Practical No: 2

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

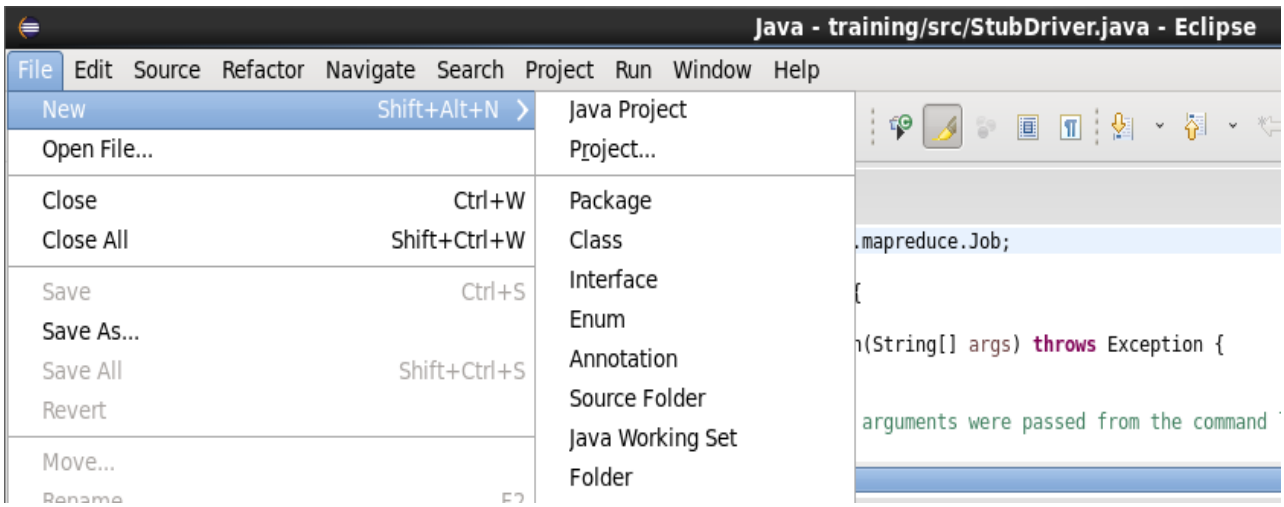
Description:

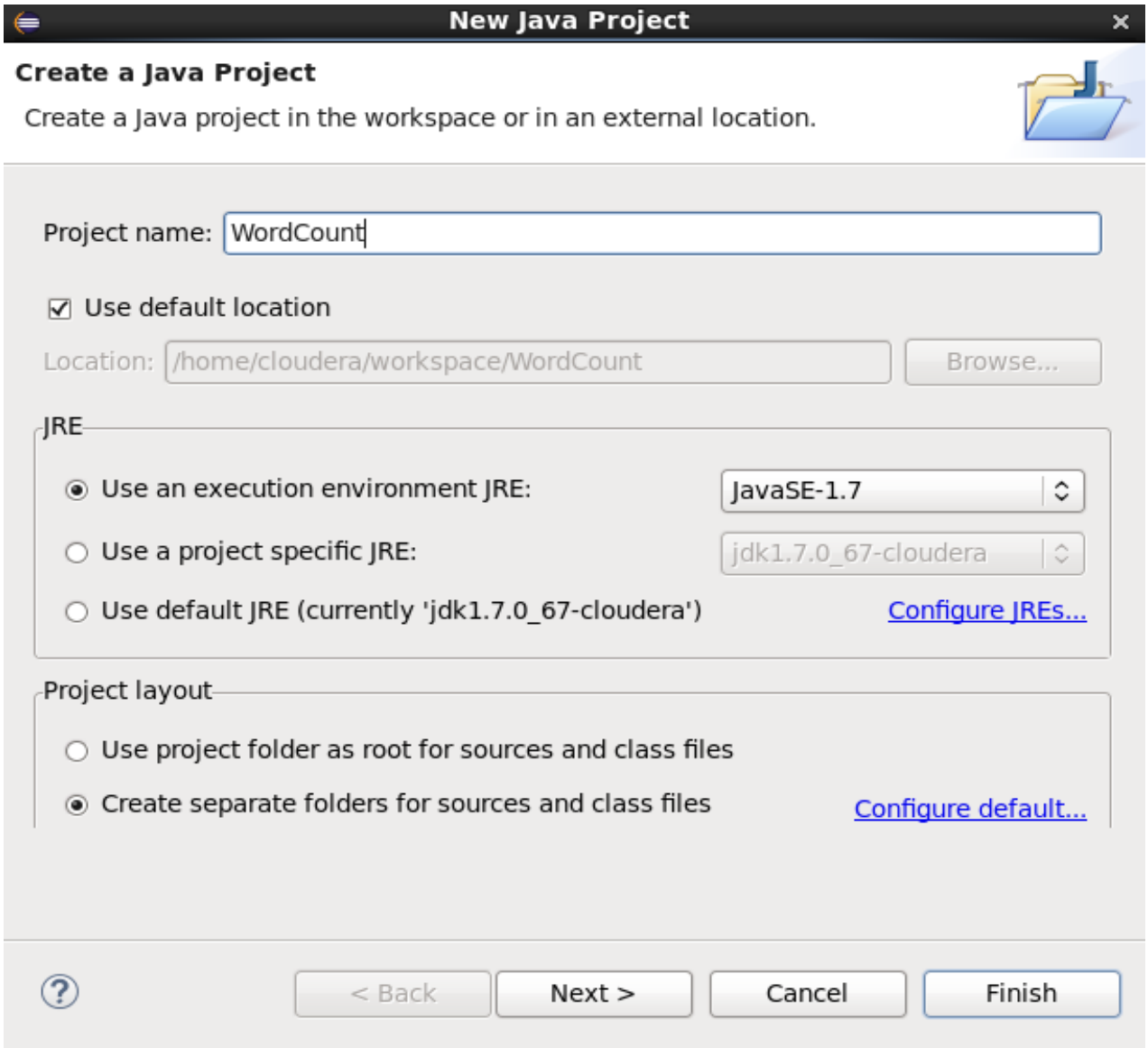
Map-Reduce is a software programming model designed in java programming language. It is combination of two individual task namely:

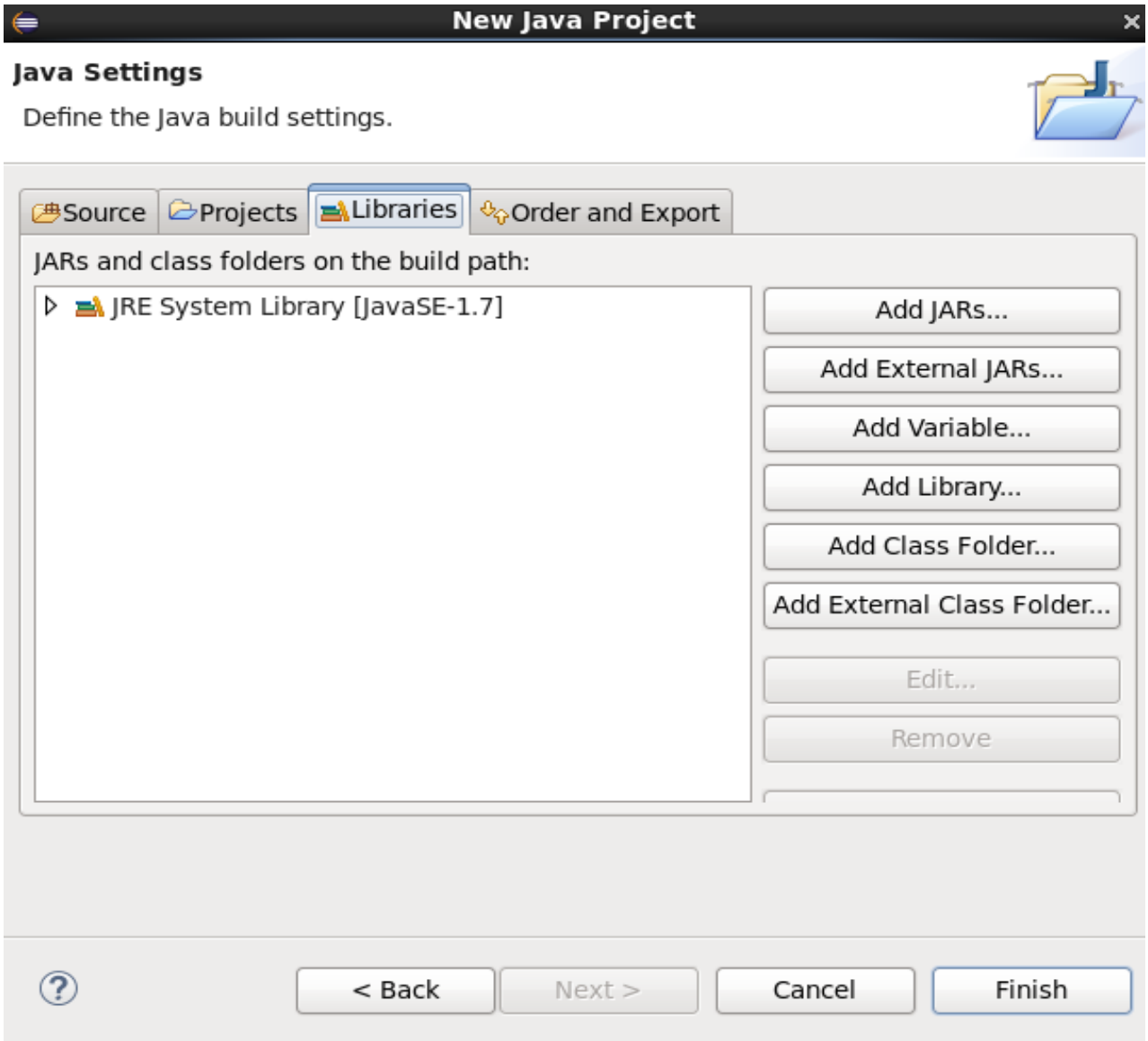
1.Map: Map takes the dataset and divides them into chunks such that they are converted into new format which should be in form of key-value pair.

