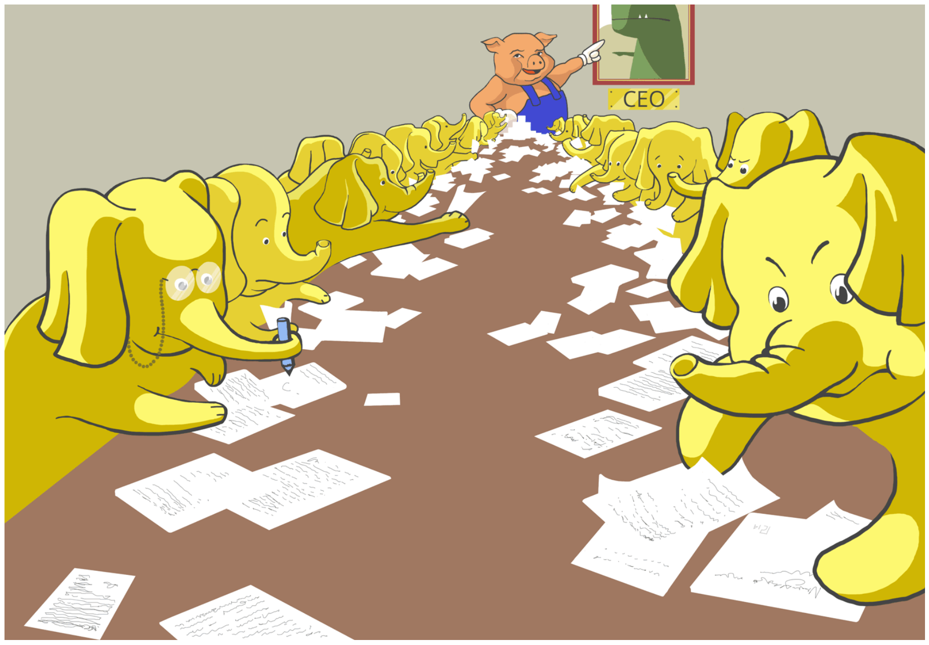


PIG USE CASE: COGNIZANT EMPLOYEE DATA ANALYSIS

BIG DATA HADOOP & SPARK TRAINING

ACADGILD

ASSIGNMENT 5.1

BY :-

SAHIL KHURANA

Problem Statement

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

Associated Data Files

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.txt

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

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

employee_expenses.txt

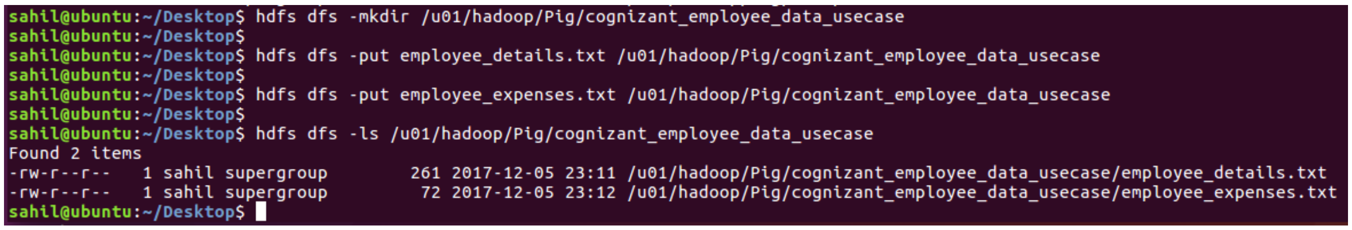
https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_expenses.txt

```
101 200
102 100
```

110 400
114 200
119 200
105 100
101 100
104 300
102 400

Note: - To solve the Assignment, I have created a VM with Ubuntu 16.04 OS and configured Hadoop 2.8.2 and pig-0.17.0 on the same

