Assignment 7 Apache Pig

Task 1

Write a program to implement wordcount using Pig.

File in HDFS

[acadgild@localhost pig]$ hdfs dfs -cat /pig/pigtestfile.txt

Hello Hello world

this is HDFS class

HDFS changes future of analytics

Pig is processing tool

To Run Pig from local

pig WordCount.pig

Pig script for word count

```
[acadgild@localhost pig]$ vi WordCount.pig
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost pig]$ cat WordCount.pig
a = load '/pig/pigtestfile.txt';
b = foreach a generate FLATTEN(TOKENIZE((chararray)$0)) as word;
c = group b by word;
d = foreach c generate group, COUNT(b);

dump d;

[acadgild@localhost pig]$ pig WordCount.pig
```

Output:

```
2018-12-23 14:09:05,080 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-12-23 14:09:05,081 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(is,2)
(of,1)
(Pig,1)
(HDFS,2)
(this,1)
(tool,1)
(Hello,2)
(class,1)
(world,1)
(future,1)
(changes,1)
(analytics,1)
(processing,1)
2018-12-23 14:09:06,065 [main] INFO  org.apache.pig.Main - Pig script completed in 1 minute, 21 seconds and 173 milliseconds (81173 ms)
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost pig]$
```

Task 2

Loading emp_det and emp_exp files.

```
grunt>
grunt> emp_det = LOAD '/home/acadgild/pig/employee_details.txt' using PigStorage(',') as (EmpID:Int, Name:chararray, Salary:Long, EmployeeRating:Int);
2018-12-30 23:14:18,280 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksu
m
2018-12-30 23:14:18,280 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> emp_exp = LOAD '/home/acadgild/pig/employee_expenses.txt' using PigStorage('\t') as (EmpID:Int, Expense:Int);
2018-12-30 23:14:24,962 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksu
m
2018-12-30 23:14:24,962 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
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)

Code

```
grunt> highestRating = order  emp_det by EmployeeRating DESC,Name;
grunt> highestRating_order = foreach highestRating generate (EmpID,Name);
grunt> FinalTopFive = LIMIT highestRating_order 5;
grunt> dump FinalTopFive;
```

Output

```
2018-12-30 20:08:03,002 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-12-30 20:08:03,003 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
((111,Tushar))
((113,Jubeen))
((101,Amitabh))
((106,Aamir))
((102,Shahrukh))
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)

Code

```
grunt> highestSalary = order emp_det by Salary DESC;
highestSalaryodd = FILTER highestSalary BY EmpID%2 != 0;
highestSalary_order = foreach highestSalaryodd generate (EmpID,Name);
FinalTopThree = LIMIT highestSalary_order 3;
dump FinalTopThree;
```

Output

```
2018-12-30 21:11:05,935 [main] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2018-12-30 21:11:05,950 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-12-30 21:11:05,950 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
((101,Amitabh))
((107,Salman))
((103,Akshay))
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)

Code

```
2018-12-30 22:18:22,648 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt> A = join emp_det by EmpID , emp_exp by EmpID;
grunt> B = ORDER A by Expense DESC,Name;
grunt> C = LIMIT B 1;
grunt>
ort MobaXterm by subscribing to the professional edition here:  https://mobaxterm.mobatek.net
```

Output

```
2018-12-30 22:19:25,986 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2018-12-30 22:19:25,986 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(110,Priyanka,2000,5,110,400)
grunt>
```

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

Code

```
grunt> joined = join emp_det by EmpID LEFT , emp_exp by EmpID;
grunt> entries = filter joined by $5 is not null;
grunt> entriesnameid = foreach entries generate $0,$1;
grunt>
```

Output

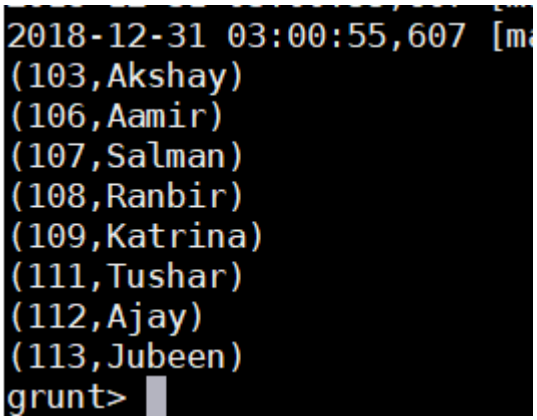
```
(101,Amitabh)
(101,Amitabh)
(102,Shahrukh)
(102,Shahrukh)
(104,Anubhav)
(105,Pawan)
(110,Priyanka)
(114,Madhuri)
```

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

Code

```
grunt>joined = join emp_det by EmpID FULL , emp_exp by EmpID;
grunt> entries = filter joined by $5 is null;
grunt> entriesnameid = foreach entries generate $0,$1;
grunt>
```

Output



Task 3

Aviation Data Analysis Using Apache Pig

There are no data sets are available for this task in the link

https://acadgild.com/blog/aviation-data-analysis-using-apache-pig So I am explaining the code here

There are two data sets Delayed_Flights and Airports in CSV format

Problem Statement 1

Find out the top 5 most visited destinations.

Source code

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_IN
PUT_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 '/home/acadgild/airline_usecase/airports.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_IN
PUT_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;
```

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 that are required for processing and explicitly typecasting each of them.
In relation C, we are filtering the null values from the “dest” column.
In relation D, we are grouping relation C by “dest.”
In relation E, we are generating the grouped column and the count of each.
Relation F and Result is used to order and limit the result to top 5.
These are the steps to find the top 5 most visited destinations. However, adding few more steps in this process, we will be using another table to find the city name and country as well.
In relation A1, we are loading another table to which we will look-up and find the city as well as the country.
In relation A2, we are generating dest, city, and country from the previous relation.
In relation joined_table, we are joining Result and A2 based on a common column, i.e., “dest”
Finally, using dump, we are printing the result.

Problem Statement 2

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

Source code

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_IN
PUT_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;
```

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., canceled = 1 means flight have been canceled and cancel_code = 'B' means the reason for cancellation is "weather." So relation C will point to the data which consists of canceled 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 canceled flights every month.
Relation F and Result is for ordering and finding the top month based on cancellation.

Problem Statement 3

Top ten origins with the highest AVG departure delay

Source code

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_IN
PUT_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/airline_usecase/airports.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_IN
PUT_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;
```

Explanation of first 3 lines are 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.
These steps are good enough to find the top ten origins with the highest average departure delay.
However, rather than generating just the code of origin, we will be following a few more steps to find some more details like country and city.
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 Lookup1, 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.

Problem Statement 4

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

Source code

```
REGISTER '/home/acadgild/airline_usecase/piggybank.jar';

A = load '/home/acadgild/airline_usecase/DelayedFlights.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',','NO_MULTILINE','UNIX','SKIP_IN
PUT_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;
```

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.