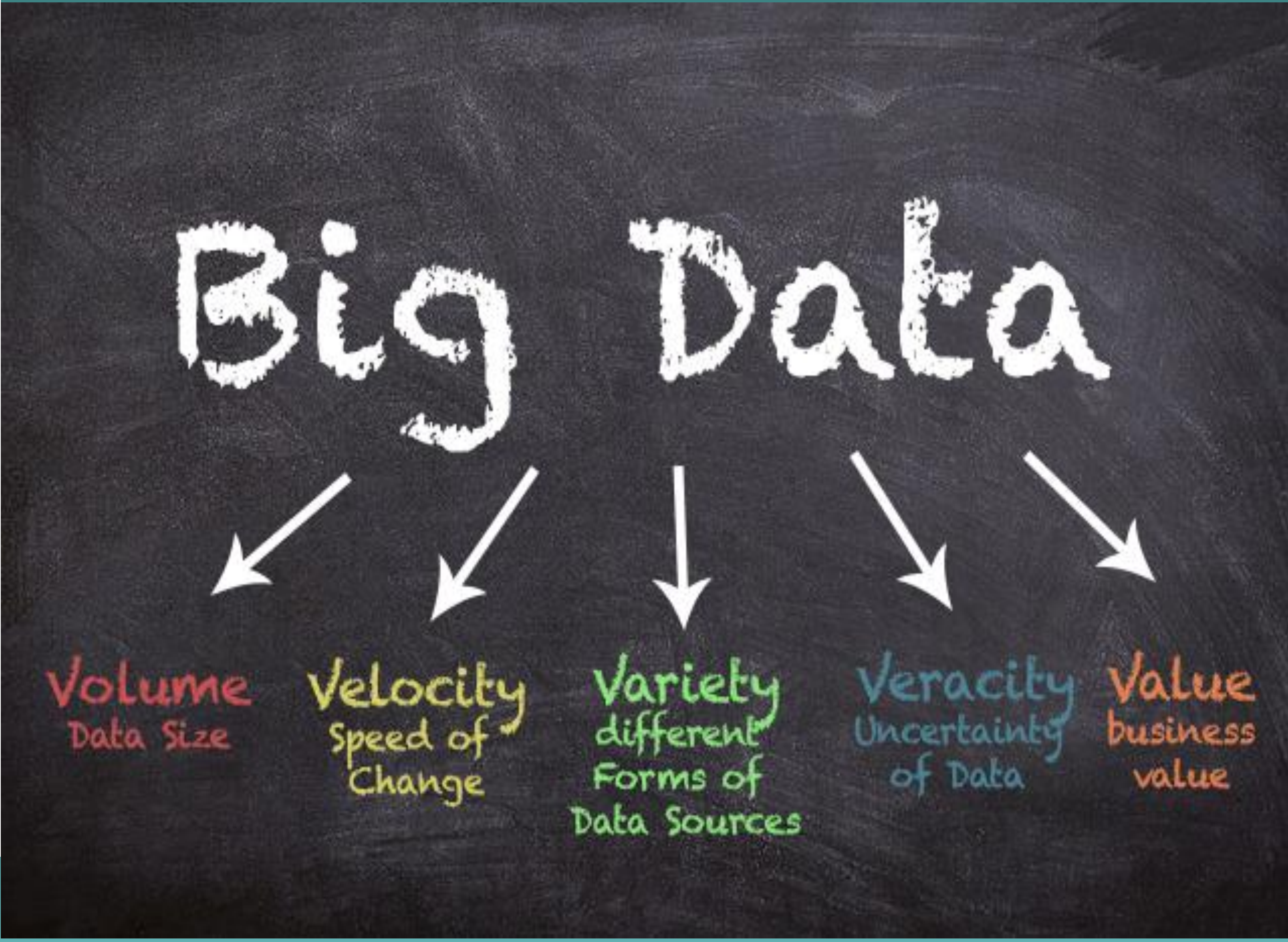
08/01/2018 | By: Ajit K Prasad



Big Data Engineering with Hadoop & Spark

Assignment on Exploring Apache Pig

Acadgild

Session 7: Assignment 7.1

This assignment is aimed at consolidating the concepts that was learnt during the Exploring Apache Pig session of the course.

