Assignment-7

1) Write a program to implement wordcount using Pig.

- In relation **lines**, we are loading the dataset.
- In relation **words**, we are splitting each line into bag of using **TOKENIZE** function and using **FLATTEN** function the bag is converted into tuple.
- In relation **grouped**, we are grouping the relation words.
- In relation **wordcount**, we are generating the result with group and count of words.
- Finally using dump, we are printing the result.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
grunt> lines = LOAD '/test.txt' AS (line:chararray);
2018-09-09 15:11:27,057 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> words = FOREACH lines GENERATE FLATTEN(TOKENIZE(line)) as word;
grunt> grouped = GROUP words by word;
grunt> wordcount = FOREACH grouped GENERATE group, COUNT(words);
grunt> dump wordcount;
```

The output of the word count is as highlighted in the below screenshot.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
job_1536468175284_0005


2018-09-09 15:14:18,778 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0
.0.1:8032
2018-09-09 15:14:18,786 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalAp
plicationStatus=SUCCEEDED. Redirecting to job history server
2018-09-09 15:14:18,842 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0
.0.1:8032
2018-09-09 15:14:18,863 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalAp
plicationStatus=SUCCEEDED. Redirecting to job history server
2018-09-09 15:14:18,922 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at localhost/127.0
.0.1:8032
2018-09-09 15:14:18,939 [main] INFO  org.apache.hadoop.mapred.ClientServiceDelegate - Application state is completed. FinalAp
plicationStatus=SUCCEEDED. Redirecting to job history server
2018-09-09 15:14:19,060 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success
!
2018-09-09 15:14:19,061 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
2018-09-09 15:14:19,064 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not g
enerate code.
2018-09-09 15:14:19,080 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-09 15:14:19,080 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(to,2)
(and,1)
(14th,2)
(July,1)
(Great,1)
(Hello,1)
(SPARK,1)
(learn,1)
(HADOOP,1)
(October,1)
(feeling,1)
(Acadgild,1)
grunt>
```

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)

- In relation **employee_details**, we are loading the dataset.
- In relation **order_by_rating**, we are ordering the data based on empRating and name.
- In relation **top_five_employees**, we are limiting the output to only top 5 records.
- Finally using dump, we are printing the result.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
grunt> employee_details = LOAD '/employee_details.txt' using PigStorage(',') as (empId:int, name:chararray, salary:int, empRa
ting:int);
2018-09-13 22:49:07,208 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> order_by_rating = ORDER employee_details BY empRating DESC, name;
grunt> top_five_employees = limit order_by_rating 5;
grunt> dump top_five_employees;
```

Required output of the script is as shown below.

```
2018-09-13 22:52:55,725 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not g
enerate code.
2018-09-13 22:52:55,750 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-13 22:52:55,750 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
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)

- In relation **employee_details**, we are loading the dataset.
- In relation **order_by_salary**, we are ordering the relation employee_details based on salary.
- In relation **filtered**, we are filtering the relation order_by_salary based on odd empId's.
- In relation **odd_output**, we are generating the odd empId output with empId, name and salary.
- In relation **limited_output**, we are limiting the output to top 3 records.
- Finally using dump, we are printing the output.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
grunt> employee_details = LOAD '/employee_details.txt' using PigStorage(',') as (empId:int, name:chararray, salary:int, empRa
ting:int);
2018-09-13 22:55:26,880 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> order_by_salary = ORDER employee_details BY salary DESC;
grunt> filtered = FILTER order_by_salary BY empId%2 ==1;
grunt> odd_output = FOREACH filtered GENERATE empId, name, salary;
grunt> limited_output = LIMIT odd_output 3;
grunt> dump limited_output;
```

The required output is as shown below.

```
2018-09-13 22:59:34,091 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-13 22:59:34,091 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(101,Amitabh,20000)
(107,Salman,17500)
(103,Akshay,11000)
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)

- In relation **employee_details** and **employee_expenses**, we are loading the dataset.
- In relation **joined**, we are joining the employee_details and employee_expenses based on common data empId.
- In relation **ordered**, we are ordering the relation joined based on expenses and name.
- In relation **limited_output**, we are limiting the no of records to 1.
- In relation **generateoutput**, we are generating the result with empId and name.
- Finally using dump, we are printing the result.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~    acadgild@localhost:~
grunt> employee_details= LOAD '/employee_details.txt' using PigStorage(',') as (empId:int, name:chararray, salary:int, empRat
ing:int);
2018-09-13 23:04:30,239 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> employee_expenses = LOAD '/employee_expenses.txt' as (expenseempId:int, expenses:int);
2018-09-13 23:04:42,313 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> joined = JOIN employee_details BY empId, employee_expenses BY expenseempId;
grunt> ordered = ORDER joined BY expenses DESC, name;
grunt> limited_output = LIMIT ordered 1;
grunt> generateoutput = FOREACH limited_output GENERATE empId, name;
grunt> dump generateoutput;
```

The required output is as shown below.

