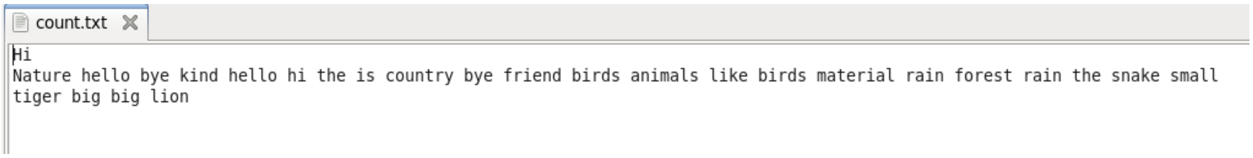
PIG-Assignment 7

Task 1

Write the WordCount Program using Pig

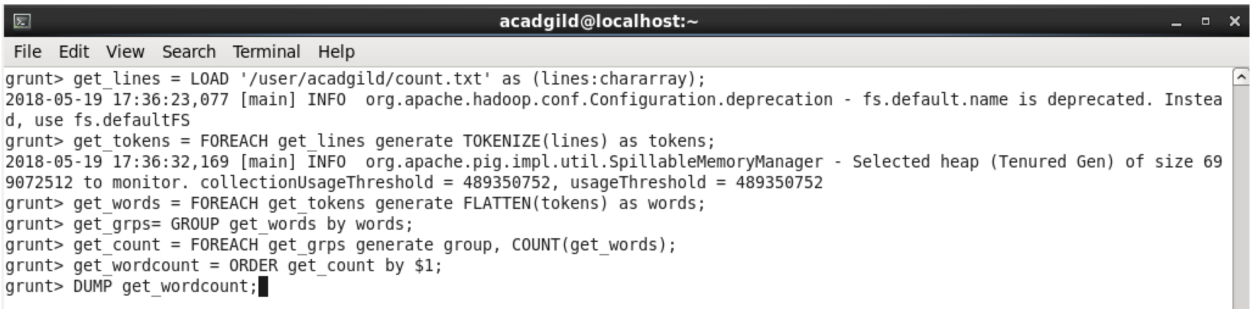
Solution:

Input WordCount File provided is as below :



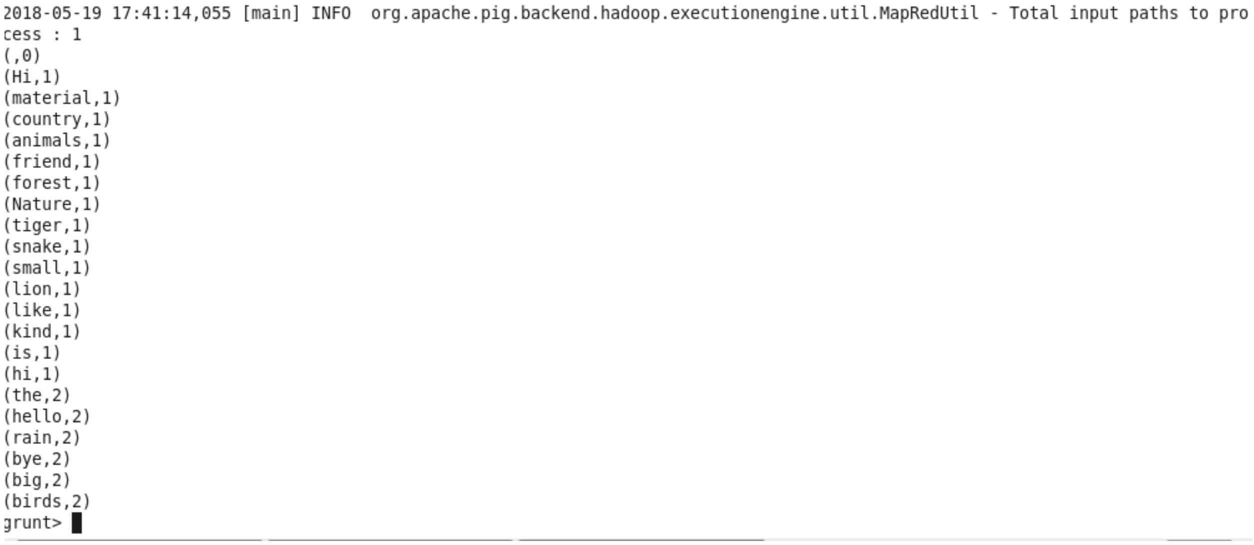
```
Hi
Nature hello bye kind hello hi the is country bye friend birds animals like birds material rain forest rain the snake small
tiger big big lion
```

