MAPREDUCE & PIG TUTORIAL

Hands-on Session

by Suchitra Jayaprakash
suchitra@cmi.ac.in
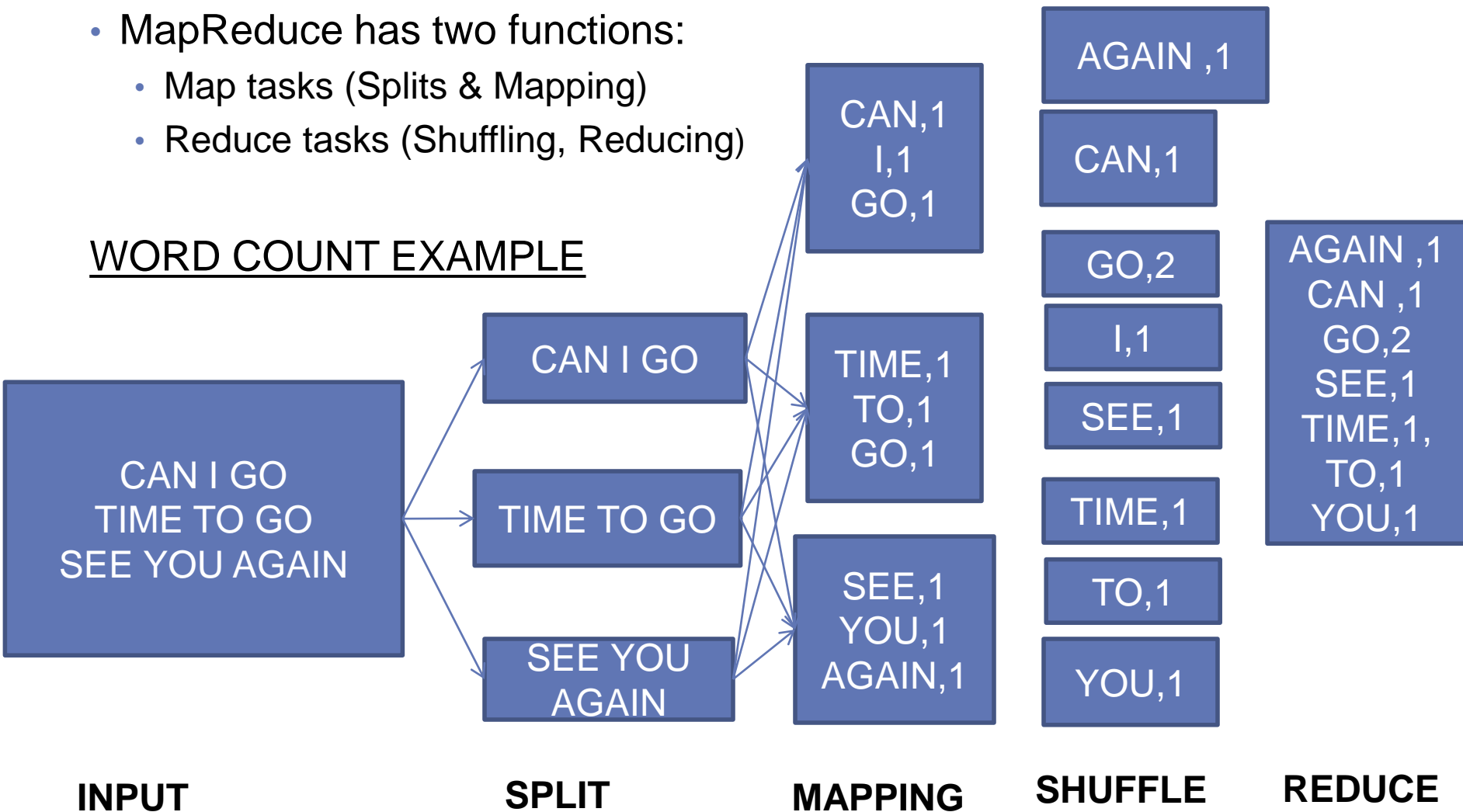
MAPREDUCE

- MapReduce is a programing model. It is specialization of the *split-apply-combine* strategy for data analysis.

- Hadoop's MapReduce implementation is based on Google's paper : MAPREDUCE: SIMPLIFIED DATA PROCESSING ON LARGE CLUSTERS" by Jeffery Dean and Sanjay Ghemawat in 2004.

- Google's proprietary MapReduce system ran on the Google File System (GFS).

- MapReduce leverages HDFS to perform high performance batch processing.