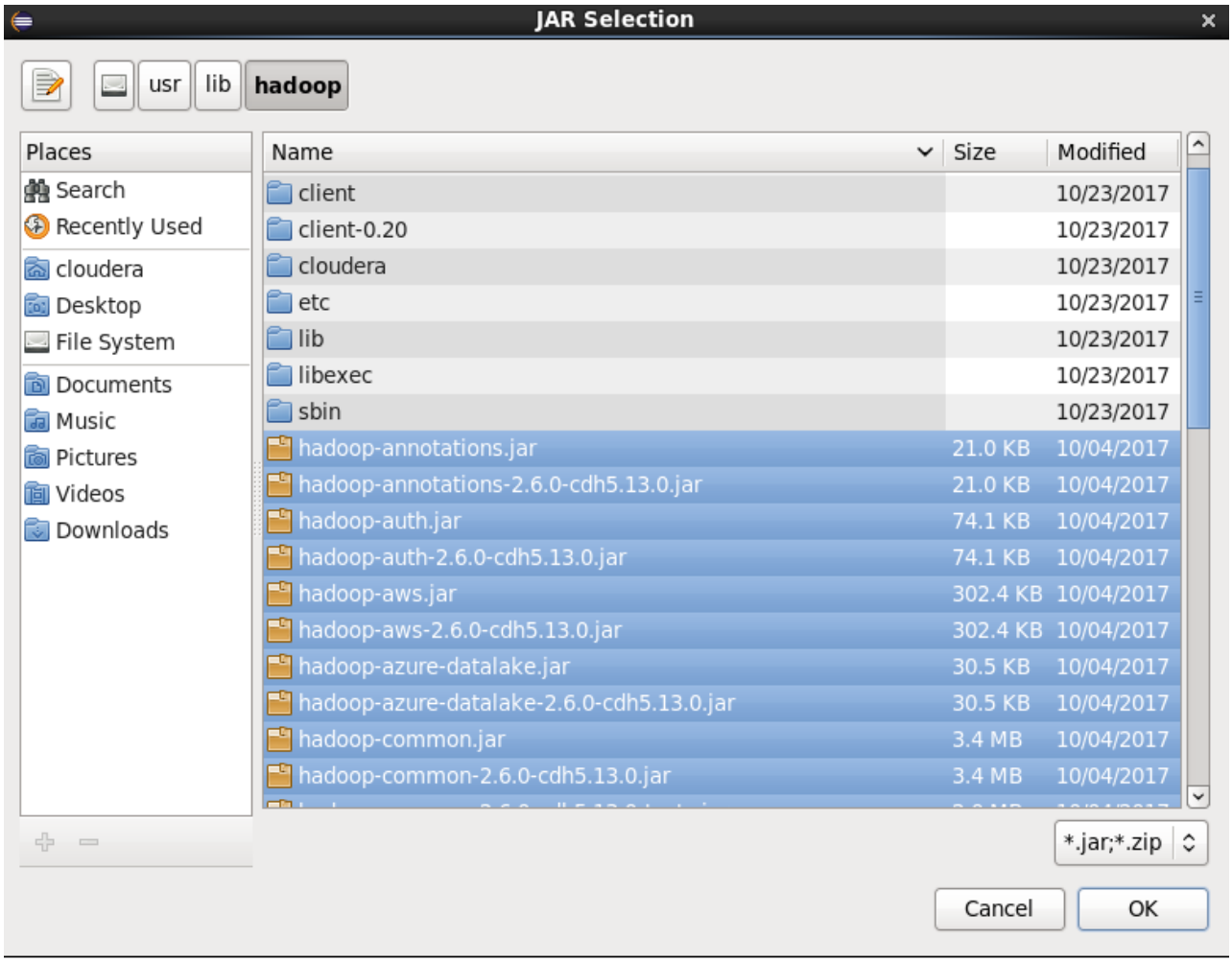
2.Reduce: Reduce is another part where key-value pair reduced to tuples.

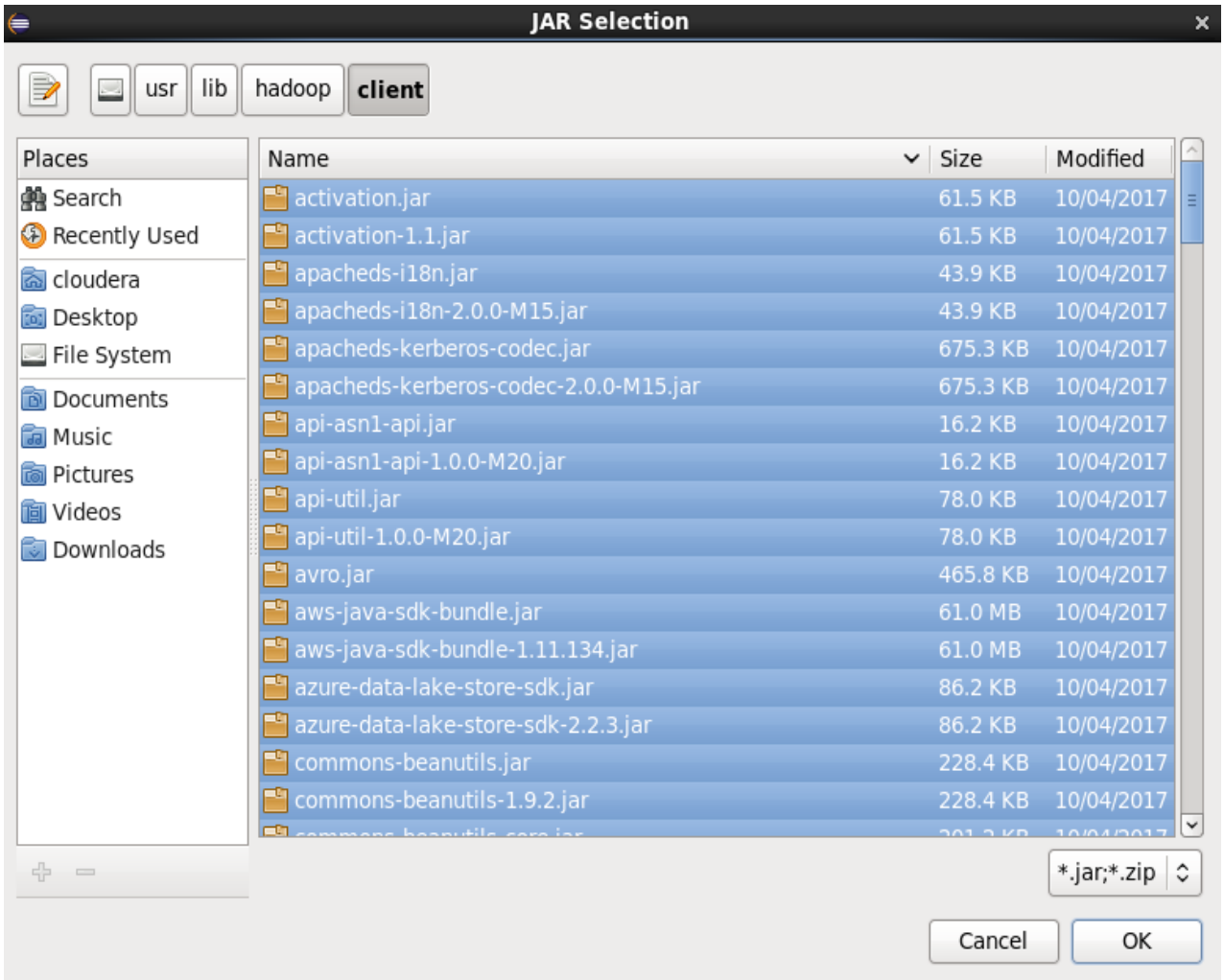
Steps & Output:

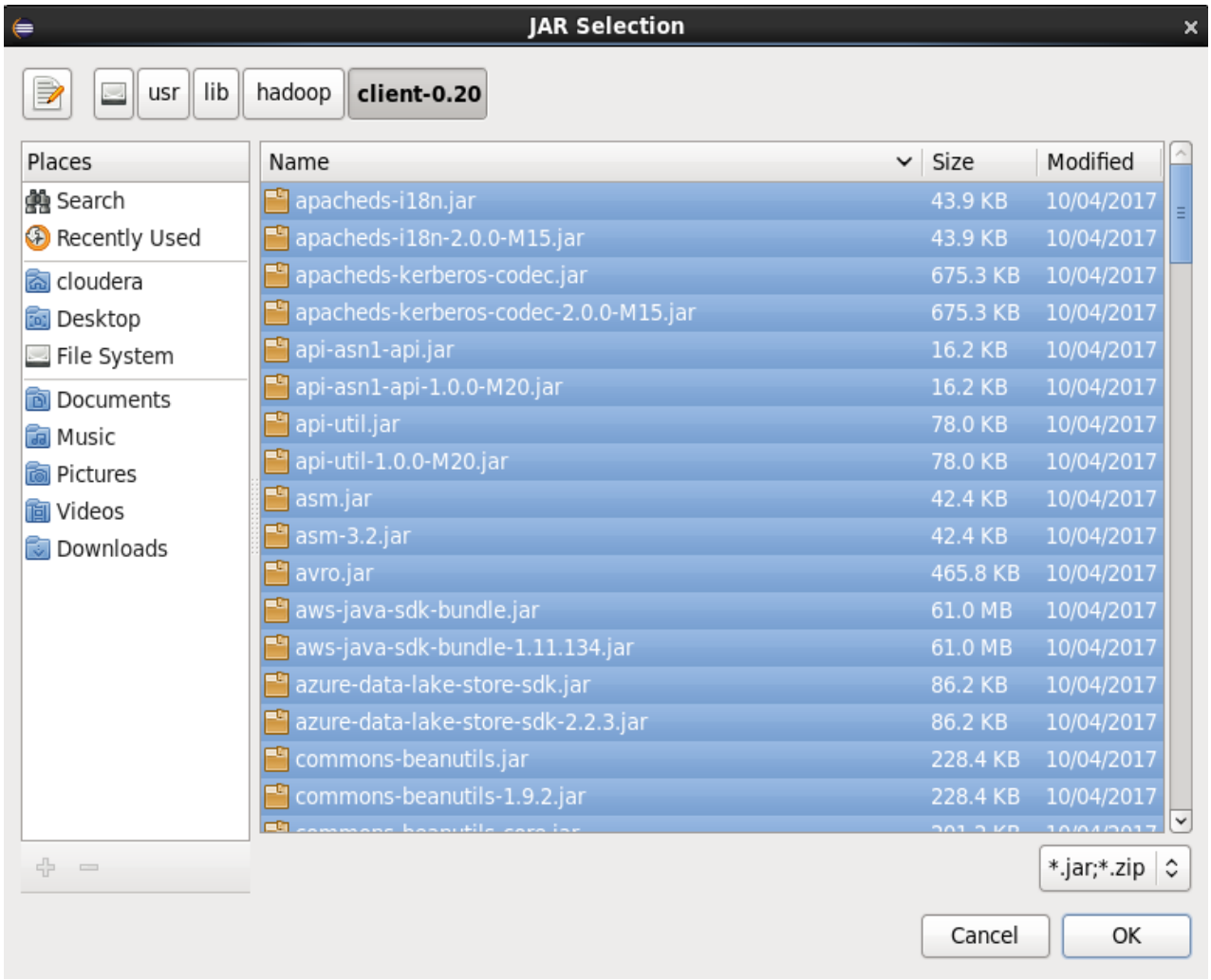


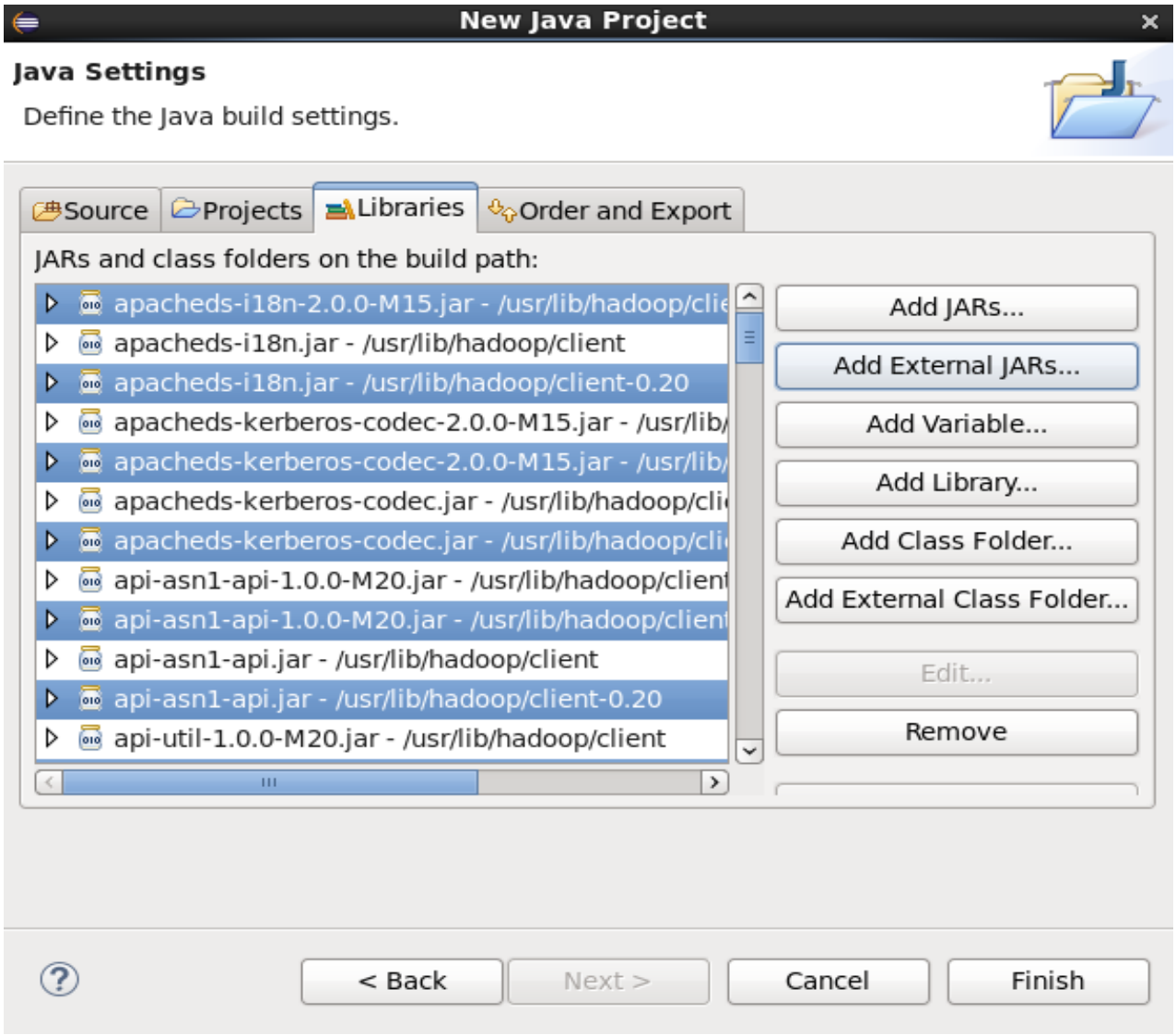

New Java Project
Create a Java Project
Create a Java project in the workspace or in an external location.
Project name: WordCount
Use default location
Location: /home/cloudera/workspace/WordCount
Browse...
JRE
Use an execution environment JRE:
JavaSE-1.7
Use a project specific JRE:
jdk1.7.0_67-cloudera
Use default JRE (currently 'jdk1.7.0_67-cloudera')
Configure JREs...
Project layout
Use project folder as root for sources and class files
Create separate folders for sources and class files
Configure default...
< Back
Next >
Cancel
Finish

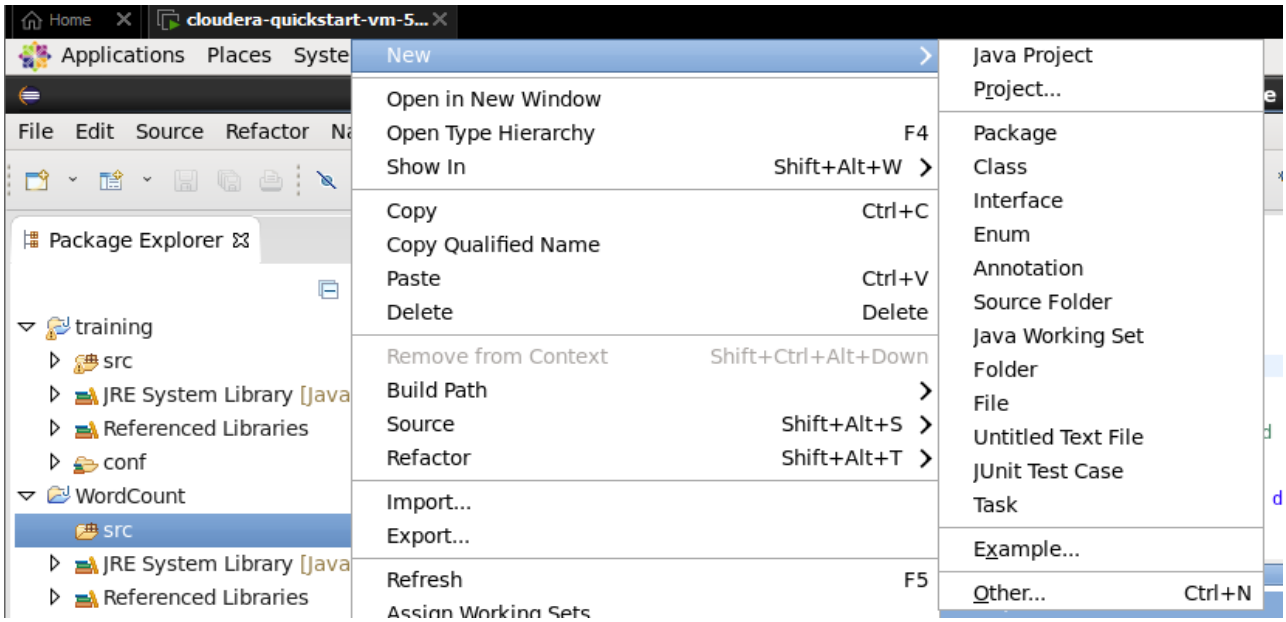

New Java Project
Java Settings
Define the Java build settings.
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
JRE System Library [JavaSE-1.7]
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
< Back
Next >
Cancel
Finish

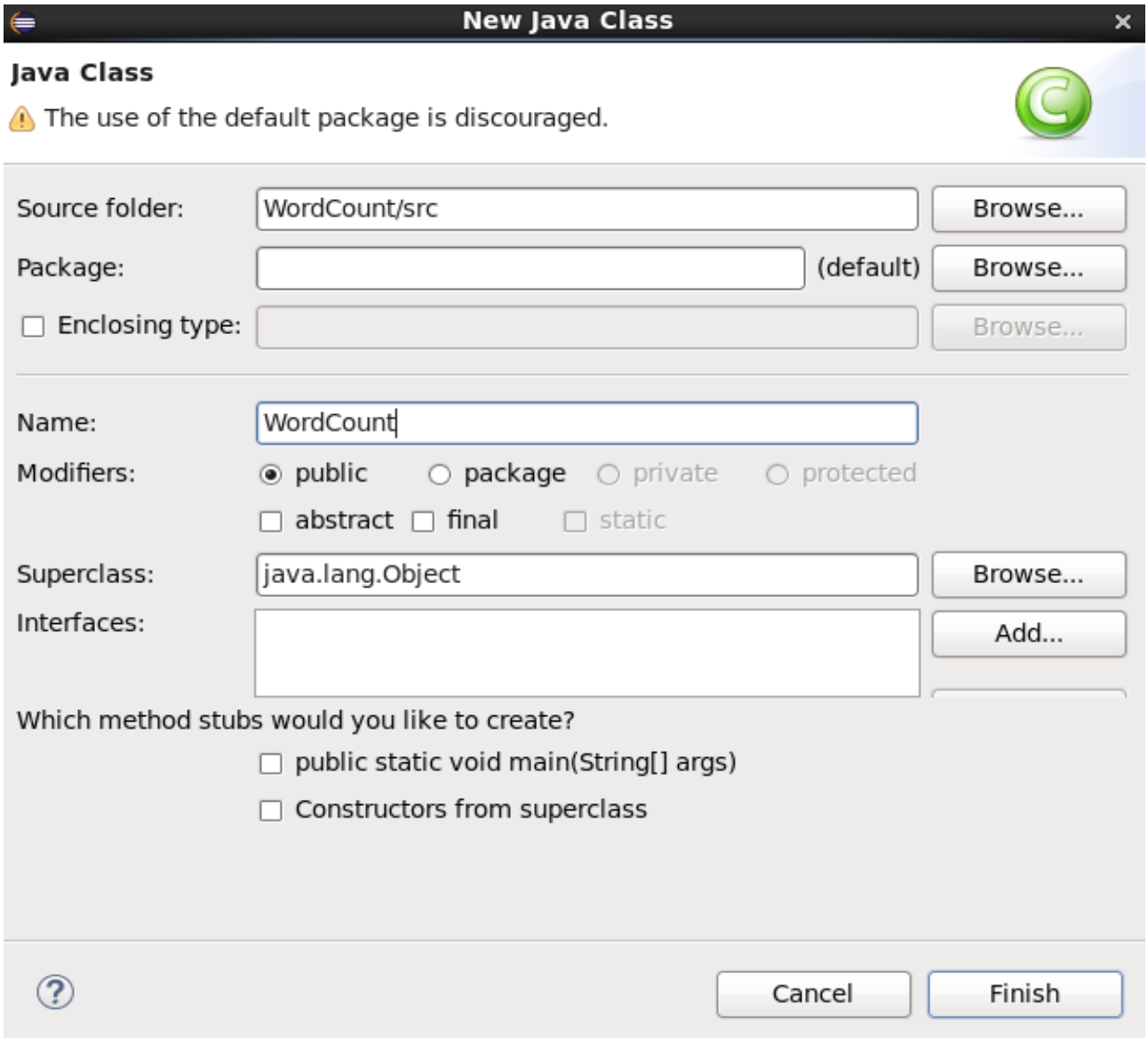

JAR Selection
usr
lib
hadoop
Places
Search
Recently Used
cloudera
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
client 10/23/2017
client-0.20 10/23/2017
cloudera 10/23/2017
etc 10/23/2017
lib 10/23/2017
libexec 10/23/2017
sbin 10/23/2017
hadoop-annotations.jar 21.0 KB 10/04/2017
hadoop-annotations-2.6.0-cdh5.13.0.jar 21.0 KB 10/04/2017
hadoop-auth.jar 74.1 KB 10/04/2017
hadoop-auth-2.6.0-cdh5.13.0.jar 74.1 KB 10/04/2017
hadoop-aws.jar 302.4 KB 10/04/2017
hadoop-aws-2.6.0-cdh5.13.0.jar 302.4 KB 10/04/2017
hadoop-azure-datalake.jar 30.5 KB 10/04/2017
hadoop-azure-datalake-2.6.0-cdh5.13.0.jar 30.5 KB 10/04/2017
hadoop-common.jar 3.4 MB 10/04/2017
hadoop-common-2.6.0-cdh5.13.0.jar 3.4 MB 10/04/2017
.jar;.zip
Cancel
OK