Below is the screen shot of wordcount Program:



```
grunt> get_lines = LOAD '/user/acadgild/count.txt' as (lines:chararray);
2018-05-19 17:36:23,077 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> get_tokens = FOREACH get_lines generate TOKENIZE(lines) as tokens;
2018-05-19 17:36:32,169 [main] INFO  org.apache.pig.impl.util.SpillableMemoryManager - Selected heap (Tenured Gen) of size 69
9072512 to monitor. collectionUsageThreshold = 489350752, usageThreshold = 489350752
grunt> get_words = FOREACH get_tokens generate FLATTEN(tokens) as words;
grunt> get_grps= GROUP get_words by words;
grunt> get_count = FOREACH get_grps generate group, COUNT(get_words);
grunt> get_wordcount = ORDER get_count by $1;
grunt> DUMP get_wordcount;
```

***Output**:*



```
2018-05-19 17:41:14,055 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(,0)
(Hi,1)
(material,1)
(country,1)
(animals,1)
(friend,1)
(forest,1)
(Nature,1)
(tiger,1)
(snake,1)
(small,1)
(lion,1)
(like,1)
(kind,1)
(is,1)
(hi,1)
(the,2)
(hello,2)
(rain,2)
(bye,2)
(big,2)
(birds,2)
grunt> 
```

TASK 2 :

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,DepartmentID)

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

employee_expenses(EmpID,Expence)

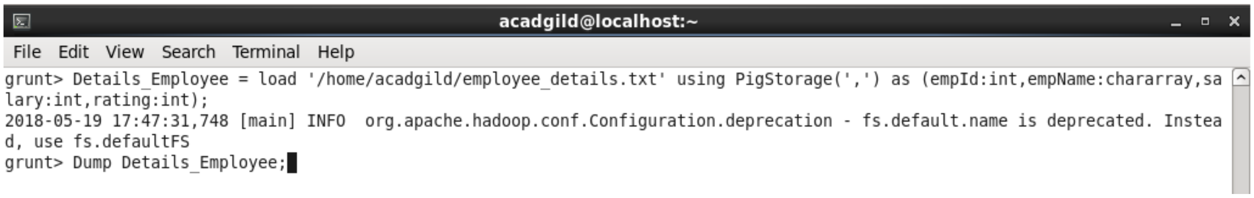
https://github.com/prateekATacadgild/DatasetsForCognizant/blob/master/employee_expenses.txt

Solution :

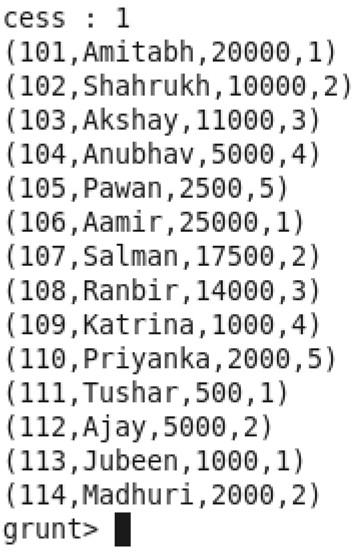
Use below command to get the desired output.

grunt> Details_Employee = load '/home/acadgild/employee_details.txt' using PigStorage(',') as (empId:int,empName:chararray,salary:int,rating:int);