How MapReduce Works?

- MapReduce has two functions:
  - Map tasks (Splits & Mapping)
  - Reduce tasks (Shuffling, Reducing)

WORD COUNT EXAMPLE

CAN I GO TIME TO GO SEE YOU AGAIN	CAN I GO	CAN,1 I,1 GO,1	AGAIN ,1 CAN,1 GO,2 I,1 SEE,1 TIME,1 TO,1 YOU,1	AGAIN ,1 CAN ,1 GO,2 SEE,1 TIME,1, TO,1 YOU,1
	TIME TO GO	TIME,1 TO,1 GO,1		
	SEE YOU AGAIN	SEE,1 YOU,1 AGAIN,1		

INPUT   **SPLIT**   **MAPPING**   **SHUFFLE**   **REDUCE**

MAPREDUCE - Run WordCount

- Start cloudera quick start

docker run --hostname=quickstart.cloudera --privileged=true -t -i --publish-all=true -p 8888:8888 -p 8080:80 -p 50070:50070 -p 8088:8088 -p 50075:50075 -p 8032:8032 -p 8042:8042 -p 19888:19888 cloudera/quickstart /usr/bin/docker-quickstart

Port	Purpose
8088	Yarn (MRv2) - job tracker
8032	ResourceManager
50070	Name node web interface
50075	Data node
8042	NodeManager
19888	JobHistory Server

MAPREDUCE - Run WordCount

- Copy text file to HDFS.
  hadoop fs -mkdir DATA

  docker cp E:/sample1.txt <containerid>:/tmp/sample1.txt
  docker cp E:/sample2.txt <containerid>:/tmp/sample2.txt

  hadoop fs -copyFromLocal /tmp/sample1.txt DATA/sample1.txt
  hadoop fs -copyFromLocal /tmp/sample2.txt DATA/sample2.txt

- Copy WordCount java to Docker .
  docker cp e:/MSc_Datascience/BigDataHadoop/WordCount.java <containerid>:/tmp/WordCount.java

- https://docs.cloudera.com/documentation/other/tutorial/CDH5/topics/ht_wordcount1_source.html

MAPREDUCE - Run WordCount

- Compile java class & create jar

  mkdir -p build

  javac -cp /usr/lib/hadoop/*:/usr/lib/hadoop-mapreduce/* /tmp/WordCount1.java -d build -Xlint

  ```
  Using CATALINA_PID:     /var/run/solr/solr.pid
  Started Impala Catalog Server (catalogd) :          [  OK  ]
  Started Impala Server (impalad):                    [  OK  ]
  [root@quickstart /]# hadoop fs -mkdir DATA
  [root@quickstart /]# hadoop fs -copyFromLocal /tmp/sample1.txt DATA/sample1.txt
  [root@quickstart /]# hadoop fs -copyFromLocal /tmp/sample2.txt DATA/sample2.txt
  [root@quickstart /]# mkdir -p build
  [root@quickstart /]# javac -cp /usr/lib/hadoop/*:/usr/lib/hadoop-mapreduce/* /tm
  p/WordCount1.java -d build -Xlint
  warning: [path] bad path element "/usr/lib/hadoop-mapreduce/jaxb-api.jar": no su
  ch file or directory
  warning: [path] bad path element "/usr/lib/hadoop-mapreduce/activation.jar": no
  such file or directory
  warning: [path] bad path element "/usr/lib/hadoop-mapreduce/jsr173_1.0_api.jar":
   no such file or directory
  warning: [path] bad path element "/usr/lib/hadoop-mapreduce/jaxb1-impl.jar": no
  such file or directory
  4 warnings
  [root@quickstart /]#
  ```

  jar -cvf wordcount.jar -C build/ .

  ```
  [root@quickstart /]# jar -cvf wordcount.jar -C build/ .
  added manifest
  adding: WordCount1$Map.class(in = 2192) (out= 982)(deflated 55%)
  adding: WordCount1$Reduce.class(in = 1630) (out= 687)(deflated 57%)
  adding: WordCount1.class(in = 1959) (out= 986)(deflated 49%)
  [root@quickstart /]#
  ```

MAPREDUCE - Run WordCount

- Execute Mapreduce Word Count job .

   hadoop jar wordcount.jar WordCount1 DATA DATA2

MAPREDUCE - Run WordCount

- Open Yarn web page in browser

http://192.168.99.100:8088/cluster

http://localhost:8088/cluster

MAPREDUCE - Run WordCount

- Click on JOB ID

		Application Ove
User:	root	
Name:	wordcount	
Application Type:	MAPREDUCE	
Application Tags:		
State:	FINISHED	
FinalStatus:	SUCCEEDED	
Started:	Wed Dec 25 14:37:12 +0000 2019	
Elapsed:	38sec	
Tracking URL:	History	
Diagnostics:		

