Caleb Post
Hadoop Lab 5

1. Pig Introduced the notion of a macro starting with Pig 0.9.1. What is a macro, explain it with an example.

A macro is a way that a user can modularize their pig scripts to be able to reuse the same function that needs to be applied over and over, or to simply make the code read better. It functions similarly to a user defined function but is written in pig script instead of another language. An example taken from http://hortonworks.com/blog/new-apache-pig-features-part-1-macro/ shows how this can be done.

Lets say we have a pig script that looks like this:

```
A = load "student.txt" as (name, student, gpa);
B = group A all;
C = foreach B generate COUNT(A);
DUMP C
D = load "students2.txt" as (name, student, gpa);
E = group D all;
F = foreach E generate COUNT(D);
DUMP F
```

We are doing two different counting operations which are in fact the exact same, leaving us with duplicated script lines. If we were to define a macro however in a separate file, or in the same file, which read

countMacro.macro

```
DEFINE row_count(X) RETURNS Z {Y = group $X all; $Z = foreach Y
generate COUNT($X);};
```

When we have defined this macro, not only do we eliminate the duplicate code from the first pig script, but we can reuse it in future pig scripts. The script above becomes

```
Import "countMacro.macro"
A = load "student.txt" as (name, student, gpa);
B = row_count(A);
DUMP B;
C = load "students2.txt" as (name, student, gpa);
D = row_count(C);
DUMP D;
```

2. Explain the following commands/functions with an example.

    a. COGROUP

COGROUP in pig functions in the capacity of both a grouping and a join. It takes the functionality of the grouping function which creates bags of records which are grouped by the same group value, and then extends this into joining it to the bags from another table which have the same group value.

Say we have a data set of people who own pets.

```
Ed, fish
Caleb, cat
Yvonne, cat
John, dog
Kyle, frog
Charlie, fish
```

Also, say we have a data set of pets with their names.

```
Bob, fish
Jim, fish
Paws, cat
Tuffy, dog
Billy, frog
```

And, if we were to load these into two different tables and run a COGROUP on the type of animal we would get these results

GROUP	OWNER	ANIMAL
fish	{(Ed, fish),(Charlie,fish)}	{(Bob, fish),(Jim, fish)}
cat	{(Caleb, cat),(Yvonne, cat)}	{(Paws, cat)}
frog	{(Kyle, frog)}	{(Billy, frog)}
dog	{(John, dog)}	{(Tuffy, dog)}

Which we can see provides both the grouping function, getting the bags from the tables, and the join function, joining the bags from the two separate tables.

b. RANK

RANK is a pig command which provides an additional element in the table which lists the order of the element. It can also work with additional ordering properties. This means that you can provide the sort order as well as getting the rank of each element.
Taking this example from pig.apache.org/docs/r0.11.0/basic.html

```
A = load 'data' AS (f1:chararray,f2:int,f3:chararray);

DUMP A;
(David,1,N)
(Tete,2,N)
(Ranjit,3,M)
(Ranjit,3,P)
(David,4,Q)
(David,4,Q)
(Jillian,8,Q)
(JaePak,7,Q)
(Michael,8,T)
(Jillian,8,Q)
(Jose,10,V)
```

Then, if we run the rank command on A we get these results.

```
B = rank A;

dump B;
(1,David,1,N)
(2,Tete,2,N)
(3,Ranjit,3,M)
(4,Ranjit,3,P)
(5,David,4,Q)
(6,David,4,Q)
(7,Jillian,8,Q)
(8,JaePak,7,Q)
(9,Michael,8,T)
(10,Jillian,8,Q)
(11,Jose,10,V)
```

However, if we wish to specify a different sort order we can get these results

```
C = rank A by f1 DESC, f2 ASC;

dump C;
(1,Tete,2,N)
(2,Ranjit,3,M)
(2,Ranjit,3,P)
(4,Michael,8,T)
(5,Jose,10,V)
(6,Jillian,8,Q)
(6,Jillian,8,Q)
(8,JaePak,7,Q)
(9,David,1,N)
(10,David,4,Q)
(10,David,4,Q)
```

c. STREAM

The STREAM operator allows us to send Pig data through to an external program or script. This functions similarly to piping in the Unix operating system. For instance if we run the command

```
A = LOAD 'data.txt';
B = STREAM A THROUGH `stream.pl -n 5`
```

Then the data in the file data.txt will be run through the perl script stream and then put back into pig.