Problem Statement

Task 1:

– Write a program to implement wordcount using Pig.

Solution:

– Along with other daemons, start ***JobHistoryServer daemon*** as it's a must for running Pig script, using the command

$ */home/acadgild/install/hadoop/hadoop-2.6.5/sbin/mr-jobhistory-daemon.sh start historyserver*

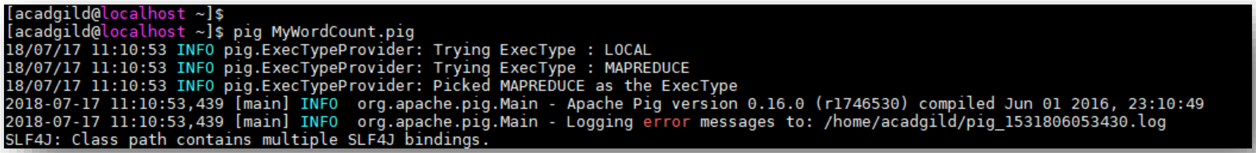
– A file named ***"file327.txt"*** was deployed on HDFS. As the name suggests, it was of ***327MB*** in size.

– Pig script was created and was saved on a file named: ***MyWordCount.pig***, given below is code of the script

```
$ lines = LOAD '/hadoopdata/wordcount/file327.txt' AS (line:chararray);
$ words = FOREACH lines GENERATE FLATTEN(TOKENIZE(line)) as word;
$ grouped = GROUP words BY word;
$ mywordcount = FOREACH grouped GENERATE group, COUNT(words);
$ DUMP mywordcount;
```

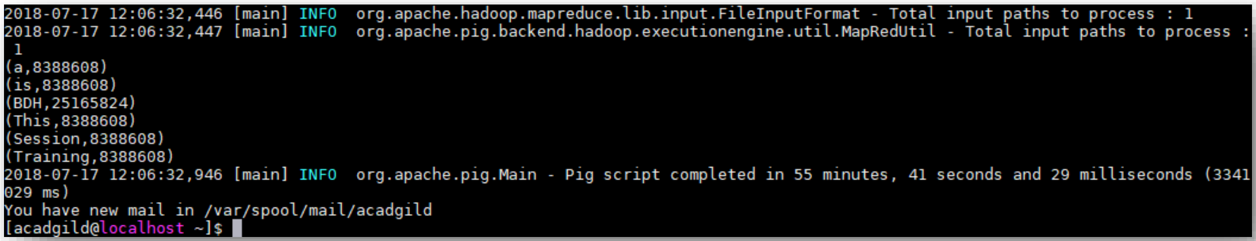
– Execute Pig Latin script from local terminal using the following command

```
$ pig MyWordCount.pig
```



```
[acadgild@localhost ~]$
[acadgild@localhost ~]$ pig MyWordCount.pig
18/07/17 11:10:53 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
18/07/17 11:10:53 INFO pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
18/07/17 11:10:53 INFO pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2018-07-17 11:10:53,439 [main] INFO  org.apache.pig.Main - Apache Pig version 0.16.0 (r1746530) compiled Jun 01 2016, 23:10:49
2018-07-17 11:10:53,439 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/acadgild/pig_1531806053430.log
SLF4J: Class path contains multiple SLF4J bindings.
```

– Output of the job



```
2018-07-17 12:06:32,446 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-07-17 12:06:32,447 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(a,8388608)
(is,8388608)
(BDH,25165824)
(This,8388608)
(Session,8388608)
(Training,8388608)
2018-07-17 12:06:32,946 [main] INFO  org.apache.pig.Main - Pig script completed in 55 minutes, 41 seconds and 29 milliseconds (3341
029 ms)
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost ~]$
```