```
2018-09-13 23:08:41,839 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-13 23:08:41,839 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(110,Priyanka)
grunt>
```

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

- In relation **employee_details** and **employee_expenses**, we are loading the dataset.
- In relation **joined**, we are joining the employee_details and employee_expenses based on common data empId.
- In relation **out**, we are generating the columns empId and name.
- In relation **Dist**, we are fetching unique records from relation out.
- Finally using dump, we are printing the result.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
grunt> employee_details= LOAD '/employee_details.txt' using PigStorage(',') as (empId:int, name:chararray, salary:int, empRat
ing:int);
2018-09-13 23:10:13,920 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> employee_expenses = LOAD '/employee_expenses.txt' as (expenseempId:int, expenses:int);
2018-09-13 23:10:24,982 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> joined = JOIN employee_details BY empId, employee_expenses BY expenseempId;
grunt> out = FOREACH joined GENERATE empId, name;
grunt> Dist = DISTINCT out;
grunt> dump Dist;
```

Required output is as shown below.

```
2018-09-13 23:12:54,761 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
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.

- In relation **employee_details** and **employee_expenses**, we are loading the dataset.
- In relation **joined**, we are joining the employee_details and employee_expenses based on common data empId.
- In relation **no_entries**, we are filtering the list of employees having no entries in employee_expenses file.
- In relation **out**, we are generating result with empId and name.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~    acadgild@localhost:~
grunt> employee_details= LOAD '/employee_details.txt' using PigStorage(',') as (empId:int, name:chararray, salary:int, empRat
ing:int);
2018-09-13 23:16:38,578 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> employee_expenses = LOAD '/employee_expenses.txt' as (expenseempId:int, expenses:int);
2018-09-13 23:16:48,870 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> joined = JOIN employee_details BY empId FULL, employee_expenses BY expenseempId;
grunt> no_entries = FILTER joined BY employee_expenses::expenseempId IS NULL;
grunt> out= FOREACH no_entries GENERATE empId, name;
grunt> dump out;
```

The required output is as shown below.

```
2018-09-13 23:18:39,812 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-13 23:18:39,812 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)
grunt>
```

3) Implementation of use case:

<u>Problem statement – 1</u>: Find out the top 5 most visited destinations.

- **In line 1**, we are registering the *piggybank* jar in order to use the CSVExcelStorage class.
- In relation **A**, we are loading the dataset using CSVExcelStorage because of its effective technique to handle double quotes and headers.
- In relation **B**, we are generating the columns with year, fligh_num,origin, and destination.
- In relation **C**, we are filtering the null from the destination column.
- In relation **D**, we are grouping relation **C** by "dest".
- In relation **E**, we are generating the grouped column and count of each.
- In relation **F** and **Result**, ordered the relation **E** and limited the result to top 5.
- In relation **A1**, we are loading another table to find the city as well as country.
- In relation **A2**, we are generating the dest, city, country from the relation **A1**.
- In relation **joined_table**, we are joining **Result** and **A2** based on a common column "dest".
- Finally, using dump, we are printing the result.

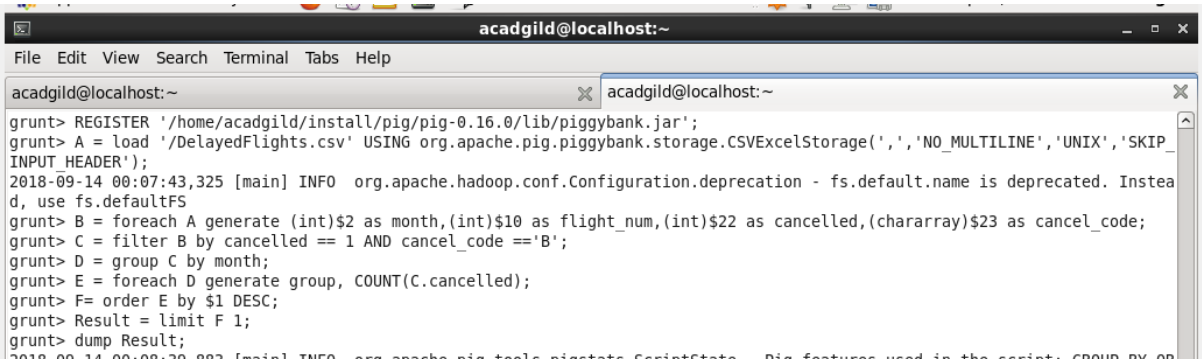
```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
grunt> REGISTER '/home/acadgild/install/pig/pig-0.16.0/lib/piggybank.jar';
2018-09-13 23:57:31,077 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, use fs.defaultFS
grunt> A = load '/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_
INPUT_HEADER');
2018-09-13 23:57:47,341 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, 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 '/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_INPUT
_HEADER');
2018-09-13 23:58:42,919 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, 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>
grunt> dump joined_table;
```

