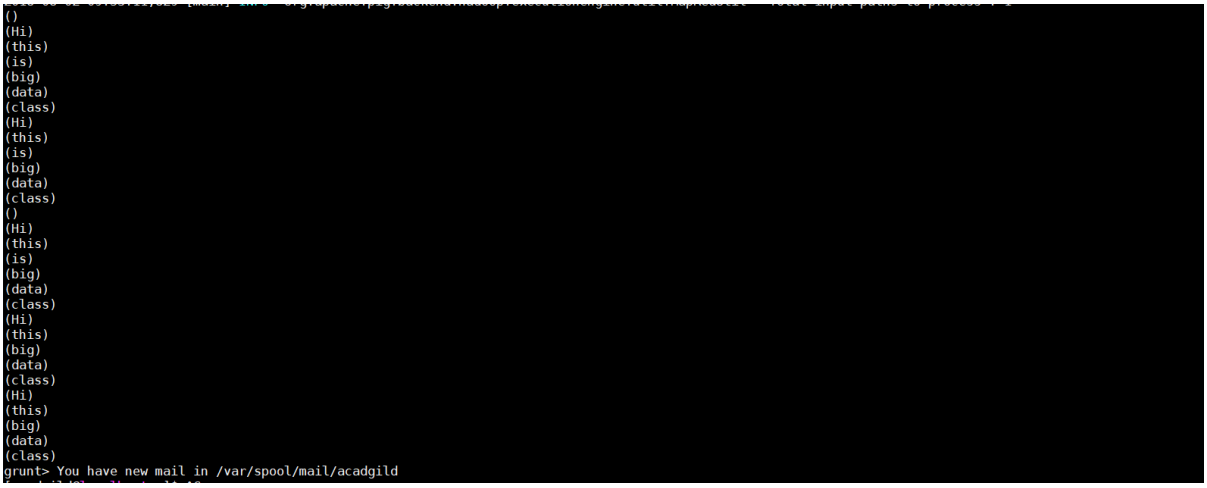
Task1:

A = load '/hadoopdata/pig/test.txt/' using PigStorage as (line:chararray);

grunt> B = FOREACH A GENERATE flatten(TOKENIZE((chararray)$0)) as word;

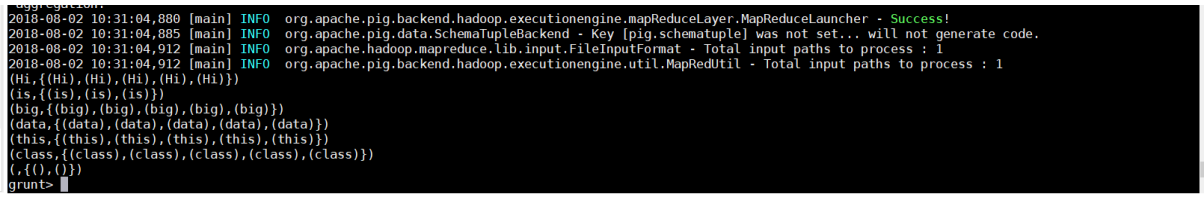
grunt> dump B;

Output of B

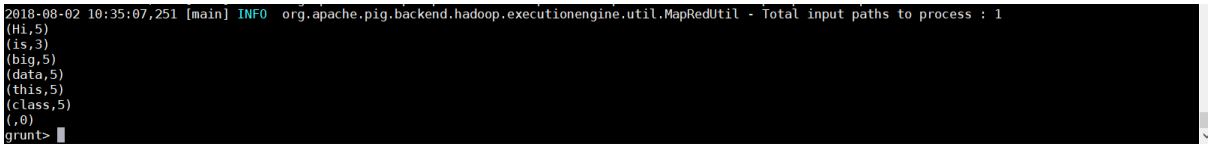


C = group B by word;

Dump C;



D = foreach C generate group, COUNT(B);



Task2: We have employee_details and employee_expenses files. Use local mode while running Pig and write Pig Latin script to get below results:

employee_details (EmpID,Name,Salary,EmployeeRating)

emp = Load '/hadoopdata/pig/employee_details.txt/' using PigStorage(',') AS (emp_id :int, emp_name:chararray,emp_salary:int, emp_rating:int);