Put the dataset in HDFS location



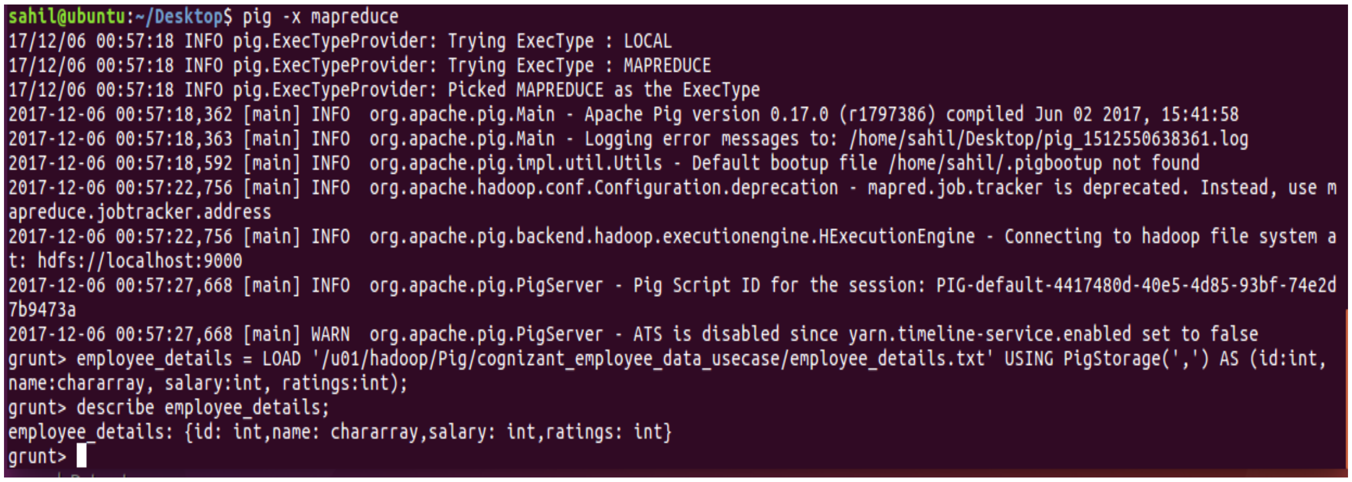
```
sahil@ubuntu:~/Desktop$ hdfs dfs -mkdir /u01/hadoop/Pig/cognizant_employee_data_usecase
sahil@ubuntu:~/Desktop$
sahil@ubuntu:~/Desktop$ hdfs dfs -put employee_details.txt /u01/hadoop/Pig/cognizant_employee_data_usecase
sahil@ubuntu:~/Desktop$
sahil@ubuntu:~/Desktop$ hdfs dfs -put employee_expenses.txt /u01/hadoop/Pig/cognizant_employee_data_usecase
sahil@ubuntu:~/Desktop$
sahil@ubuntu:~/Desktop$ hdfs dfs -ls /u01/hadoop/Pig/cognizant_employee_data_usecase
Found 2 items
-rw-r--r--   1 sahil supergroup        261 2017-12-05 23:11 /u01/hadoop/Pig/cognizant_employee_data_usecase/employee_details.txt
-rw-r--r--   1 sahil supergroup         72 2017-12-05 23:12 /u01/hadoop/Pig/cognizant_employee_data_usecase/employee_expenses.txt
sahil@ubuntu:~/Desktop$
```

Load dataset in Pig

employee_details.txt

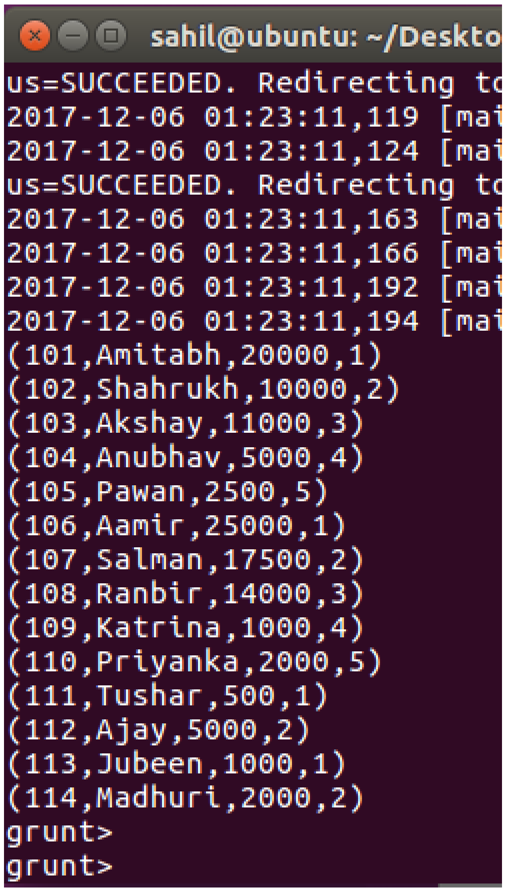
grunt> employee_details = LOAD '/u01/hadoop/Pig/cognizant_employee_data_usecase/employee_details.txt' USING PigStorage(',') AS (id:int, name:chararray, salary:int, ratings:int);

