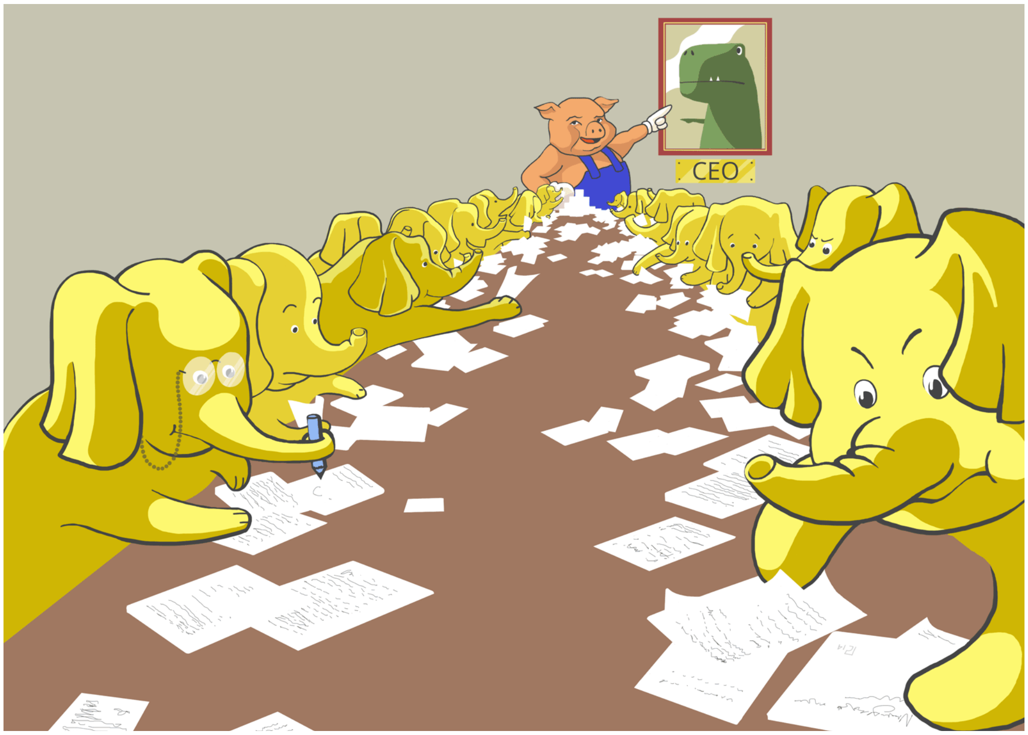


Data Hadoop & Spark Training - ACADGILD Assignment 4.3

26-Nov-17

BIG DATA – PIG ASSIGNMENT

BY – SAHIL KHURANA

Problem Statement

Write a program to implement wordcount using Pig.

Share the screenshots of the commands used with its Associated output.

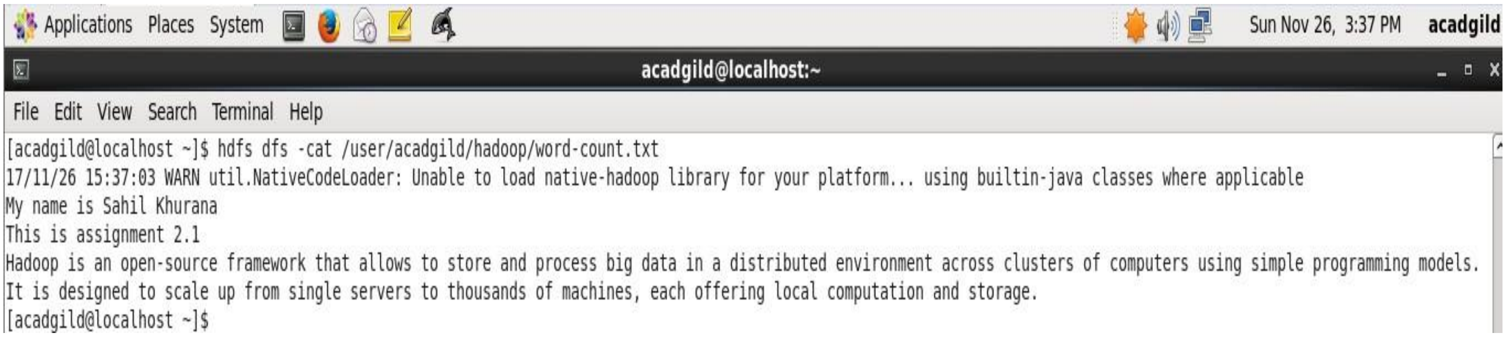
As not dataset is given by Acadgild. So, I will be using the following dataset to implement wordcount using Pig

So, I decided to use the “word-count.txt” which I created in Assignment 2.1 Task 2.