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:
DataSets:
- 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*

Prerequisites:

```
$ hadoop fs -mkdir /hadoopdata/pig/EmployeeDatasets
$ hadoop fs -mkdir /hadoopdata/pig/EmployeeDatasets/Input
$ hadoop fs -put employee_details.txt /hadoopdata/pig/EmployeeDatasets/Input
$ hadoop fs -put employee_expenses.txt /hadoopdata/pig/EmployeeDatasets/Input
$ hadoop fs -cat /hadoopdata/pig/EmployeeDatasets/Input/employee_details.txt
$ hadoop fs -cat /hadoopdata/pig/EmployeeDatasets/Input/employee_expenses.txt
```

1. 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)

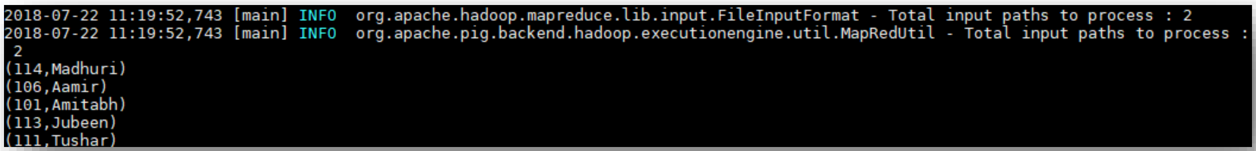
Job Execution:

– Created scripts was saved in a file **“Query1.pig”**

```
$ EmpDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_details.txt' USING PigStorage(',') AS(EmpID:int,name:chararray,salary:int,DepartmentID:int);
$ ExpenseDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_expenses.txt' USING PigStorage(' ') AS (id:int,expense:int);
$ ratingOrder = ORDER EmpDetails BY DepartmentID asc;
$ limit5 = limit ratingOrder 5;
```

$ *SPLIT limit5 into rating1 if DepartmentID == 1, rating2 if DepartmentID == 2;*
$ *orderedRating1 = ORDER rating1 by name asc;*
$ *Sol1 = UNION rating2,orderedRating1;*
$ *SolFinal = FOREACH Sol1 GENERATE EmpID,name;*
$ *dump SolFinal;*

– Command to run the script file
$ ***pig Query1.pig***



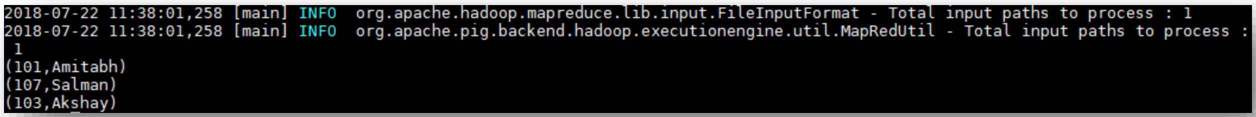
```
2018-07-22 11:19:52,743 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 2
2018-07-22 11:19:52,743 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 2
(114,Madhuri)
(106,Aamir)
(101,Amitabh)
(113,Jubeen)
(111,Tushar)
```

2. 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)

<u>Job Execution:</u>

– Created scripts was saved in a file **"Query2.pig"**
$ *EmpDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_details.txt' USING PigStorage(',') AS(EmpID:int,name:chararray,salary:int,DepartmentID:int);*
$ *ExpenseDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_expenses.txt ' USING PigStorage(' ') AS (id:int,expense:int);*
$ *Odd = FILTER EmpDetails BY (EmpID%2 != 0);*
$ *SalOrder = ORDER Odd BY salary desc;*
$ *Limit3 = limit SalOrder 3;*
$ *Sol2 = FOREACH Limit3 GENERATE EmpID,name;*
$ *dump Sol2;*