Dump Details_Employee



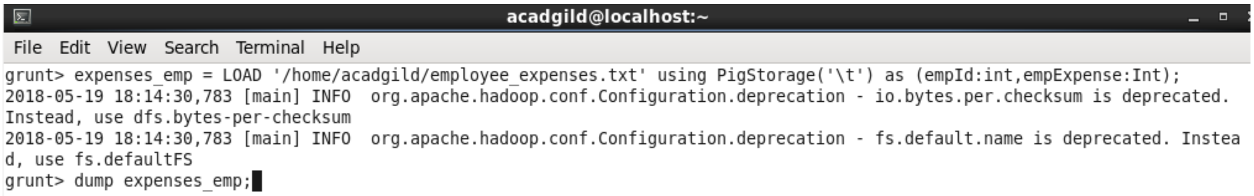
```
grunt> Details_Employee = load '/home/acadgild/employee_details.txt' using PigStorage(',') as (empId:int,empName:chararray,sa
lary:int,rating:int);
2018-05-19 17:47:31,748 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> Dump Details_Employee;
```



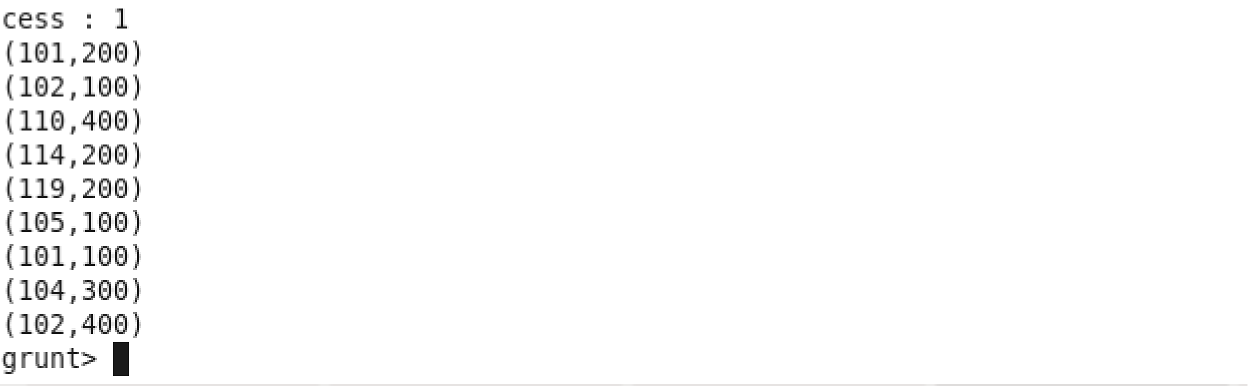
```
cess : 1
(101,Amitabh,20000,1)
(102,Shahrukh,10000,2)
(103,Akshay,11000,3)
(104,Anubhav,5000,4)
(105,Pawan,2500,5)
(106,Aamir,25000,1)
(107,Salman,17500,2)
(108,Ranbir,14000,3)
(109,Katrina,1000,4)
(110,Priyanka,2000,5)
(111,Tushar,500,1)
(112,Ajay,5000,2)
(113,Jubeen,1000,1)
(114,Madhuri,2000,2)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

- expenses_emp = LOAD '/home/acadgild/employee_expenses.txt' using PigStorage('\t') as (empId:int,empExpense:Int);

-dump expenses_emp



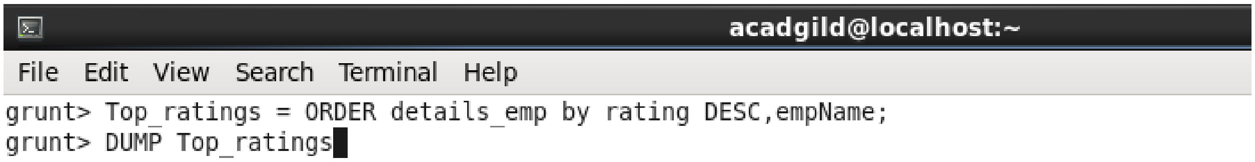
```
grunt> expenses_emp = LOAD '/home/acadgild/employee_expenses.txt' using PigStorage('\t') as (empId:int,empExpense:Int);
2018-05-19 18:14:30,783 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated.
Instead, use dfs.bytes-per-checksum
2018-05-19 18:14:30,783 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> dump expenses_emp;
```



