Zhihao Xue
CM 3357
Lab5
CSSE490 Dr. Mohan
Fri,Oct 16, 2015

- Introduce notion of am macro.
  Pig Lation supports the definition, expansion, and import of macros.

  – Macro Definition
  A macro definition can appear anywhere in a Pig script as long as it appears prior to the first use. A macro definition can include references to other macros as long as the referenced macros are defined prior to the macro definition. Recursive references are not allowed.
  The following example the macro is named my_macro:

```
DEFINE my_macro(A, sortkey) RETURNS C {
     B = FILTER $A BY my_filter(*);
     $C = ORDER B BY $sortkey;
}
```

  – Macro Expansion
  A macro can be expanded inline using the macro expansion syntax. Note the following:

    * Any alias in the macro which isn't visible from the outside will be prefixed with a macro name and suffixed with an instance id to avoid namespace collision.
    * Macro expansion is not a complete replacement for function calls. Recursive expansions are not supported.

  In this example my_macro (defined above) is expanded. Because alias B is not visible from the outside it is renamed macro_my_macro_B_0.

```
/* These statements ... */

X = LOAD 'users' AS (user, address, phone);
Y = my_macro(X, user);
STORE Y into 'bar';

/* Are expanded into these statements ... */

X = LOAD 'users' AS (user, address, phone);
macro_my_macro_B_0 = FILTER X BY my_filter(*);
Y = ORDER macro_my_macro_B_0 BY user;
STORE Y INTO 'output';
```

  – Macro Import
  A macro can be imported from another Pig script. Splitting your macros from

your main Pig script is useful for making reusable code. In this example no parameters are passed to the macro.

```
DEFINE my_macro() returns B {
   D = LOAD 'data' AS (a0:int, a1:int, a2:int);
   $B = FILTER D BY ($1 == 8) OR (NOT ($0+$2 > $1));
};

X = my_macro();
STORE X INTO 'output';
```

- Explain the following commands/functions with an example

  - COGROUP
    The GROUP and COGROUP operators are identical. Both operators work with one or more relations. For readability GROUP is used in statements involving one relation and COGROUP is used in statements involving two or more relations. You can COGROUP up to but no more than 127 relations at a time.
    Examples:
    Suppose we have relations A and B.

```
A = LOAD 'data1' AS (owner:chararray,pet:chararray);

DUMP A;
(Alice,turtle)
(Alice,goldfish)
(Alice,cat)
(Bob,dog)
(Bob,cat)

B = LOAD 'data2' AS (friend1:chararray,friend2:chararray);

DUMP B;
(Cindy,Alice)
(Mark,Alice)
(Paul,Bob)
(Paul,Jane)

X = COGROUP A BY owner, B BY friend2;

DESCRIBE X;
```

```
X: {group: chararray,A: {owner: chararray,pet: chararray},B: {friend1:
    chararray,friend2: chararray}}

DUMP X;
(Alice,{(Alice,turtle),(Alice,goldfish),(Alice,cat)},{(Cindy,Alice),(Mark,Alice)})
(Bob,{(Bob,dog),(Bob,cat)},{(Paul,Bob)})
(Jane,{},{(Paul,Jane)})
```

– RANK

Returns each tuple with the rank within a relation.

When specifying no field to sort on, the RANK operator simply prepends a sequential value to each tuple.

Otherwise, the RANK operator uses each field (or set of fields) to sort the relation. The rank of a tuple is one plus the number of different rank values preceding it. If two or more tuples tie on the sorting field values, they will receive the same rank.

Example:

Suppose we have relation A.

```
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)
```

In this example, the RANK operator does not change the order of the relation and simply prepends to each tuple a sequential value.

```
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)
```

In this example, the RANK operator works with f1 and f2 fields, and each one with different sorting order. RANK sorts the relation on these fields and prepends the rank value to each tuple. Otherwise, the RANK operator uses each field (or set of fields) to sort the relation. The rank of a tuple is one plus the number of different rank values preceding it. If two or more tuples tie on the sorting field values, they will receive the same rank.

```
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)
```

– STREAM

Sends data to an external script or program.

Use the STREAM operator to send data through an external script or program. Multiple stream operators can appear in the same Pig script. The stream operators can be adjacent to each other or have other operations in between.

When used with a command, a stream statement could look like this:

```
A = LOAD 'data';

B = STREAM A THROUGH `stream.pl -n 5`;
```

When used with a cmd_alias, a stream statement could look like this, where mycmd is the defined alias.

```
A = LOAD 'data';

DEFINE mycmd `stream.pl ?n 5`;

B = STREAM A THROUGH mycmd;
```