– Command to run the script file
$ ***pig Query2.pig***



```
2018-07-22 11:38:01,258 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-07-22 11:38:01,258 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(101,Amitabh)
(107,Salman)
(103,Akshay)
```

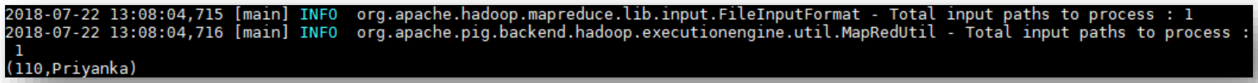
3. Employee (employee id and employee name) with maximum expense (In case two employees have same expense, employee with name coming first in dictionary should get preference)

Job Execution:

– Created scripts was saved in a file **"Query3.pig"**

$ EmpDetail = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_details.txt' USING PigStorage(',') AS (EmpID:int,name:chararray,salary:int,DepartmentID:int);
$ ExpenseDetail = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_expenses.txt ' USING PigStorage('\t') AS (id:int,expense:int);
$ JoinExpense = JOIN EmpDetail by EmpID, ExpenseDetail by id;
$ MaxExpense = ORDER JoinExpense by expense desc;
$ LimitMaxExpense = LIMIT MaxExpense 1;
$ FinalMaxExpense = FOREACH LimitMaxExpense GENERATE EmpID,name;
$ DUMP FinalMaxExpense;

– Command to run the script file
$ ***pig Query3.pig***



```
2018-07-22 13:08:04,715 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-07-22 13:08:04,716 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(110,Priyanka)
```

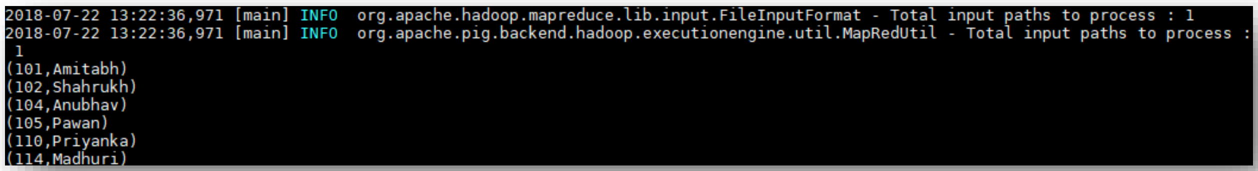
4. List of employees (employee id and employee name) having entries in ***"employee_expenses"*** file.

Job Execution:

– Created scripts was saved in a file **"Query4.pig"**

$ EmpDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_details.txt' USING PigStorage(',') AS(EmpID:int,name:chararray,salary:int,DepartmentID:int);
$ ExpenseDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_expenses.txt ' USING PigStorage('\t') AS (id:int,expense:int);
$ CommonExpense = JOIN EmpDetails by EmpID, ExpenseDetails by id;
$ ListExpense = FOREACH CommonExpense GENERATE EmpID,name;
$ FinalEntry = DISTINCT ListExpense;
$ DUMP FinalEntry;

– Command to run the script file
  $ ***pig Query4.pig***



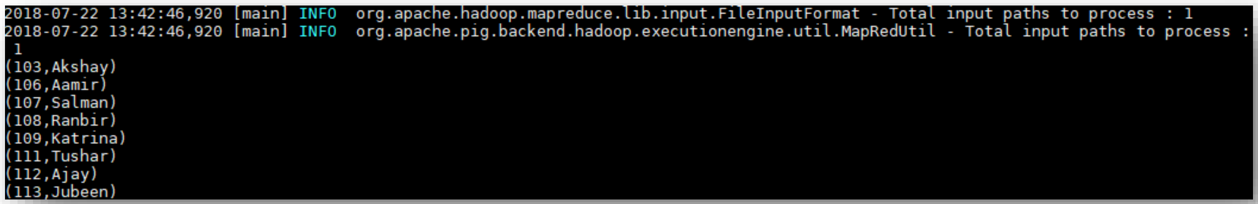
```
2018-07-22 13:22:36,971 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-07-22 13:22:36,971 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(101,Amitabh)
(102,Shahrukh)
(104,Anubhav)
(105,Pawan)
(110,Priyanka)
(114,Madhuri)
```