grunt> describe employee_details;



```
sahil@ubuntu:~/Desktop$ pig -x mapreduce
17/12/06 00:57:18 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
17/12/06 00:57:18 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
17/12/06 00:57:18 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2017-12-06 00:57:18,362 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.0 (r1797386) compiled Jun 02 2017, 15:41:58
2017-12-06 00:57:18,363 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/sahil/Desktop/pig_1512550638361.log
2017-12-06 00:57:18,592 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/sahil/.pigbootup not found
2017-12-06 00:57:22,756 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use m
apreduce.jobtracker.address
2017-12-06 00:57:22,756 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system a
t: hdfs://localhost:9000
2017-12-06 00:57:27,668 [main] INFO  org.apache.pig.PigServer - Pig Script ID for the session: PIG-default-4417480d-40e5-4d85-93bf-74e2d
7b9473a
2017-12-06 00:57:27,668 [main] WARN  org.apache.pig.PigServer - ATS is disabled since yarn.timeline-service.enabled set to false
grunt> employee_details = LOAD '/u01/hadoop/Pig/cognizant_employee_data_usecase/employee_details.txt' USING PigStorage(',') AS (id:int,
name:chararray, salary:int, ratings:int);
grunt> describe employee_details;
employee_details: {id: int,name: chararray,salary: int,ratings: int}
grunt>
```

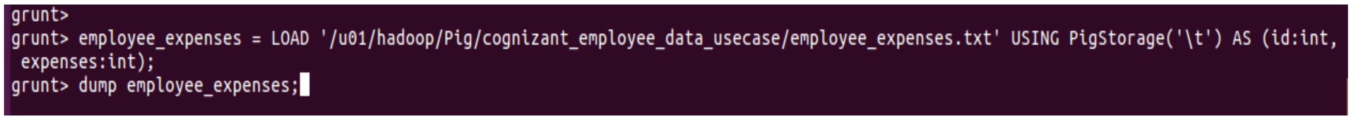
grunt> dump employee_details;



```
sahil@ubuntu: ~/Deskto
us=SUCCEEDED. Redirecting to
2017-12-06 01:23:11,119 [mai
2017-12-06 01:23:11,124 [mai
us=SUCCEEDED. Redirecting to
2017-12-06 01:23:11,163 [mai
2017-12-06 01:23:11,166 [mai
2017-12-06 01:23:11,192 [mai
2017-12-06 01:23:11,194 [mai
(101,Amitabh,20000,1)
(102,Shahrukh,10000,2)
(103,Akshay,11000,3)
(104,Anubhav,5000,4)
(105,Pawan,2500,5)
(106,Aamir,25000,1)
(107,Salman,17500,2)
(108,Ranbir,14000,3)
(109,Katrina,1000,4)
(110,Priyanka,2000,5)
(111,Tushar,500,1)
(112,Ajay,5000,2)
(113,Jubeen,1000,1)
(114,Madhuri,2000,2)
grunt>
grunt>
```

