**PROBLEM #1:**

Select frequent words (whose count is equal or greater than 50,000). Hint. Use ‘filter’.

Display the frequent words in descending order. (Hint: ORDER .. BY..DESC)

lines = load 'eBooks/\*.txt' as (line:chararray);

words = foreach lines generate FLATTEN(TOKENIZE(line)) as word;

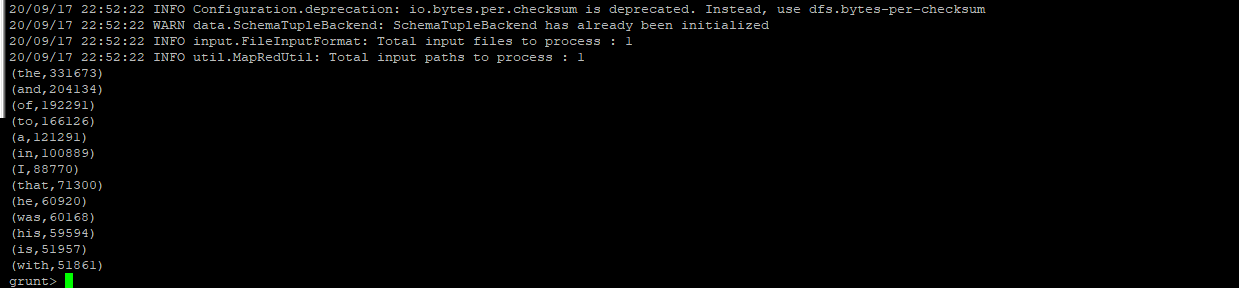
grouped = group words by word;

wordcount = foreach grouped generate group, COUNT(words);

filtered = FILTER wordcount by $1 >= 50000;

ordered\_records = ORDER filtered BY $1 DESC;

dump ordered\_records;



**PROBLEM #2**: → simple change from the above example  
Get groups of words by their length (Hint: use the built-in function SIZE) and count each group.

eg)

grunt> DUMP grouped;  
(1,258.946) → means there are 258946 occurence of words that have one character.

(2,1.096.049) → means there are 1096049 occurence of words that have two characters.

(3,1.444.799)

(4,1.096.414)

(5,711.919)

...

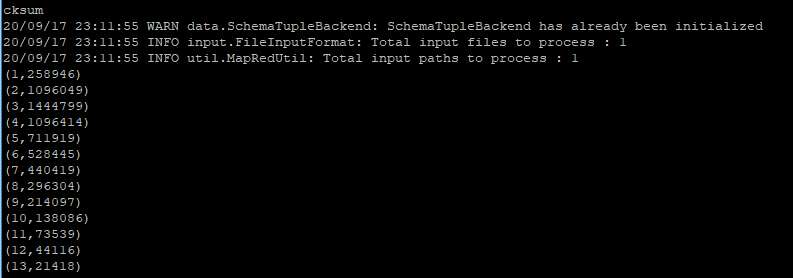
lines = load 'eBooks/\*.txt' as (line:chararray);

words = foreach lines generate FLATTEN(TOKENIZE(line)) as word;

grouped = group words by SIZE($0);

wordcount = foreach grouped generate group, COUNT(words);

dump wordcount;



PROBLEM #3:  
Find out the effect of passenger\_count on trip\_distance, fare\_amount, and tip\_rate.

(a) Create a new record ‘records2’ that has passenger\_count, trip\_distance, fare\_amount, tip\_rate (tip\_amount/total\_amount)

(b) Filter records2 by passenger\_count ( 0 < passenger\_count < 10 ) and create ‘records3’

(c) Group records3 by passenger\_count.   
(d) Display the average trip\_distance, average fare\_amount, and average tip\_rate per each group of passenger\_count.

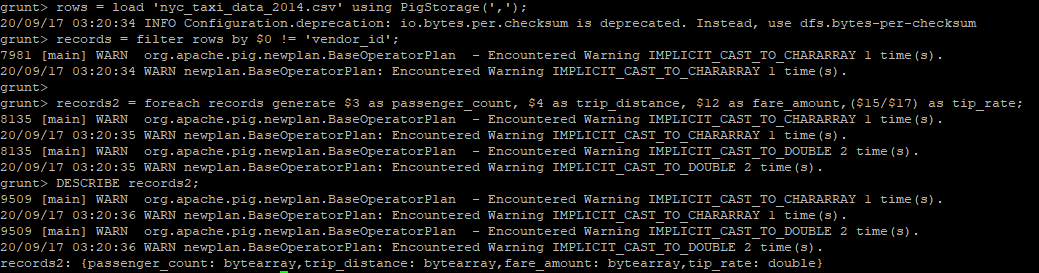
a)

grunt> rows = load 'nyc\_taxi\_data\_2014.csv' using PigStorage(',');

grunt> records = filter rows by $0 != 'vendor\_id';

grunt> records2 = foreach records generate $3 as passenger\_count, $4 as trip\_distance, $12 as fare\_amount,($15/$17) as tip\_rate;

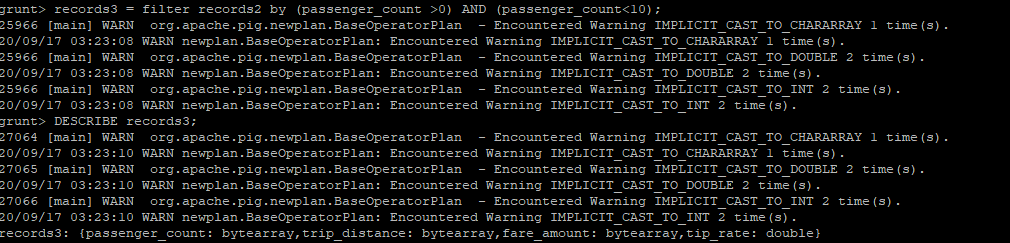
grunt> DESCRIBE records2;



b)

grunt> records3 = filter records2 by (passenger\_count >0) AND (passenger\_count<10);

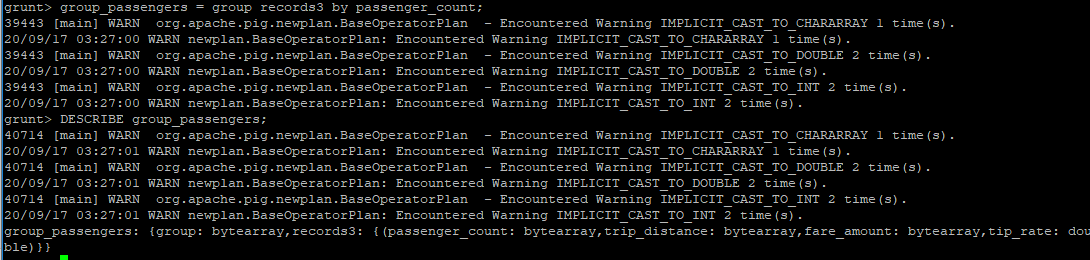
grunt> DESCRIBE records3;



c)

grunt> group\_passengers = group records3 by passenger\_count;

grunt> DESCRIBE group\_passengers;



d)

grunt> final\_result = foreach group\_passengers generate group, AVG(records3.trip\_distance), AVG(records3.fare\_amount),AVG(records3.tip\_rate);

grunt>DUMP final\_result;