Problem Statement of Assignment 2.1 Task 2.

Create a file in HDFS under directory /user/acadgild/hadoop, with name word-count.txt. Whatever we type on screen should get appended to the file. Try to type (on screen) few lines from any online article or textbook.

Dataset - File Name “word-count.txt”



Applications Places System
Sun Nov 26, 3:37 PM acadgild
acadgild@localhost:~
File Edit View Search Terminal Help

```
[acadgild@localhost ~]$ hdfs dfs -cat /user/acadgild/hadoop/word-count.txt
17/11/26 15:37:03 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
My name is Sahil Khurana
This is assignment 2.1
Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models.
It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.
[acadgild@localhost ~]$
```

word-count.txt

My name is Sahil Khurana

This is assignment 2.1

Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.

Now we will see in steps how to generate the output using Pig Latin.

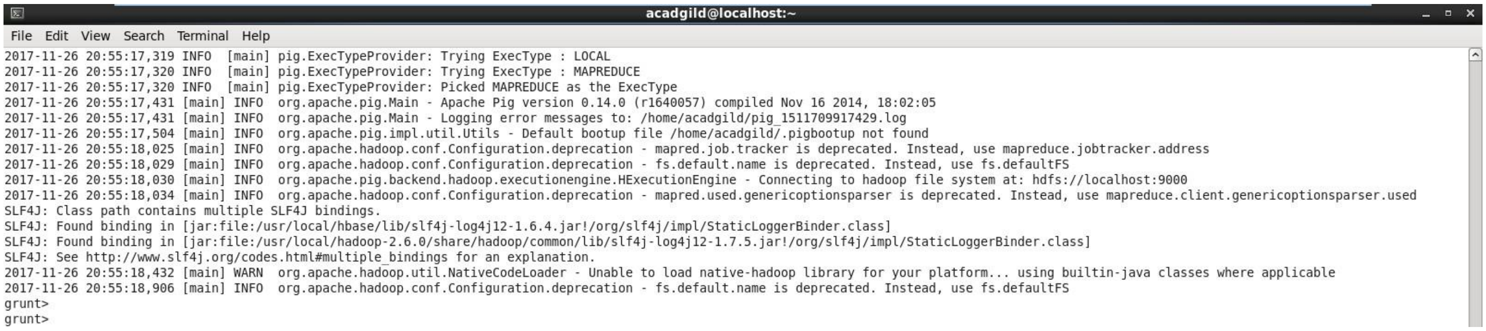
Step 1: - Put the dataset in HDFS.

In our case dataset is already in HDFS location “/user/acadgild/hadoop/”.

Here are major steps to develop Pig word count application.

MapReduce Mode - To run Pig in mapreduce mode, you need access to a Hadoop cluster and HDFS installation. You can specify mapreduce mode using the -x flag

pig -x mapreduce



```
2017-11-26 20:55:17,319 INFO  [main] pig.ExecTypeProvider: Trying ExecType : LOCAL
2017-11-26 20:55:17,320 INFO  [main] pig.ExecTypeProvider: Trying ExecType : MAPREDUCE
2017-11-26 20:55:17,320 INFO  [main] pig.ExecTypeProvider: Picked MAPREDUCE as the ExecType
2017-11-26 20:55:17,431 [main] INFO  org.apache.pig.Main - Apache Pig version 0.14.0 (r1640057) compiled Nov 16 2014, 18:02:05
2017-11-26 20:55:17,431 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/acadgild/pig_1511709917429.log
2017-11-26 20:55:17,504 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/acadgild/.pigbootup not found
2017-11-26 20:55:18,025 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-11-26 20:55:18,029 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-11-26 20:55:18,030 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: hdfs://localhost:9000
2017-11-26 20:55:18,034 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.used.genericoptionsparser is deprecated. Instead, use mapreduce.client.genericoptionsparser.used
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hbase/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
2017-11-26 20:55:18,432 [main] WARN  org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2017-11-26 20:55:18,906 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt>
grunt>
```

Step 2: - Load the data from HDFS.

grunt>Input_Dataset = LOAD '/user/acadgild/hadoop/word-count.txt' AS(line:Chararray);

Step 3: - Transforming Sentence into words and Column into rows

Convert the Sentence into words.

