Bigdata Assignment 2.4

Entered into pig shell – **pig -x local**

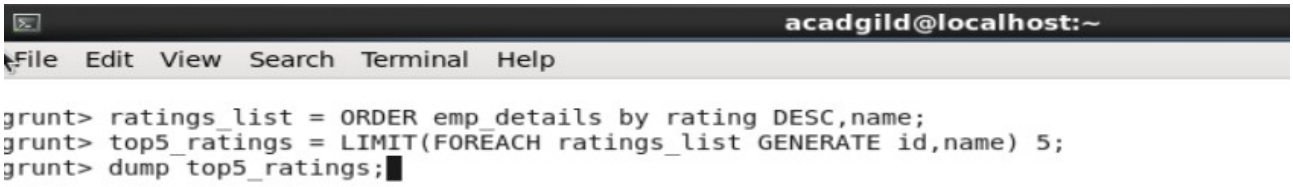
Loaded the text file into a relation – **emp_details = LOAD '/home/acadgild/INPUT/PIG/employee_details.txt' using PigStorage(',') as (id:int , name:chararray , salary:int , rating:int);**
emp_expenses = LOAD '/home/acadgild/INPUT/PIG/employee_expenses.txt' as (id:int , expenses:int);

(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)

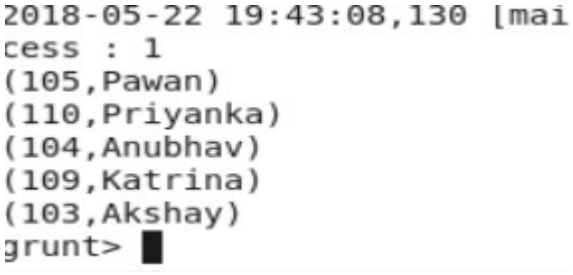
Ans) We first ordered the by rating in descending order then by name in ascending order.Then selected top 5 employees based on the given sorting.

ratings_list = ORDER emp_details by rating DESC , name;

top5_ratings = LIMIT(FOREACH ratings_list GENERATE id,name) 5;
dump top5_ratings;



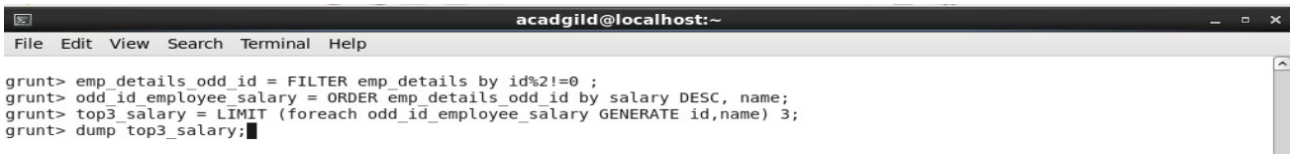
OUTPUT- As we can see in the below screenshot of the output we got the top 5 employee with topratings.



(b) 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)

Ans) Filtered out the odd id employees , then sorted them by salary in descending order and then by name in ascending order. Thwn selected top 3 employees of the list.

emp_details_odd_id = FILTER emp_details by id%2!=0 ;
odd_id_employee_salary = ORDER emp_details_odd_id by salary DESC , name;
top3_salary = LIMIT (foreach odd_id_empolyee_salary GENERATE id , name) 3;
dump top3_salary;



OUTPUT -

We got the employee id and its name with highest salary whose id is odd no.

```
2018-05-22 13:21:27,743 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(101,Amitabh)
(107,Salman)
(103,Akshay)
grunt>
```

(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)

Ans - We made a inner join table of emp_details and emp_expenses.Then grouped them by id. Then made a list of each employee's id , name and its total expenses. Then we flattened the data and made a distinct list of it. Selected the top expense by sorting it in descending order.

emp_join = JOIN emp_details by id , emp_expenses by id;
emp_join = group emp_join by emp_details::id;
emp_list = foreach emp_join generate emp_join.emp_details::id as id , emp_join.emp_details::name as name , SUM)emp_join.emp_expenses::expenses) as total_expenses;
emp_list = foreach emp_list generate flatten(id) , flatten(total_expenses);
emp_list = distinct emp_list;
dump emp_list;
top_expense = LIMIT (ORDER emp_list by total_expenses DESC) 1;
dump top_expense;


```
grunt> top_expense = LIMIT (ORDER emp_list by total_expenses DESC) 1;
grunt> dump top_expense;
```

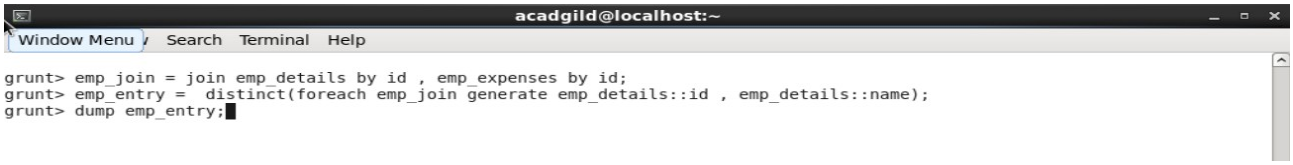
OUTPUT – Got the highest expense employee id and name.

```
cess : 1
(102,Shahrukh,500)
grunt>
```

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

Ans- Made an inner join by id of the details and expense tables. Then made an disticnt list of it each employee name and it's id.

emp_join = join emp_details by id;
emp_entry = distinct (foreach emp_join generate emp_details::id , emp_details::name);
dump emp_entry;

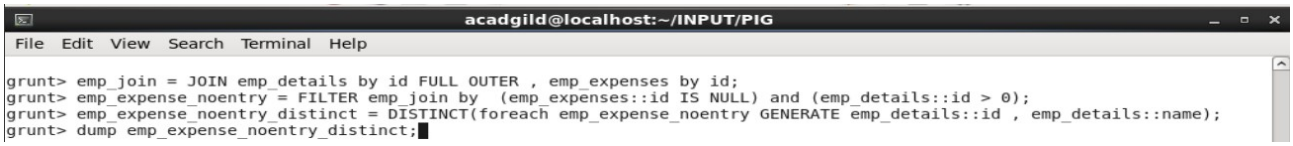


OUTPUT - The list of the employees having entry in expenses file.

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

Ans - Made a full outer join by id of tables emp_details and emP_expenses. Then filtered out the tuples which have id as NULL in emp_expenses. Then made a distinct list of it having employee name and its id.

emp_join = JOIN emp_details by id FULL OUTER, emp_expenses by id;
emp_expense_noentry = FILTER emp_join by (emp_expenses::id IS NULL) and (emp_details::id > 0);
emp_expense_noentry_distinct = DISTINCT(foreach emp_expense_noentry GENERATE emp_details::id, emp_details::name);
dump emp_expense_noentery_distinct;



OUTPUT - It shows the details of employee having deyails in emp-details.txt but not in emp_expenses.txt.

```
cess : 1
(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)
grunt>
```