

Name : Shubham Lad

Subject: Business Intelligence & Big Data Analytics

Class: MSC-I (Computer Science) Semester 2

Exam Seat no: 512

Academic Year: 2022-2023

		INDEX		
No	Date	TITLE	Page No	SIGN
1	06/01/23	Installing and setting environment variables for Working with Apache Hadoop.		
2	06/01/23	Implementing Map-Reduce Program for Word Count problem.		
3	19/01/23	Write a Pig Script for solving counting problems.		
4	20/01/23	Install HBase and use the HBase Data model Store and retrieve data.		
5	20/01/23	Install Hive and use Hive Create and store structured databases.		
6	25/01/23	Write a program to construct different types of k-shingles for a given document.		
7	25/01/23	Write a program for measuring similarity among documents and detecting passages which have been reused.		
8	02/02/23	Write a program to compute the n- moment for a given stream where n is given.		
9	02/02/23	Write a program to demonstrate the Alon-Matias-Szegedy Algorithm for second moments.		

# Practical 01

---

Aim : Installing and setting environment variables for Working with Apache Hadoop

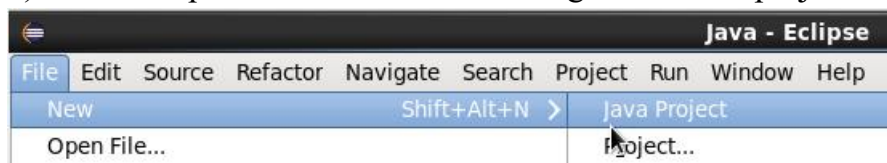
# Practical 02

---

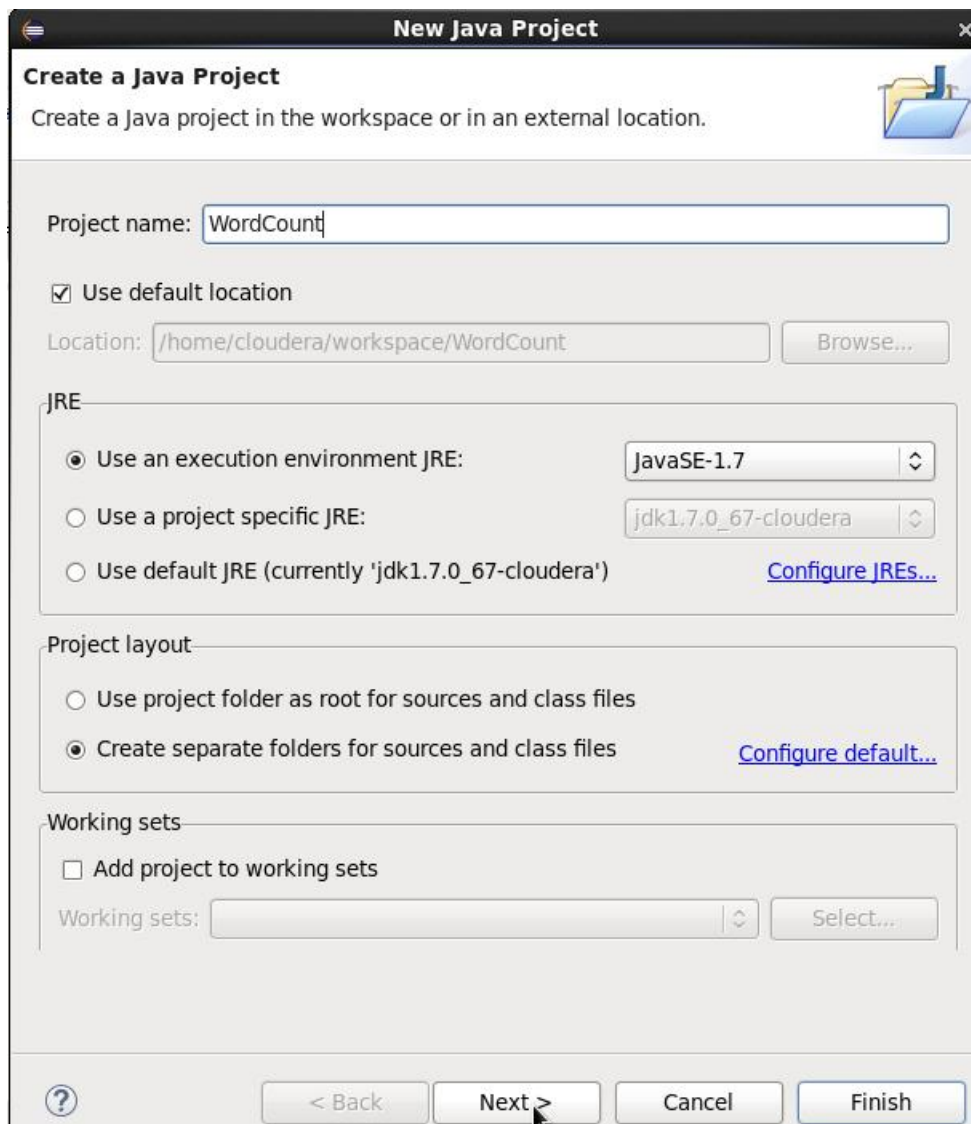
Aim: Implementing Map-Reduce Program for Word Count problem.

Steps:

1) Goto Eclipse in Cloudera and creating a new Java project.



2) Now under project name enter “WordCount” and click on next.