JAR Selection
usr
lib
hadoop
client
Places
Search
Recently Used
cloudera
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
activation.jar 61.5 KB 10/04/2017
activation-1.1.jar 61.5 KB 10/04/2017
apacheds-i18n.jar 43.9 KB 10/04/2017
apacheds-i18n-2.0.0-M15.jar 43.9 KB 10/04/2017
apacheds-kerberos-codec.jar 675.3 KB 10/04/2017
apacheds-kerberos-codec-2.0.0-M15.jar 675.3 KB 10/04/2017
api-asn1-api.jar 16.2 KB 10/04/2017
api-asn1-api-1.0.0-M20.jar 16.2 KB 10/04/2017
api-util.jar 78.0 KB 10/04/2017
api-util-1.0.0-M20.jar 78.0 KB 10/04/2017
avro.jar 465.8 KB 10/04/2017
aws-java-sdk-bundle.jar 61.0 MB 10/04/2017
aws-java-sdk-bundle-1.11.134.jar 61.0 MB 10/04/2017
azure-data-lake-store-sdk.jar 86.2 KB 10/04/2017
azure-data-lake-store-sdk-2.2.3.jar 86.2 KB 10/04/2017
commons-beanutils.jar 228.4 KB 10/04/2017
commons-beanutils-1.9.2.jar 228.4 KB 10/04/2017
.jar;.zip
Cancel
OK


JAR Selection
usr
lib
hadoop
client-0.20
Places
Search
Recently Used
cloudera
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
apacheds-i18n.jar 43.9 KB 10/04/2017
apacheds-i18n-2.0.0-M15.jar 43.9 KB 10/04/2017
apacheds-kerberos-codec.jar 675.3 KB 10/04/2017
apacheds-kerberos-codec-2.0.0-M15.jar 675.3 KB 10/04/2017
api-asn1-api.jar 16.2 KB 10/04/2017
api-asn1-api-1.0.0-M20.jar 16.2 KB 10/04/2017
api-util.jar 78.0 KB 10/04/2017
api-util-1.0.0-M20.jar 78.0 KB 10/04/2017
asm.jar 42.4 KB 10/04/2017
asm-3.2.jar 42.4 KB 10/04/2017
avro.jar 465.8 KB 10/04/2017
aws-java-sdk-bundle.jar 61.0 MB 10/04/2017
aws-java-sdk-bundle-1.11.134.jar 61.0 MB 10/04/2017
azure-data-lake-store-sdk.jar 86.2 KB 10/04/2017
azure-data-lake-store-sdk-2.2.3.jar 86.2 KB 10/04/2017
commons-beanutils.jar 228.4 KB 10/04/2017
commons-beanutils-1.9.2.jar 228.4 KB 10/04/2017
.jar;.zip
Cancel
OK

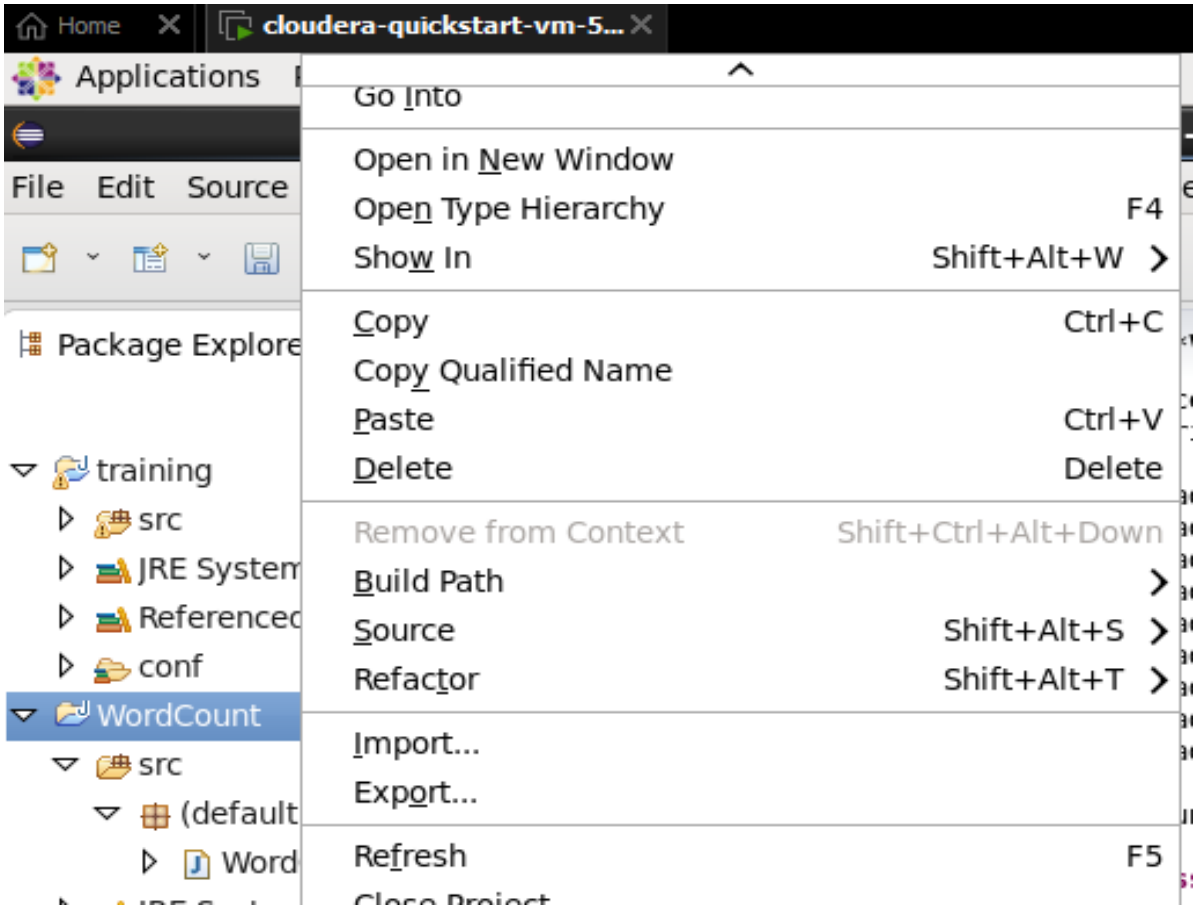

New Java Project
Java Settings
Define the Java build settings.
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
apacheds-i18n-2.0.0-M15.jar - /usr/lib/hadoop/clie
apacheds-i18n.jar - /usr/lib/hadoop/client
apacheds-i18n.jar - /usr/lib/hadoop/client-0.20
apacheds-kerberos-codec-2.0.0-M15.jar - /usr/lib/
apacheds-kerberos-codec-2.0.0-M15.jar - /usr/lib/
apacheds-kerberos-codec.jar - /usr/lib/hadoop/cli
apacheds-kerberos-codec.jar - /usr/lib/hadoop/cli
api-asn1-api-1.0.0-M20.jar - /usr/lib/hadoop/client
api-asn1-api-1.0.0-M20.jar - /usr/lib/hadoop/client
api-asn1-api.jar - /usr/lib/hadoop/client
api-asn1-api.jar - /usr/lib/hadoop/client-0.20
api-util-1.0.0-M20.jar - /usr/lib/hadoop/client
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
< Back
Next >
Cancel
Finish

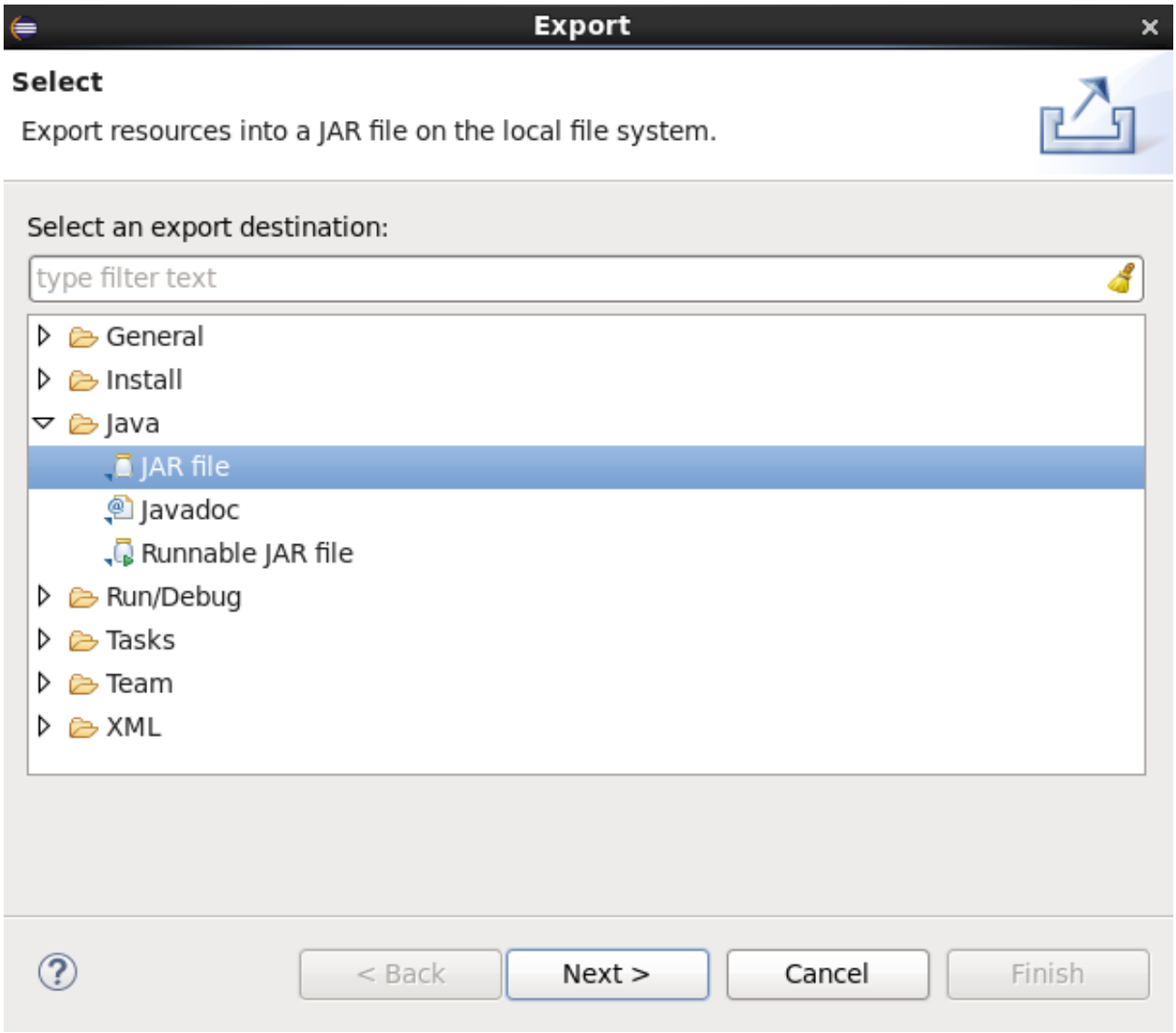

Home
cloudera-quickstart-vm-5...
Applications Places Syste
File Edit Source Refactor Na
Package Explorer
training
src
JRE System Library [Java
Referenced Libraries
conf
WordCount
src
JRE System Library [Java
Referenced Libraries
New
Open in New Window
Open Type Hierarchy F4
Show In Shift+Alt+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Shift+Ctrl+Alt+Down
Build Path
Source Shift+Alt+S
Refactor Shift+Alt+T
Import...
Export...
Refresh F5
Assign Working Sets
Java Project
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Java Working Set
Folder
File
Untitled Text File
JUnit Test Case
Task
Example...
Other... Ctrl+N