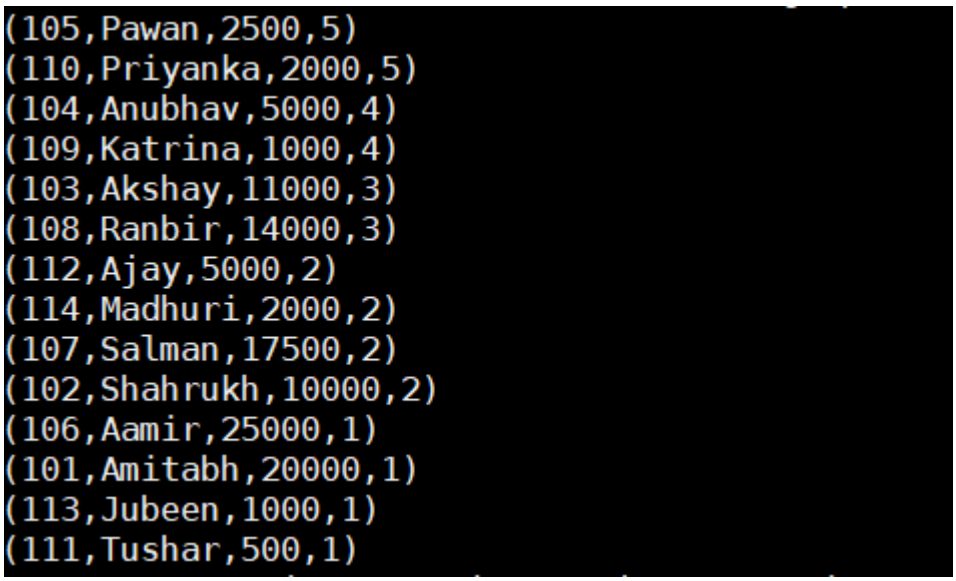
dump emp;

employee_expenses(EmpID,Expense)

emp_expenses = LOAD '/hadoopdata/pig/employee_expense.txt/' AS (emp_id: int, expenses: int);

(a) Top 5 employees (employee id and employee name) with highest rating. (In case two employees have same rating, employee with name coming first in dictionary should get preference)

emp_rating1 = order emp by emp_rating DESC, emp_name ASC;

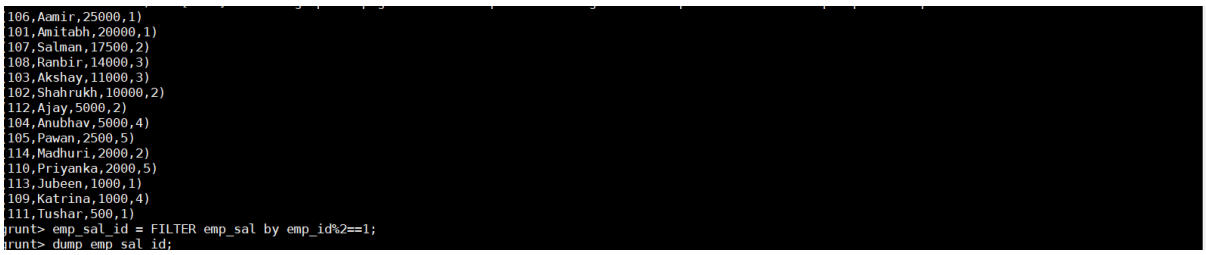


emp_rating_limit = LIMIT emp_rating1 5;

Dump emp_rating _limit;

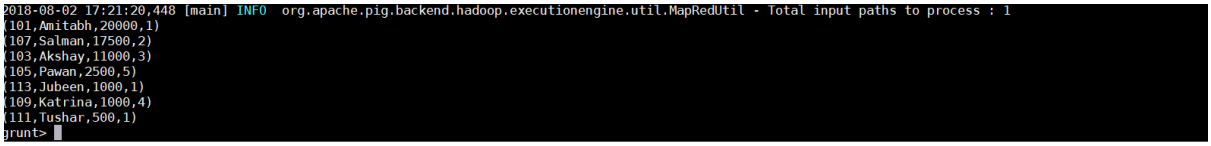
Top 3 employees (employee id and employee name) with highest salary, whose employee id is an odd number. (In case two employees have same salary, employee with name coming first in dictionary should get preference)

emp_sal = ORDER emp by emp_salary DESC, emp_name ASC;



emp_sal_id = FILTER emp_sal by emp_id%2==1;

dump emp_sal_id;



emp_final_limit = LIMIT emp_sal_id 3;

grunt> dump emp_final_id ;