The data we have is in sentences. So we have to convert that data into words using

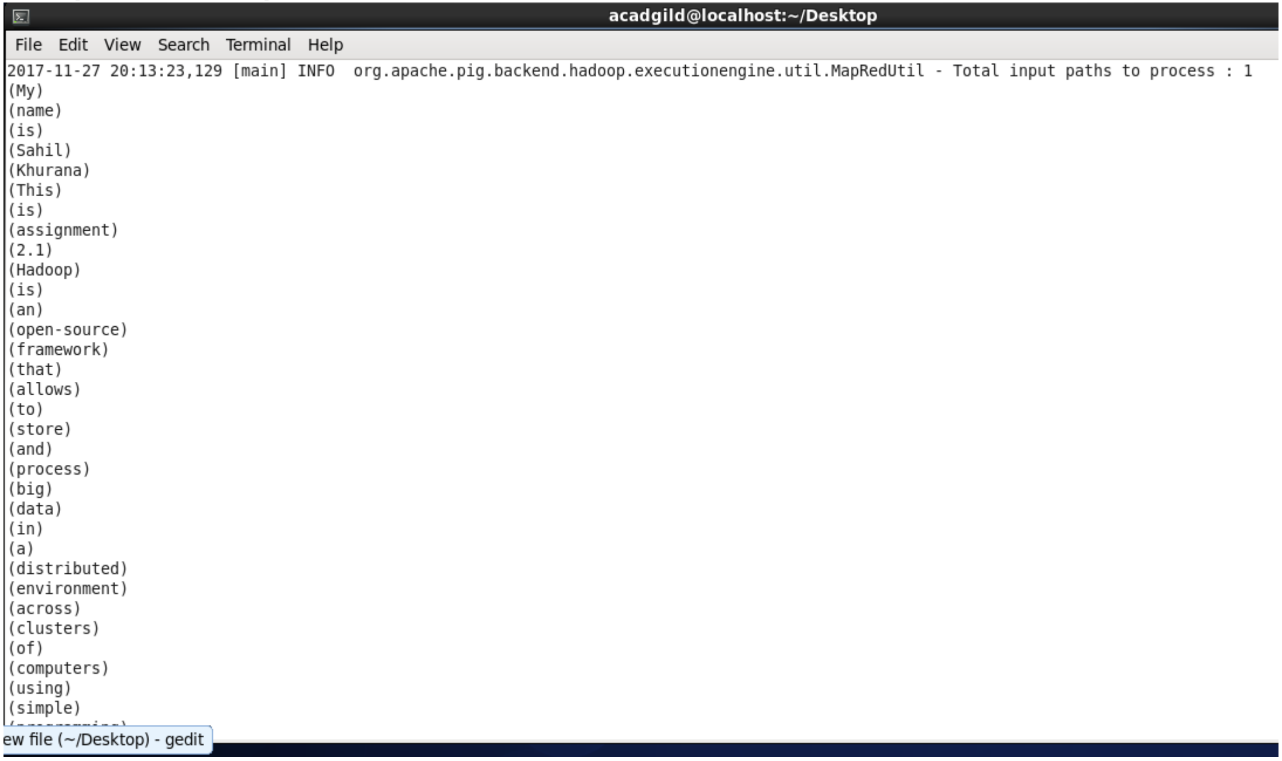
TOKENIZE Function and delimiter like space can specify as (TOKENIZE(line,' '));

Convert Column into Rows

To convert every line of data into multiple rows, for this we have function called FLATTEN in pig. Using FLATTEN function the bag is converted into tuple, means the array of strings converted into multiple rows.

grunt>words = FOREACH Input_Dataset GENERATE FLATTEN(TOKENIZE(line,' ')) AS word;

Output of Step 3


acadgild@localhost:~/Desktop
File Edit View Search Terminal Help
ew file (~/Desktop) - gedit

```
2017-11-27 20:13:23,129 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(My)
(name)
(is)
(Sahil)
(Khurana)
(This)
(is)
(assignment)
(2.1)
(Hadoop)
(is)
(an)
(open-source)
(framework)
(that)
(allows)
(to)
(store)
(and)
(process)
(big)
(data)
(in)
(a)
(distributed)
(environment)
(across)
(clusters)
(of)
(computers)
(using)
(simple)
(programming)
```

```
(My)
(name)
(is)
(Sahil)
(Khurana)
(This)
(is)
(assignment)
(2.1)
(Hadoop)
(is)
(an)
(open-source)
(framework)
(that)
(allows)
(to)
(store)
(and)
(process)
(big)
(data)
(in)
(a)
(distributed)
(environment)
(across)
(clusters)
(of)
(computers)
(using)
(simple)
(programming)
(models.)
(It)
(is)
(designed)
(to)
(scale)
(up)
(from)
(single)
(servers)
(to)
(thousands)
(of)
(machines,)
(each)
```

(offering)

(local)

(computation)

(and)

(storage.)