5. List of employees (employee id and employee name) having no entry in **“employee_expenses”** file.

<u>Job Execution:</u>

– Created scripts was saved in a file **“Query5.pig”**
  $ *EmpDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_details.txt' USING PigStorage(',') AS (EmpID:int,name:chararray,salary:int,DepartmentID:int);*
  $ *ExpenseDetails = LOAD '/hadoopdata/pig/EmployeeDatasets/Input/employee_expenses.txt ' USING PigStorage('\t') AS (id:int,expense:int);*
  $ *NonCommonEmp = JOIN EmpDetails by EmpID LEFT OUTER, ExpenseDetails by id;*
  $ *NoExpenseEntry = FILTER NonCommonEmp BY id is null;*
  $ *FinalNoEntry = FOREACH NoExpenseEntry GENERATE EmpID,name;*
  $ *DUMP FinalNoEntry;*

– Command to run the script file
  $ ***pig Query5.pig***



```
2018-07-22 13:42:46,920 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-07-22 13:42:46,920 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(103,Akshay)
(106,Aamir)
(107,Salman)
(108,Ranbir)
(109,Katrina)
(111,Tushar)
(112,Ajay)
(113,Jubeen)
```

Task 3:

– Implement the use case present in below blog link and share the complete steps along with screenshot(s) from your end.
https://acadgild.com/blog/aviation-data-analysis-using-apache-pig/

Prerequisites:

– Create a directory in the HDFS to copy the provided ***"csv"*** files.

```
$	hadoop fs -mkdir /hadoopdata/pig/AirlineDataAnalysis
```

– Copy the ***"csv"*** files from local to the HDFS directory.

```
$	hadoop fs -copyFromLocal /home/acadgild/Airline/*.csv
	/hadoopdata/pig/AirlineDataAnalysis/
```

– Check the HDFS directory to list its contents.

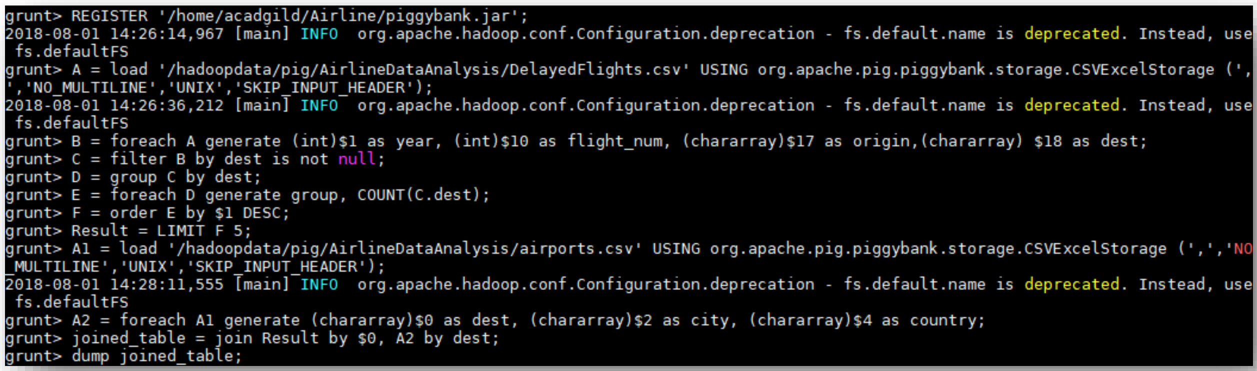
```
$	hadoop fs -ls /hadoopdata/pig/AirlineDataAnalysis/
```

Note: Required changes to the provided source codes were made to suit the user HDFS setup.

