Pig Installation and Working

1. Download the pig software version 0.15.0 from apache.org into Downloads folder.
2. Extract pig software and rename it as Pig.
3. now we have to set the path for the pig installation

for that type in terminal as

$sudo gedit .bashrc

It will open one file and there at the end of the .bashrc file add these lines at end

export PIG\_HOME=/home/user/Downloads/pig-0.15.0

export PATH=$PATH:/home/user/Downloads/pig-0.15.0/bin

export PIG\_CLASSPATH=$HADOOP\_HOME/conf/

export PATH=$PATH:$PIG\_HOME/bin

After adding save the .bashrc file and then close the terminal one time and then open again.

1. Now type in the terminal as

$pig -version

then it will display as

cselab@cselab-lenovo:~$ pig -version

Apache Pig version 0.15.0 (r1682971)

compiled Jun 01 2015, 11:44:35

1. First you have to start all the nodes present in the hadoop. For that type in terminal as

cselab@cselab-lenovo:~$start-all.sh

1. Whether the nodes are started are not you have to test in the terminal by typing in

cselab@cselab-lenovo:~$jps

1. after that you have to create file in the current directory

for that follow the steps

~$gedit sample.txt

then you enter some name like this given below in sample.txt

1,rafi,guntur

2,riz,nrt

3,ria,hyd

after that save and exit the file

1. we have to create a directory in hadoop file system type the follwing in terminal as

~$: hadoop fs -mkdir /Pig\_Data

1. Now verify dir is created or not for that go to mozilla firefox and type there

localhost:50070-> in that utilities there u can check.

1. Now move the sample.txt file from current directory to hadoop directory for that purpose we have to do like this in terminal

~$hadoop fs -put /home/cselab/sample.txt /Pig\_Data/sample.txt

1. Now the file is shown in the Pig\_data directory
2. now we have to do some pig scripts for that purpose we use to execute type in terminal as

pig -x mapreduce

then it will enter in to grunt>

1. There we use LOAD statement for executing the written script

grunt> RELATION = LOAD 'hdfs://localhost:9000/Pig\_Data/sample.txt' USING PigStorage(',') as (id:int,name:chararray,city:chararray);

1. To view the output in active shell use the following cmd

Dump relation;

1. if you want to see output for that purpose we have to create one output folder in hadoop file system for that purpose the follwoing command is useful with store command

grunt> store su\_data into 'hdfs://localhost:9000/Pig\_Out/' using PigStorage(',');

1. come out from the grunt shell as

grunt>control+D

Diagnostic operators : There are mainly two types of diagnostic operator

1.Dump

2.Describe

3.illustrate

4.explain

now I would like to use the dump operator

1. Dump Operator :

The Dump operator is used to run the Pig Latin statements and display the results on

The screen. It is generally used for debugging Purpose.

grunt>dump du\_data;

1. Describe: The describe operator is used to view the schema of a relation afterwards I would like to describe the statement in which form it is stored. For that we use describe mode

grunt> describe RELATION

GROUP ING OPERATORS

The group operator is used to group the data in one or more relations. It collects the

data

having the same key.

grunt> group\_data = GROUP du\_data by id;

Grouping by Multiple Columns

grunt> group\_multiple = GROUP du\_data by (id, city);

Group All

You can group a relation by all the columns as shown below.

grunt> group\_all = GROUP du\_data All;

cogroup

The cogroup operator works more or less in the same way as the group operator. The

only difference between the two operators is that the group operator is normally used

with one relation, while the cogroup operator is used in statements involving two or

more

relations.

Grouping Two Relations using Cogroup

Assume that we have two files namely student\_details.txt and employee\_details.txt

in the HDFS directory /pig\_data/ as shown below.

~$gedit emp.txt

1,smr,gnt

2,em,hyd

3,mcp,nrt

after that this emp.txt file should be moved to hadoop file system by the follwing

command in the terminal as

~$hadoop fs -put /home/user/emp.txt /Pig\_Data/emp.txt

now this file would be moved to the hadoop file system and u have to see in the

browser also.

After that load two files in the grunt shell

student\_details = LOAD 'hdfs://localhost:9000/Pig\_data/student\_details.txt'

USING PigStorage(',')as (id:int, firstname:chararray, lastname:chararray,

age:int, phone:chararray, city:chararray);

employee\_details = LOAD 'hdfs://localhost:9000/Pig\_data/employee\_details.txt'

USING PigStorage(',')as (id:int, name:chararray, age:int, city:chararray);

-> For this the code for cogroup is as follows

grunt> cogroup\_data = cogroup su\_da by name, em\_da by city;

grunt> dump cogroup\_data;

grunt> co\_group = cogroup su\_da by 3, em\_da by 1;

**Join**

The join operator is used to combine records from two or more relations. While Performing a join operation, we declare one (or a group of) tuple(s) from each relation, as keys. When these keys match, the two particular tuples are matched, else the records are dropped.