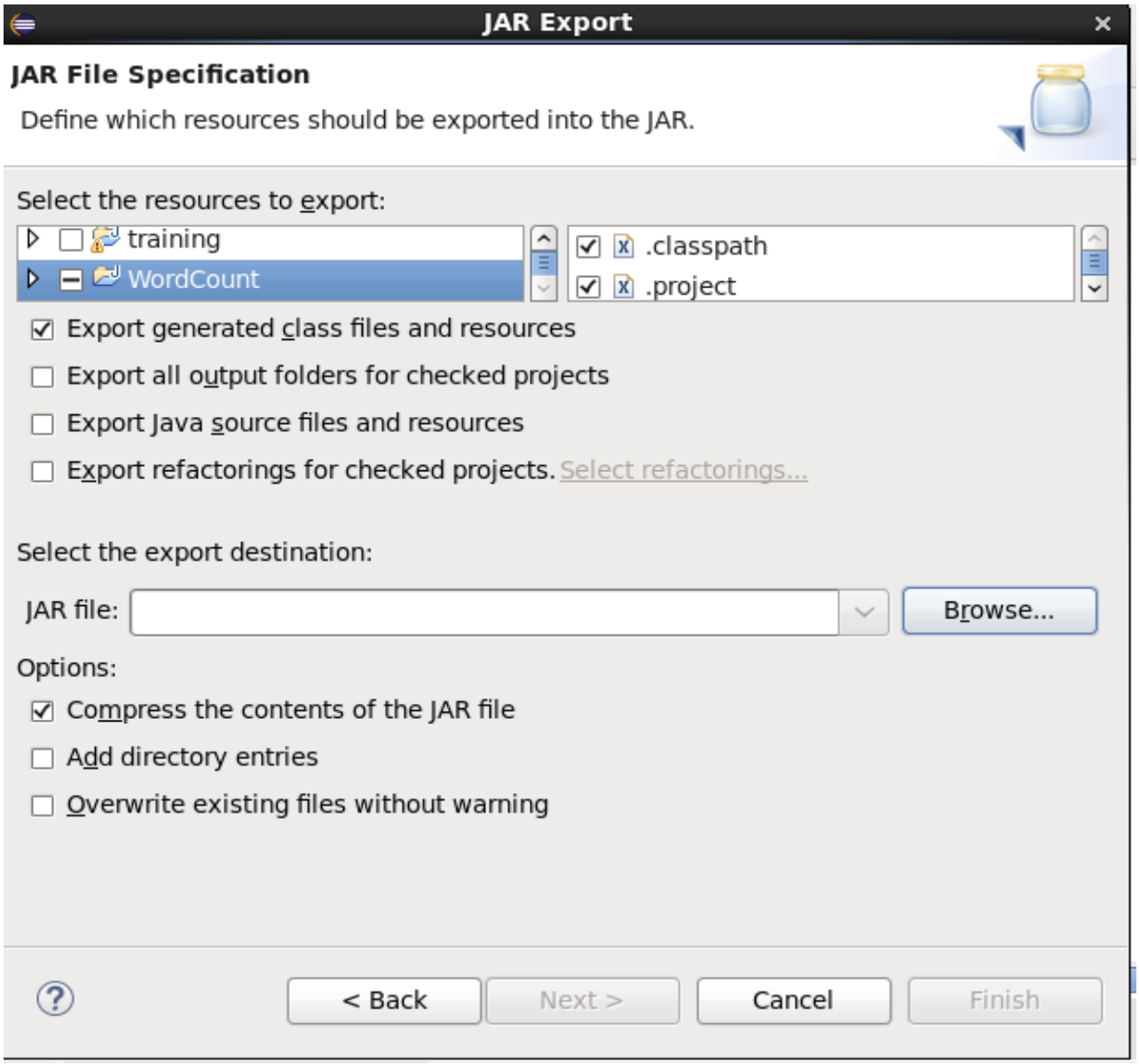

New Java Class
Java Class
The use of the default package is discouraged.
Source folder:
WordCount/src
Browse...
Package:
(default)
Browse...
Enclosing type:
Browse...
Name:
WordCount
Modifiers:
public
package
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Cancel
Finish