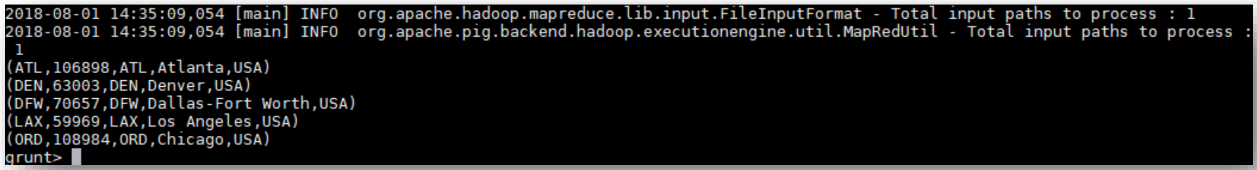
1. Find out the top 5 most visited destinations.

Source Code & Output:

```
$	REGISTER '/home/acadgild/Airline/piggybank.jar';
$	A = load '/hadoopdata/pig/AirlineDataAnalysis/DelayedFlights.csv'
	USING org.apache.pig.piggybank.storage.CSVExcelStorage
	(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
$	B = foreach A generate (int)$1 as year, (int)$10 as flight_num,
	(chararray)$17 as origin,(chararray) $18 as dest;
$	C = filter B by dest is not null;
$	D = group C by dest;
$	E = foreach D generate group, COUNT(C.dest);
$	F = order E by $1 DESC;
$	Result = LIMIT F 5;
$	A1 = load '/hadoopdata/pig/AirlineDataAnalysis/airports.csv'
	USING org.apache.pig.piggybank.storage.CSVExcelStorage
	(',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
$	A2 = foreach A1 generate (chararray)$0 as dest, (chararray)$2 as
	city, (chararray)$4 as country;
$	joined_table = join Result by $0, A2 by dest;
$	dump joined_table;
```



```
grunt> REGISTER '/home/acadgild/Airline/piggybank.jar';
2018-08-01 14:26:14,967 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use
 fs.defaultFS
grunt> A = load '/hadoopdata/pig/AirlineDataAnalysis/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage (',
','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-08-01 14:26:36,212 [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 '/hadoopdata/pig/AirlineDataAnalysis/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage (',','NO
_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-08-01 14:28:11,555 [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;
```

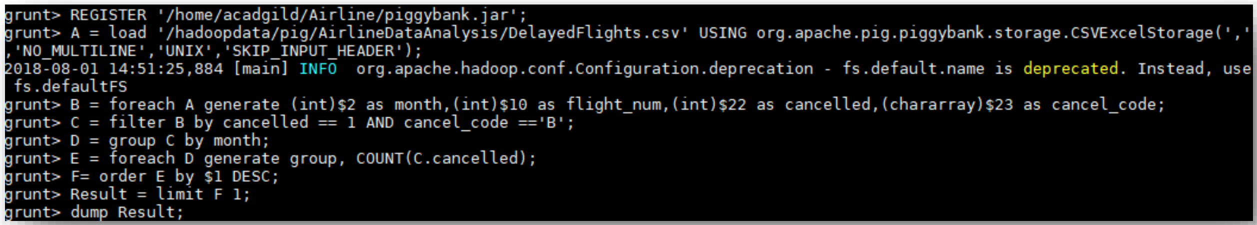


```
2018-08-01 14:35:09,054 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-08-01 14:35:09,054 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(ATL,106898,ATL,Atlanta,USA)
(DEN,63003,DEN,Denver,USA)
(DFW,70657,DFW,Dallas-Fort Worth,USA)
(LAX,59969,LAX,Los Angeles,USA)
(ORD,108984,ORD,Chicago,USA)
grunt>
```

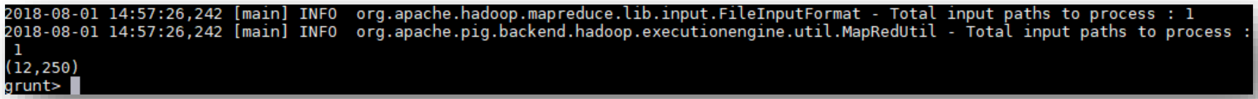
2. Which month has seen the most number of cancellations due to bad weather?