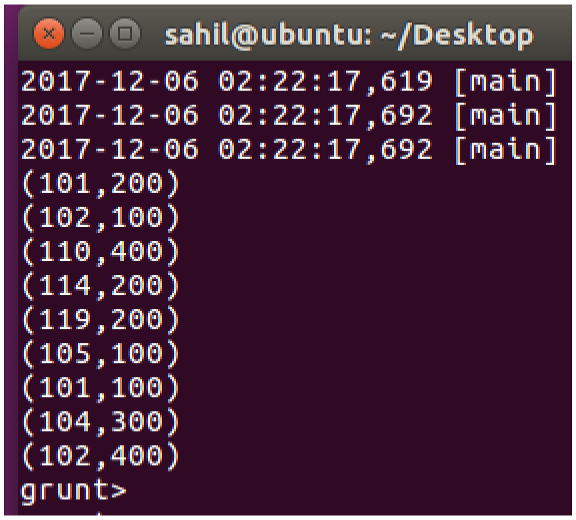
employee_expenses.txt

grunt> employee_expenses = LOAD '/u01/hadoop/Pig/cognizant_employee_data_usecase/employee_expenses.txt' USING PigStorage('\t') AS (id:int, expenses:int);



```
grunt>
grunt> employee_expenses = LOAD '/u01/hadoop/Pig/cognizant_employee_data_usecase/employee_expenses.txt' USING PigStorage('\t') AS (id:int,
 expenses:int);
grunt> dump employee_expenses;
```

grunt> dump employee_expenses;



```
sahil@ubuntu: ~/Desktop
2017-12-06 02:22:17,619 [main]
2017-12-06 02:22:17,692 [main]
2017-12-06 02:22:17,692 [main]
(101,200)
(102,100)
(110,400)
(114,200)
(119,200)
(105,100)
(101,100)
(104,300)
(102,400)
grunt>
```

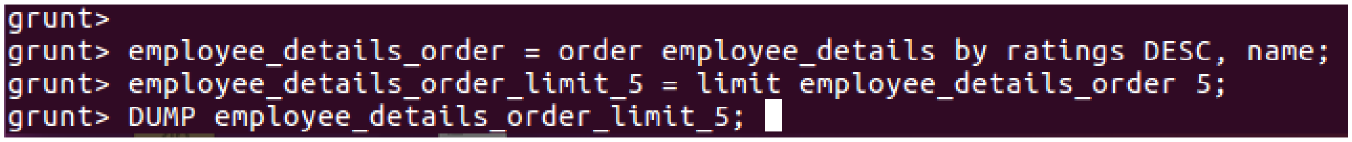
(a) Top 5 employees (employee id and employee name) with highest rating. (In case two employees have same rating, employee with name coming first in dictionary should get preference)

Commands Used:-

grunt> employee_details_order = order employee_details by ratings DESC, name;

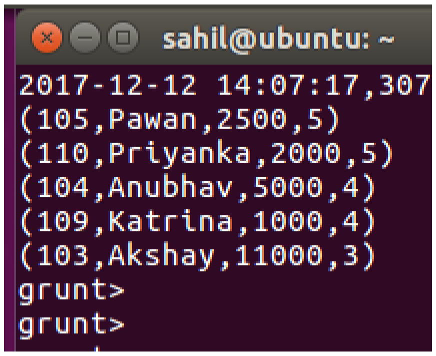
grunt> employee_details_order_limit_5 = limit employee_details_order 5;

grunt> DUMP employee_details_order_limit_5;



```
grunt>
grunt> employee_details_order = order employee_details by ratings DESC, name;
grunt> employee_details_order_limit_5 = limit employee_details_order 5;
grunt> DUMP employee_details_order_limit_5;
```

Result:-



```
sahil@ubuntu: ~
2017-12-12 14:07:17,307
(105,Pawan,2500,5)
(110,Priyanka,2000,5)
(104,Anubhav,5000,4)
(109,Katrina,1000,4)
(103,Akshay,11000,3)
grunt>
grunt>
```

(105,Pawan,2500,5)
(110,Priyanka,2000,5)
(104,Anubhav,5000,4)
(109,Katrina,1000,4)
(103,Akshay,11000,3)

(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)

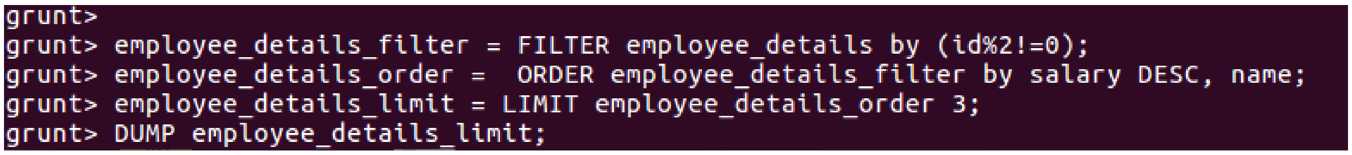
Commands Used:-

grunt> employee_details_filter = FILTER employee_details by (id%2!=0);

grunt> employee_details_order = ORDER employee_details_filter by salary DESC, name;