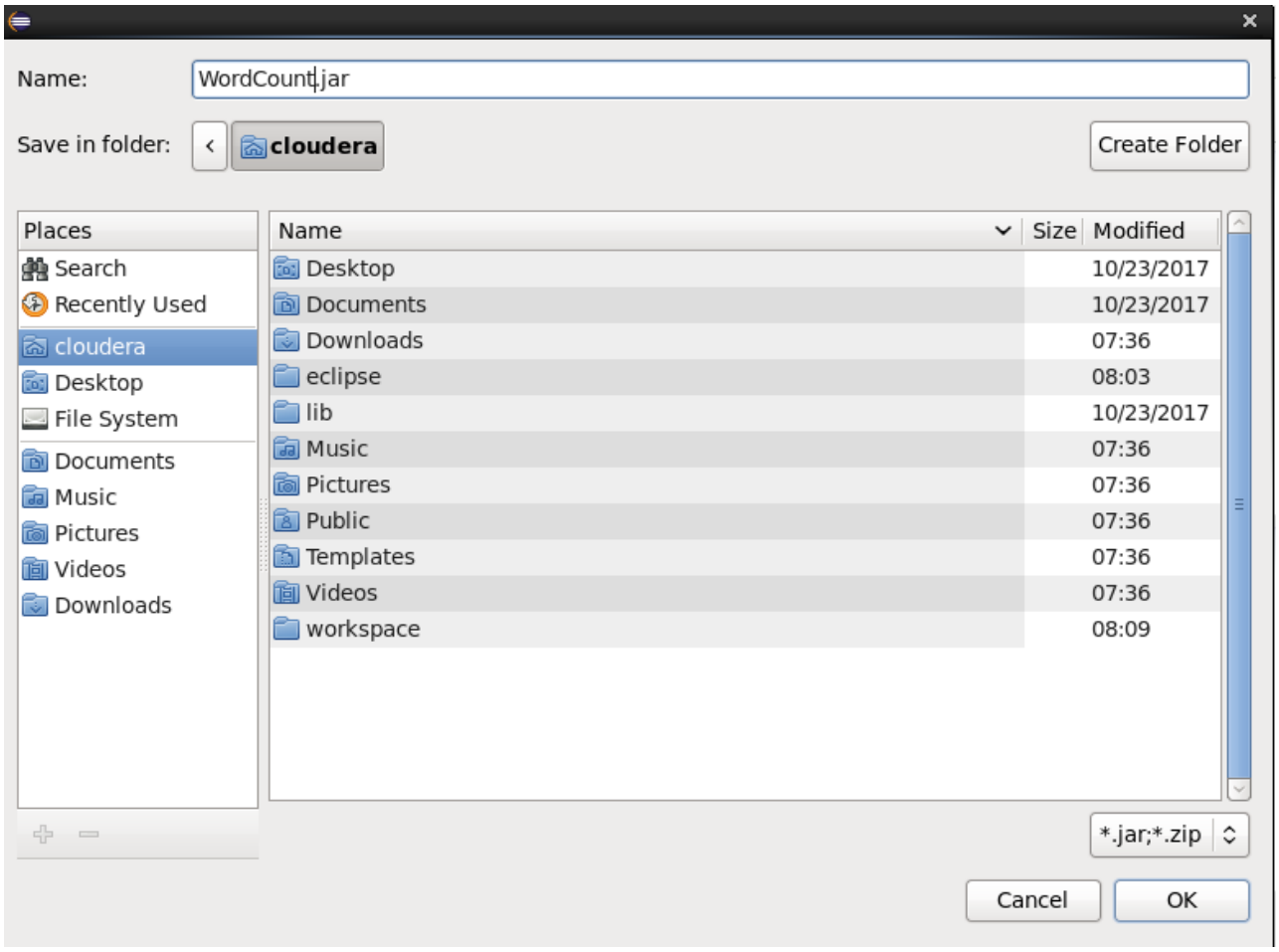
StubDriver.java | *WordCount.java

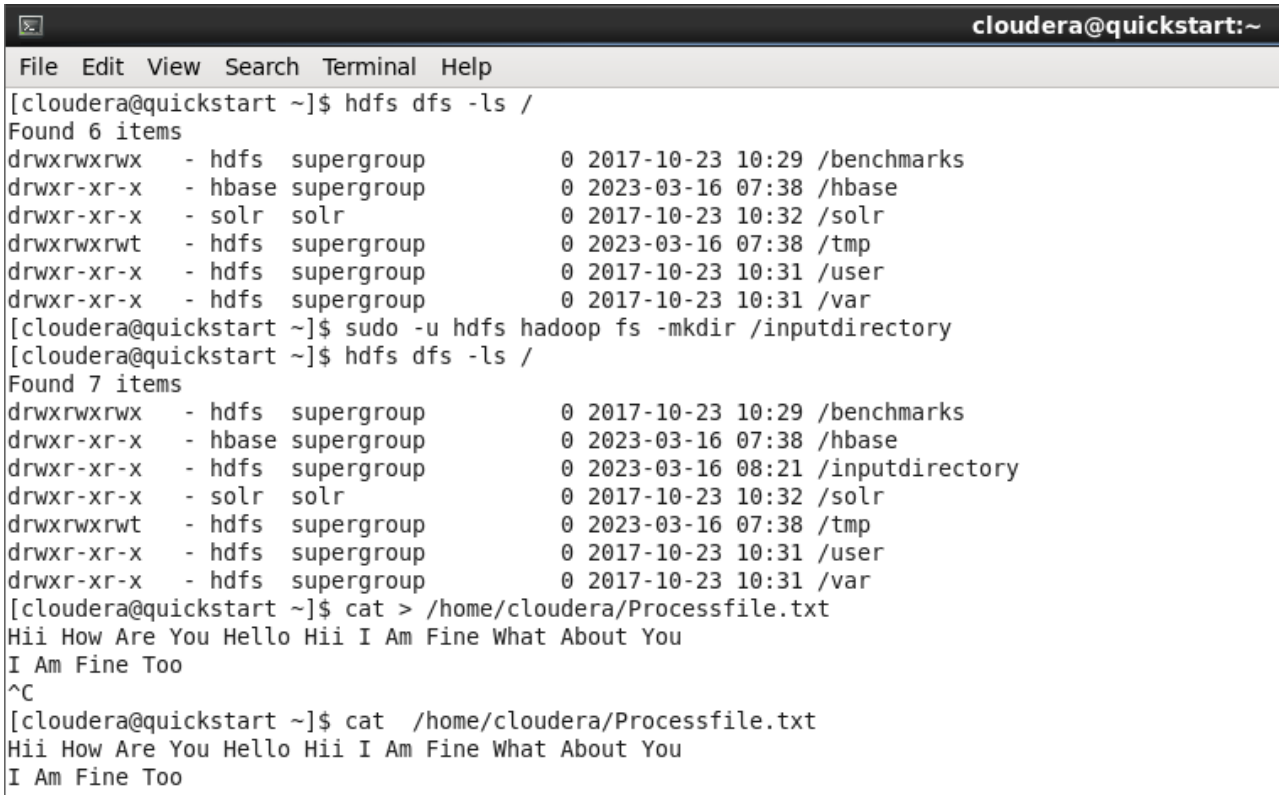
```java
import java.io.IOException;
import java.util.StringTokenizer;

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);
      }
```




Export
Select
Export resources into a JAR file on the local file system.
Select an export destination:
type filter text
General
Install
Java
JAR file
Javadoc
Runnable JAR file
Run/Debug
Tasks
Team
XML
< Back
Next >
Cancel
Finish


JAR Export
JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
training
WordCount
.classpath
.project
Export generated class files and resources
Export all output folders for checked projects
Export Java source files and resources
Export refactorings for checked projects. Select refactorings...
Select the export destination:
JAR file:
Browse...
Options:
Compress the contents of the JAR file
Add directory entries
Overwrite existing files without warning
< Back
Next >
Cancel
Finish


Name:
WordCount.jar
Save in folder:
cloudera
Create Folder
Places
Search
Recently Used
cloudera
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
Desktop 10/23/2017
Documents 10/23/2017
Downloads 07:36
eclipse 08:03
lib 10/23/2017
Music 07:36
Pictures 07:36
Public 07:36
Templates 07:36
Videos 07:36
workspace 08:09
.jar;.zip
Cancel
OK


cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 6 items
drwxrwxrwx - hdfs supergroup 0 2017-10-23 10:29 /benchmarks
drwxr-xr-x - hbase supergroup 0 2023-03-16 07:38 /hbase
drwxr-xr-x - solr solr 0 2017-10-23 10:32 /solr
drwxrwxrwt - hdfs supergroup 0 2023-03-16 07:38 /tmp
drwxr-xr-x - hdfs supergroup 0 2017-10-23 10:31 /user
drwxr-xr-x - hdfs supergroup 0 2017-10-23 10:31 /var
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -mkdir /inputdirectory
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 7 items
drwxrwxrwx - hdfs supergroup 0 2017-10-23 10:29 /benchmarks
drwxr-xr-x - hbase supergroup 0 2023-03-16 07:38 /hbase
drwxr-xr-x - hdfs supergroup 0 2023-03-16 08:21 /inputdirectory
drwxr-xr-x - solr solr 0 2017-10-23 10:32 /solr
drwxrwxrwt - hdfs supergroup 0 2023-03-16 07:38 /tmp
drwxr-xr-x - hdfs supergroup 0 2017-10-23 10:31 /user
drwxr-xr-x - hdfs supergroup 0 2017-10-23 10:31 /var
[cloudera@quickstart ~]$ cat > /home/cloudera/Processfile.txt
Hii How Are You Hello Hii I Am Fine What About You
I Am Fine Too
^C
[cloudera@quickstart ~]$ cat /home/cloudera/Processfile.txt
Hii How Are You Hello Hii I Am Fine What About You
I Am Fine Too

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -chmod -R 777 /inputdirectory
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -put /home/cloudera/Processfile.txt /inputdirectory
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -put /home/cloudera/Processfile.txt /inputdirectory
[cloudera@quickstart ~]$ hdfs dfs -ls /inputdirectory
Found 1 items
-rwxrwxrwx   1 hdfs supergroup         65 2023-03-16 08:24 /inputdirectory/Processfile.txt
[cloudera@quickstart ~]$ hadoop jar /home/cloudera/WordCount.jar WordCount /inputdirectory/Processfile.txt /out1
23/03/16 08:29:26 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
```

```
[cloudera@quickstart ~]$ hdfs dfs -ls /out1
Found 2 items
-rw-r--r--   1 cloudera supergroup          0 2023-03-16 08:30 /out1/_SUCCESS
-rw-r--r--   1 cloudera supergroup         69 2023-03-16 08:30 /out1/part-r-00000
[cloudera@quickstart ~]$ hdfs dfs -cat /out1/part-r-00000
About   1
Am      2
Are     1
Fine    2
Hello   1
Hii     2
How     1
I       2
Too     1
What    1
You     2
[cloudera@quickstart ~]$
```

Practical No: 3

Aim: Write a Pig Script for solving counting problems.

Description:

Apache Pig is an abstraction over MapReduce. It is a tool/platform which is used to analyze larger sets of data representing them as data flows. Pig is generally used with **Hadoop**; we can perform all the data manipulation operations in Hadoop using Apache Pig.

To write data analysis programs, Pig provides a high-level language known as **Pig Latin**. This language provides various operators using which programmers can develop their own functions for reading, writing, and processing data.

To analyze data using **Apache Pig**, programmers need to write scripts using Pig Latin language. All these scripts are internally converted to Map and Reduce tasks. Apache Pig has a component known as **Pig Engine** that accepts the Pig Latin scripts as input and converts those scripts into MapReduce jobs.

Steps & Output:

cat> /home/cloudera/input.csv

cat /home/cloudera/input.csv

pig -x local

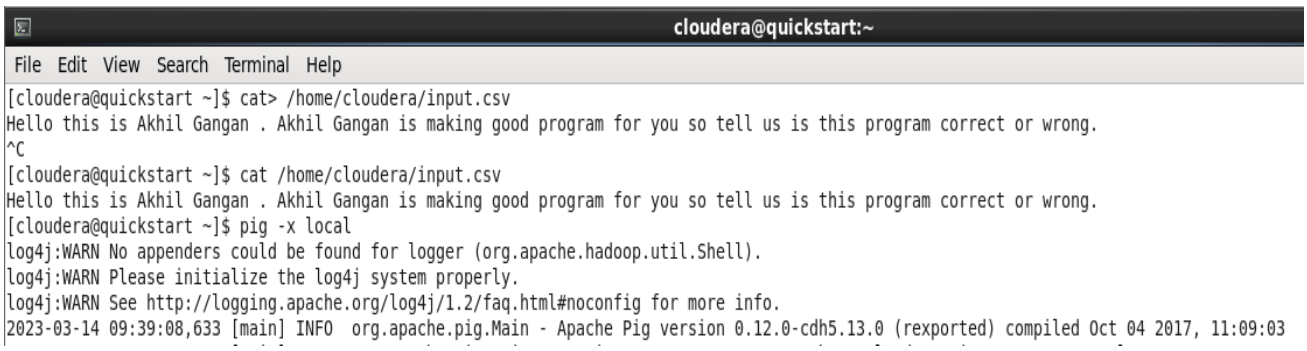
lines = load '/home/cloudera/input.csv' as (line:chararray);

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

grouped = GROUP words by woed;

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

dump wordcount;



```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ cat> /home/cloudera/input.csv
Hello this is Akhil Gangan . Akhil Gangan is making good program for you so tell us is this program correct or wrong.
^C
[cloudera@quickstart ~]$ cat /home/cloudera/input.csv
Hello this is Akhil Gangan . Akhil Gangan is making good program for you so tell us is this program correct or wrong.
[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.
2023-03-14 09:39:08,633 [main] INFO  org.apache.pig.Main - Apache Pig version 0.12.0-cdh5.13.0 (rexported) compiled Oct 04 2017, 11:09:03
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
grunt> lines = load '/home/cloudera/input.csv' as (line:chararray);
grunt> words = foreach lines GENERATE FLATTEN(TOKENIZE(line)) as woed;
grunt> grouped = GROUP words by woed;
grunt> wordcount = foreach grouped GENERATE group,COUNT(words);
grunt> dump wordcount;
2023-03-14 09:45:53,673 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.13.0 0.12.0-cdh5.13.0        cloudera        2023-03-14 09:45:54     2023-03-14 09:46:12     GROUP_BY

Success!

Job Stats (time in seconds):
JobId   Alias   Feature Outputs
job_local1455401358_0001        grouped,lines,wordcount,words   GROUP_BY,COMBINER       file:/tmp/temp-2043429157/tmp-4282069,

Input(s):
Successfully read records from: "/home/cloudera/input.csv"

Output(s):
Successfully stored records in: "file:/tmp/temp-2043429157/tmp-4282069"

Job DAG:
job_local1455401358_0001


2023-03-14 09:46:18,701 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2023-03-14 09:46:18,824 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2023-03-14 09:46:18,825 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-03-14 09:46:18,826 [main] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2023-03-14 09:46:19,132 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2023-03-14 09:46:19,132 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(.,1)
(is,3)
(or,1)
(so,1)
(us,1)
(for,1)
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help

Output(s):
Successfully stored records in: "file:/tmp/temp-2043429157/tmp-4282069"

Job DAG:
job_local1455401358_0001


2023-03-14 09:46:18,701 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2023-03-14 09:46:18,824 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2023-03-14 09:46:18,825 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-03-14 09:46:18,826 [main] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2023-03-14 09:46:19,132 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2023-03-14 09:46:19,132 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(.,1)
(is,3)
(or,1)
(so,1)
(us,1)
(for,1)
(you,1)
(good,1)
(tell,1)
(this,2)
(Akhil,2)
(Hello,1)
(Gangan,2)
(making,1)
(wrong.,1)
(correct,1)
(program,2)
grunt>
```

Practical No: 4

Aim: Install HBase and use the HBase Data model Store and retrieve data.

Description:

HBase is a distributed column-oriented database built on top of the Hadoop file system. It is an open-source project and is horizontally scalable.

HBase is a data model that is similar to Google's big table designed to provide quick random access to huge amounts of structured data.

It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System.

Steps & Output:

//Start HBase

hbase shell

//HBase Commands

status

version,

table_help

whoami

//Data Definition Language

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

//Verify created table

list

//Disable single table

disable 'employee'

```
scan ‘employee’

 //or
is_disable ‘employee’
//Disable multiple tables
disable_all ‘e.*’