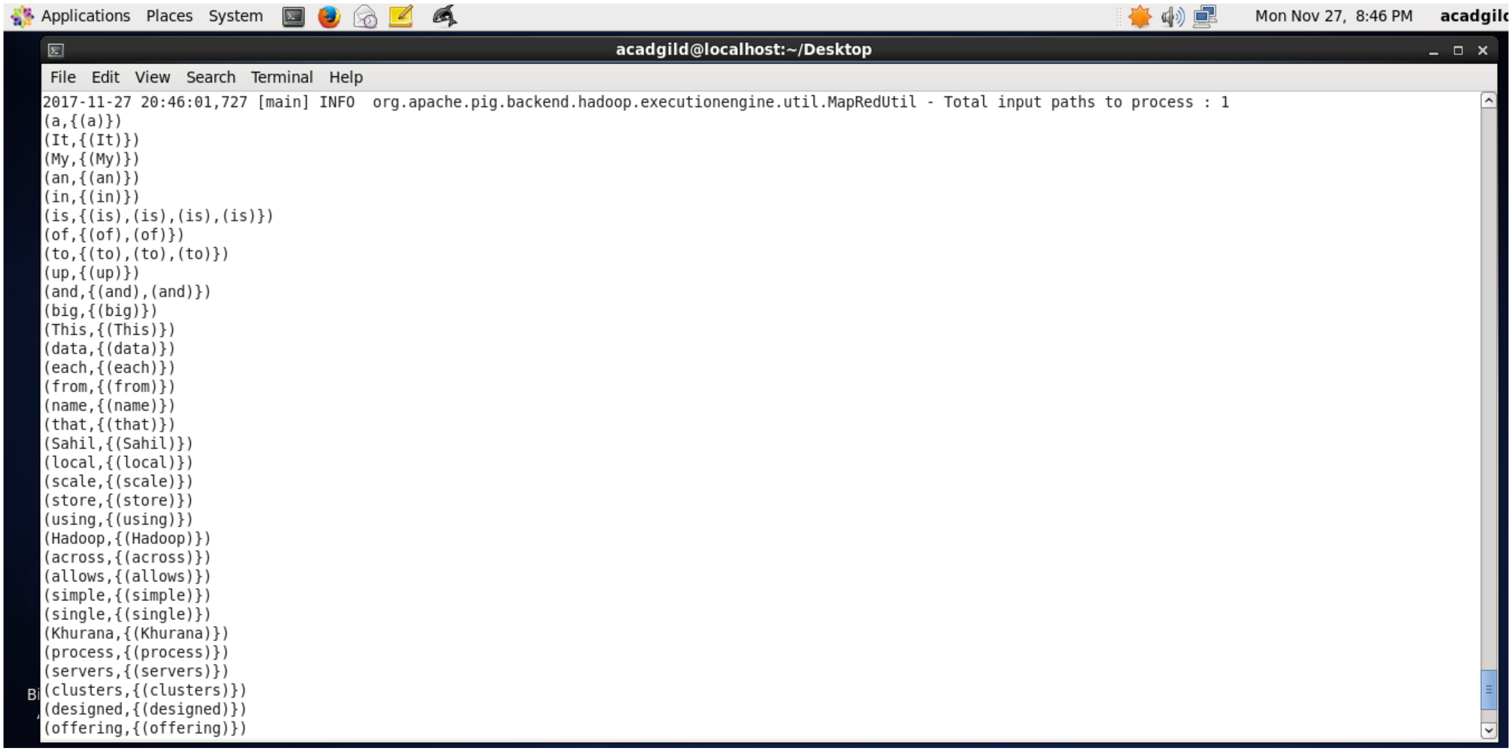
Step 4: - The words are filtered to remove any spaces in the file.

```
grunt>filtered_words = FILTER words BY word MATCHES '\\w+';
```

Step 5: - To count each word occurrences, for that we have to group all the words.

```
grunt>word_groups = GROUP filtered_words BY word;
```

Output of Step 5



```
2017-11-27 20:46:01,727 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(a,{(a)})
(It,{(It)})
(My,{(My)})
(an,{(an)})
(in,{(in)})
(is,{(is),(is),(is),(is)})
(of,{(of),(of)})
(to,{(to),(to),(to)})
(up,{(up)})
(and,{(and),(and)})
(big,{(big)})
(This,{(This)})
(data,{(data)})
(each,{(each)})
(from,{(from)})
(name,{(name)})
(that,{(that)})
(Sahil,{(Sahil)})
(local,{(local)})
(scale,{(scale)})
(store,{(store)})
(using,{(using)})
(Hadoop,{(Hadoop)})
(across,{(across)})
(allows,{(allows)})
(simple,{(simple)})
(single,{(single)})
(Khurana,{(Khurana)})
(process,{(process)})
(servers,{(servers)})
(clusters,{(clusters)})
(designed,{(designed)})
(offering,{(offering)})
```