		Application M
Total Resource Preempted:	<memory:0, vCores:0>	
Total Number of Non-AM Containers Preempted:	0	
Total Number of AM Containers Preempted:	0	
Resource Preempted from Current Attempt:	<memory:0, vCores:0>	
Number of Non-AM Containers Preempted from Current Attempt:	0	
Aggregate Resource Allocation:	137655 MB-seconds, 86 vcore-seconds	

Cluster
- About
- Nodes
- Applications
  - NEW
  - NEW_SAVING
  - SUBMITTED
  - ACCEPTED
  - RUNNING
  - FINISHED
  - FAILED
  - KILLED
- Scheduler

Tools

ApplicationMaster

Attempt Number	Start Time	Node	Lo
1	Wed Dec 25 14:37:12 +0000 2019	quickstart.cloudera:8042	logs

- To View Node Manager
- http://192.168.99.100:8042/node/node

MAPREDUCE - Run WordCount

- View Job history

http://192.168.99.100:19888/jobhistory

	Job Overview
Job Name:	wordcount
User Name:	root
Queue:	root.root
State:	SUCCEEDED
Uberized:	false
Submitted:	Wed Dec 25 14:37:12 UTC 2019
Started:	Wed Dec 25 14:37:25 UTC 2019
Finished:	Wed Dec 25 14:37:51 UTC 2019
Elapsed:	25sec
Diagnostics:	
Average Map Time	13sec
Average Shuffle Time	5sec
Average Merge Time	0sec
Average Reduce Time	0sec

Application
- Job
    - Overview
    - Counters
    - Configuration
    - Map tasks
    - Reduce tasks
- Tools

ApplicationMaster

Attempt Number	Start Time	Node	Logs
1	Wed Dec 25 14:37:18 UTC 2019	quickstart.cloudera:8042	logs

Task Type	Total	Complete
Map	2	2
Reduce	1	1

Attempt Type	Failed	Killed	Successful
Maps	0	0	2
Reduces	0	0	1

MAPREDUCE - Run WordCount

- Open DATA2 folder in Namenode to view mapreduce output

http://192.168.99.100:50070/explorer.html#/user/root/DATA2

Hadoop   Overview   Datanodes   Snapshot   Startup Progress   Utilities ▾

Browse Directory

/user/root/DATA2								Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	root	supergroup	0 B	Wed Dec 25 20:07:51 +0530 2019	1	128 MB	_SUCCESS
-rw-r--r--	root	supergroup	39 B	Wed Dec 25 20:07:50 +0530 2019	1	128 MB	part-r-00000

MAPREDUCE - Run WordCount

- Input file

This is sample1 text
This is sample2 text

- Final Output

```
1    This    2
2    is  2
3    sample1 1
4    sample2 1
5    text    2
6
```

Apache PIG

- PIG is an abstraction over MapReduce.

- It uses high level language - Pig Latin.

- PIG Latin is easy to learn if programmer knows procedural language. Programmer can do MapReduce task without having to write complex Java code.

- Pig has many built-in functions & data types for doing operations like joins, filters, ordering, grouping. Thus reducing code length.

- Apache PIG has a internal component called PIG engine.

- PIG engine converts Pig Latin scripts to Map Reduce Task.

- Apache Pig was developed as a research project at Yahoo in 2006. First release of Apache Pig came out in 2008.

Apache PIG – Run Mode

- Pig can be run in various modes:
  - Local mode : Run PIG in local host and file system
  - MapReduce mode : Run PIG on Hadoop cluster and HDFS. It is the default mode.
  - Interactive mode : Run Pig using Grunt shell.
  - Batch mode : Run Pig using PIG script.

	Interactive Mode	Batch Mode
Local Mode	$ pig -x local grunt> <pig Latin statement>	$ pig -x local ScriptFile.pig
MapReduce Mode	$ pig grunt> <pig Latin statement>	$ pig ScriptFile.pig
	$ pig -x mapreduce grunt> <pig Latin statement>	$ pig -x mapreduce ScriptFile.pig

PIG Latin Statement

- Pig Latin statement structure is:

- **LOAD** statement to read data from the file system.
- **Transformation** statements to process the data.
- **DUMP** statement to view results or **STORE** statement to save the results.