```
cess : 1
(101,200)
(102,100)
(110,400)
(114,200)
(119,200)
(105,100)
(101,100)
(104,300)
(102,400)
grunt>
```

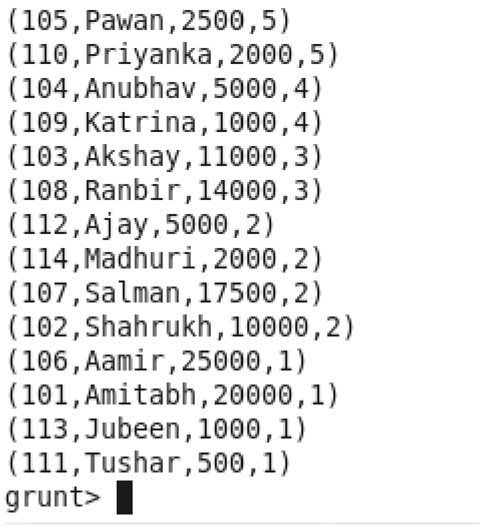
a) ***Top 5 employees (employee id and employee name) with highest rating. (In case two employees have same rating, employee with name coming first in dictionary should get preference)***

SOLUTION:

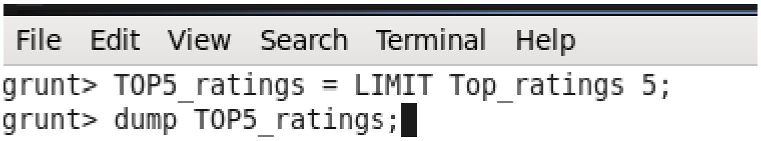


```
grunt> Top_ratings = ORDER details_emp by rating DESC,empName;
grunt> DUMP Top_ratings
```

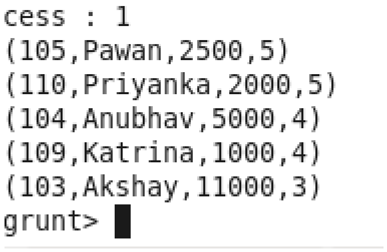
Output:



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



```
grunt> TOP5_ratings = LIMIT Top_ratings 5;
grunt> dump TOP5_ratings;
```



```
cess : 1
(105,Pawan,2500,5)
(110,Priyanka,2000,5)
(104,Anubhav,5000,4)
(109,Katrina,1000,4)
(103,Akshay,11000,3)
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)

Solution:

1)emp_oddID = FILTER employee_details by empId%2==1;

2)dump emp_oddID;

Output :

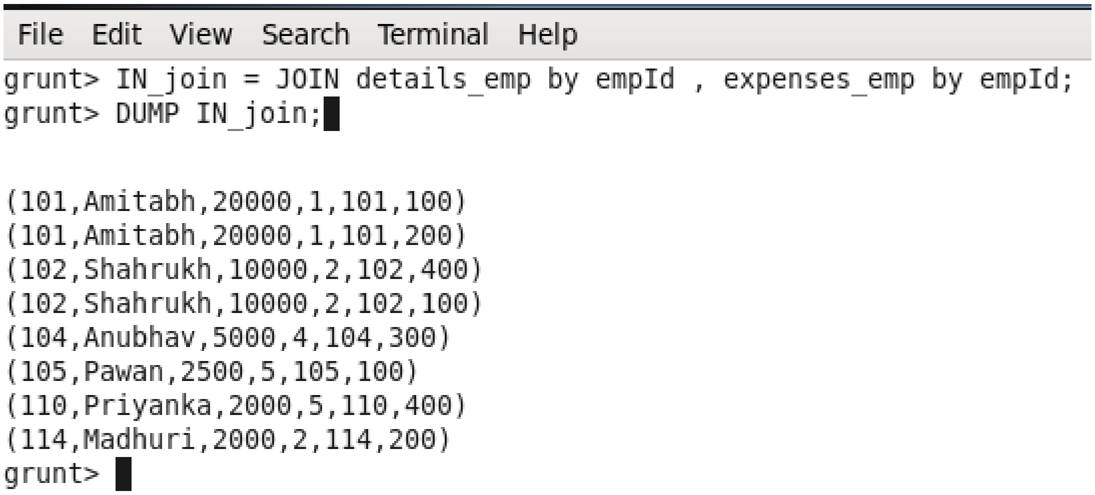
(101,Amitabh,20000,1)

```
(103,Akshay,11000,3)

(105,Pawan,2500,5)

(107,Salman,17500,2)

(109,Katrina,1000,4)

(111,Tushar,500,1)

(113,Jubeen,1000,1)
```