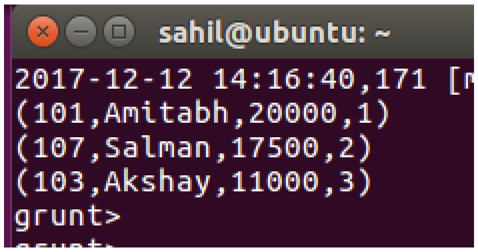
grunt> employee_details_limit = LIMIT employee_details_order 3;

grunt> DUMP employee_details_limit;



```
grunt>
grunt> employee_details_filter = FILTER employee_details by (id%2!=0);
grunt> employee_details_order =  ORDER employee_details_filter by salary DESC, name;
grunt> employee_details_limit = LIMIT employee_details_order 3;
grunt> DUMP employee_details_limit;
```

Result:-



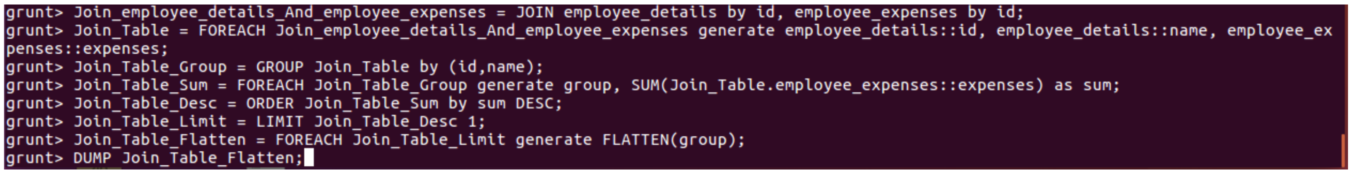
```
2017-12-12 14:16:40,171 [r
(101,Amitabh,20000,1)
(107,Salman,17500,2)
(103,Akshay,11000,3)
grunt>
```

(101,Amitabh,20000,1)
(107,Salman,17500,2)
(103,Akshay,11000,3)

(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)

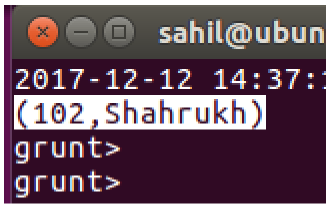
Commands Used:-

grunt> Join_employee_details_And_employee_expenses = JOIN employee_details by id, employee_expenses by id;
grunt> Join_Table = FOREACH Join_employee_details_And_employee_expenses generate employee_details::id, employee_details::name, employee_expenses::expenses;
grunt> Join_Table_Group = GROUP Join_Table by (id,name);
grunt> Join_Table_Sum = FOREACH Join_Table_Group generate group, SUM(Join_Table.employee_expenses::expenses) as sum;
grunt> Join_Table_Desc = ORDER Join_Table_Sum by sum DESC;
grunt> Join_Table_Limit = LIMIT Join_Table_Desc 1;
grunt> Join_Table_Flatten = FOREACH Join_Table_Limit generate FLATTEN(group);
grunt> DUMP Join_Table_Flatten;



```
grunt> Join_employee_details_And_employee_expenses = JOIN employee_details by id, employee_expenses by id;
grunt> Join_Table = FOREACH Join_employee_details_And_employee_expenses generate employee_details::id, employee_details::name, employee_ex
penses::expenses;
grunt> Join_Table_Group = GROUP Join_Table by (id,name);
grunt> Join_Table_Sum = FOREACH Join_Table_Group generate group, SUM(Join_Table.employee_expenses::expenses) as sum;
grunt> Join_Table_Desc = ORDER Join_Table_Sum by sum DESC;
grunt> Join_Table_Limit = LIMIT Join_Table_Desc 1;
grunt> Join_Table_Flatten = FOREACH Join_Table_Limit generate FLATTEN(group);
grunt> DUMP Join_Table_Flatten;
```

Result:-



```
2017-12-12 14:37:
(102,Shahrukh)
grunt>
grunt>
```

(102,Shahrukh)