Joins can be of the following types:

 Self-join

 Inner-join

 Outer-join : left join, right join, and full join

Inner Join

Inner Join is used quite frequently; it is also referred to as equijoin. An inner join returns rows when there is a match in both tables. It creates a new relation by combining column values of two relations (say A and B) based upon the join-predicate. The query compares each row of A with each row of B to find all pairs of rows which satisfy the join-predicate. When the join-predicate is satisfied, the column values for each matched pair of rows of A and B are combined into a result row.

Now i would like to perfomr en exmaple given below

grunt> jo\_data = join su\_data by id, du\_data by id;

grunt> dump jo\_data;

then it produces the following output

(1,rafi,guntur,1,rah,nrt)

(2,riz,narasaraopet,2,rev,gnt)

(3,ria,hyderabad,3,mcp,narasa)

(4,amaan,vijayawada,4,ant,hyd)

(5,ayaan,banglore,5,mem,ban)

Self - join

Self-join is used to join a table with itself as if the table were two relations,

temporarily

renaming at least one relation.

Generally, in Apache Pig, to perform self-join, we will load the same data multiple

times,

under different aliases (names).

grunt> jo\_data1 = join su\_data by name, du\_data by name;

grunt> dump jo\_data1;

**Outer Join**

Unlike inner join, outer join returns all the rows from at least one of the relations. An

outer join operation is carried out in three ways –

 Left outer join

 Right outer join

 Full outer join

Left Outer Join

The left outer Join operation returns all rows from the left table, even if there are no

matches in the right relation.

grunt> out\_left = join su\_data by name left outer, du\_data by name;

grunt> dump out\_left;

out put

(2,riz,narasaraopet,,,)

(4,amaan,vijayawada,,,)

(1,rafi,guntur,,,)

(3,ria,hyderabad,,,)

(5,ayaan,banglore,,,)

grunt>

Right Outer Join

The right outer join operation returns all rows from the right table, even if there are

no matches in the left table.

grunt> out\_right = join su\_data by name right, du\_data by name;

grunt> dump out\_right;

**output**

(,,,4,ant,hyd)

(,,,3,mcp,narasa)

(,,,5,mem,ban)

(,,,1,rah,nrt)

(,,,2,rev,gnt)

grunt>

Full Outer Join

The full outer join operation returns rows when there is a match in one of the

relations.

grunt> out\_full = join su\_data by id full outer, du\_data by id;

grunt> dump out\_full;

output

(1,rafi,guntur,1,rah,nrt)

(2,riz,narasaraopet,2,rev,gnt)

(3,ria,hyderabad,3,mcp,narasa)

(4,amaan,vijayawada,4,ant,hyd)

(5,ayaan,banglore,5,mem,ban)

grunt>

Using Multiple Keys

We can perform JOIN operation using multiple keys.

grunt> multiple\_keys = join su\_data by (id,name), du\_data by(id,name);

grunt> dump multiple\_keys;

FILTER OPERATORS:

There are mainly two types of filter operators

1. filter

2. limit

Filter:The filter operator is used to select the required tuples from a relation based on

a condition.

grunt>filter\_data = FILTER student\_details BY city == 'guntur';

LIMIT: the limit operator is used to limit the no of statements in the relation

grunt>limit\_data = LIMIT student\_details by sno == 2

**output**

(1,rafi,guntur)

SORTING OPERATORS

1.SORT

2.FOREACH

The ORDER BY operator is used to display the contents of a relation in a sorted

order based on one or more fields

grunt> order\_data = order su\_data by city;

grunt>dump order\_data;

foreach

The FOREACH operator is used to generate specified data transformations based on

the

column data.

grunt>for\_each = FOREACH su\_data GENERATE id,city;

output

(1,guntur)

(2,narasaraopet)

(3,hyderabad)

(4,vijayawada)

(5,banglore)

grunt>

**1. PIG script to perform word count**

wordcount

=========

myinput = load 'hdfs://localhost:9000/cse/f1' as (line);

//TOKENIZE splits the line into a field for each word.

//flatten will take the collection of records returned by TOKENIZE and

//produce a separate record for each one, calling the single field in the

//record word.

words = foreach myinput generate flatten(TOKENIZE(line)) as word;

grpd = group words by word;

cntd = foreach grpd generate group, COUNT(words);

dump cntd;