CSP554—Big Data Technologies

Assignment #5 (Modules 05)

Exercise 1) 2 points

Create new versions of the foodratings and foodplaces files by using TestDataGen (as described in assignment #4) and copy them to HDFS (say into /user/hadoop).

```
[hadoop@ip-172-31-6-108 ~]$ java TestDataGen
Magic Number = 22585
[hadoop@ip-172-31-6-108 ~]$
```

```
[hadoop@ip-172-31-6-108 ~]$ ls
foodplaces22585.txt  foodratings22585.txt  TestDataGen.class
[hadoop@ip-172-31-6-108 ~]$
```

```
Magic Number: 22585

hdfs dfs -copyFromLocal foodratings22585.txt /user/csp554;
hdfs dfs -copyFromLocal foodplaces22585.txt hdfs:///user/csp554;

food_ratings    =    LOAD    '/user/csp554/foodratings22585.txt'    USING
PigStorage(',')
AS
(name:chararray,
f1:int,
f2:int,
f3:int,
f4:int,
placeid:int);

Describe food_ratings;
```

```
grunt> describe food_ratings
food_ratings: {name: chararray,f1: int,f2: int,f3: int,f4: int,placeid: int}
grunt>
```

Exercise 2) 2 points

Now create another relation with two fields of the initial (food_ratings) relation: ‘name’ and ‘f4’. Call this relation ‘food_ratings_subset’.

Store this last relation, food_ratings_subset, back to HDFS (perhaps as the file /user/hadoop/fr_subset)

```
food_ratings_subset = foreach food_ratings generate name, f4;

store food_ratings_subset into '/home/hadoop/fr_subset' USING PigStorage(',');
```

```
Input(s):
Successfully read 1000 records (17508 bytes) from: "/user/csp554/foodratings22585.txt"

Output(s):
Successfully stored 1000 records (7022 bytes) in: "/home/hadoop/fr_subset"

grunt>
```

Also write 6 records of this relation out to the console.

```
six_food_ratings_subset = LIMIT food_ratings_subset 6;

dump six_food_ratings_subset;
```

```
22/10/05 19:55:40 INFO u
(Joy,7)
(Joy,16)
(Joy,47)
(Sam,10)
(Mel,43)
(Jill,19)
grunt>
```

Exercise 3) 2 points

Now create another relation using the initial (food_ratings) relation. Call this relation 'food_ratings_profile'. The new relation should only have one record. This record should hold the minimum, maximum and average values for the attributes 'f2' and 'f3'. (So this one record will have 6 fileds).

Write the record of this relation out to the console.

```
food_ratings_profile = FOREACH (GROUP food_ratings ALL) GENERATE
MIN(food_ratings.f2), MAX(food_ratings.f2), AVG(food_ratings.f2),
MIN(food_ratings.f3), MAX(food_ratings.f3),
AVG(food_ratings.f3);

Describe food_ratings_profile;

DUMP food_ratings_profile;
```

```
grunt> describe food_ratings_profile
food_ratings_profile: {int,int,double,int,int,double}
grunt>
```

```
      HadoopVersion: 2.10.1-amzn-4
         PigVersion: 0.17.0
         TezVersion: 0.9.2
             UserId: hadoop
           FileName:
          StartedAt: 2022-10-05 20:00:32
         FinishedAt: 2022-10-05 20:00:47
           Features: GROUP_BY

Success!


DAG 0:
                                Name: PigLatin:DefaultJobName-0_scope-2
                       ApplicationId: job_1664998331522_0002
                   TotalLaunchedTasks: 2
                       FileBytesRead: 151
                    FileBytesWritten: 87
                       HdfsBytesRead: 17508
                    HdfsBytesWritten: 28
     SpillableMemoryManager spill count: 0
              Bags proactively spilled: 0
            Records proactively spilled: 0

DAG Plan:
Tez vertex scope-70     ->      Tez vertex scope-71,
Tez vertex scope-71

Vertex Stats:
VertexId Parallelism TotalTasks   InputRecords   ReduceInputRecords  OutputRecords  FileBytesRead FileBytesWritten  HdfsBytesRead HdfsBytesWritten Alias        Feature Outp
uts
scope-70          1          1          1000                   0          1000           64              87          17508              0 1-38,food_ratings,food_ra
tings_profile
scope-71          1          1             0                1000             1           87               0              0             28 food_ratings_profile GROU
P_BY    hdfs://ip-172-31-6-108.ec2.internal:8020/tmp/temp-1629842855/tmp-918684770,

Input(s):
Successfully read 1000 records (17508 bytes) from: "/user/csp554/foodratings22585.txt"

Output(s):
Successfully stored 1 records (28 bytes) in: "hdfs://ip-172-31-6-108.ec2.internal:8020/tmp/temp-1629842855/tmp-918684770"

22/10/05 20:00:47 INFO input.FileInputFormat: Total input files to process : 1
486696 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil  - Total input paths to process : 1
22/10/05 20:00:47 INFO util.MapRedUtil: Total input paths to process : 1
(1,50,25.244,1,50,26.495)
grunt>
```

```
(1,50,25.244,1,50,26.495)
grunt>
```