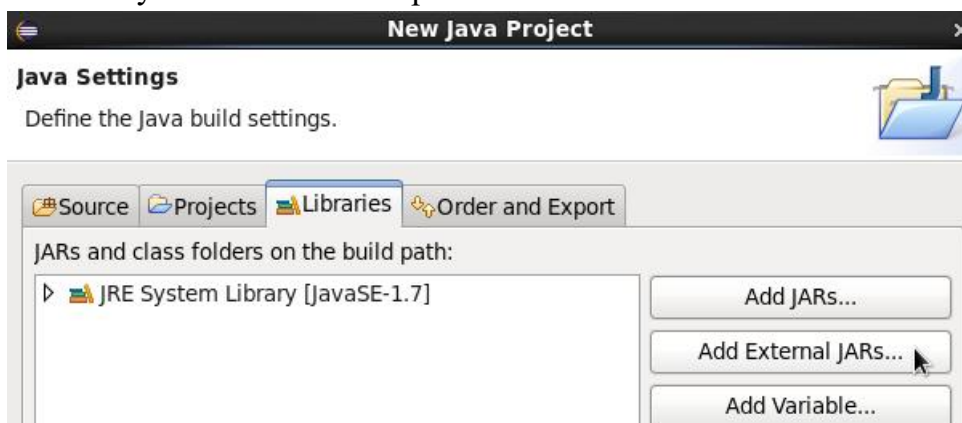


3) Now under “Add External JARs” add all the jar files present under the following locations:

File System/usr/lib/hadoop

File System/usr/lib/hadoop/client

File System/usr/lib/hadoop/client-0.20



After adding all the JAR files click on Finish.

- 4) Now create a new Java class “WordCount.java” and add the following lines of code in it.

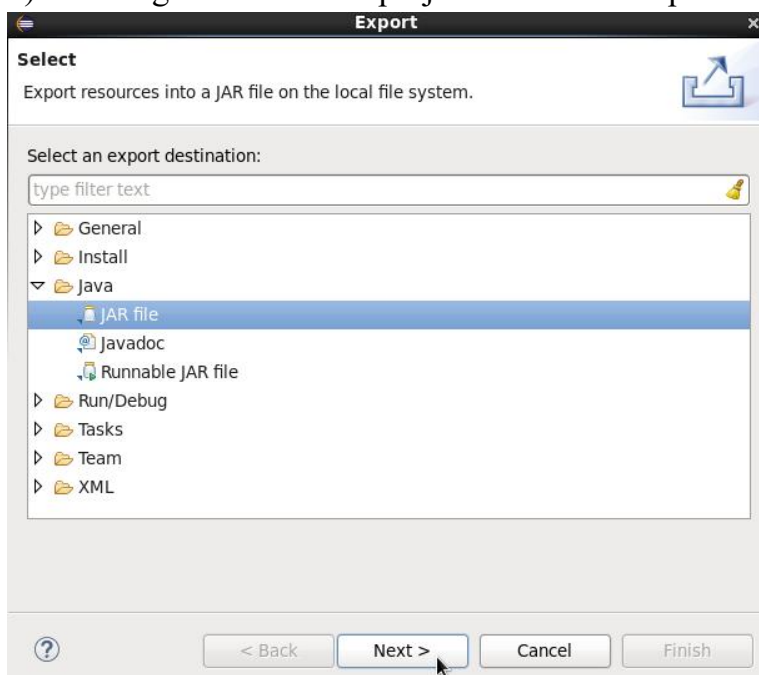
```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
public class WordCount {
    public static class TokenizerMapper
        extends Mapper<Object, Text, Text, IntWritable>{
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();
        public void map(Object key, Text value, Context context
            ) throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                context.write(word, one);
            }
        }
    }

    public static class IntSumReducer
        extends Reducer<Text,IntWritable,Text,IntWritable> {
        private IntWritable result = new IntWritable();
        public void reduce(Text key, Iterable<IntWritable> values,
            Context context
            ) throws IOException, InterruptedException {
            int sum = 0;
            for (IntWritable val : values) {
                sum += val.get();
            }
            result.set(sum);
            context.write(key, result);
        }
    }

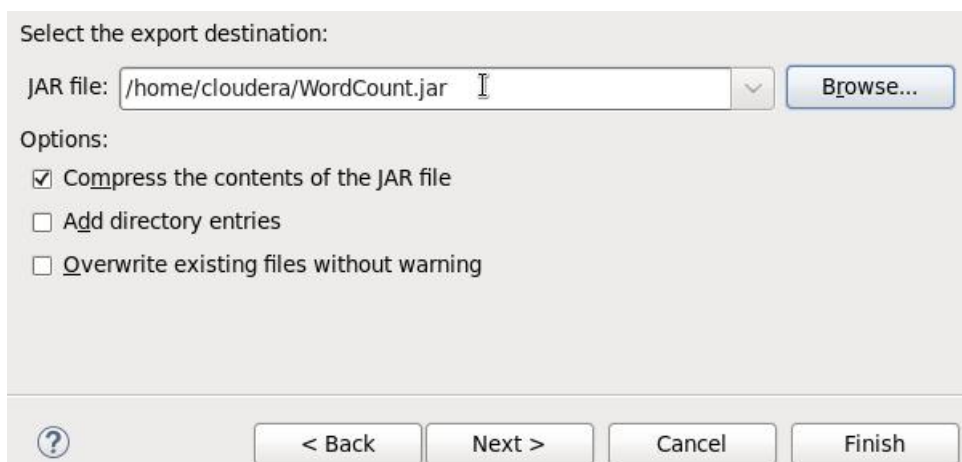
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "word count");
```

```
job.setJarByClass(WordCount.class);
job.setMapperClass(TokenizerMapper.class);
job.setCombinerClass(IntSumReducer.class);
job.setReducerClass(IntSumReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```

5) Now right-click on the project and under Export select Jar file and click on next



6) Now give the same file destination and file name as shown below and click on Finish



7) Perform the following commands by opening a new terminal.

Commands:

\$ hdfs dfs -ls /

