Task1- Write a Program to implement word count using Pig.

Code for wordcount:

```
lines =load 'demo.txt' as (line:chararray);
words =foreach lines generate TOKENIZE(line) as word;
dump words;
res =foreach words generate Flatten(word) as word1;
dump res;
grp =group res by word1;
dump grp;
describe grp;
res2 =foreach grp generate group as word, COUNT(res.word1) as cnt;
dump res2;
```

Input File

```
[acadgild@localhost ~]$ cat demo.txt
Java is simple
Java is secure
Java is compiled
Java is interpreted[acadgild@localhost ~]$
```

Output

```
(is,4)
(Java,4)
(secure,1)
(simple,1)
(compiled,1)
(interpreted,1)
grunt>
```

Task 2:

(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)

Source Code:

```
data = LOAD 'employee_details.txt' using PigStorage(',') AS (eid:int,ename:chararray,sal:int,erate:int);
grp = group data by eid;
maxrate  = FOREACH grp Generate FLATTEN(data.eid),FLATTEN(data.ename) as name,MAX(data.erate) as maxrt;
orddata = ORDER maxrate BY maxrt desc,name;
STORE orddata into 'MAXRATE';
```

Input:

```
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
~
~
```

Output:

```
105     Pawan   5
110     Priyanka        5
104     Anubhav 4
109     Katrina 4
103     Akshay  3
108     Ranbir  3
112     Ajay    2
114     Madhuri 2
107     Salman  2
102     Shahrukh        2
106     Aamir   1
101     Amitabh 1
113     Jubeen  1
111     Tushar  1
~
~
~
```

(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)

Source Code:

```
data = LOAD 'employee_details.txt' using PigStorage(',') AS (eid:int,ename:chararray,sal:int,did:int);
filterdata = FILTER data BY (eid%2==1);
grp = group filterdata by eid;
maxsal = FOREACH grp Generate FLATTEN(filterdata.eid),FLATTEN(filterdata.ename),MAX(filterdata.sal) as maxsl;
orddata = ORDER maxsal BY maxsl desc;
limidata = limit orddata 3;
DUMP limidata;
STORE limidata into 'MAXSAL';
~
~
~
```

Output:

```
1       ramesh  50000
7       r       34000
5       t       30000
~
~
```

(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)

Source Code:

```
data = LOAD 'employee_details.txt' using PigStorage(',') AS (eid:int,ename:chararray,sal:int,did:int);
data1 = LOAD 'employee_expenses.txt' using PigStorage('\t') AS (eid:int,expense:chararray);
joindata = JOIN data BY eid,data1 BY eid;
fdata = foreach joindata generate data1::eid,data1::expense,data::ename;
grpdata = group fdata  BY eid;
accdata = foreach grpdata Generate FLATTEN(fdata.eid) AS eid,FLATTEN(fdata.ename) AS name,MAX(fdata.expense) AS highexpense;
disdata = DISTINCT accdata;
orddata = ORDER disdata BY name;
store orddata into 'EXPENSE';
```

Output:

```
101     Amitabh 200
104     Anubhav 300
114     Madhuri 200
105     Pawan   100
110     Priyanka        400
102     Shahrukh        400
~
```

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

Source code:

```
data = LOAD 'employee_details.txt' using PigStorage(',') AS (eid:int,ename:chararray,sal:int,did:int);
data1 = LOAD 'employee_expenses.txt' using PigStorage('\t') AS (eid:int,expense:chararray);
joindata = JOIN data BY eid,data1 BY eid;
dump joindata;
fdata = foreach joindata generate data1::eid,data::ename;
result = distinct fdata;
dump result;
```

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 file.

Source Code:

```
data = LOAD 'employee_details.txt' using PigStorage(',') AS (eid:int,ename:chararray,sal:int,did:int);
data1 = LOAD 'employee_expenses.txt' using PigStorage('\t') AS (eid:int,expense:chararray);
joindata = JOIN data BY eid left outer,data1 BY eid;
fdata = filter joindata BY data1::eid is null;
STORE fdata INTO 'FDATA';
```

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

Task 3:

Problem statement 1: Find out the top 5 most visited destinantions.