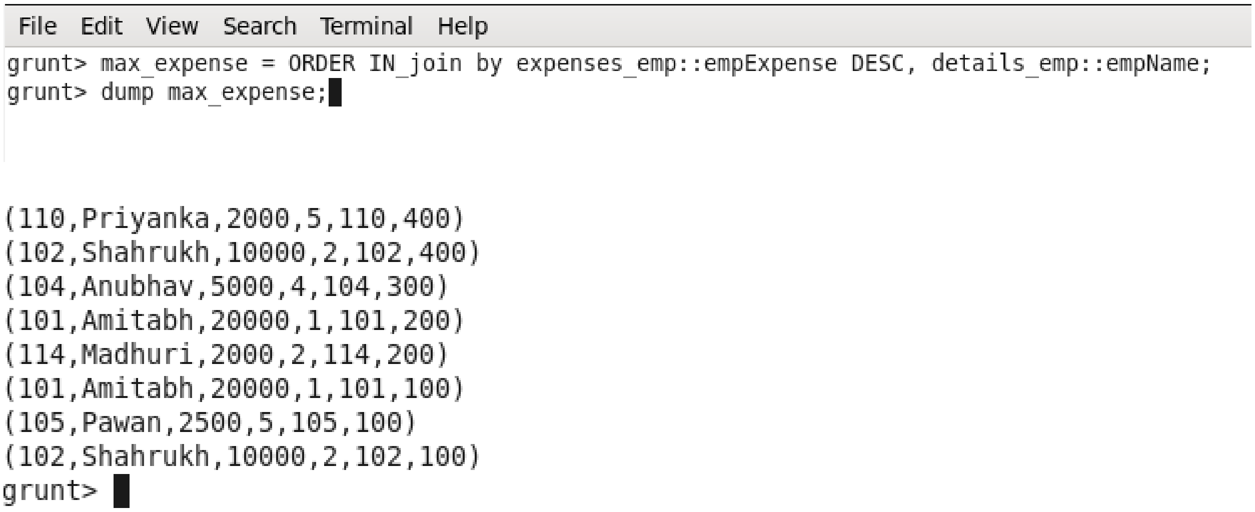
```
3)highest_salary= ORDER emp_oddID by salary DESC,empName;

4)dump highest_salary;

(101,Amitabh,20000,1)

(107,Salman,17500,2)

(103,Akshay,11000,3)

(105,Pawan,2500,5)

(113,Jubeen,1000,1)

(109,Katrina,1000,4)

(111,Tushar,500,1)

5)ouput_b= LIMIT highest_salary 3;

6)dump ouput_b------------------------------------------------->RESULT

(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)

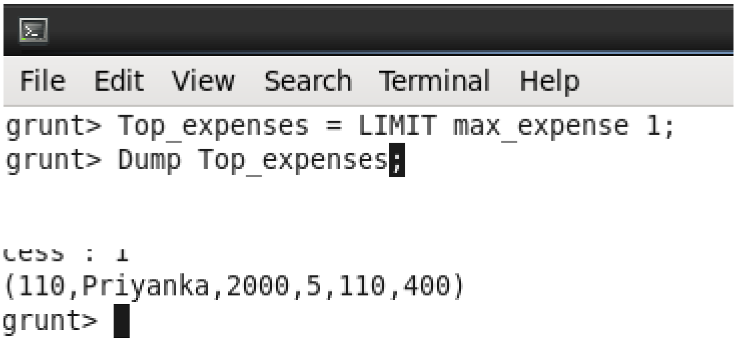
Solution:



```
grunt> IN_join = JOIN details_emp by empId , expenses_emp by empId;
grunt> DUMP IN_join;