```
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 6 items
drwxrwxrwx - hdfs supergroup 0 2017-10-23 09:15 /benchmarks
drwxr-xr-x - hbase supergroup 0 2023-01-22 05:23 /hbase
drwxr-xr-x - solr solr 0 2017-10-23 09:18 /solr
drwxrwxrwt - hdfs supergroup 0 2023-01-22 04:29 /tmp
drwxr-xr-x - hdfs supergroup 0 2017-10-23 09:17 /user
drwxr-xr-x - hdfs supergroup 0 2017-10-23 09:17 /var
[cloudera@quickstart ~]$
```

\$ sudo -u hdfs hadoop fs -mkdir /inputdirectory

```
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -mkdir /inputdirectory
[cloudera@quickstart ~]$
```

\$ hdfs dfs -ls /

```
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 7 items
drwxrwxrwx - hdfs supergroup 0 2017-10-23 09:15 /benchmarks
drwxr-xr-x - hbase supergroup 0 2023-01-22 05:23 /hbase
drwxr-xr-x - hdfs supergroup 0 2023-01-22 05:36 /inputdirectory
drwxr-xr-x - solr solr 0 2017-10-23 09:18 /solr
drwxrwxrwt - hdfs supergroup 0 2023-01-22 05:35 /tmp
drwxr-xr-x - hdfs supergroup 0 2017-10-23 09:17 /user
drwxr-xr-x - hdfs supergroup 0 2017-10-23 09:17 /var
[cloudera@quickstart ~]$
```

\$ cat>/home/cloudera/ProcessFile.txt

```
[cloudera@quickstart ~]$ cat>/home/cloudera/ProcessFile.txt
We all face challenges and difficulties in life.
Positivity helps to live life happily.
^C
```

\$ cat /home/cloudera/ProcessFile.txt

```
[cloudera@quickstart ~]$ cat /home/cloudera/ProcessFile.txt
We all face challenges and difficulties in life.
Positivity helps to live life happily.
```

\$ sudo -u hdfs hadoop fs -chmod -R 777 /inputdirectory

```
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -chmod -R 777 /inputdirectory
[cloudera@quickstart ~]$
```

\$ hdfs dfs -ls /

```
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 7 items
drwxrwxrwx - hdfs supergroup          0 2017-10-23 09:15 /benchmarks
drwxr-xr-x - hbase supergroup          0 2023-01-22 05:23 /hbase
drwxrwxrwx - hdfs supergroup          0 2023-01-22 05:36 /inputdirectory
drwxr-xr-x - solr solr                 0 2017-10-23 09:18 /solr
drwxrwxrwt - hdfs supergroup          0 2023-01-22 05:35 /tmp
drwxr-xr-x - hdfs supergroup          0 2017-10-23 09:17 /user
drwxr-xr-x - hdfs supergroup          0 2017-10-23 09:17 /var
[cloudera@quickstart ~]$
```

\$ sudo -u hdfs hadoop fs -put /home/cloudera/ProcessFile.txt /inputdirectory

```
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -put /home/cloudera/ProcessFile.txt /inputdirectory
[cloudera@quickstart ~]$
```

\$ hdfs dfs -ls /inputdirectory

```
[cloudera@quickstart ~]$ hdfs dfs -ls /inputdirectory
Found 1 items
-rw-r--r-- 1 hdfs supergroup          88 2023-01-22 05:43 /inputdirectory/ProcessFile.txt
[cloudera@quickstart ~]$
```

\$ hadoop jar /home/cloudera/WordCount.jar WordCount  
/inputdirectory/ProcessFile.txt /out1

```
[cloudera@quickstart ~]$ hadoop jar /home/cloudera/WordCount.jar WordCount /inputdirectory/Processfile.txt /out4
23/02/08 22:30:40 INFO client.RMPProxy: Connecting to ResourceManager at quickstart.cloudera/127.0.0.1:8032
23/02/08 22:30:42 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your ap
plication with ToolRunner to remedy this.
23/02/08 22:30:42 INFO input.FileInputFormat: Total input paths to process : 1
23/02/08 22:30:42 INFO mapreduce.JobSubmitter: number of splits:1
23/02/08 22:30:43 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1675923935139_0002
23/02/08 22:30:45 INFO impl.YarnClientImpl: Submitted application application_1675923935139_0002
23/02/08 22:30:46 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1675923935139_0002/
23/02/08 22:30:46 INFO mapreduce.Job: Running job: job_1675923935139_0002

23/02/08 22:31:23 INFO mapreduce.Job: Counters: 2
Job Counters
  Total time spent by all maps in occupied slots (ms)=0
  Total time spent by all reduces in occupied slots (ms)=0
[cloudera@quickstart ~]$
```

\$ hdfs dfs -ls /out1

```
[cloudera@quickstart ~]$ hdfs dfs -ls /out1
Found 2 items
-rw-r--r-- 1 cloudera supergroup          0 2023-01-05 22:27 /out1/_SUCCESS
-rw-r--r-- 1 cloudera supergroup        71 2023-01-05 22:27 /out1/part-r-00000
[cloudera@quickstart ~]$
```

\$ hdfs dfs -cat /out1/part-r-00000

```
[cloudera@quickstart ~]$ hdfs dfs -ls /out1/part-r-00000
-rw-r--r-- 1 cloudera supergroup        71 2023-01-05 22:27 /out1/part-r-00000
[cloudera@quickstart ~]$
```



# Practical 03

---

**Aim:** Write a Pig Script for solving counting problems.

## Commands:

\$ cat >/home/cloudera/input.csv

