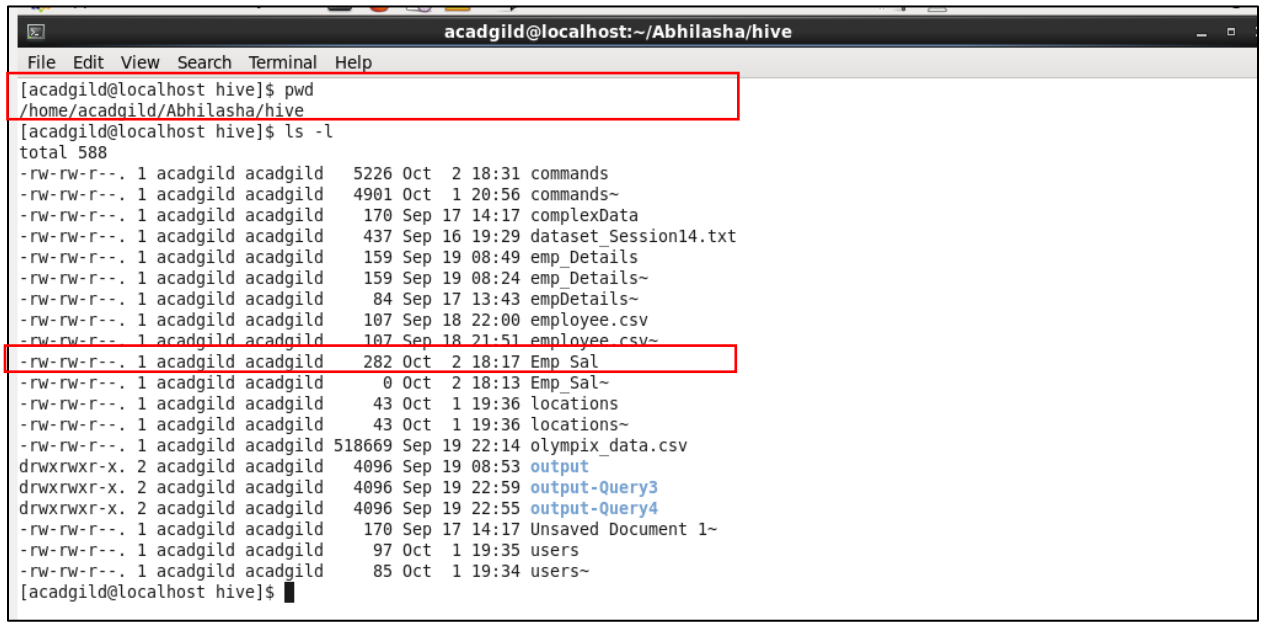
Session 16 - Assignment 1

Problem Statement 1:

Get a list of employees who receive a salary less than 100, compared to their immediate employee with higher salary in the same unit
Solution:

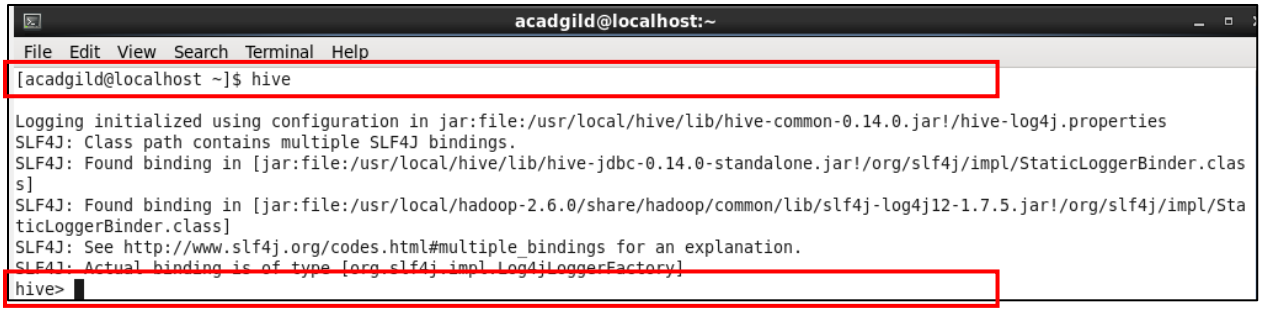
Input File

The input file is present on the local file system at **/home/acadgild/Abhilasha/hive** as follows:



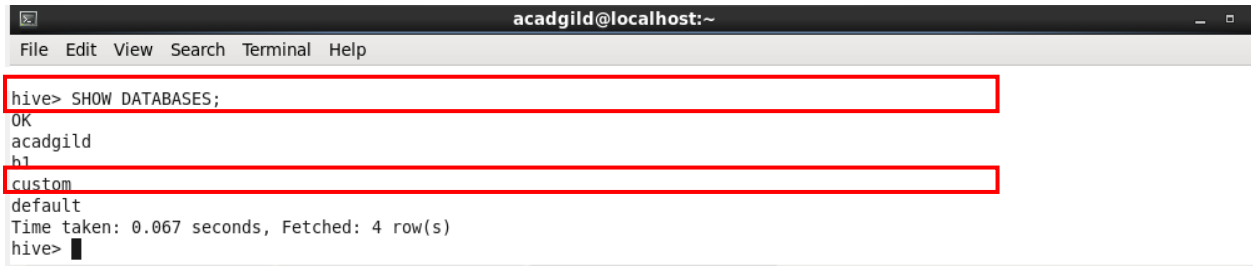
The content of the input file can be seen using the **cat** command as follows:

Start hive: We start the hive command line by executing the command hive as shown below:

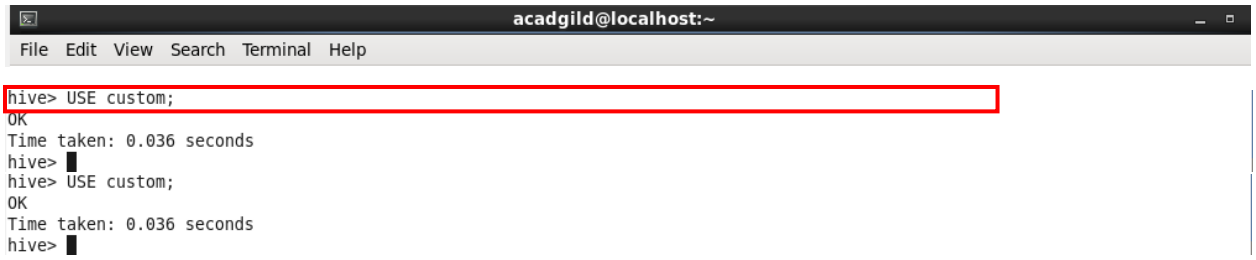


The above snapshot also shows that hive prompt has started. A pre-requisite to use hive is to start mysql server. This was done using the command sudo service mysqld start.

Step 1: We use **SHOW DATABASES** command to list the databases present. The database we will be using is **custom** as shown below:

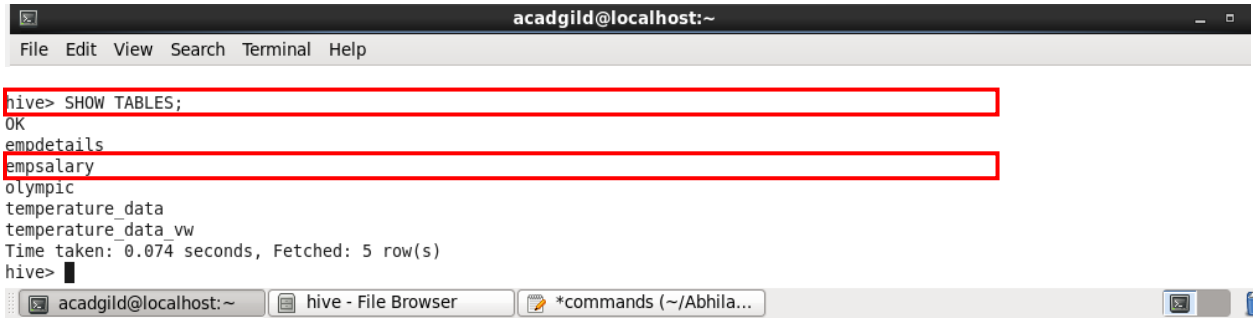


Step 2: We use **USE custom** command to make use of custom database, as shown below:

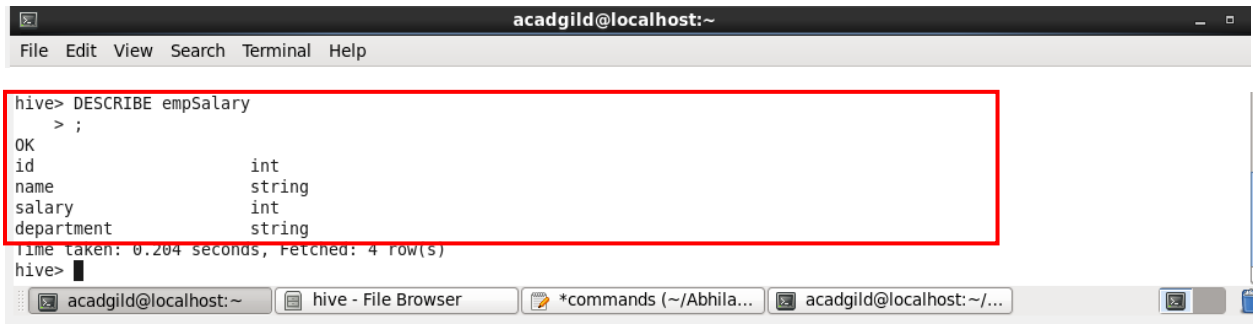


Step 3: We create the table using **CREATE TABLE** command. The fields of the table are: id, name, salary, and department.

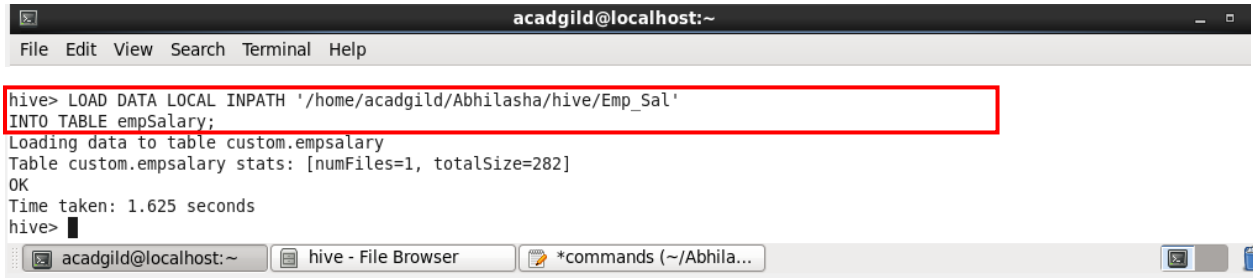
Step 4: **SHOW TABLES** command will help us verify that the table is created.



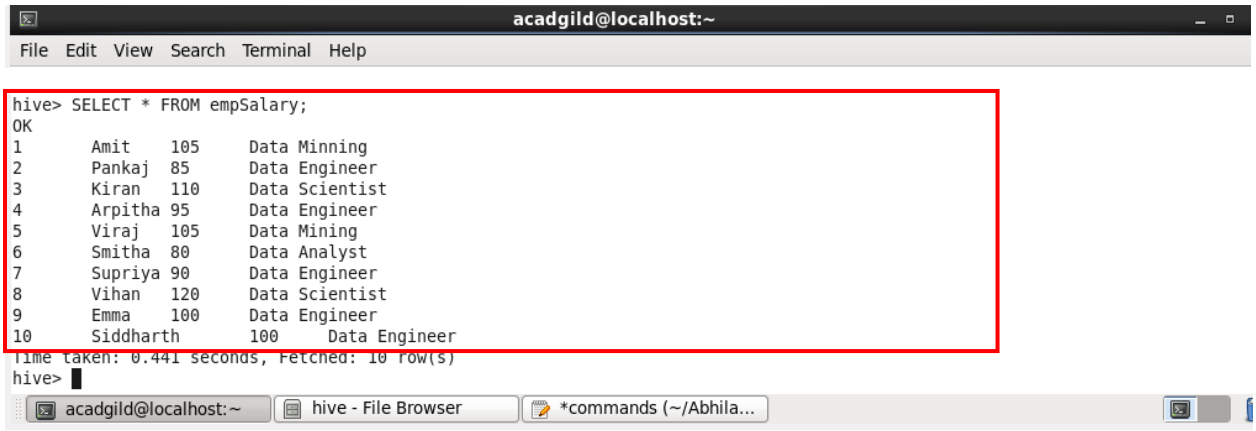
Step 5: **DESCRIBE** command will help us verify the schema of the table as follows:



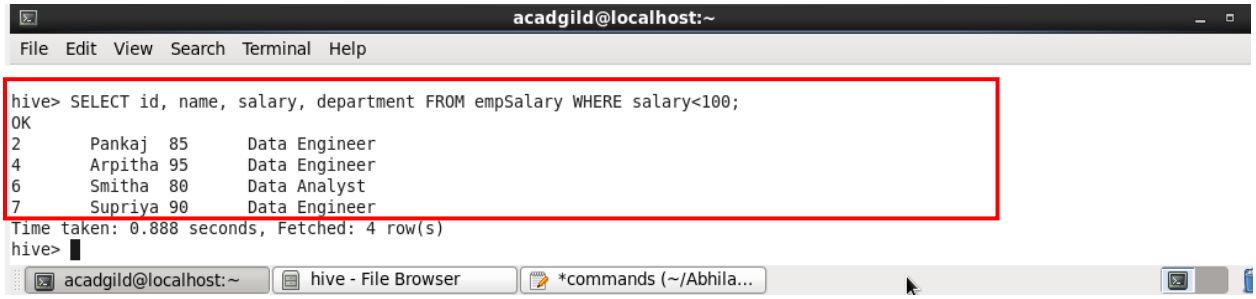
Step 6: Next is to load the data from input file, which is located at **/home/acadgild/Abhilasha/hive** as follows. We use the **LOAD** command and use the keyword **LOCAL** to specify that the file is present in the local file system and not HDFS.



Step 7: Using the **SELECT *** query, we can display the complete data as follows:

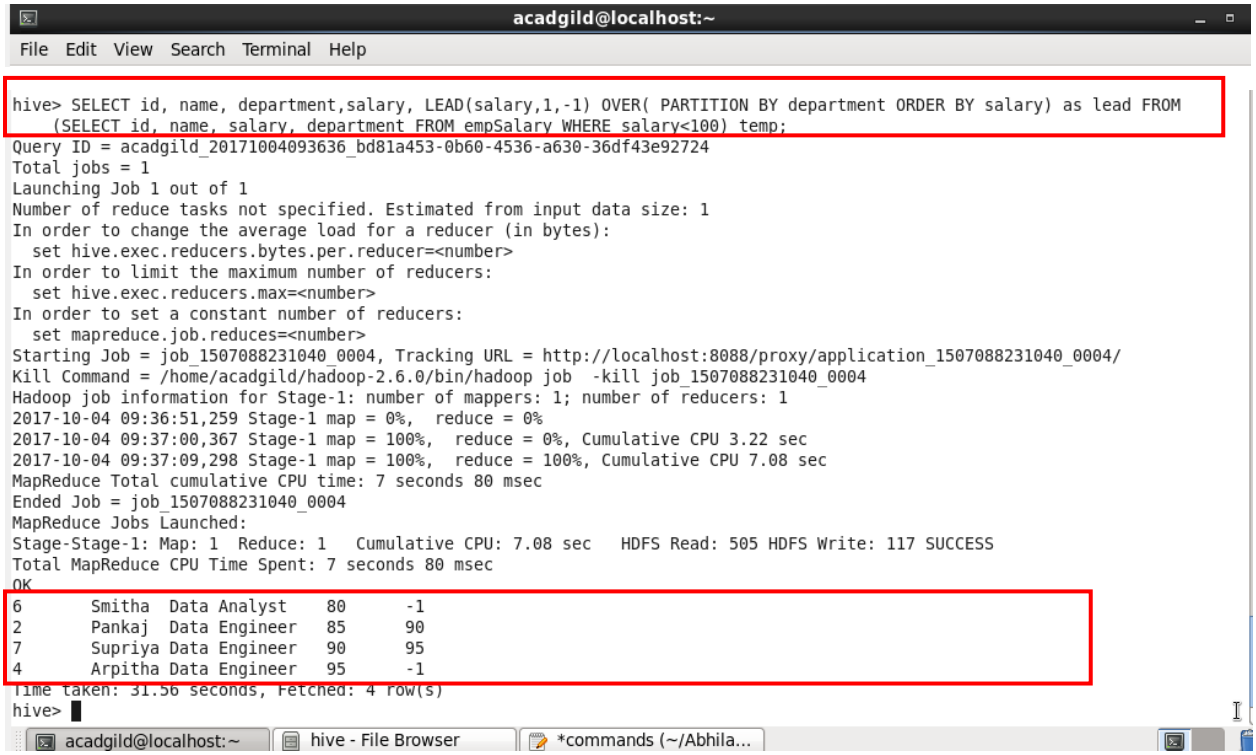


Step 8: We need only those employees that have salary less than 100. Hence, we apply a predicate **salary<100** as follows:



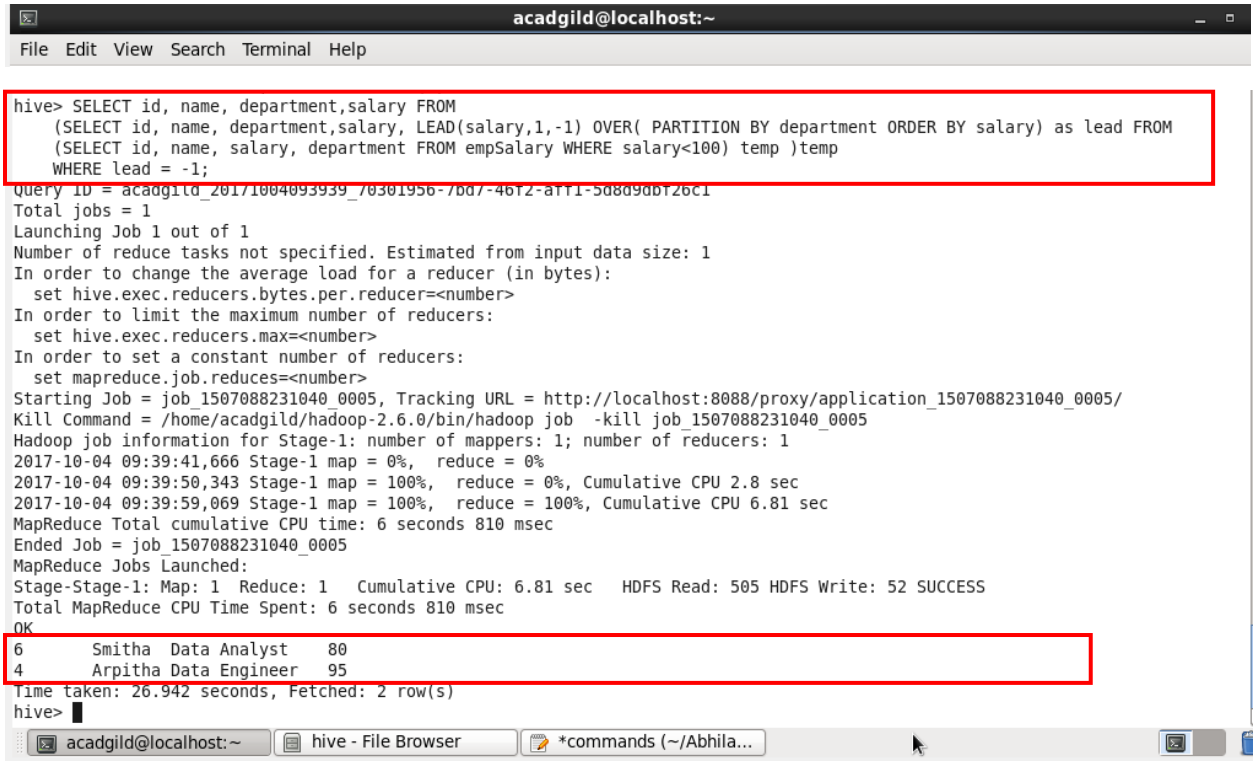
The screenshot also mentions the output of the query executed.

Step 9: We add a use another query on top of the query in step 8, to make use of **LEAD** function. LEAD function will give us the immediate higher salary. If the employee itself has the highest salary in the department, it has no other lead and hence mentioning the default value to be -1. We need this data per department and hence, using the clause **PARTITION BY department** as follows:



The screenshot also mentions the output of the query executed.

Step 10: Now we need only those employees that have highest salary in the department. This can be achieved by using another query on top of the query in step 9. The rows with lead column having value -1 will have highest salary in respective departments. So the final query is as follows:



The above screen shot mentions the final output.