(101,Amitabh,20000,1,101,100)
(101,Amitabh,20000,1,101,200)
(102,Shahrukh,10000,2,102,400)
(102,Shahrukh,10000,2,102,100)
(104,Anubhav,5000,4,104,300)
(105,Pawan,2500,5,105,100)
(110,Priyanka,2000,5,110,400)
(114,Madhuri,2000,2,114,200)
grunt>
```



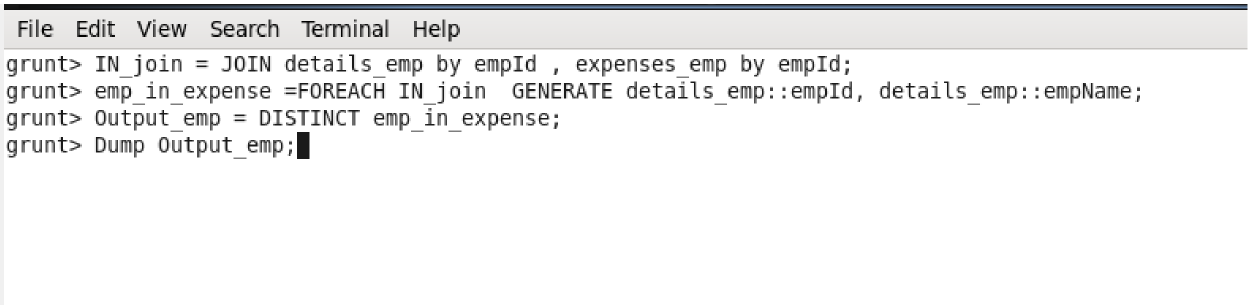
```
grunt> max_expense = ORDER IN_join by expenses_emp::empExpense DESC, details_emp::empName;
grunt> dump max_expense;


(110,Priyanka,2000,5,110,400)
(102,Shahrukh,10000,2,102,400)
(104,Anubhav,5000,4,104,300)
(101,Amitabh,20000,1,101,200)
(114,Madhuri,2000,2,114,200)
(101,Amitabh,20000,1,101,100)
(105,Pawan,2500,5,105,100)
(102,Shahrukh,10000,2,102,100)
grunt>
```



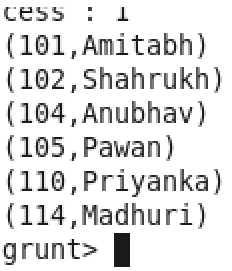
```
grunt> Top_expenses = LIMIT max_expense 1;
grunt> Dump Top_expenses;