```
[cloudera@quickstart ~]$ cat >/home/cloudera/input.csv
Hello Welcome to my world
I'm Shubham Lad, a MSc Computer Science student.
```

\$ cat /home/cloudera/input.csv

```
[cloudera@quickstart ~]$ cat /home/cloudera/input.csv
Hello Welcome to my world
I'm Shubham Lad, a MSc Computer Science student.
[cloudera@quickstart ~]$
```

\$ pig -x local

```
[cloudera@quickstart ~]$ pig -x local
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.Shell).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
2023-01-18 19:46:48,251 [main] INFO org.apache.pig.Main - Apache Pig version 0.12.0-cdh5.13.0 (reexported) compiled Oct 04 2017, 11:09:03
2023-01-18 19:46:48,252 [main] INFO org.apache.pig.Main - Logging error messages to: /home/cloudera/pig 1674100008239.log
2023-01-18 19:46:48,350 [main] INFO org.apache.pig.impl.util.Utils - Default bootstrap file /home/cloudera/.pigbootstrap not found
2023-01-18 19:46:48,717 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2023-01-18 19:46:48,717 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-01-18 19:46:48,726 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: file:///
2023-01-18 19:46:49,245 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-01-18 19:46:49,422 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2023-01-18 19:46:50,302 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-01-18 19:46:50,405 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2023-01-18 19:46:50,408 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2023-01-18 19:46:50,413 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
grunt>
```

Grunt shell is Pig's interactive shell which is used to execute all Pig's scripts.

## Scripting Code:

\$ lines = load '/home/cloudera/input.csv' as (line:chararray);



```
grunt> lines = load '/home/cloudera/input.csv' as (line:chararray);
grunt> █
```

\$ words = foreach lines GENERATE FLATTEN(TOKENIZE(line)) as woed;

```
grunt> words = foreach lines GENERATE FLATTEN(TOKENIZE(line)) as woed;
grunt> █
```

\$ grouped = GROUP words by woed;

```
grunt> grouped = GROUP words by woed;
grunt> █
```

\$ wordcount = foreach grouped GENERATE group, COUNT(words);

```
grunt> wordcount = foreach grouped GENERATE group, COUNT(words);
grunt> █
```

\$ dump wordcount;

```
grunt> dump wordcount;
2023-01-18 20:06:22,660 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP BY
2023-01-18 20:06:22,793 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]}
2023-01-18 20:06:22,961 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? false
2023-01-18 20:06:22,993 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.CombinerOptimizer - Choosing to move algebraic foreach to combiner
2023-01-18 20:06:23,039 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2023-01-18 20:06:23,039 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
```

HadoopVersion	PigVersion	UserId	StartedAt	FinishedAt	Features
2.6.0-cdh5.13.0	0.12.0-cdh5.13.0	cloudera	2023-01-18 20:06:23	2023-01-18 20:06:36	GROUP_BY

Success!

Job Stats (time in seconds):

JobId	Alias	Feature	Outputs
job_local110405554_0001	grouped,lines,wordcount,words	GROUP_BY,COMBINER	file:/tmp/temp-122151954/tmp511558588,

Input(s):

Successfully read records from: "/home/cloudera/input.csv"

Output(s):

Successfully stored records in: "file:/tmp/temp-122151954/tmp511558588"

Job DAG:

job\_local110405554\_0001

```
2023-01-18 20:06:42,590 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2023-01-18 20:06:42,856 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2023-01-18 20:06:42,856 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(a,1)
(my,1)
(to,1)
(I'm,1)
(Lad,1)
(MSc,1)
(Hello,1)
(world,1)
(Science,1)
(Shubham,1)
(Welcome,1)
(Computer,1)
(student.,1)
grunt> █
```

# Practical 04

---

**Aim:** Install HBase and use the HBase Data Model store and retrieve data.

# Start HBase

\$ hbase shell

```
[cloudera@quickstart ~]$ hbase shell
2023-01-19 21:20:50,141 INFO [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.13.0, rUnknown, Wed Oct 4 11:16:18 PDT 2017

hbase(main):001:0> █
```

//HBase Commands

\$ status

```
hbase(main):001:0> status
1 active master, 0 backup masters, 1 servers, 0 dead, 2.0000 average load
```

\$ version

```
hbase(main):002:0> version
1.2.0-cdh5.13.0, rUnknown, Wed Oct 4 11:16:18 PDT 2017
```

\$ table\_help

```
hbase(main):003:0> table_help
Help for table-reference commands.

You can either create a table via 'create' and then manipulate the table via commands like 'put', 'get', etc.
See the standard help information for how to use each of these commands.

However, as of 0.96, you can also get a reference to a table, on which you can invoke commands.
For instance, you can get create a table and keep around a reference to it via:

    hbase> t = create 't', 'cf'

Or, if you have already created the table, you can get a reference to it:

    hbase> t = get_table 't'

You can do things like call 'put' on the table:

    hbase> t.put 'r', 'cf:q', 'v'

which puts a row 'r' with column family 'cf', qualifier 'q' and value 'v' into table t.