Source Code & Output:

$ REGISTER '/home/acadgild/Airline/piggybank.jar';
$ A = load '/hadoopdata/pig/AirlineDataAnalysis/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;



```
grunt> REGISTER '/home/acadgild/Airline/piggybank.jar';
grunt> A = load '/hadoopdata/pig/AirlineDataAnalysis/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(','
,'NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-08-01 14:51:25,884 [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;
```

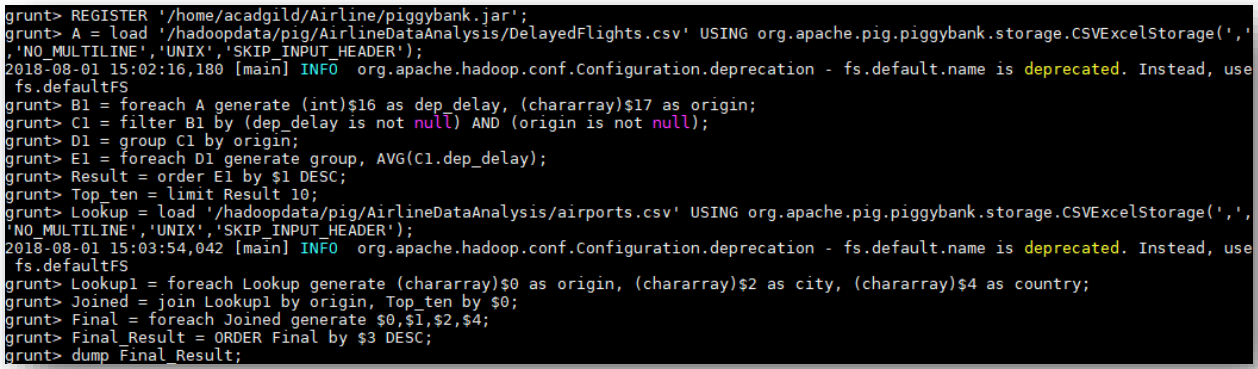


```
2018-08-01 14:57:26,242 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-08-01 14:57:26,242 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(12,250)
grunt> 
```

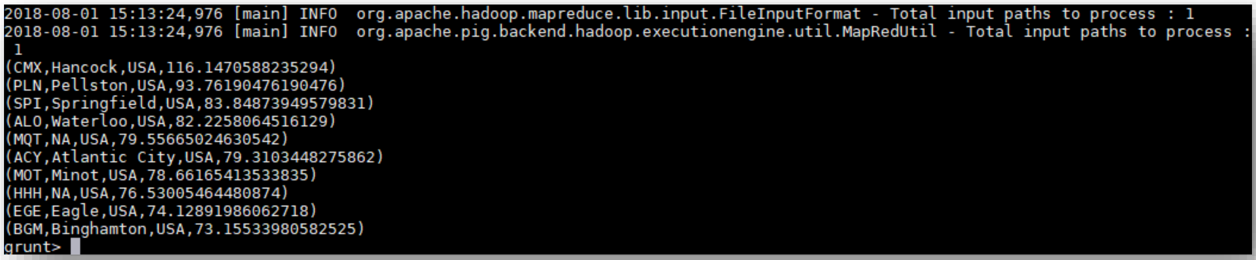
3. Top 10 origins with the highest AVG departure delay.