Exercise 4) 2 points

Now create yet another relation from the initial (food_ratings) relation. This new relation should only include tuples (records) where f1 < 20 and f3 > 5. Call this relation 'food_ratings_filtered'.

Write 6 records of this relation out to the console.

```
food_ratings_filtered = FILTER food_ratings BY (f1 < 20) AND (f3 > 5);

food_ratings_filtered_6 = LIMIT food_ratings_filtered 6;

DESCRIBE food_ratings_filtered_6;

DUMP food_ratings_filtered_6;
```

```
grunt> DESCRIBE food_ratings_filtered_6;
food_ratings_filtered_6: {name: chararray,f1: int,f2: int,f3: int,f4: int,placeid: int}
grunt>
```

```
Output(s):
Successfully stored 6 records (117 bytes) in: "hdfs://ip-172-31-6-108.ec2.internal:8020/tmp/temp-1629842855/tmp574601619"

22/10/05 20:04:58 INFO input.FileInputFormat: Total input files to process : 1
737408 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil  - Total input paths to process : 1
22/10/05 20:04:58 INFO util.MapRedUtil: Total input paths to process : 1
(Joy,3,4,23,16,5)
(Sam,3,24,17,10,5)
(Mel,5,5,47,43,4)
(Joy,9,41,29,1,3)
(Joe,1,24,18,44,2)
(Joe,17,11,45,34,1)
grunt>
```

Exercise 5) 2 points

Using the initial (food_ratings) relation, write and execute a sequence of pig latin statements that creates

another relation, call it ‘food_ratings_2percent’, holding a random selection of 2% of the records in the initial relation.

Write 10 of the records out to the console.

```
food_ratings_2percent = SAMPLE food_ratings 0.02;

describe food_ratings_2percent;
```

```
grunt> describe food_ratings_2percent;
food_ratings_2percent: {name: chararray,f1: int,f2: int,f3: int,f4: int,placeid: int}
grunt>
```

```
food_ratings_2percent_10 = LIMIT food_ratings_2percent 10;

describe food_ratings_2percent_10;

dump food_ratings_2pecent_10;
```

```
Input(s):
Successfully read 432 records (17508 bytes) from: "/user/csp554/foodratings22585.txt"

Output(s):
Successfully stored 10 records (201 bytes) in: "hdfs://ip-172-31-6-108.ec2.internal:8020/tmp/temp-1629842855/tmp-1295801858"

22/10/05 20:10:41 INFO input.FileInputFormat: Total input files to process : 1
1080280 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil  - Total input paths to process : 1
22/10/05 20:10:41 INFO util.MapRedUtil: Total input paths to process : 1
(Joe,20,37,14,4,2)
(Joy,23,43,49,35,4)
(Jill,36,22,36,14,5)
(Joy,43,22,3,17,4)
(Mel,5,37,45,16,3)
(Sam,26,6,46,44,5)
(Mel,2,21,47,2,5)
(Joe,44,41,5,22,5)
(Joe,19,17,33,30,4)
(Sam,5,22,47,18,2)
grunt>
```

Exercise 6) 2 points

Write and execute a sequence of pig latin statements that loads the foodplaces file as a relation. Call the relation ‘food_places’. The load command should associate a schema with this relation where the first attribute is referred to as ‘placeid’ and is of type int and the second attribute is referred to as ‘placename’ and is of type chararray.

Execute the describe command on this relation.