- **Load Operator**

    Relation_name = LOAD 'Input file path' USING function as schema;

PIG Latin Statement

Transformation statements :

- FILTER operator to remove unwanted rows of data.
- FOREACH , GENERATE operator to perform data transformations on columns of data.

- GROUP operator to group data in a single relation.
- COGROUP, inner JOIN, and outer JOIN operators to group or join data in two or more relations.

- ORDER operator to sort data.
- UNION operator to merge the contents of two or more relations.
- SPLIT operator to partition the contents of a relation into multiple relations.

Exercise

- Find average sepal length for each flower class in IRIS DATASET

- Reference :

  http://archive.cloudera.com/cdh5/cdh/5/pig-0.12.0-cdh5.15.0/func.html

  http://archive.cloudera.com/cdh5/cdh/5/pig-0.12.0-cdh5.15.0/udf.html

Run PIG

- **Start Cloudera server**

docker run --hostname=quickstart.cloudera --privileged=true -t -i --publish-all=true -p 8888:8888 -p 8080:80 -p 50070:50070 -p 8088:8088 -p 50075:50075 -p 8032:8032 -p 8042:8042 -p 19888:19888 cloudera/quickstart /usr/bin/docker-quickstart

- **Copy Text file to docker container**

docker cp E:/iris.txt <containerid>:/tmp/iris.txt

- **Copy Text file to HDFS**
hadoop fs -mkdir DATA
hadoop fs -copyFromLocal /tmp/iris.txt DATA/iris.txt

#view output
hdfs dfs -cat DATA/iris.txt

Run PIG

- Start PIG
- type pig and press enter to get PIG command prompt

```
flower = LOAD 'DATA/iris.txt' USING PigStorage(',') as ( sepal_length:int,
sepal_width:int, petal_length:int, petal_width:int, flower_class:chararray);

 DUMP flower;


B = GROUP flower BY flower_class;
DUMP B;


Result = FOREACH B GENERATE flower.flower_class,
AVG(flower.sepal_length);


DUMP Result;
```

OUTPUT


```
Success!

Job Stats (time in seconds):
JobId      Maps      Reduces  MaxMapTime            MinMapTIme            AvgMapTime            MedianMa
pTime      MaxReduceTime      MinReduceTime         AvgReduceTime         MedianReducetime               A
lias       Feature  Outputs
job_1581917396709_0001   1          0          47        47        47        47        n/a       n
/a        n/a       n/a       flower  MAP_ONLY              hdfs://quickstart.cloudera:8020/
tmp/temp1319724132/tmp-899714232,

Input(s):
Successfully read 151 records (4925 bytes) from: "hdfs://quickstart.cloudera:802
0/user/root/DATA/iris.txt"

Output(s):
Successfully stored 151 records (4088 bytes) in: "hdfs://quickstart.cloudera:802
0/tmp/temp1319724132/tmp-899714232"

Counters:
Total records written : 151
Total bytes written : 4088
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1581917396709_0001


2020-02-17 05:50:51,141 [main] WARN  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Encountered Warning ACCESSING_NON_EXISTENT
_FIELD 4 time(s).
2020-02-17 05:50:51,144 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2020-02-17 05:50:51,161 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2020-02-17 05:50:51,164 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2020-02-17 05:50:51,169 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2020-02-17 05:50:51,235 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
```

OUTPUT


```
(6,3,5,2,Iris-virginica)
(6,3,5,1,Iris-virginica)
(6,3,4,1,Iris-virginica)
(6,3,5,2,Iris-virginica)
(6,3,5,2,Iris-virginica)
(6,3,5,2,Iris-virginica)
(5,2,5,1,Iris-virginica)
(6,3,5,2,Iris-virginica)
(6,3,5,2,Iris-virginica)
(6,2,5,1,Iris-virginica)
(6,3,5,2,Iris-virginica)
(6,3,5,2,Iris-virginica)
(5,3,5,1,Iris-virginica)
(,,,,)
grunt> B = GROUP flower BY flower_class;
2020-02-17 05:53:12,148 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2020-02-17 05:53:12,150 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
grunt> Result = FOREACH B GENERATE flower.flower_class, AVG(flower.sepal_length)
;
grunt> DUMP Result;
2020-02-17 05:53:42,813 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: GROUP_BY
2020-02-17 05:53:42,826 [main] INFO  org.apache.pig.newplan.logical.optimizer.Lo
gicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateFor
EachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptim
izer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimiz
er, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter],
RULES_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]}
2020-02-17 05:53:42,909 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
2020-02-17 05:53:42,959 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2020-02-17 05:53:42,962 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
2020-02-17 05:53:43,080 [main] INFO  org.apache.hadoop.yarn.client.RMProxy - Con
necting to ResourceManager at /0.0.0.0:8032
2020-02-17 05:53:43,097 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig script settings are added to the job
2020-02-17 05:53:43,190 [main] INFO  org.apache.pig.backend.hadoop.executionengi
```