To read the data out, you can scan the table:

    hbase> t.scan
```

\$ whoami

```
hbase(main):006:0> whoami
cloudera (auth:SIMPLE)
groups: cloudera, default
```

//Data definition Language

\$ create 'employee','Name','ID','Designation','Salary','Department'

```
hbase(main):007:0> create 'employee','Name','ID','Designation','Salary','Department'
0 row(s) in 1.6630 seconds

=> Hbase::Table - employee
hbase(main):008:0> █
```

// Verify created table

\$ list

```
hbase(main):009:0> list
TABLE
employee
1 row(s) in 0.0790 seconds

=> ["employee"]
hbase(main):010:0> █
```

// Disable single table

\$ disable 'employee'

```
hbase(main):010:0> disable 'employee'
0 row(s) in 2.5020 seconds
```

\$ scan 'employee'

```
hbase(main):011:0> scan 'employee'
ROW                                COLUMN+CELL

ERROR: employee is disabled.
```

// Enabling table

\$ enable 'employee'

```
hbase(main):018:0> enable 'employee'
0 row(s) in 1.3220 seconds
```

\$ scan 'employee'

```
hbase(main):019:0> scan 'employee'
ROW                                COLUMN+CELL
0 row(s) in 0.0090 seconds
```

//To insert record

\$ create 'student','name','age','course'

```
hbase(main):020:0> create 'student','name','age','course'
0 row(s) in 1.2500 seconds
=> Hbase::Table - student
```

\$ put 'student','shubham','name:fullname','shubham lad'

```
hbase(main):021:0> put 'student','shubham','name:fullname','shubham lad'
0 row(s) in 0.0850 seconds
```

\$ put 'student','shubham','age:presentage','21'

```
hbase(main):022:0> put 'student','shubham','age:presentage','21'
0 row(s) in 0.0080 seconds
```

\$ put 'student','shubham','course:pursuing','Hadoop'

```
hbase(main):023:0> put 'student','shubham','course:pursuing','Hadoop'
0 row(s) in 0.0050 seconds
```

\$ put 'student','mayuresh','name:fullname','mayuresh mhatre'

```
hbase(main):027:0> put 'student','mayuresh','name:fullname','mayuresh mhatre'
0 row(s) in 0.0070 seconds
```

```
$ put 'student','mayuresh','age:presentage','25'
```

```
hbase(main):028:0> put 'student','mayuresh','age:presentage','25'  
0 row(s) in 0.0060 seconds
```

```
$ put 'student','mayuresh','course:pursuing','Hadoop'
```

```
hbase(main):029:0> put 'student','mayuresh','course:pursuing','Hadoop'  
0 row(s) in 0.0070 seconds
```

```
// Get Information
```

```
$ get 'student','shubham'
```

```
hbase(main):030:0> get 'student','shubham'  
COLUMN                                CELL  
age:presentage                        timestamp=1674193412702, value=21  
course:pursuing                       timestamp=1674193451670, value=Hadoop  
name:fullname                         timestamp=1674193387146, value=shubham lad  
3 row(s) in 0.0250 seconds
```

```
$ get 'student','mayuresh'
```

```
hbase(main):001:0> get 'student','mayuresh'  
COLUMN                                CELL  
age:presentage                        timestamp=1674193535683, value=25  
course:pursuing                       timestamp=1674193550842, value=Hadoop  
name:fullname                         timestamp=1674193518601, value=mayuresh mhatre  
3 row(s) in 0.2170 seconds
```

```
$ get 'student','shubham','course'
```

```
hbase(main):002:0> get 'student','shubham','course'  
COLUMN                                CELL  
course:pursuing                       timestamp=1674193451670, value=Hadoop  
1 row(s) in 0.0100 seconds
```



\$ get 'student','mayuresh','course'

```
hbase(main):005:0> get 'student','mayuresh','course'
COLUMN                                CELL
  course:pursuing                      timestamp=1674193550842, value=Hadoop
1 row(s) in 0.0050 seconds
```

\$ get 'student','shubham','name'

```
hbase(main):006:0> get 'student','shubham','name'
COLUMN                                CELL
  name:fullname                        timestamp=1674193387146, value=shubham lad
1 row(s) in 0.0080 seconds
```

\$ scan 'student'

```
hbase(main):008:0> scan 'student'
ROW                                  COLUMN+CELL
 mayuresh                           column=age:presentage, timestamp=1674193535683, value=25
 mayuresh                           column=course:pursuing, timestamp=1674193550842, value=Hadoop
 mayuresh                           column=name:fullname, timestamp=1674193518601, value=mayuresh mhatre
 shubham                            column=age:presentage, timestamp=1674193412702, value=21
 shubham                            column=course:pursuing, timestamp=1674193451670, value=Hadoop
 shubham                            column=name:fullname, timestamp=1674193387146, value=shubham lad
2 row(s) in 0.0170 seconds
```

\$ alter 'student',NAME=>'name',VERSIONS=>5

```
hbase(main):013:0> alter 'student',NAME=>'name',VERSIONS=>5
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9250 seconds
```

\$ put 'student','mayuresh','name:fullname','mayuresh rao'

```
hbase(main):014:0> put 'student','mayuresh','name:fullname','mayuresh rao'
0 row(s) in 0.0460 seconds
```

\$ scan 'student'

```
hbase(main):015:0> scan 'student'
ROW                                COLUMN+CELL
mayuresh                          column=age:presentage, timestamp=1674193535683, value=25
mayuresh                          column=course:pursuing, timestamp=1674193550842, value=Hadoop
mayuresh                          column=name:fullname, timestamp=1674194392029, value=mayuresh rao
shubham                            column=age:presentage, timestamp=1674193412702, value=21
shubham                            column=course:pursuing, timestamp=1674193451670, value=Hadoop
shubham                            column=name:fullname, timestamp=1674193387146, value=shubham lad
2 row(s) in 0.0450 seconds
```

# Deletion of record

\$ delete 'student','mayuresh','name:fullname'

```
hbase(main):016:0> delete 'student','mayuresh','name:fullname'
0 row(s) in 0.0540 seconds
```

\$ scan 'student'