Source Code & Output:

$ *REGISTER '/home/acadgild/Airline/piggybank.jar';*
$ *A = load '/hadoopdata/pig/AirlineDataAnalysis/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 '/hadoopdata/pig/AirlineDataAnalysis/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;*



```
grunt> REGISTER '/home/acadgild/Airline/piggybank.jar';
grunt> A = load '/hadoopdata/pig/AirlineDataAnalysis/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(','
,'NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-08-01 15:02:16,180 [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 '/hadoopdata/pig/AirlineDataAnalysis/airports.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(',',
'NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-08-01 15:03:54,042 [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;
```

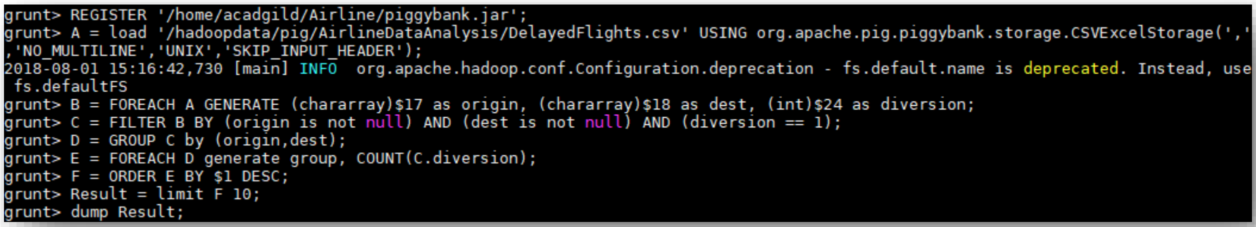


```
2018-08-01 15:13:24,976 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-08-01 15:13:24,976 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process :
 1
(CMX,Hancock,USA,116.1470588235294)
(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)
grunt>
```

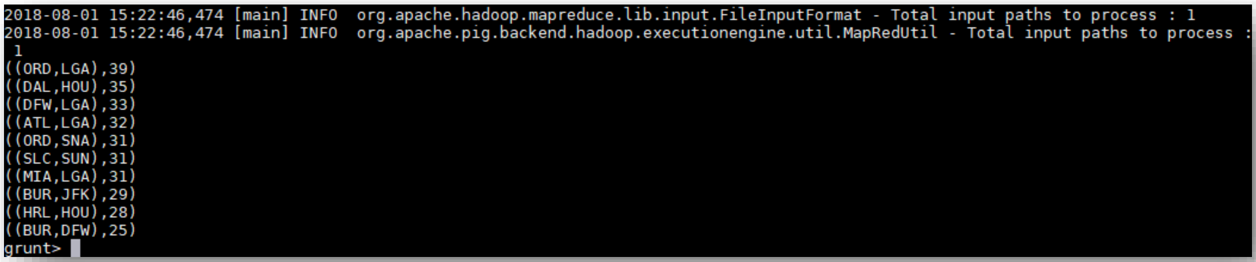
4. Which route (origin & destination) has seen the maximum diversion?

<u>Source Code & Output:</u>

$ *REGISTER '/home/acadgild/Airline/piggybank.jar';*
$ *A = load '/hadoopdata/pig/AirlineDataAnalysis/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage (',','NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');*
$ *B = FOREACH A GENERATE (chararray)$17 as origin, (chararray)$18 as dest, (int)$24 as diversion;*
$ *C = FILTER B BY (origin is not null) AND (dest is not null) AND (diversion == 1);*
$ *D = GROUP C by (origin,dest);*
$ *E = FOREACH D generate group, COUNT(C.diversion);*
$ *F = ORDER E BY $1 DESC;*
$ *Result = limit F 10;*
$ *dump Result;*



```
grunt> REGISTER '/home/acadgild/Airline/piggybank.jar';
grunt> A = load '/hadoopdata/pig/AirlineDataAnalysis/DelayedFlights.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(','
,'NO_MULTILINE','UNIX','SKIP_INPUT_HEADER');
2018-08-01 15:16:42,730 [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;
```



```
2018-08-01 15:22:46,474 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-08-01 15:22:46,474 [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>
```