(a,{(a)})

(It,{(It)})

(My,{(My)})

(an,{(an)})

(in,{(in)})

(is,{(is),(is),(is),(is)})

(of,{(of),(of)})

(to,{(to),(to),(to)})

(up,{(up)})

(and,{(and),(and)})

(big,{(big)})

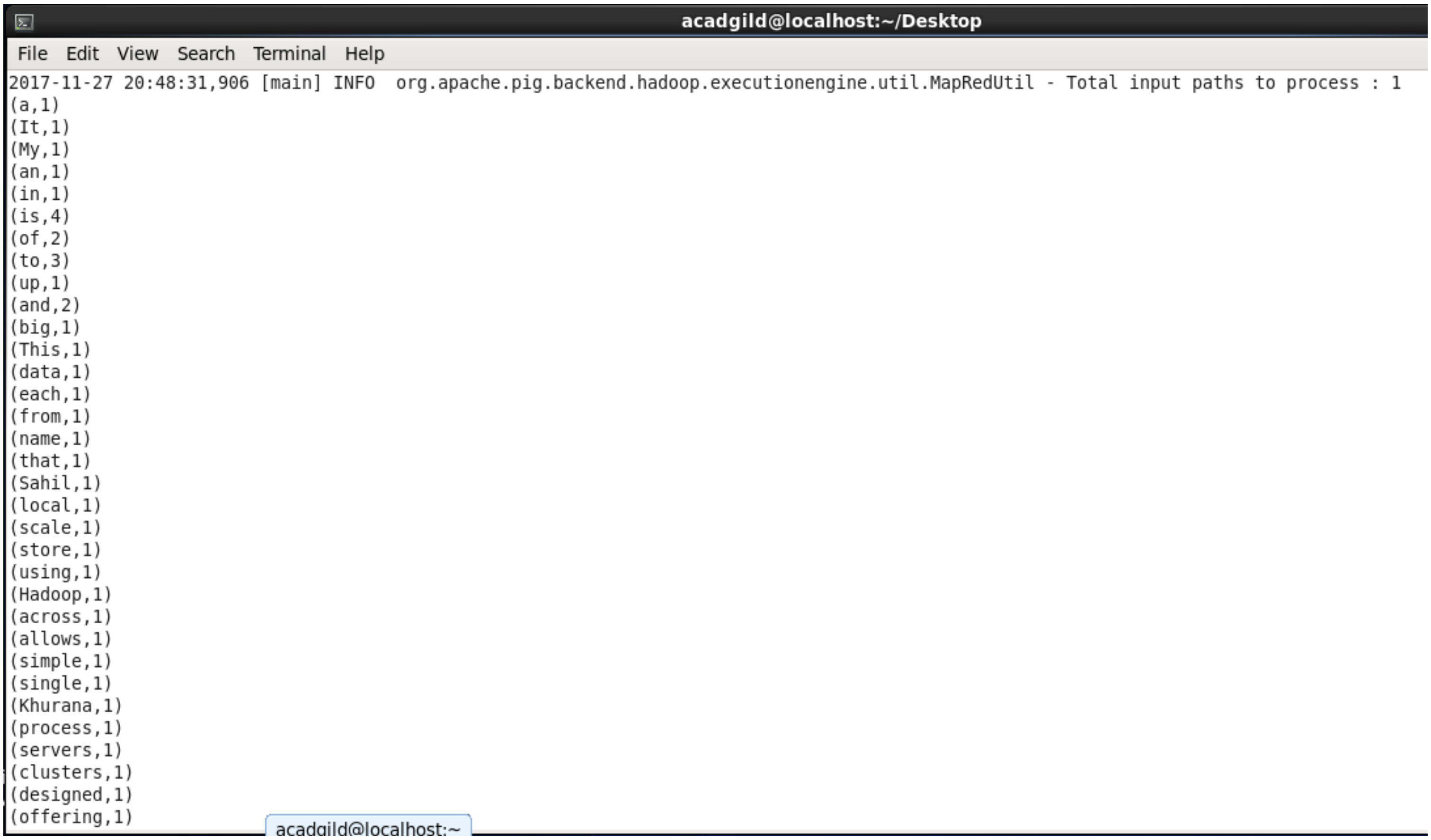
(This,{(This)})

```
(data,{(data)})
(each,{(each)})
(from,{(from)})
(name,{(name)})
(that,{(that)})
(Sahil,{(Sahil)})
(local,{(local)})
(scale,{(scale)})
(store,{(store)})
(using,{(using)})
(Hadoop,{(Hadoop)})
(across,{(across)})
(allows,{(allows)})
(simple,{(simple)})
(single,{(single)})
(Khurana,{(Khurana)})
(process,{(process)})
(servers,{(servers)})
(clusters,{(clusters)})
(designed,{(designed)})
(offering,{(offering)})
(computers,{(computers)})
(framework,{(framework)})
(thousands,{(thousands)})
(assignment,{(assignment)})
(computation,{(computation)})
(distributed,{(distributed)})
(environment,{(environment)})
(programming,{(programming)})
```