// Enabling table
enable‘employee’
//Or
is_enabled ‘employee’

//create new table
create‘student’, ‘name’, ‘age’, ‘course’
put ‘student’, ‘sharath’, ‘name:fullname’, ‘sharathkumar’
put ‘student’, ‘sharath’, ‘age:presentage’, ‘24’
put ‘student’, ‘sharath’, ‘course:pursuing’, ‘Hadoop’
put ‘student’, ‘shashank’, ‘name:fullname’, ‘shashank R
put ‘student’, ‘shashank’, ‘age:presentage’, ‘23’
put ‘student’, ‘shashank’, ‘course:pursuing’, ‘Java’

//Get Information
get ‘student’, ‘shashank’
get ‘student’, ‘sharath’
get ‘student’, ‘sharath’, ‘course’
get ‘student’, ‘shashank’, ‘course’
```

```
get ‘student’, ‘sharath’, ‘name’

//Scan
scan ‘student’
//Count
Count ‘student’

//Alter
alter ‘student’, NAME=>’name’, VERSIONS=>5
put ‘student’, ‘shashank’, ‘name:fullname’, ‘shashank Rao’
scan ‘student’

//Delete
delete ‘student’, ‘shashank’, ‘name:fullname’
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ hbase shell
2023-03-13 22:39:08,413 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> create 'employee','Name','ID','Designation','Salary','Department'
0 row(s) in 1.8340 seconds

=> Hbase::Table - employee
hbase(main):002:0> list
TABLE
employee
1 row(s) in 0.0200 seconds

=> ["employee"]
hbase(main):003:0> disable 'employee'
0 row(s) in 2.5030 seconds

hbase(main):004:0> scan 'employee'
ROW                                    COLUMN+CELL

ERROR: employee is disabled.

Scan a table; pass table name and optionally a dictionary of scanner
specifications.  Scanner specifications may include one or more of:
TIMERANGE, FILTER, LIMIT, STARTROW, STOPROW, ROWPREFIXFILTER, TIMESTAMP,
MAXLENGTH or COLUMNS, CACHE or RAW, VERSIONS, ALL_METRICS or METRICS

If no columns are specified, all columns will be scanned.
To scan all members of a column family, leave the qualifier empty as in
'col_family'.

The filter can be specified in two ways:
1. Using a filterString - more information on this is available in the
Filter Language document attached to the HBASE-4176 JIRA
2. Using the entire package name of the filter.
```

```
hbase(main):002:0> disable_all 'e.*'
employee

Disable the above 1 tables (y/n)?
y
1 tables successfully disabled

hbase(main):003:0> enable 'employee'
0 row(s) in 1.3970 seconds

hbase(main):004:0> scan 'employee'
ROW                                    COLUMN+CELL
0 row(s) in 0.0160 seconds

hbase(main):005:0> is_enabled 'employee'
true
0 row(s) in 0.0210 seconds

hbase(main):006:0> create 'student','name','age','course'
0 row(s) in 1.2930 seconds
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
0 row(s) in 0.0210 seconds

hbase(main):006:0> create 'student','name','age','course'
0 row(s) in 1.2930 seconds

=> Hbase::Table - student
hbase(main):007:0> put 'student','Akhil','name:fullname','AkhilGangan'
0 row(s) in 0.1370 seconds

hbase(main):008:0> put 'student','Akhil','age:presentage','18'
0 row(s) in 0.0110 seconds

hbase(main):009:0> put 'student','Akhil','course:pursing','Hadoop'
0 row(s) in 0.0160 seconds

hbase(main):010:0> put 'student','Faizan','name:fullname','FaizanSiddique'
0 row(s) in 0.0110 seconds

hbase(main):011:0> put 'student','Faizan','age:presentage','19'
0 row(s) in 0.0090 seconds

hbase(main):012:0> put 'student','Faizan','course:pursuing','Java'
0 row(s) in 0.0080 seconds

hbase(main):013:0> get 'student','Akhil'
COLUMN                                  CELL
 age:presentage                         timestamp=1678773345390, value=18
 course:pursing                         timestamp=1678773478617, value=Hadoop
 name:fullname                          timestamp=1678773233107, value=AkhilGangan
3 row(s) in 0.0210 seconds

hbase(main):014:0> get 'student','Faizan'
COLUMN                                  CELL
 age:presentage                         timestamp=1678773880393, value=19
 course:pursuing                        timestamp=1678773957087, value=Java
 name:fullname                          timestamp=1678773840797, value=FaizanSiddique
3 row(s) in 0.0140 seconds
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help

hbase(main):014:0> get 'student','Faizan'
COLUMN                                  CELL
 age:presentage                         timestamp=1678773880393, value=19
 course:pursuing                        timestamp=1678773957087, value=Java
 name:fullname                          timestamp=1678773840797, value=FaizanSiddique
3 row(s) in 0.0140 seconds

hbase(main):015:0> get 'student','Faizan','course'
COLUMN                                  CELL
 course:pursuing                        timestamp=1678773957087, value=Java
1 row(s) in 0.0090 seconds

hbase(main):016:0> get 'student','Akhil','course'
COLUMN                                  CELL
 course:pursing                         timestamp=1678773478617, value=Hadoop
1 row(s) in 0.0080 seconds

hbase(main):017:0> get 'student','Akhil','name'
COLUMN                                  CELL
 name:fullname                          timestamp=1678773233107, value=AkhilGangan
1 row(s) in 0.0090 seconds

hbase(main):018:0> scan 'student'
ROW                                     COLUMN+CELL
 Akhil                                  column=age:presentage, timestamp=1678773345390, value=18
 Akhil                                  column=course:pursing, timestamp=1678773478617, value=Hadoop
 Akhil                                  column=name:fullname, timestamp=1678773233107, value=AkhilGangan
 Faizan                                 column=age:presentage, timestamp=1678773880393, value=19
 Faizan                                 column=course:pursuing, timestamp=1678773957087, value=Java
 Faizan                                 column=name:fullname, timestamp=1678773840797, value=FaizanSiddique
2 row(s) in 0.0630 seconds

hbase(main):019:0> Count 'student'
NoMethodError: undefined method `Count' for #<Object:0x37e16599>

hbase(main):020:0> count 'student'
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help

hbase(main):019:0> Count 'student'
NoMethodError: undefined method `Count' for #<Object:0x37e16599>

hbase(main):020:0> count 'student'
2 row(s) in 0.0460 seconds

=> 2
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
   { MAX_FILESIZE => '134217728' }, { METHOD => 'delete', NAME => 'f2' },
   OWNER => 'johndoe', METADATA => { 'mykey' => 'myvalue' }


hbase(main):022:0> alter 'student',NAME=>'name',VERSIONS=>5
Updating all regions with the new schema...
0/1 regions updated.
1/1 regions updated.
Done.
0 row(s) in 3.1780 seconds

hbase(main):023:0> put 'student','Akhil','name:fullname','Akhil Lokande'
0 row(s) in 0.0120 seconds

hbase(main):024:0> scan 'student'
ROW                                    COLUMN+CELL
 Akhil                                 column=age:presentage, timestamp=1678773345390, value=18
 Akhil                                 column=course:pursing, timestamp=1678773478617, value=Hadoop
 Akhil                                 column=name:fullname, timestamp=1678774746855, value=Akhil Lokande
 Faizan                                column=age:presentage, timestamp=1678773880393, value=19
 Faizan                                column=course:pursuing, timestamp=1678773957087, value=Java
 Faizan                                column=name:fullname, timestamp=1678773840797, value=FaizanSiddique
2 row(s) in 0.0430 seconds

hbase(main):025:0> delete 'student','Akhil','name:fullname'
0 row(s) in 0.0450 seconds

hbase(main):026:0> scan 'student'
ROW                                    COLUMN+CELL
 Akhil                                 column=age:presentage, timestamp=1678773345390, value=18
 Akhil                                 column=course:pursing, timestamp=1678773478617, value=Hadoop
 Faizan                                column=age:presentage, timestamp=1678773880393, value=19
 Faizan                                column=course:pursuing, timestamp=1678773957087, value=Java
 Faizan                                column=name:fullname, timestamp=1678773840797, value=FaizanSiddique
2 row(s) in 0.0160 seconds

hbase(main):027:0>
```

Practical No: 5

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

Description:

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analyzing easy.

Initially Hive was developed by Facebook, later the Apache Software Foundation took it up and developed it further as an open source under the name Apache Hive. It is used by different companies. For example, Amazon uses it in Amazon Elastic MapReduce.

It is a platform used to develop SQL type scripts to do MapReduce operations.

Steps & Output:

```
cat > /home/cloudera/employee.txt
        1~Sachine~Pune~Product Engineering~100000~Big Data
        2~Gaurav~Banglore~Sales~90000~CRM
        3~Manish~Chennai~Recruiter~125000~HR
        4~Bhushan~Hyderabad~Developer~50000~BFSI
cat  /home/cloudera/employee.txt
sudo -u hdfs hadoop fs -put /home/cloudera/employee.txt /inputdirectroy
hdfs dfs -ls /
hdfs dfs -ls /inputdirectory
hadoop fs -cat  /inputdirectory/employee.txt
hive
show databases;
create database organization;
show databases;
use organization;
show tables;
```

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

show tables;
select * from employee;
show tables;
load data inpath '/inputdirectory/employee.txt' overwrite into table employee;
show tables;
select * from employee;
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[cloudera@quickstart ~]$ cat > /home/cloudera/employee.txt
1~Akhil~Pune~Product Engineering~100000~Big Data
2~Tasneem~Bangalore~Sales~90000~CRM
3~Praju~Chennai~Recruiter~125000~HR
4~Faizan~Hyderabad~Developer~50000~BFSI
^C
[cloudera@quickstart ~]$ cat  /home/cloudera/employee.txt
1~Akhil~Pune~Product Engineering~100000~Big Data
2~Tasneem~Bangalore~Sales~90000~CRM
3~Praju~Chennai~Recruiter~125000~HR
4~Faizan~Hyderabad~Developer~50000~BFSI
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -put /home/cloudera/employee.txt /inputdirectory
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 8 items
drwxrwxrwx   - hdfs     supergroup          0 2017-10-23 10:29 /benchmarks
drwxr-xr-x   - hbase    supergroup          0 2023-03-16 15:26 /hbase
drwxrwxrwx   - hdfs     supergroup          0 2023-03-16 15:31 /inputdirectory
drwxr-xr-x   - cloudera supergroup          0 2023-03-16 08:30 /out1
drwxr-xr-x   - solr     solr                0 2017-10-23 10:32 /solr
drwxrwxrwt   - hdfs     supergroup          0 2023-03-16 07:38 /tmp
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 10:31 /user
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 10:31 /var
[cloudera@quickstart ~]$ hdfs dfs -ls /inputdirectory
Found 2 items
-rwxrwxrwx   1 hdfs supergroup         65 2023-03-16 08:24 /inputdirectory/Processfile.txt
-rw-r--r--   1 hdfs supergroup        161 2023-03-16 15:31 /inputdirectory/employee.txt
[cloudera@quickstart ~]$ hadoop fs -cat /inputdirectory/employee.txt
1~Akhil~Pune~Product Engineering~100000~Big Data
2~Tasneem~Bangalore~Sales~90000~CRM
3~Praju~Chennai~Recruiter~125000~HR
4~Faizan~Hyderabad~Developer~50000~BFSI
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
[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.
hive> show databases;
OK
default
Time taken: 1.779 seconds, Fetched: 1 row(s)
hive> create database organization;
OK
Time taken: 3.643 seconds
hive> show databases;
OK
default
organization
Time taken: 0.07 seconds, Fetched: 2 row(s)
hive> use organization;
OK
Time taken: 0.177 seconds
hive> show tables;
OK
Time taken: 0.089 seconds
hive> create table employee(
    > id int,
    > name string,
    > city string,
    > department string,
    > salary int,
    > domain string )
    > row format delimited
    > fields terminated by '~';
OK
```

```
cloudera@quickstart:~
File  Edit  View  Search  Terminal  Help
    > salary int,
    > domain string )
    > row format delimited
    > fields terminated by '~';
OK
Time taken: 0.732 seconds
hive> show tables;
OK
employee
Time taken: 0.038 seconds, Fetched: 1 row(s)
hive> select * from employee;
OK
Time taken: 1.356 seconds
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': Permission denied. user=cloudera is not the
owner of inode=employee.txt
Table organization.employee stats: [numFiles=1, numRows=0, totalSize=161, rawDataSize=0]
OK
Time taken: 1.276 seconds
hive> show tables;
OK
employee
Time taken: 0.077 seconds, Fetched: 1 row(s)
hive> select * from employee;
OK
1       Akhil   Pune    Product Engineering     100000  Big Data
2       Tasneem Bangalore       Sales   90000   CRM
3       Praju   Chennai Recruiter       125000  HR
4       Faizan  Hyderabad       Developer       50000   BFSI
Time taken: 0.184 seconds, Fetched: 4 row(s)
hive>
```