cess : 1
(110,Priyanka,2000,5,110,400)
grunt>
```

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

Solution:



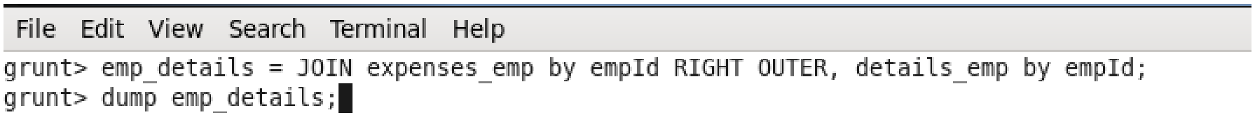
```
grunt> IN_join = JOIN details_emp by empId , expenses_emp by empId;
grunt> emp_in_expense =FOREACH IN_join  GENERATE details_emp::empId, details_emp::empName;
grunt> Output_emp = DISTINCT emp_in_expense;
grunt> Dump Output_emp;
```



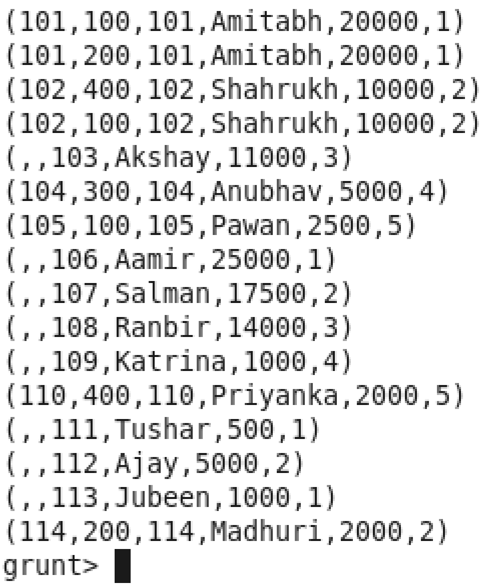
```
cess : 1
(101,Amitabh)
(102,Shahrukh)
(104,Anubhav)
(105,Pawan)
(110,Priyanka)
(114,Madhuri)
grunt>
```

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

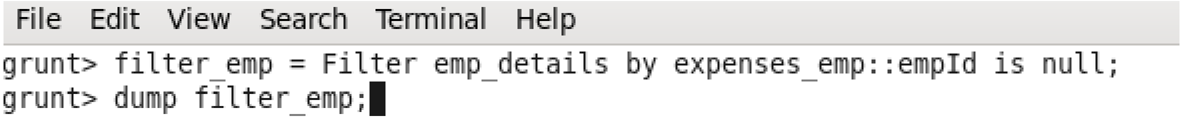
Solution:



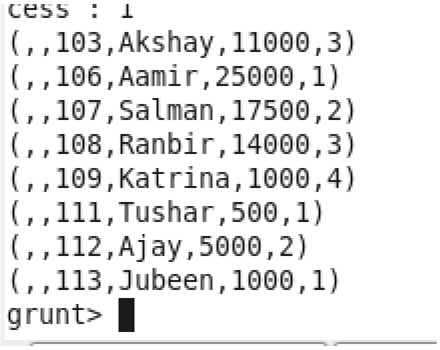
```
grunt> emp_details = JOIN expenses_emp by empId RIGHT OUTER, details_emp by empId;
grunt> dump emp_details;
```



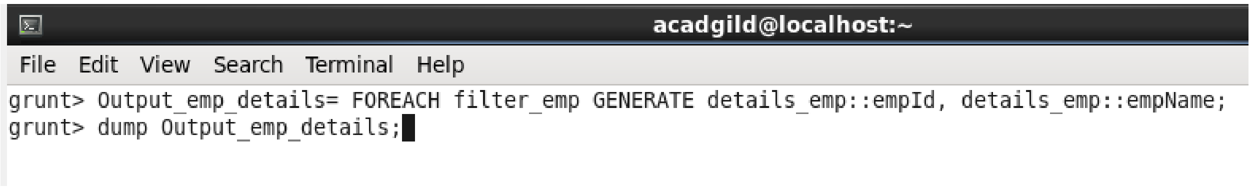
```
(101,100,101,Amitabh,20000,1)
(101,200,101,Amitabh,20000,1)
(102,400,102,Shahrukh,10000,2)
(102,100,102,Shahrukh,10000,2)
(,,103,Akshay,11000,3)
(104,300,104,Anubhav,5000,4)
(105,100,105,Pawan,2500,5)
(,,106,Aamir,25000,1)
(,,107,Salman,17500,2)
(,,108,Ranbir,14000,3)
(,,109,Katrina,1000,4)
(110,400,110,Priyanka,2000,5)
(,,111,Tushar,500,1)
(,,112,Ajay,5000,2)
(,,113,Jubeen,1000,1)
(114,200,114,Madhuri,2000,2)
grunt>
```



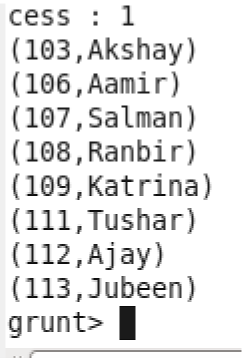
```
grunt> filter_emp = Filter emp_details by expenses_emp::empId is null;
grunt> dump filter_emp;
```



```
cess : 1
(,,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)
grunt>
```



```
grunt> Output_emp_details= FOREACH filter_emp GENERATE details_emp::empId, details_emp::empName;
grunt> dump Output_emp_details;
```



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

Task 3:

Implement the use case present in below blog link and share the complete steps along with Screen shot(s) from your end.

https://acadgild.com/blog/aviation-data-analysis-using-apache-pig/

Solution

Problem Statement 1 : Find out the top 5 most visited destinations.

grunt> history

1 A= LOAD '/home/acadgild/Downloads/DelayedFlights.csv' Using org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');

2 B= foreach A generate (int)$1 as year, (int)$10 as flight_num, (chararray)$17 as origin, (chararray)$18 as dest;

3 C =filter B by dest is not null;

4 D = GROUP C by dest;

```
E =foreach D generate group, COUNT(C.dest);

F =order E by $1 DESC;

Result = LIMIT F 5;

dump Result;

2018-05-13 22:15:46,787 [main] INFO

org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1


(ORD,108984)


(ATL,106898)


(DFW,70657)


(DEN,63003)


(LAX,59969)
```

Problem Statement 2 : Which month has been the most number of cancellations due to bad weather.