Step 6: - Generate word count

```
grunt>word_count = FOREACH word_groups GENERATE group AS word , COUNT(filtered_words) AS count;
```

Output of Step 6


acadgild@localhost:~/Desktop
File Edit View Search Terminal Help
acadgild@localhost:~

```
2017-11-27 20:48:31,906 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(a,1)
(It,1)
(My,1)
(an,1)
(in,1)
(is,4)
(of,2)
(to,3)
(up,1)
(and,2)
(big,1)
(This,1)
(data,1)
(each,1)
(from,1)
(name,1)
(that,1)
(Sahil,1)
(local,1)
(scale,1)
(store,1)
(using,1)
(Hadoop,1)
(across,1)
(allows,1)
(simple,1)
(single,1)
(Khurana,1)
(process,1)
(servers,1)
(clusters,1)
(designed,1)
(offering,1)
```

```
(a,1)
(It,1)
(My,1)
(an,1)
(in,1)
(is,4)
(of,2)
(to,3)
(up,1)
(and,2)
(big,1)
(This,1)
(data,1)
(each,1)
(from,1)
(name,1)
(that,1)
(Sahil,1)
(local,1)
(scale,1)
(store,1)
(using,1)
(Hadoop,1)
(across,1)
(allows,1)
(simple,1)
```

(single,1)

(Khurana,1)

(process,1)

(servers,1)

(clusters,1)

(designed,1)

(offering,1)

(computers,1)

(framework,1)

(thousands,1)

(assignment,1)

(computation,1)

(distributed,1)

(environment,1)

(programming,1)

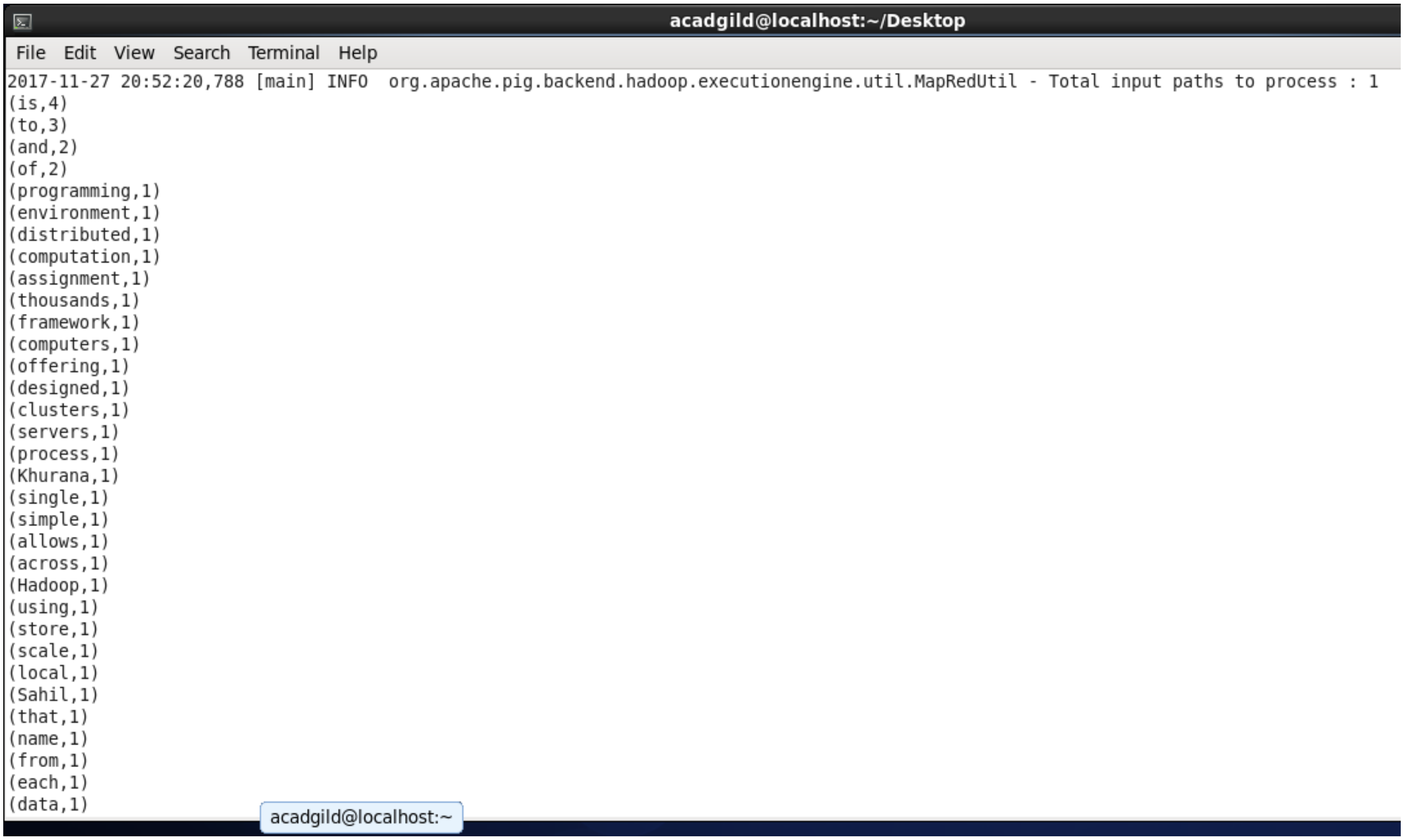
Step 7: - Arrange the word count by descending order

```
grunt>ordered_word_count = ORDER word_count BY count DESC;
```