OUTPUT

```
2020-02-17 05:56:39,541 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - fs.default.name is deprecated. Instead, use fs.defaultFS
2020-02-17 05:56:39,544 [main] INFO  org.apache.hadoop.conf.Configuration.deprec
ation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.addr
ess
2020-02-17 05:56:39,551 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2020-02-17 05:56:39,583 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2020-02-17 05:56:39,585 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(((Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-se
tosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iri
s-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),
(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-seto
sa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-
setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(I
ris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa
),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-se
tosa),(Iris-setosa),(Iris-setosa),(Iris-setosa),(Iris-setosa)},4.6)
(((Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virg
inica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris
-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),
(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virgin
ica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-v
irginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(I
ris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginic
a),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-vir
ginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iri
s-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica),(Iris-virginica)
,(Iris-virginica),(Iris-virginica),(Iris-virginica)},6.08)
(((Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-
versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicol
or),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iri
s-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versic
olor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(I
ris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-vers
icolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),
(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-ve
rsicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor
),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-
versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicolor),(Iris-versicol
or),(Iris-versicolor)},5.48)
((()},)
grunt>
```

USER DEFINED FUNCTION

- **Find Volume Weighted Average Price of a stock per year.**

Date	Adj Close	Close	High	Low	Open	Volume
31-12-2007	92.64	92.64	94.37	92.45	93.81	5755200
02-01-2008	96.25	96.25	97.43	94.7	95.35	13858700
03-01-2008	95.21	95.21	97.25	94.52	96.06	9122500
04-01-2008	88.79	88.79	93.4	88.5	93.26	10270000

VWAP= sum(Price * Volume)/ sum(Volume)

It is a measure of the volume weighted average price at which a stock is traded over the trading horizon.

USER DEFINED FUNCTION

- **Find Volume Weighted Average Price of a stock**

Copy content to docker:

docker cp E:/MSc_Datascience/BigDataHadoop/Slides/stock.csv
<containerid>:/tmp/stock.csv

docker cp E:/MSc_Datascience/BigDataHadoop/Slides/pig_UDF.py
<containerid>:/tmp/pig_UDF.py

docker cp E:/MSc_Datascience/BigDataHadoop/Slides/piggybank-0.15.0.jar
<containerid>:/tmp/piggybank-0.15.0.jar

hadoop fs -mkdir DATA

hadoop fs -copyFromLocal /tmp/stock.csv DATA/stock.csv

USER DEFINED FUNCTION

Execute in PIG Interface:

REGISTER '/tmp/pig_UDF.py' using jython as myudfs;
REGISTER '/tmp/piggybank-0.15.0.jar';

records = LOAD 'DATA/stock.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'NO_MULTILINE', 'UNIX',
'SKIP_INPUT_HEADER') AS
(Date:datetime,AdjClose:int,Close:int,High:int,Low:int,Open:int,Volume:int);

record= foreach records GENERATE Volume, Close, GetYear(Date) as Year;

yearData = GROUP record BY Year;

 vol_wt_avg_price = FOREACH yearData GENERATE
myudfs.calVWAP(record.Year,record.Volume,record.Close);

STORE vol_wt_avg_price INTO '/user/root/PRICE';

hadoop fs -cat /user/root/PRICE/part-r-00000

QUIZ 5

```
emp_data.txt
001,Mary,32,Delhi
002,Ram,33,Bangalore
|
```

Q) Each row of the 'emp_data' file stored in HDFS contains the id, name, age and city of an employee. Data transformation is performed using PIG script below.

emp_data = LOAD 'emp_data.txt' USING PigStorage(',')as (id:int, name:chararray, age:int, city:chararray);

substring_data = FOREACH emp_data GENERATE (id,name), SUBSTRING (city, 0, 3);

Dump substring_data ;

What is the output of the code?

A) (Del)
   (Ban)

C) (001,Mary,Del)
   (002,Ram, Ban)

B) (1,Del)
   (2, Ban)

D) ((1,Mary),Del)
   ((2,Ram), Ban)

THANK YOU