Now perform a join between the initial place_ratings relation and the food_places relation on the placeid attributes to create a new relation called ‘food_ratings_w_place_names’. This new relation should have all the attributes (columns) of both relations. The new relation will allow us to work with place ratings and place names together.

Write 6 records of this relation out to the console.

```
food_places = LOAD '/user/csp554/foodplaces22585.txt' USING PigStorage(',') AS (placeid:int, placename:chararray);
```

```
food_ratings_w_place_names = join food_ratings by placeid, food_places
by placeid;

food_ratings_w_place_names_6 = LIMIT food_ratings_w_place_names 6;

describe food_ratings_w_place_names_6;

dump food_ratings_w_place_names_6;
```

```
grunt> describe food_places
food_places: {placeid: int,placename: chararray}
grunt>
```

```
grunt> describe food_ratings_w_place_names_6;
food_ratings_w_place_names_6: {food_ratings::name: chararray,food_ratings::f1: int,food_ratings::f2: int,food_ratings::f3: int,food_ratings::f4: int,food_ratings::placeid:
int,food_places::placeid: int,food_places::placename: chararray}
grunt>
```

```
Input(s):
Successfully read 1000 records (17508 bytes) from: "/user/csp554/foodratings22585.txt"
Successfully read 5 records (59 bytes) from: "/user/csp554/foodplaces22585.txt"

Output(s):
Successfully stored 6 records (213 bytes) in: "hdfs://ip-172-31-6-108.ec2.internal:8020/tmp/temp-1629842855/tmp277443011"

22/10/05 20:24:23 INFO input.FileInputFormat: Total input files to process : 1
1902469 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil  - Total input paths to process : 1
22/10/05 20:24:23 INFO util.MapRedUtil: Total input paths to process : 1
(Joy,48,29,41,10,1,1,China Bistro)
(Joy,19,19,32,30,1,1,China Bistro)
(Jill,22,17,26,39,1,1,China Bistro)
(Jill,3,34,24,45,1,1,China Bistro)
(Mel,5,7,4,26,1,1,China Bistro)
(Jill,29,11,43,17,1,1,China Bistro)
grunt>
```

Exercise 7) (3 points) Identify the one correct answer for each the following questions. These questions are similar to the ones you might find on the mid-term covering Pig. Each is worth ½ point.

I. Which keyword is used to select a certain number of rows from a relation when forming a new relation?

Answer: LIMIT

Choices:

A. LIMIT
B. DISTINCT
C. UNIQUE
D. SAMPLE

II. Which keyword returns only unique rows for a relation when forming a new relation?

Choices:

Answer: DISTINCT

A. SAMPLE
B. FILTER
C. DISTINCT

D. SPLIT

III. Assume you have an HDFS file with a large number of records similar to the examples below

- Mel, 1, 2, 3
- Jill, 3, 4, 5

Which of the following would NOT be a correct pig schema for such a file?

Choices:

Answer: (f1: STRING, f: INT, f3: INT, f4: INT)

A. (f1: CHARARRY, f2: INT, f3: INT, f4: INT)
B. (f1: STRING, f2: INT, f3: INT, f4: INT)
C. (f1, f2, f3, f4)
D. (f1: BYTEARRAY, f2: INT, f3: BYTEARRAY, f4: INT)

IV. Which one of the following statements would create a relation (relB) with two columns from a relation (relA) with 4 columns? Assume the pig schema for relA is as follows:

(f1: INT, f2, f3, f4: FLOAT)

Answer: relB = FOREACH relA GENERATE $0, f3;

Choices:

A. relB = GROUP relA GENERATE f1, f3;
B. relB = FOREACH relA GENERATE $0, f3;
C. relB = FOREACH relA GENERATE f1, f5;
D. relB = FOREACH relA SELECT f1, f3;

V. Pig Latin is a **dataflow** language. Select the best choice to fill in the blank.

Choices:

A. functional
B. data flow
C. procedural
D. declarative

VI. Given a relation (relA) with 4 columns and pig schema as follows: (f1: INT, f2, f3, f4: FLOAT) which one statement will create a relation (relB) having records all of whose first field is less than 20

Answer: relB = FILTER relA by $0 < 20

Choices:

A. relB = FILTER relA by $0 < 20
B. relB = GROUP relA by f1 < 20
C. relB = FILTER relA by $1 < 20
D. relB = FOREACH relA GENERATE f1 < 20