Session 5: Exploring Pig

Assignment 5.1

We have employee\_details and employee\_expenses files. Use local mode while running Pig and

write Pig Latin script to get below results:

<https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_details.txt>

<https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_expeses.txt>

1. 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)

Solution:-

grunt> emp\_det = LOAD 'employee\_details.txt' USING PigStorage(',') as (emp\_id:int,emp\_name:chararray,salary:int,rating:int);

grunt> emp\_exp = LOAD 'employee\_expenses.txt' as (emp\_id:int,exp:int);

grunt> emp\_det\_sorted = ORDER emp\_det by rating desc, emp\_name;

dump emp\_det\_sorted;



grunt> five\_top\_emp = limit emp\_det\_sorted 5;

grunt> result= foreach five\_top\_emp generate emp\_id, emp\_name;

grunt> dump result;



1. 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)

Solution:-

grunt> emp\_det = LOAD 'employee\_details.txt' USING PigStorage(',') as (emp\_id:int,emp\_name:chararray,salary:int,rating:int);

grunt> emp\_exp = LOAD 'employee\_expenses.txt' as (emp\_id:int,exp:int);

grunt> emp\_det\_odd = filter emp\_det by emp\_id%2 != 0;

grunt> emp\_det\_odd\_sorted = ORDER emp\_det\_odd by salary desc, emp\_name;

grunt> dump emp\_det\_odd\_sorted;



grunt> three\_top\_odd\_emp = limit emp\_det\_odd\_sorted 3;

grunt> result= foreach three\_top\_odd\_emp generate emp\_id, emp\_name;

grunt> dump result;



1. 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)

Solution:-

grunt> emp\_det = LOAD 'employee\_details.txt' USING PigStorage(',') as (emp\_id:int,emp\_name:chararray,salary:int,rating:int);

grunt> emp\_exp = LOAD 'employee\_expenses.txt' as (emp\_id:int,exp:int);

grunt> emp\_joined = join emp\_det by emp\_id inner, emp\_exp by emp\_id inner;

grunt> emp\_joined\_selected = foreach emp\_joined generate emp\_det::emp\_id as emp\_id, emp\_name as emp\_name, exp as exp;

grunt> emp\_joined\_sorted = ORDER emp\_joined\_selected by exp desc, emp\_name;

grunt> emp\_group = GROUP emp\_joined\_sorted by (emp\_id, emp\_name);

grunt> emp\_exp\_sum = foreach emp\_group generate flatten(group), MAX(emp\_joined\_sorted.exp) as exp;

grunt> emp\_exp\_sorted = ORDER emp\_exp\_sum by exp desc, emp\_name;

grunt> dump emp\_exp\_sorted;



1. List of employees (employee id and employee name) having entries in employee\_expenses file.

Solution:-

grunt> emp\_det = LOAD 'employee\_details.txt' USING PigStorage(',') as (emp\_id:int,emp\_name:chararray,salary:int,rating:int);

grunt> emp\_exp = LOAD 'employee\_expenses.txt' as (emp\_id:int,exp:int);

grunt> emp\_joined = join emp\_det by emp\_id outer, emp\_exp by emp\_id ;

grunt> emp\_grp = cogroup emp\_det by emp\_id , emp\_exp by emp\_id;

grunt> emp\_filtered = filter emp\_grp by not IsEmpty(emp\_exp) and not IsEmpty(emp\_det);

grunt> result = foreach emp\_filtered generate group, flatten(emp\_det.emp\_name);

grunt> dump result;



1. List of employees (employee id and employee name) having no entry in employee\_expenses file.

Solution:-

grunt> emp\_det = LOAD 'employee\_details.txt' USING PigStorage(',') as (emp\_id:int,emp\_name:chararray,salary:int,rating:int);

grunt> emp\_exp = LOAD 'employee\_expenses.txt' as (emp\_id:int,exp:int);

grunt> emp\_joined = join emp\_det by emp\_id outer, emp\_exp by emp\_id ;

grunt> emp\_grp = cogroup emp\_det by emp\_id , emp\_exp by emp\_id;

grunt> emp\_filtered = filter emp\_grp by IsEmpty(emp\_exp) and not IsEmpty(emp\_det);

grunt> result = foreach emp\_filtered generate group, flatten(emp\_det.emp\_name);

grunt> dump result;