Step 8: - Print the word count on console

```
grunt>dump ordered_word_count
```

Output of Step 7 and Step 8

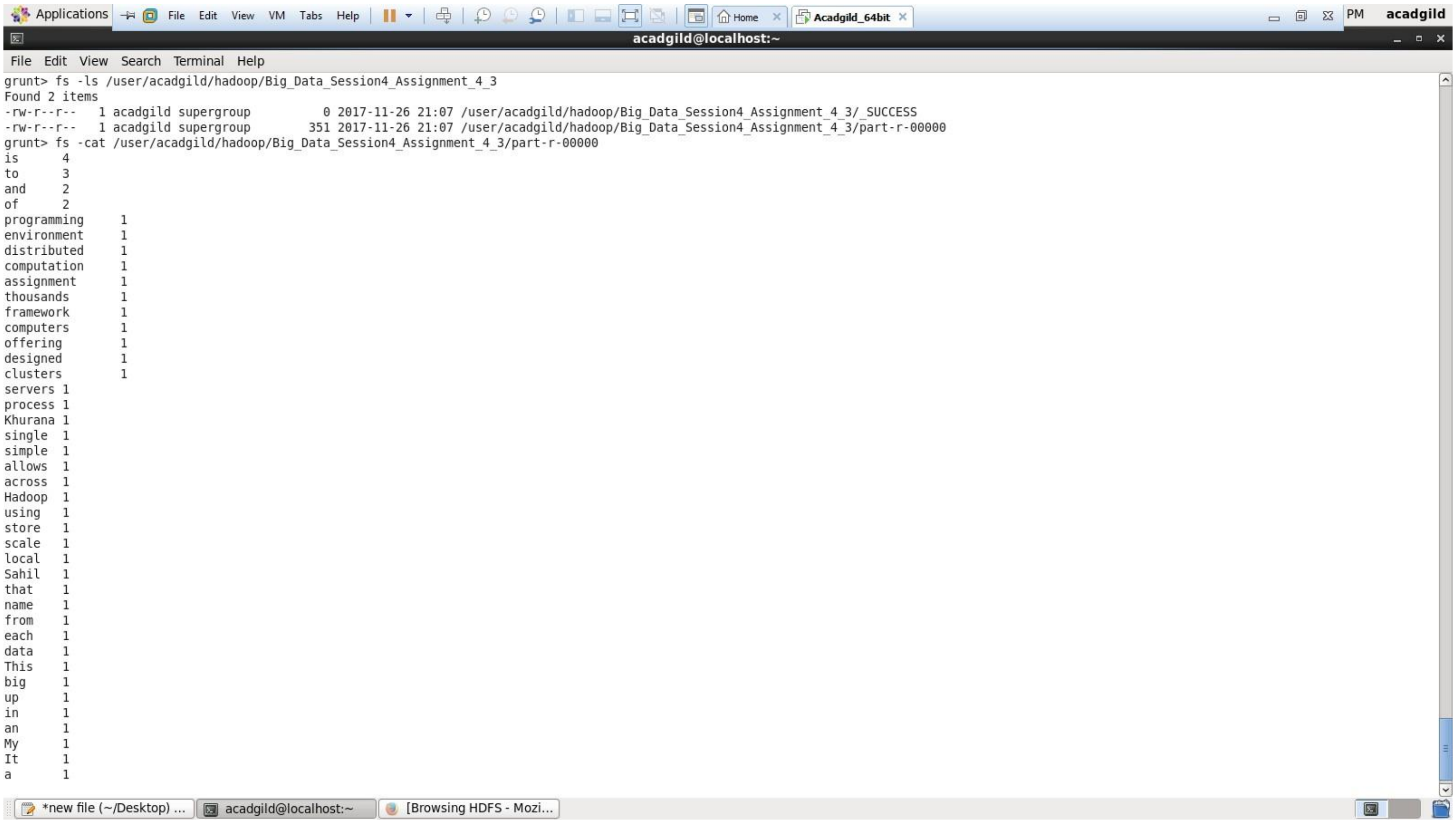

acadgild@localhost:~/Desktop
File Edit View Search Terminal Help
acadgild@localhost:~

```
2017-11-27 20:52:20,788 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(is,4)
(to,3)
(and,2)
(of,2)
(programming,1)
(environment,1)
(distributed,1)
(computation,1)
(assignment,1)
(thousands,1)
(framework,1)
(computers,1)
(offering,1)
(designed,1)
(clusters,1)
(servers,1)
(process,1)
(Khurana,1)
(single,1)
(simple,1)
(allows,1)
(across,1)
(Hadoop,1)
(using,1)
(store,1)
(scale,1)
(local,1)
(Sahil,1)
(that,1)
(name,1)
(from,1)
(each,1)
(data,1)
```

```
(is,4)
(to,3)
(and,2)
(of,2)
(programming,1)
(environment,1)
(distributed,1)
(computation,1)
(assignment,1)
(thousands,1)
(framework,1)
(computers,1)
(offering,1)
(designed,1)
(clusters,1)
(servers,1)
(process,1)
(Khurana,1)
(single,1)
(simple,1)
(allows,1)
(across,1)
(Hadoop,1)
(using,1)
(store,1)
(scale,1)
(local,1)
(Sahil,1)
(that,1)
(name,1)
(from,1)
(each,1)
(data,1)
(This,1)
(big,1)
(up,1)
(in,1)
(an,1)
(My,1)
(It,1)
(a,1)
```

Step 9: - Store the output grunt>STORE ordered_word_count INTO '/user/acadgild/hadoop/Big_Data_Session4_Assignment_4_3';

Final Output



```
grunt> fs -ls /user/acadgild/hadoop/Big_Data_Session4_Assignment_4_3
Found 2 items
-rw-r--r--   1 acadgild supergroup          0 2017-11-26 21:07 /user/acadgild/hadoop/Big_Data_Session4_Assignment_4_3/_SUCCESS
-rw-r--r--   1 acadgild supergroup        351 2017-11-26 21:07 /user/acadgild/hadoop/Big_Data_Session4_Assignment_4_3/part-r-00000