Problem Statement 2:

List of all employees who draw higher salary than the average salary of that department
Solution:

In the problem statement 1, we have mentioned of the input file. Same file is being used for this problem statement as well.

Steps 1 to 7 are applicable for this problem statement as well, hence, not repeating them again.

The first part of the solution is to find the average salary per department. If we use GROUP BY and aggregate function AVG, we will get one row per department. However, we need average salary on every row, for every employee. Hence, we use analytic and windowing in hive.

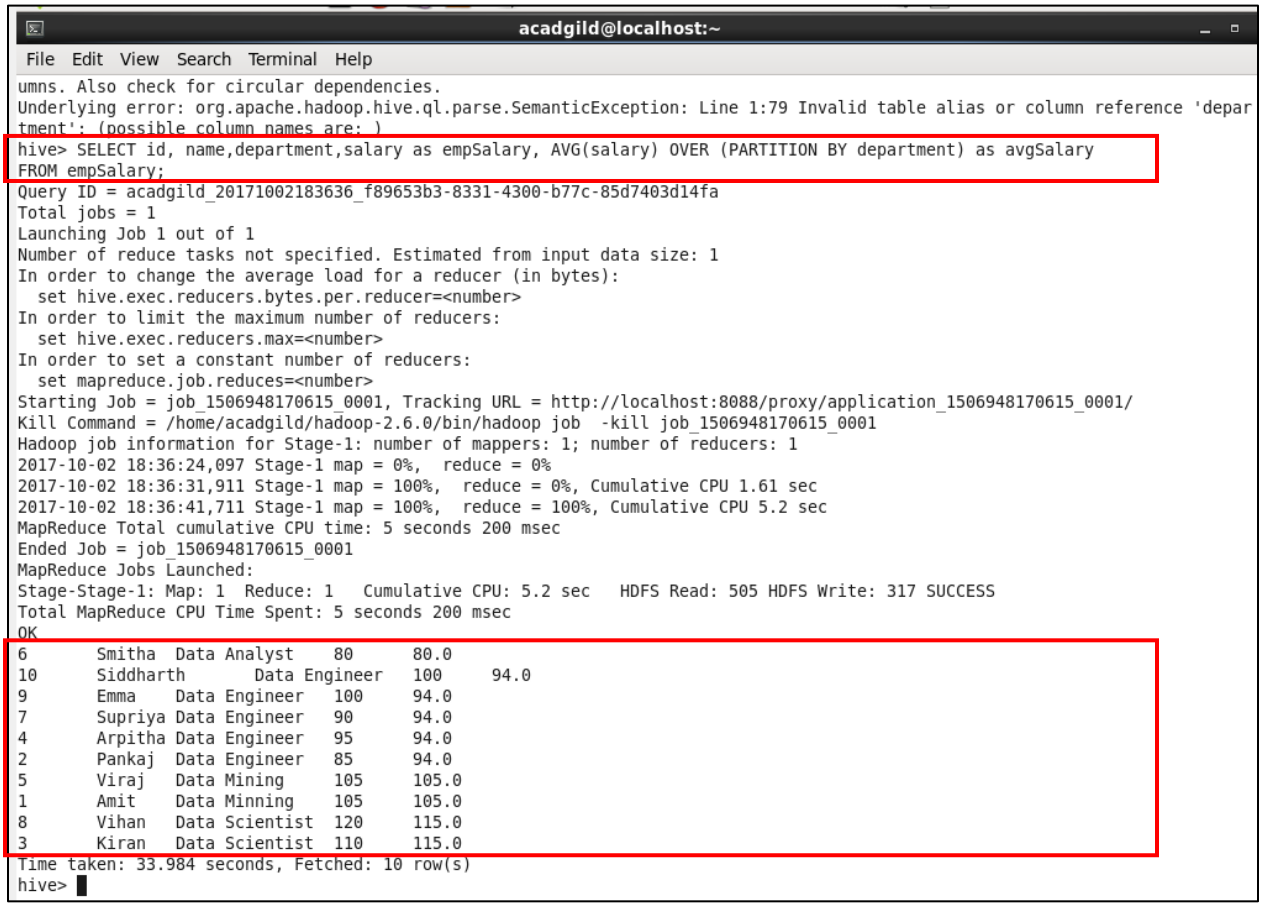
The query used to achieve this is

Select id, name, department, salary as empSalary, AVG (salary) OVER (PARTITION BY department) as avgSalary FROM empSalary;

Here, OVER clause is used to produce results per row, based on the computations of the values of the rows in a particular window. We decide the window based on the department. This is similar to

grouping the rows by department. Then we have made use of the aggregate function AVG to get average salary per department.