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

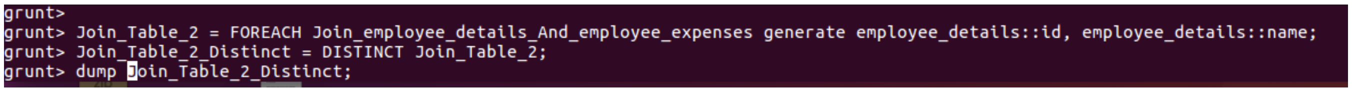
Commands Used:-

grunt> Join_employee_details_And_employee_expenses = JOIN employee_details by id, employee_expenses by id;

grunt> Join_Table_2 = FOREACH Join_employee_details_And_employee_expenses generate employee_details::id, employee_details::name;

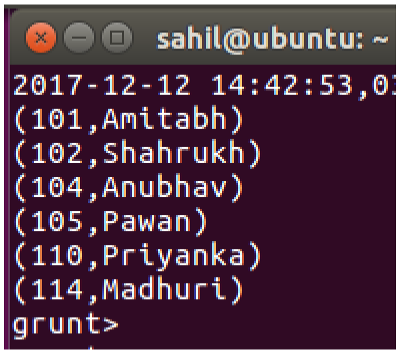
grunt> Join_Table_2_Distinct = DISTINCT Join_Table_2;

grunt> dump Join_Table_2_Distinct;



```
grunt>
grunt> Join_Table_2 = FOREACH Join_employee_details_And_employee_expenses generate employee_details::id, employee_details::name;
grunt> Join_Table_2_Distinct = DISTINCT Join_Table_2;
grunt> dump Join_Table_2_Distinct;
```

Result:-



```
sahil@ubuntu: ~
2017-12-12 14:42:53,01
(101,Amitabh)
(102,Shahrukh)
(104,Anubhav)
(105,Pawan)
(110,Priyanka)
(114,Madhuri)
grunt>
```

(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 file.

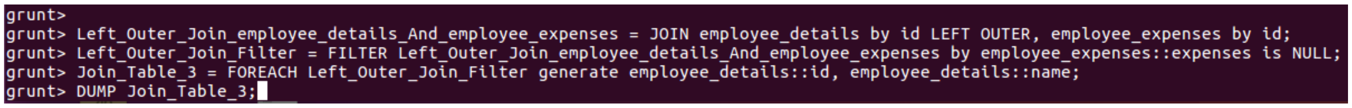
Commands Used:-

grunt> Left_Outer_Join_employee_details_And_employee_expenses = JOIN employee_details by id LEFT OUTER, employee_expenses by id;

grunt> Left_Outer_Join_Filter = FILTER Left_Outer_Join_employee_details_And_employee_expenses by employee_expenses::expenses is NULL;

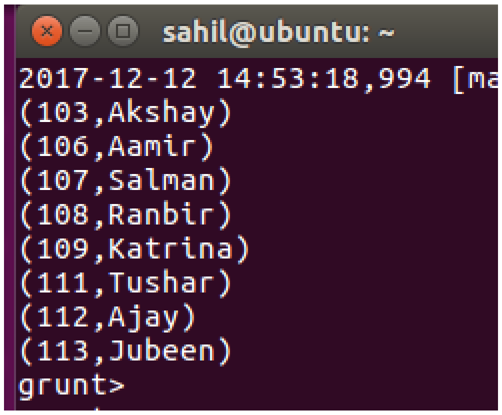
grunt> Join_Table_3 = FOREACH Left_Outer_Join_Filter generate employee_details::id, employee_details::name;

grunt> DUMP Join_Table_3;



```
grunt>
grunt> Left_Outer_Join_employee_details_And_employee_expenses = JOIN employee_details by id LEFT OUTER, employee_expenses by id;
grunt> Left_Outer_Join_Filter = FILTER Left_Outer_Join_employee_details_And_employee_expenses by employee_expenses::expenses is NULL;
grunt> Join_Table_3 = FOREACH Left_Outer_Join_Filter generate employee_details::id, employee_details::name;
grunt> DUMP Join_Table_3;
```

Result:-



```
2017-12-12 14:53:18,994 [ma
(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)
grunt>
```

(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)