```
A= LOAD '/home/acadgild/Downloads/DelayedFlights.csv' Using org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
```

```
B= FOREACH A generate (int)$2 as month, (int)$10 as flight_num, (int)$22 as cancelled,
(chararray)$23 as cancel_code;
```

```
C= filter B by cancelled==1 AND cancel_code=='B';

D =group C by month;

E =FOREACH D generate group,COUNT(C.cancelled);

F =order E by $1 DESC;

REsult = limit F 1;

Dump REsult;
```

```
2018-05-13 22:26:08,460 [main] INFO

org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to
process : 1


(12,250)
```

PROBLEM STATEMENT 3 : Top ten origins with the highest AVG departure delay.

```
A= LOAD '/home/acadgild/Downloads/DelayedFlights.csv' Using
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_
HEADER');

B1 = FOREACH A GENERATE (int)$16 as dep_delay, (chararray)$17 as origin;

C1 = filter B1 by (dep_delay is not null) AND (origin is not null);
```

```
D1= group C1 by origin;


E1= FOREACH D1 generate

group,AVG(C1.dep_delay); Result =order E1 by

$1 DESC; top_ten =limit Result 10;


Lookup = load '/home/acadgild/Downloads/airports.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');


Lookup1 =FOREACH Lookup generate (chararray)$0 as origin, (chararray)$2 as city,
(chararray)$4 as country;


Joined =join Lookup1 by origin, top_ten by $0;


Final =Foreach Joined generate $0,$1,$2,$4;


Final_Result = ORDER Final by $3 DESC;


DUMP Final_Result
```

```
2018-05-13 22:50:42,423 [main] WARN
org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has
already been initialized


2018-05-13 22:50:42,491 [main] INFO
org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process :
1


2018-05-13 22:50:42,491 [main] INFO


org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to
process : 1


(CMX,Hancock,USA,116.1470588235294)


(PLN,Pellston,USA,93.76190476190476)


(SPI,Springfield,USA,83.84873949579831)


(ALO,Waterloo,USA,82.2258064516129)


(MQT,NA,USA,79.55665024630542)


(ACY,Atlantic City,USA,79.3103448275862)


(MOT,Minot,USA,78.66165413533835)


(HHH,NA,USA,76.53005464480874)
```

```
(EGE,Eagle,USA,74.12891986062718)

(BGM,Binghamton,USA,73.15533980582525)
```

PROBLME STATEMENT 4 : Which route (origin&destination) has been the maximum diversion.

```
grunt> A= LOAD '/home/acadgild/Downloads/DelayedFlights.csv' Using
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_
HEADER');

2018-05-13 22:54:32,442 [main] INFO
org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is
deprecated. Instead, use dfs.bytes-per-checksum

2018-05-13 22:54:32,443 [main] INFO
org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is
deprecated. Instead, use fs.defaultFS

grunt> B =FOREACH A GENERATE (chararray)$17 as origin, (chararray)$18 as dest,
(int)$24 as diversion;

grunt> C = FILTER B BY (origin is not null) AND (dest is not null) AND (diversion ==1);

grunt> D = GROUP C by (origin,dest);

grunt> E =FOREACH D generate group,COUNT(C.diversion);
grunt> F = ORDER E by $1 DESC;

grunt> Res = limit F 1;

grunt> Result = limit F 10;

grunt> dump Result;



2018-05-13 23:04:24,145 [main] WARN
org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has
already been initialized

2018-05-13 23:04:24,182 [main] INFO
org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process :
1

2018-05-13 23:04:24,182 [main] INFO

org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to
process : 1

((ORD,LGA),39)

((DAL,HOU),35)

((DFW,LGA),33)

((ATL,LGA),32)
```

```
((ORD,SNA),31)

((SLC,SUN),31)

((MIA,LGA),31)

((BUR,JFK),29)

((HRL,HOU),28)

((BUR,DFW),25)

grunt>
```