grunt> fs -cat /user/acadgild/hadoop/Big_Data_Session4_Assignment_4_3/part-r-00000
is      4
to      3
and     2
of      2
programming     1
environment     1
distributed     1
computation     1
assignment      1
thousands       1
framework       1
computers       1
offering        1
designed        1
clusters        1
servers 1
process 1
Khurana 1
single  1
simple  1
allows  1
across  1
Hadoop  1
using   1
store   1
scale   1
local   1
Sahil   1
that    1
name    1
from    1
each    1
data    1
This    1
big     1
up      1
in      1
an      1
My      1
It      1
a       1
```

is 4
to 3
and 2
of 2
programming 1
environment 1
distributed 1
computation 1
assignment 1
thousands 1
framework 1
computers 1
offering 1
designed 1

```
clusters	1
servers	1
process	1
Khurana	1
single	1
simple	1
allows	1
across	1
Hadoop	1
using	1
store	1
scale	1
local	1
Sahil	1
that	1
name	1
from	1
each	1
data	1
This	1
big	1
up	1
in	1
an	1
My	1
It	1
a	1
```

Complete Script

```
grunt>Input_Dataset = LOAD '/user/acadgild/hadoop/word-count.txt' AS(line:Chararray);

grunt>words = FOREACH Input_Dataset GENERATE FLATTEN(TOKENIZE(line,' ')) AS word;

grunt>filtered_words = FILTER words BY word MATCHES '\\w+';

grunt>word_groups = GROUP filtered_words BY word;

grunt>word_count = FOREACH word_groups GENERATE group AS word , COUNT(filtered_words)
AS count;

grunt>ordered_word_count = ORDER word_count BY count DESC;

grunt>dump ordered_word_count

grunt>STORE ordered_word_count INTO
'/user/acadgild/hadoop/Big_Data_Session4_Assignment_4_3';
```