Source Code:

```
grunt> register '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/piggybank-0.17.0.jar';
grunt> A = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/DelayedFlights.csv'
USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEA
DER');
grunt> B = foreach A generate (int)$1 as year, (int)$10 as flight_num, (chararray)$17 as
origin,(chararray) $18 as dest;
grunt> C = filter B by dest is not null;
grunt> D = group C by dest;
grunt> E = foreach D generate group, COUNT(C.dest);
grunt> F = order E by $1 DESC;
grunt> Result = LIMIT F 5;
grunt> dump Result;
```

```
(ORD,108984)
(ATL,106898)
(DFW,70657)
(DEN,63003)
(LAX,59969)
```

```
grunt> A1 = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
grunt> A2 = foreach A1 generate (chararray)$0 as dest, (chararray)$2 as city, (chararray)$4 as country;
grunt> joined_table = join Result by $0, A2 by dest;
grunt> dump joined_table;
```

```
grunt> A = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-11-13 03:58:05,702 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-11-13 03:58:05,702 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> B = foreach A generate (int)$1 as year, (int)$10 as flight_num, (chararray)$17 as origin,(chararray) $18 as dest;
grunt> C = filter B by dest is not null;
grunt> D = group C by dest;
grunt> E = foreach D generate group, COUNT(C.dest);
grunt> F = order E by $1 DESC;
grunt> Result = LIMIT F 5;
grunt> A1 = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-11-13 03:59:32,256 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-11-13 03:59:32,257 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> A2 = foreach A1 generate (chararray)$0 as dest, (chararray)$2 as city, (chararray)$4 as country;
grunt> joined_table = join Result by $0, A2 by dest;
grunt> dump joined_table;
```

Output:

```
(ATL,106898,ATL,Atlanta,USA)
(DEN,63003,DEN,Denver,USA)
(DFW,70657,DFW,Dallas-Fort Worth,USA)
(LAX,59969,LAX,Los Angeles,USA)
(ORD,108984,ORD,Chicago,USA)
```

Problem statement 2: Which month has seen the most number of cancellations due to bad weather?

Source Code:

```
grunt> REGISTER '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/piggybank-0.17.0.jar';
grunt> A = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTIL
INE','UNIX','SKIP_INPUT_HEADER');
2018-11-13 04:03:46,903 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-c
hecksum
2018-11-13 04:03:46,903 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> B = foreach A generate (int)$2 as month,(int)$10 as flight_num,(int)$22 as cancelled,(chararray)$23 as cancel_code;
grunt> C = filter B by cancelled == 1 AND cancel_code =='B';
grunt> D = group C by month;
grunt> E = foreach D generate group, COUNT(C.cancelled);
grunt> F= order E by $1 DESC;
grunt> Result = limit F 1;
grunt> dump Result;
```

Output:

```
2018-11-13 04:07:59,506 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(12,250)
grunt>
```

Problem statement 3: Top ten origins with the highest AVG departure delay.

Source Code:

```
grunt> REGISTER '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/piggybank-0.17.0.jar';
grunt> A = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-11-13 04:14:30,448 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-11-13 04:14:30,448 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> B1 = foreach A generate (int)$16 as dep_delay, (chararray)$17 as origin;
grunt> C1 = filter B1 by (dep_delay is not null) AND (origin is not null);
grunt> D1 = group C1 by origin;
grunt> E1 = foreach D1 generate group, AVG(C1.dep_delay);
grunt> Result = order E1 by $1 DESC;
grunt> Top_ten = limit Result 10;
grunt> Lookup = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-11-13 04:15:52,182 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2018-11-13 04:15:52,182 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> Lookup1 = foreach Lookup generate (chararray)$0 as origin, (chararray)$2 as city, (chararray)$4 as country;
grunt> Joined = join Lookup1 by origin, Top_ten by $0;
grunt> Final = foreach Joined generate $0,$1,$2,$4;
grunt> Final_Result = ORDER Final by $3 DESC;
grunt> dump Final_Result;
```

Output:

```
(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)
grunt>
```

Problem statement 4: Which route (origin & destination) has seen the maximum diversion?

Source Code:

```
grunt> REGISTER '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/piggybank-0.17.0.jar';
grunt> A = load '/home/acadgild/Desktop/Practise/PIG/ASSIGNMENT/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTIL
INE','UNIX','SKIP_INPUT_HEADER');
2018-11-13 04:25:27,266 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-c
hecksum
2018-11-13 04:25:27,267 [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> Result = limit F 10;
grunt> dump Result;
```

Output:

```
((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>
```