Practical No: 6

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

Description:

A k -shingle (or k -gram) for a document is a sequence of k characters that appears in the document.

Example: k=2; doc = abcab. Set of 2- shingles = {ab, bc, ca, ab}.

Steps & Output:

```
install.packages("tm")
require("tm")
install.packages("devtools")
readinteger <-function()
{
  n<-readline(prompt="Enter value of k-1:")
  k<- as.integer(n)
  u1<- readLines("C:/MSC Notes/file.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()
```

```
+ }
> if(interactive())readinteger()
Enter value of k-1:2
character(0)
[1] "Hii"
[1] "ii "
[1] "i H"
[1] " Ho"
[1] "How"
[1] "ow "
[1] "w A"
[1] " Ar"
[1] "Are"
[1] "re "
[1] "e Y"
[1] " Yo"
[1] "You"
[1] "ou "
[1] "u I"
[1] " I "
[1] "I A"
[1] " Am"
[1] "Am "
[1] "m F"
[1] " Fi"
[1] "Fin"
[1] "ine"
[1] "ne "
[1] "e W"
[1] " Wh"
[1] "Wha"
[1] "hat"
[1] "at "
[1] "t A"
[1] " Ab"
[1] "Abo"
[1] "bou"
[1] "out"
[1] "ut "
[1] "t Y"
[1] " Yo"
[1] "You"
```

Practical No: 7

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

Description:

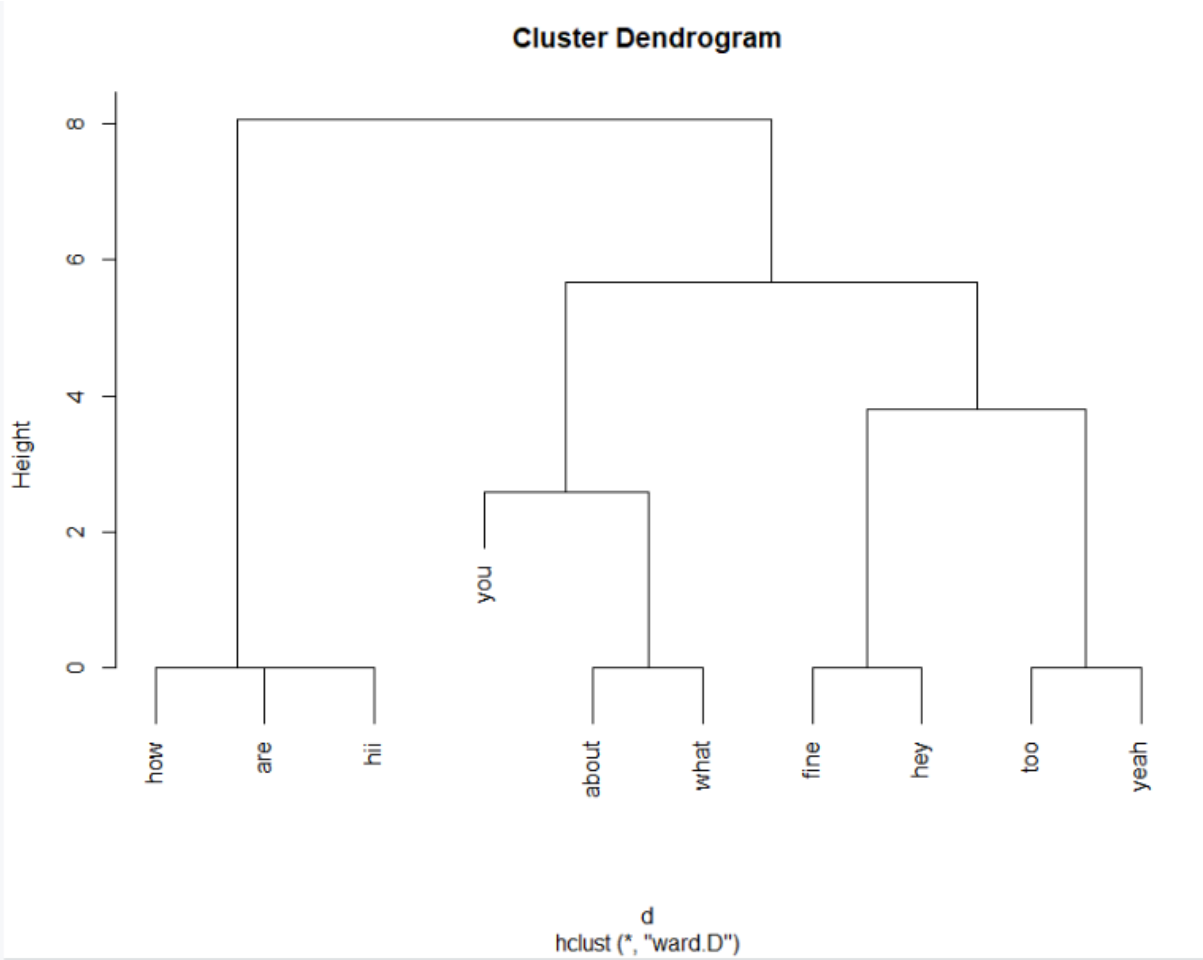
Document similarity, as the name suggests determines how similar are the two given documents. By “documents”, we mean a collection of strings. For example, an essay or a .txt file. Many organizations use this principle of document similarity to check plagiarism. It is also used by many exams conducting institutions to check if a student cheated from the other.

Steps & Output:

```
install.packages("tm")

require("tm")

install.packages(“ggplot2”)

install.packages(“textreuse”)

install.packages("devtools")

my.corpus <- Corpus(DirSource("C:/MSC Notes/r-corpus"))

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))

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

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

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

plot(fit)
```

```
> my.corpus <- Corpus(DirSource("C:/MSC Notes/r-corpus"))
> 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: 10, documents: 3)>>
Non-/sparse entries: 13/17
Sparsity           : 57%
Maximal term length: 5
Weighting          : term frequency (tf)
Sample             :
       Docs
Terms   file1.txt file2.txt file3.txt
  about         0         1         0
  are           1         0         0
  fine          0         1         1
  hey           0         1         1
  hii           1         0         0
  how           1         0         0
  too           0         0         1
  what          0         1         0
  yeah          0         0         1
  you           1         1         0
> my.df.scale<- scale(my.df)
> d<-dist(my.df.scale,method="euclidean")
> fit<-hclust(d,method = "ward")
The "ward" method has been renamed to "ward.D"; note new "ward.D2"
> plot(fit)
>
```

Name: Akhil R Gangan **Roll No:** 530



Course: MSc CS-I **Subject:** BIBD

Practical No: 8

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

Description:

For a random variable x, its Nth moment is the expected value of the Nth power of x, where N is a positive integer. The Nth moment of the deviation of x from its mean is called "the Nth central moment".

The 1st moment is the mean, the 2nd central moment is the variance.

Steps & Output:

```
import java.io.*;
import java.util.*;
public class n_moment
{
	public static void main(String args[]) {
		int n=15;
		String stream[]= {"a","b","c","b","d","a","c","d","a","b","d","c","a","a","b"};
		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
                {
                    //System.out.println("Hello"+i);
                    arrlist.add(count);
                    count=1;
                }
            }
            arrlist.add(count);

            zero_moment=arrlist.size();
            System.out.println("\n\n\nValue of Zeroth moment for given
stream::"+zero_moment);
            for(int i=0;i<arrlist.size();i++)
            {
                first_moment+=arrlist.get(i);
            }
            System.out.println("\n\nValue of First moment for given
stream::"+first_moment);
            for (int i=0;i<arrlist.size();i++)
            {
                int j=arrlist.get(i);
                second_moment+=(j*j);
            }
            System.out.println("\n\nValue of Second moment for given
stream::"+second_moment);

        }
```

}

```
<terminated> n_moment [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (14-Mar-2023, 9:33:40 pm – 9:33:40 pm) [pid: 8140]
Arraylist elements are::
a
b
c
b
d
a
c
d
a
b
d
c
a
a
b


Value of Zeroth moment for given stream::4


Value of First moment for given stream::15


Value of Second moment for given stream::59
```

Practical No: 9

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

Description:

The Alon-Matias-Szegedy Algorithm (AMS algorithm), that estimate the second moment using this formula:

E= (n *(2 * X.value − 1))

In which X is an univocal element of the stream, randomically selected, and X.value is a counter, that, as we read the stream, add to 1 each time we encounter another occurrence of the x element from the time we selected it.

n represents the length of the data stream

Steps & Output:

```
import java.io.*;

import java.util.*;

class AMSA

{

  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[])
 {
  int n=15;
  String stream="abcbdacdabdcaab";
  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+" "+XE2+"="+XV2+" "+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-matis-szegedy
is::"+apprSecondMomentValue);

}

}
```

```
<terminated> AMSA [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (14-Mar-2023, 9:29:10 pm – 9:29:11 pm) [pid: 10652]
c=3 d=2 a=2
Expected value forc is::75
Expected value ford is::45
Expected value fora is::45
approximate second moment value using alon-matis-szegedy is::55
```