The execution of this command and its result are as follows:



Now, we need list of those employees whose salary is greater than average salary of the department. For this, we will use a nested query and apply the predicate on top of the above query as follows:

To the output of above query, we have applied the predicate empSalary > avgSalary as follows

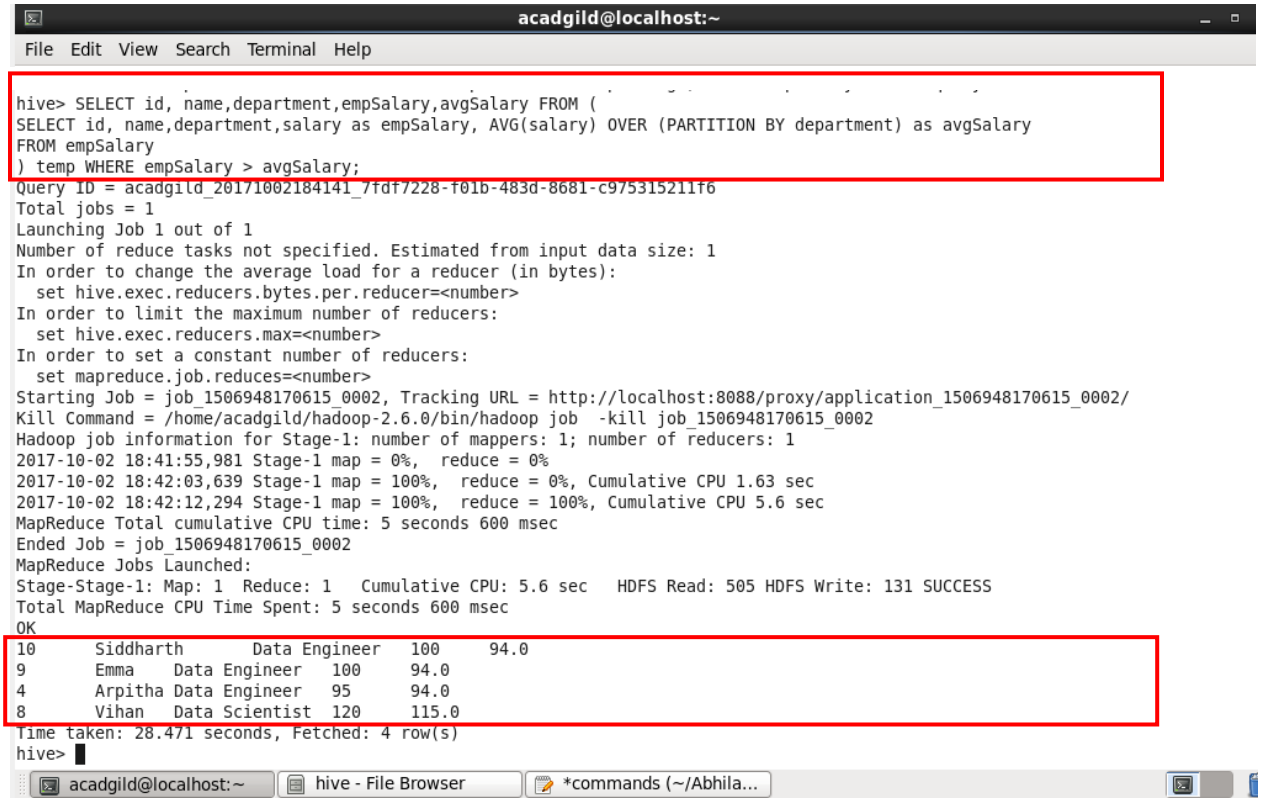
Select id, name, department, empSalary, avgSalary FROM

(

Select id, name, department, salary as empSalary, AVG (salary) OVER (PARTITION BY department) as avgSalary FROM empSalary

) temp

where empSalary>avgSalary;



The output shows the result containing the employees with their salary greater than average salary. The rows of the output are id, name, department, salary of employee and average salary of the department.