```
hbase(main):017:0> scan 'student'
ROW                                COLUMN+CELL
mayuresh                          column=age:presentage, timestamp=1674193535683, value=25
mayuresh                          column=course:pursuing, timestamp=1674193550842, value=Hadoop
shubham                            column=age:presentage, timestamp=1674193412702, value=21
shubham                            column=course:pursuing, timestamp=1674193451670, value=Hadoop
shubham                            column=name:fullname, timestamp=1674193387146, value=shubham lad
2 row(s) in 0.0130 seconds
```



## Practical 05

---

**Aim:** Install Hive and use Hive Create and store structured databases

**Steps:**

```
$ cat>/home/cloudera/employee.txt
```

```
1~Sachin~Pune~Product Enginnering~100000~Big Data
```

```
2~Gaurav~Bangalore~Sales~90000~CRM
```

```
3~Manish~Chennai~Recruiter~125000~HR
```

```
4~Bhushan~Hyderabad~Developer~50000~BFSI
```

```
[cloudera@quickstart ~]$ cat>/home/cloudera/employee.txt
1~Sachin~Pune~Product Enginnering~100000~Big Data
2~Gaurav~Bangalore~Sales~90000~CRM
3~Manish~Chennai~Recruiter~125000~HR
4~Bhushan~Hyderabad~Developer~50000~BFSI
^C
```

```
$ cat /home/cloudera/employee.txt
```

```
[cloudera@quickstart ~]$ cat /home/cloudera/employee.txt
1~Sachin~Pune~Product Enginnering~100000~Big Data
2~Gaurav~Bangalore~Sales~90000~CRM
3~Manish~Chennai~Recruiter~125000~HR
4~Bhushan~Hyderabad~Developer~50000~BFSI
[cloudera@quickstart ~]$ █
```

```
$ sudo -u hdfs hadoop fs -put /home/cloudera/employee.txt /inputdirectory
```

```
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -put /home/cloudera/employee.txt /inputdirectory
[cloudera@quickstart ~]$ █
```

```
$ hdfs dfs -ls
```

```
[cloudera@quickstart ~]$ hdfs dfs -ls
```

\$ hdfs dfs -ls /inputdirectory

```
[cloudera@quickstart ~]$ hdfs dfs -ls /inputdirectory
Found 2 items
-rw-r--r--  1 hdfs supergroup      76 2023-01-05 22:18 /inputdirectory/Processfile.txt
-rw-r--r--  1 hdfs supergroup    163 2023-01-19 22:23 /inputdirectory/employee.txt
```

\$ hadoop fs -cat /inputdirectory/employee.txt

```
[cloudera@quickstart ~]$ hadoop fs -cat /inputdirectory/employee.txt
1~Sachin~Pune~Product Enginnering~100000~Big Data
2~Gaurav~Bangalore~Sales~90000~CRM
3~Manish~Chennai~Recruiter~125000~HR
4~Bhushan~Hyderabad~Developer~50000~BFSI
[cloudera@quickstart ~]$
```

\$ hive

```
[cloudera@quickstart ~]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
```

\$ show databases;

```
hive> show databases;
OK
default
Time taken: 6.472 seconds, Fetched: 1 row(s)
hive>
```

\$ create database organization;

```
hive> create database organization;
OK
Time taken: 2.367 seconds
```

\$ show databases;

```
hive> show databases;
OK
default
organization
Time taken: 0.034 seconds, Fetched: 2 row(s)
hive>
```

\$ use organization;

```
hive> use organization;  
OK  
Time taken: 0.104 seconds
```

\$ show tables;

```
hive> show tables;  
OK  
Time taken: 0.084 seconds
```

\$ create table employee(

- > id int,
- > name string,
- > city string,
- > department string,
- > salary int,
- > domain string)
- > row format delimited
- > fields terminated by '~';

```
hive> create table employee(  
  > id int,  
  > name string,  
  > city string,  
  > department string,  
  > salary int,  
  > domain string)  
  > row format delimited  
  > fields terminated by '~';  
OK  
Time taken: 0.607 seconds
```

\$ show tables;

\$ select \* from employee;

\$ show tables;

```
hive> show tables;
OK
employee
Time taken: 0.023 seconds, Fetched: 1 row(s)
hive> select * from employee;
OK
Time taken: 0.618 seconds
hive> show tables;
OK
employee
Time taken: 0.014 seconds, Fetched: 1 row(s)
hive> █
```

\$ load data inpath '/inputdirectory/employee.txt' overwrite into table employee;

```
hive> load data inpath '/inputdirectory/employee.txt' overwrite into table employee;
Loading data to table organization.employee
chmod: changing permissions of 'hdfs://quickstart.cloudera:8020/user/hive/warehouse/organization.db/employee/employee.txt':
dera is not the owner of inode=employee.txt
Table organization.employee stats: [numFiles=1, numRows=0, totalSize=163, rawDataSize=0]
OK
Time taken: 0.555 seconds
```

\$ show tables;

```
hive> show tables;
OK
employee
Time taken: 0.025 seconds, Fetched: 1 row(s)
```

\$ select \* from employee;

```
hive> select * from employee;
OK
1      Sachin  Pune      Product Enginnering    100000  Big Data
2      Gaurav  Bangalore Sales      90000   CRM
3      Manish  Chennai  Recruiter    125000  HR
4      Bhushan Hyderabad Developer    50000   BFSI
Time taken: 0.06 seconds, Fetched: 4 row(s)
```

## Practical 06

---

**Aim:** Write a program to construct different types of shingles for given document.

Commands for installation of required packages before executing program:

```
> install.packages("tm")
> require("tm")
> install.packages("devtools")
```

Code:

```
readinteger <- function()
{
  n <- readline(prompt="Enter value of k-1: ")
  k<-as.integer(n)
  u1 <- readLines("File1.txt")
  Shingle<-0
  i <-0
  while(i<nchar(u1)-k+1){
    Shingle[i] <- substr(u1, start=i, stop=i+k)
    print(Shingle[i])
    i=i+1
  }
}
if(interactive()) readinteger()
```

File1.txt:

Life is not a problem to be solved, but a reality to be experienced.

Output:

```
Enter value of k-1: 5
character(0)
[1] "Life i"
[1] "ife is"
[1] "fe is "
[1] "e is n"
[1] " is no"
[1] "is not"
[1] "s not "
[1] " not a"
[1] "not a "
[1] "ot a p"
[1] "t a pr"
[1] " a pro"
[1] "a prob"
[1] " probl"
[1] "proble"
[1] "roblem"
[1] "oblem "
[1] "blem t"
[1] "lem to"
[1] "em to "
[1] "m to b"
```

```
[1] " to be"
[1] "to be "
[1] "o be s"
[1] " be so"
[1] "be sol"
[1] "e solv"
[1] " solve"
[1] "solved"
[1] "olved,"
[1] "lved, "
[1] "ved, b"
[1] "ed, bu"
[1] "d, but"
[1] ", but "
[1] " but a"
[1] "but a "
[1] "ut a r"
[1] "t a re"
[1] " a rea"
[1] "a real"
[1] " reali"
[1] "realit"
[1] "eality"
[1] "ality "
[1] "lity t"
[1] "ity to"
[1] "ty to "
[1] "y to b"
[1] " to be"
[1] "to be "
[1] "o be e"
[1] " be ex"
[1] "be exp"
[1] "e expe"
[1] " exper"
[1] "experi"
[1] "xperie"
[1] "perien"
[1] "erience"
[1] "rience"
[1] "ienced"
[1] "enced."
```

## Practical 07

---

**Aim:** Write a program for measuring similarity among documents and detecting passages which have been reused.

Commands for installation of required packages before executing program:

```
> install.packages("tm")
> require("tm")
> install.packages("devtools")
> install.packages("ggplot2")
> install.packages("textreuse")
```

**Code:**

```
> my.corpus <- Corpus(DirSource("C:/Users/shubh/OneDrive/Documents/R
Studio"))

> my.corpus <- tm_map(my.corpus, removewords, stopwords("english"))

> my.tdm <- TermDocumentMatrix(my.corpus)

> my.dtm <- DocumentTermMatrix(my.corpus, control = list(weighting =
weightTfIdf, stopwords = TRUE))

> my.df <- as.data.frame(inspect(my.tdm))
<-TermDocumentMatrix (terms: 21, documents: 3)>>
Non-/sparse entries: 22/41
Sparsity           : 65%
Maximal term length: 12
weighting          : term frequency (tf)
Sample            :
Terms              Docs
File1.txt File2.txt File3.txt
experienced.      1      0      0
goal,             0      1      0
happy            0      1      0
life             1      0      1
life,            0      1      0
live             0      1      0
people           0      1      0
problem          1      0      0
reality          1      0      0
solved,          1      0      0

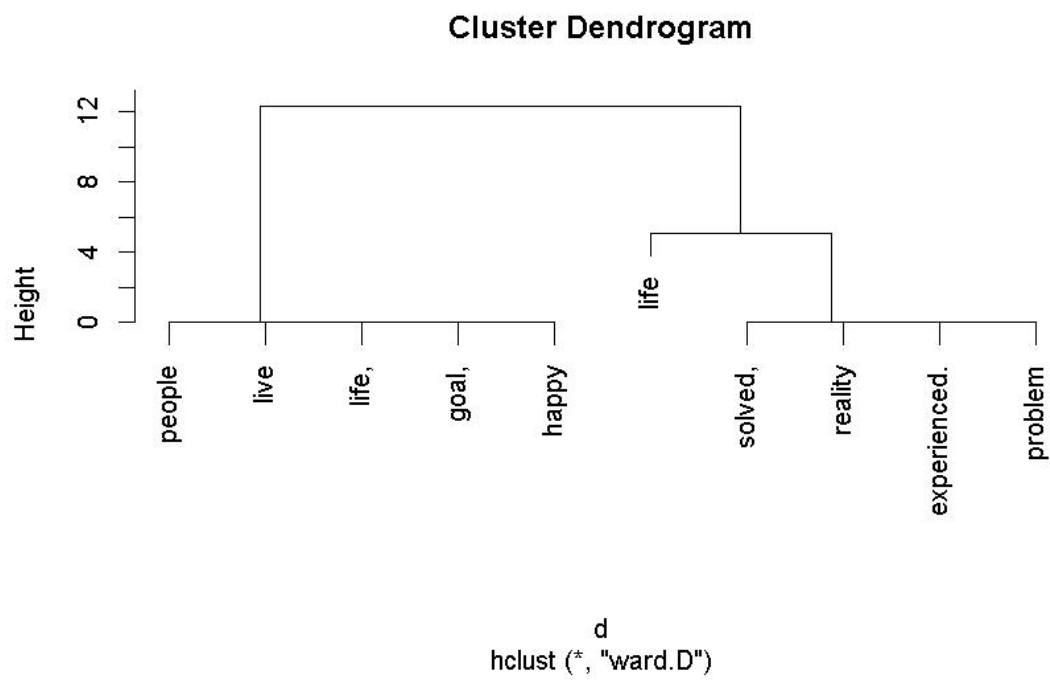
> my.df.scale <- scale(my.df)

> d <- dist(my.df.scale,method="euclidean")

> fit <- hclust(d, method="ward")