(c) Employee (employee id and employee name) with maximum expense (In case two employees have same expense, employee with name coming first in dictionary should get preference)

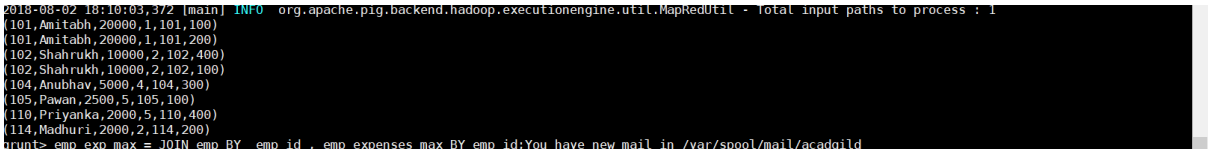
emp = Load '/hadoopdata/pig/employee_details.txt/' using PigStorage(',') AS (emp_id :int, emp_name:chararray,emp_salary:int, emp_rating:int);

empexpenses = LOAD '/hadoopdata/pig/employee_expense.txt/' AS (emp_id :int, expense_emp :int);

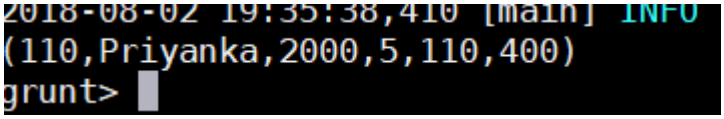
emp_exp = JOIN emp BY emp_id , empexpenses BY emp_id;

grunt> maxexpense = ORDER emp_exp BY empexpenses::expense_emp DESC;

grunt> dump maxexpense ;

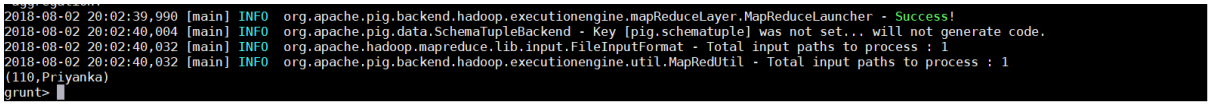



```
Limitmaxepnse = LIMIT maxexpense 1;
```




```
Limitmaxexpensefinal= FOREACH Limitmaxexpense generate emp:emp_id, emp:emp_name;
```

Dump Limitmaxexpensefinal;



d) (d) List of employees (employee id and employee name) having entries in employee_expenses file.

grunt> A = Load '/hadoopdata/pig/employee_details.txt/' using PigStorage(',') AS (emp_id :int, emp_name:chararray,emp_salary:int, emp_rating:int);

B = LOAD '/hadoopdata/pig/employee_expense.txt/' AS (emp_id :int, expense_emp :int);

emp_expenses = JOIN A BY emp_id, B BY emp_id;

emp_expense_data = FOREACH emp_expenses GENERATE emp::emp_id,emp:: emp_name;

```
grunt> emp_expenses = JOIN A BY emp_id, B BY emp_id;
grunt> emp_expense_data = FOREACH emp_expenses GENERATE A::emp_id, A::emp_name;
grunt> emp_exp_distinct_data = DISTINCT emp_expense_data;
grunt> dump emp_exp_distinct_data;
```

```
(101,Amitabh)
(102,Shahrukh)
(104,Anubhav)
(105,Pawan)
(110,Priyanka)
(114,Madhuri)
grunt>
```

(e) List of employees (employee id and employee name) having no entry in employee_expenses file.

A = Load '/hadoopdata/pig/employee_details.txt/' using PigStorage(',') AS (emp_id :int, emp_name:chararray,emp_salary:int, emp_rating:int);

B = LOAD '/hadoopdata/pig/employee_expense.txt/' AS (emp_id :int, expense_emp :int);

```
emp_without_exp = JOIN A BY emp_id LEFT OUTER, B BY emp_id;

emp_without_exp_filter = FILTER emp_without_exp BY B::emp_id is null;
emp_without_exp_filter_data = FOREACH emp_without_exp_filter  GENERATE  A::emp_id, A::emp_name;

dump emp_without_exp_filter_data;
```

```
2018-08-02 21:04:51,987 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)
grunt>
```