The required output is as shown below:

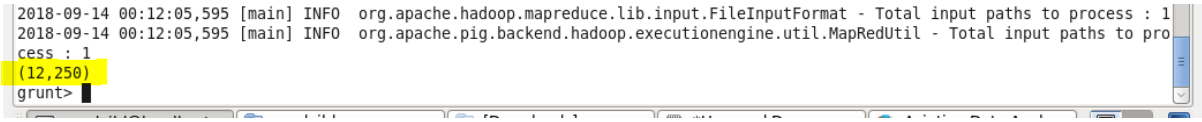
```
!
2018-09-14 00:03:47,974 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Key [pig.schematuple] was not set... will not g
enerate code.
2018-09-14 00:03:47,985 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-14 00:03:47,985 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 1
(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)
grunt>
```

<u>Problem statement – 2</u>: Which month has seen the most number of cancellations due to bad weather?

- **In Line 1**, we are registering *piggybank* jar in order to use the CSVExcelStorage class.
  In relation **A**, we are loading the dataset using CSVExcelStorage because of its effective technique to handle double quotes and header.
  In relation **B**, we are generating the columns which are required for processing and explicitly typecasting each of them.
  In relation **C**, we are filtering the data based on cancellation and cancellation code, i.e., cancelled = 1 means flight have been cancelled and cancel_code = 'B' means the reason for cancellation is "weather." So relation **C** will point to the data which consists of cancelled flights due to bad weather.
  In relation **D**, we are grouping the relation **C** based on every month.
  In relation **E**, we are finding the count of cancelled flights every month.
  Relation **F** and **Result** is for ordering and finding the top month based on cancellation.



The required output is as shown below.



<u>Problem statement – 3</u>: Top ten origins with the highest AVG departure delay.

- Explanation of first 3 lines is the same as explained in the previous 2 problem statements.
- In relation **C1**, we are removing the null values fields present if any.
- In relation **D1**, we are grouping the data based on column "origin."
- In relation **E1**, we are finding average delay from each unique origin.
- Relations named **Result** and **Top_ten** are ordering the results in descending order and printing the top ten values.
- In the relation **Lookup**, we are loading another table to which we will look up and find the city as well as the country.
- In the relation L**ookup1,** we are generating the destination, city, and country from the previous relation.
- In the relation **Joined**, we are joining relation Top_ten and Lookup1 based on common a column, i.e., "origin."
- In the relation **Final,** we are generating required columns from the Joined table.
- Finally, we are ordering and printing the results.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
grunt> REGISTER '/home/acadgild/install/pig/pig-0.16.0/lib/piggybank.jar';
grunt> A = load '/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_
INPUT_HEADER');
2018-09-14 00:15:10,647 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, 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 '/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_I
NPUT_HEADER');
2018-09-14 00:16:12,090 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, 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;
```

The required output is as shown below.

```
2018-09-14 00:23:37,744 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-14 00:23:37,744 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 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)
grunt>
*Unsaved Document 1 - gedit
```

<u>Problem statement – 4</u>: Which route (origin & destination) has seen the maximum diversion?

- **In Line 1**: We are registering *piggybank* jar in order to use CSVExcelStorage class.
- In relation **A**, we are loading the dataset using CSVExcelStorage because of its effective technique to handle double quotes and headers.
- In relation **B**, we are generating the columns which are required for processing and explicitly type-casting each of them.
- In relation **C**, we are filtering the data based on "not null" and diversion =1. This will remove the null records, if any, and give the data corresponding to the diversion taken.
- In relation **D**, we are grouping the data based on origin and destination.
- Relation **D** finds the count of diversion taken per unique origin and destination.
- Relations **F** and **Result** orders the result and produces top 10 results.

```
acadgild@localhost:~
File  Edit  View  Search  Terminal  Tabs  Help
acadgild@localhost:~        acadgild@localhost:~
grunt> REGISTER '/home/acadgild/install/pig/pig-0.16.0/lib/piggybank.jar';
grunt> A = load '/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_
INPUT_HEADER');
2018-09-14 00:26:17,616 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instea
d, 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;
```

The required output is as shown below.

```
2018-09-14 00:30:55,793 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-09-14 00:30:55,793 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to pro
cess : 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>
*Unsaved Document 1 - gedit
```