Assignment 2.4

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_expenses.txt

Loading data to use in pig.

```
[acadgild@localhost ~]$ pig -x LOCAL
```

```
grunt> employee_details = LOAD '/home/acadgild/employee_details.txt' USING PigStorage(',') as (id:int, name:chararray
, salary:int, rating:int);
```

```
grunt> employee_expenses = LOAD '/home/acadgild/employee_expenses.txt' as (id:int, expense: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)

We will order the employees based on rating and then name.

```
grunt> rating_order = ORDER employee_details BY rating,name;
grunt> dump rating_order;
```

```
(106,Aamir,25000,1)
(101,Amitabh,20000,1)
(113,Jubeen,1000,1)
(111,Tushar,500,1)
(112,Ajay,5000,2)
(114,Madhuri,2000,2)
(107,Salman,17500,2)
(102,Shahrukh,10000,2)
(103,Akshay,11000,3)
(108,Ranbir,14000,3)
(104,Anubhav,5000,4)
(109,Katrina,1000,4)
(105,Pawan,2500,5)
(110,Priyanka,2000,5)
```

```
grunt> top_5_rating = LIMIT (foreach rating_order GENERATE id,name) 5;
grunt> dump top_5_rating
```

OUTPUT:

```
(106,Aamir)
(101,Amitabh)
(113,Jubeen)
(111,Tushar)
(112,Ajay)
grunt>
```

(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)

Filtering Odd ID employees.
Ordering them by salary in decreasing order, also to keep name preference name has been ordered.

```
grunt> employee_odd_id = FILTER employee_details BY id%2 != 0;
grunt> odd_employee_salary_order = ORDER (ORDER employee_odd_id BY name) BY salary DESC;
```

```
grunt> top_3_salary = LIMIT (foreach odd_employee_salary_order GENERATE id,name) 3;
grunt> dump top_3_salary;
```

OUTPUT:

```
(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)

Joining both tables

```
grunt> employee_join = JOIN employee_details by id FULL OUTER, employee_expenses by id;
```

Grouping by ID since there are repetitions.

```
grunt> employee_join_group = GROUP employee_join BY employee_details::id;
```

Filtering data, Summing up total expense and cleaning up list by removing duplicate.

```
grunt> employee_total_expenses = foreach employee_join_group GENERATE employee_join.employee_details::id as id, emplo
yee_join.employee_details::name as name, SUM(employee_join.employee_expenses::expense) as total_expense;
```

```
grunt> expense_list = foreach employee_total_expenses GENERATE FLATTEN(id),FLATTEN(name),FLATTEN(total_expense);
```

```
grunt> expense_list = DISTINCT expense_list;
grunt> dump expense_list;
```

```
grunt> top_expense = LIMIT (ORDER expense_list BY total_expense DESC) 1;
grunt> dump top_expense;
```

OUTPUT:

```
(102,Shahrukh,500)
```

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

Reusing the joined table.
Filtering out the entries will null id from expense list.

```
grunt> employee_expense_entry = FILTER employee_join BY (employee_expenses::id > 0) AND (employee_details::id > 0);
```

```
grunt> employee_expense_entry_names = DISTINCT(foreach employee_expense_entry GENERATE employee_details::id,employee_
details::name);
grunt> dump employee_expense_entry_names
```

OUTPUT:

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

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

Checking for null entries in expense columns from joined table

```
grunt> employee_expense_no_entry = FILTER employee_join BY (employee_expenses::id IS NULL) AND (employee_details::id
> 0);
```

```
grunt> dump employee_expense_no_entry;
```

OUTPUT:

```
(103,Akshay,11000,3,,)
(106,Aamir,25000,1,,)
(107,Salman,17500,2,,)
(108,Ranbir,14000,3,,)
(109,Katrina,1000,4,,)
(111,Tushar,500,1,,)
(112,Ajay,5000,2,,)
(113,Jubeen,1000,1,,)
```