> plot(fit)
```



**Output:**

## Practical 08

---

**Aim:** Write a program to compute the n-moment for a given stream where n is given.

**Code:**

```
package moment;

import java.util.ArrayList;
import java.util.Arrays;

public class Moment {

    public static void main(String[] args) {

        String stream[]={"a","b","c","b","d","a","c","d","a","b","d","c","a","a","b"};

        int n=15;

        int zero_moment=0,first_moment=0,second_moment=0,count=1,flag=0;

        ArrayList<Integer> arrlist=new ArrayList();

        System.out.println("Arraylist elements are: ");

        for (int i=0;i<15;i++){

            System.out.println(stream[i]+ " ");

        }

        Arrays.sort(stream);

        for (int i=1;i<n;i++){

            if (stream[i]==stream[i-1]){

                count++;

            }

            else {

                arrlist.add(count);

                count=1;

            }

        }

        arrlist.add(count);

        zero_moment=arrlist.size();

    }

}
```

```
System.out.println("\nZeroth moment for the given stream is: "+zero_moment);

for (int i=0;i<arrlist.size();i++){
    first_moment+=arrlist.get(i);
}

System.out.println("\nFirst moment for the given stream is: "+first_moment);

for (int i=0;i<arrlist.size();i++){
    int j=arrlist.get(i);
    second_moment+=(j*j);
}

System.out.println("\nSecond moment for the given stream is: "+second_moment);
}
```

### Output:

```
Arraylist elements are:
a
b
c
b
d
a
c
d
a
b
d
c
a
a
b

Zeroth moment for the given stream is: 4

First moment for the given stream is: 15

Second moment for the given stream is: 59
BUILD SUCCESSFUL (total time: 0 seconds)
|
```

## Practical 09

---

**Aim:** Write a program to demonstrate the Alon-Matias-Szegedy Algorithm for second moments.

**Code:**

```
package amsa;

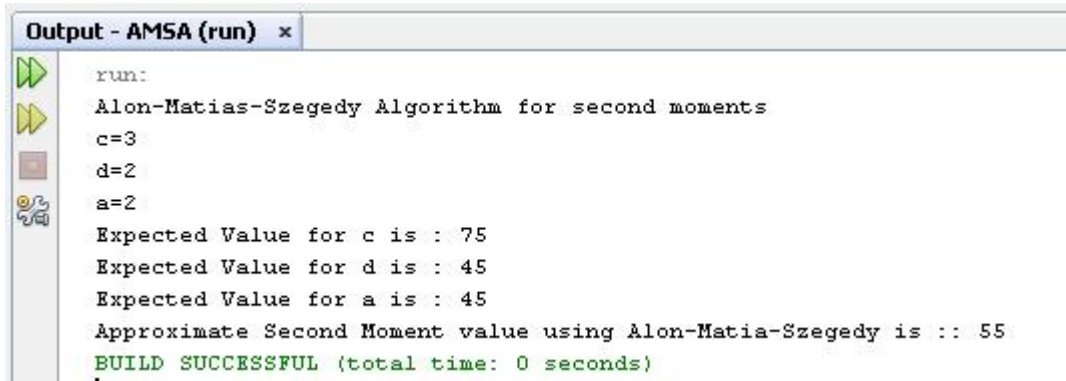
/**
 *
 * @author shubham
 */
public class AMSA {
    /**
     * @param stream
     * @param XE
     * @param random
     * @param n
     * @return
     */
    public static int findCharCount(String stream, char XE, int random, int n){
        int countOccurance=0;
        for (int i = random; i<n; i++){
            if (stream.charAt(i)==XE){
                countOccurance++;
            }
        }
        return countOccurance;
    }
}
```

```
public static int estimateValue(int XV1,int n){
    int ExpValue;
    ExpValue=n*(2*XV1-1);
    return ExpValue;
}

public static void main(String[] args) {
    System.out.println("Alon-Matias-Szegedy Algorithm for second moments");
    String stream="abcbdacdabdcaab";
    int n = stream.length();
    int random1=3,random2=8,random3=13;
    char XE1,XE2,XE3;
    int XV1,XV2,XV3;
    int ExpValuXE1,ExpValuXE2,ExpValuXE3;
    int apprSecondMomentValue;
    XE1=stream.charAt(random1-1);
    XE2=stream.charAt(random2-1);
    XE3=stream.charAt(random3-1);
    XV1=findCharCount(stream,XE1,random1-1,n);
    XV2=findCharCount(stream,XE2,random2-1,n);
    XV3=findCharCount(stream,XE3,random3-1,n);
    System.out.println(XE1+"="+XV1+"\n"+XE2+"="+XV2+"\n"+XE3+"="+XV3);
    ExpValuXE1=estimateValue(XV1,n);
    ExpValuXE2=estimateValue(XV2,n);
    ExpValuXE3=estimateValue(XV3,n);
    System.out.println("Expected Value for "+XE1+" is : "+ExpValuXE1);
    System.out.println("Expected Value for "+XE2+" is : "+ExpValuXE2);
    System.out.println("Expected Value for "+XE3+" is : "+ExpValuXE3);
    apprSecondMomentValue=(ExpValuXE1+ExpValuXE2+ExpValuXE3)/3;
    System.out.println("Approximate Second Moment value using Alon-Matia-Szegedy is ::
"+apprSecondMomentValue);
```

```
}  
}
```

### Output:



```
run:  
Alon-Matias-Szegedy Algorithm for second moments  
c=3  
d=2  
a=2  
Expected Value for c is : 75  
Expected Value for d is : 45  
Expected Value for a is : 45  
Approximate Second Moment value using Alon-Matia-Szegedy is :: 55  
BUILD SUCCESSFUL (total time: 0 seconds)
```