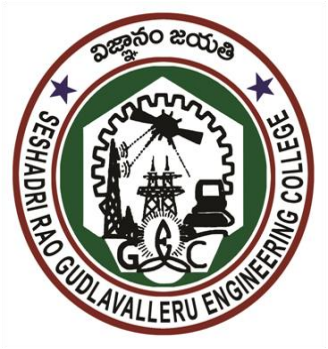
BIG DATA

FACULTY LAB MANUAL

IV Year I Semester



Prepared by

Dr.K.BALA BRAHMESWARA	Sr.Scale.Gr.Asst.Professor
Mr.K.BHASKAR	Assistant Professor
Mrs.T.NAGA MANI	Assistant Professor
Mrs. G.KEERTHI	Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE GUDLAVALLERU

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision

To be a Centre of Excellence in computer science and engineering education and training to meet the challenging needs of the industry and society.

Mission

- To impart quality education through well-designed curriculum in tune with the growing software needs of the industry.
- To serve our students by inculcating in them problem solving, leadership, teamwork skills and the value of commitment to quality, ethical behavior & respect for others.
- To foster industry-academia relationship for mutual benefit and growth.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO 1: Identify, analyze, formulate and solve computing problems both independently and in a team environment using appropriate modern tools.

PEO 2: Develop software systems with significant technical, legal, ethical, social, environmental and economic considerations.

PEO 3: Exhibit commitment in lifelong learning, professional development and leadership and communicate effectively with professional clients and the public.

PROGRAM SPECIFIC OUTCOMES (PSOs)

1. Design, develop, test and maintain reliable software systems and intelligent systems.

2. Design and develop websites, web apps and mobile apps.

Program Outcomes

Computer Science and Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Department of Computer Science and Engineering

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis / program / experiment details.
3. Student should enter into the laboratory with:
    a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
    b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab. c. Proper Dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

COURSE NAME: BIG DATA ANALYTICS LAB

COURSE CODE: CT3565

COURSE OBJECTIVES:

- To familiarize the basic concepts of Hadoop and its eco system.
- To develop programs using Map Reduce, PIG and HIVE.

COURSE OUTCOMES:

Upon successful completion of the course, the students will be able to

- apply suitable LINUX commands to work in Hadoop environment.
- use HDFS file structure and Map Reduce framework to solve complex problems.
- analyze data using Pig and Hive.

MAPPING OF COURSE OUTCOMES WITH PROGRAM OUTCOMES:

COURSE OUTCOMES	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	P10	P11	P12
CO1	√	√		√	√							√
CO2	√	√			√							√

BIG DATA ANALYTICS LAB INDEX

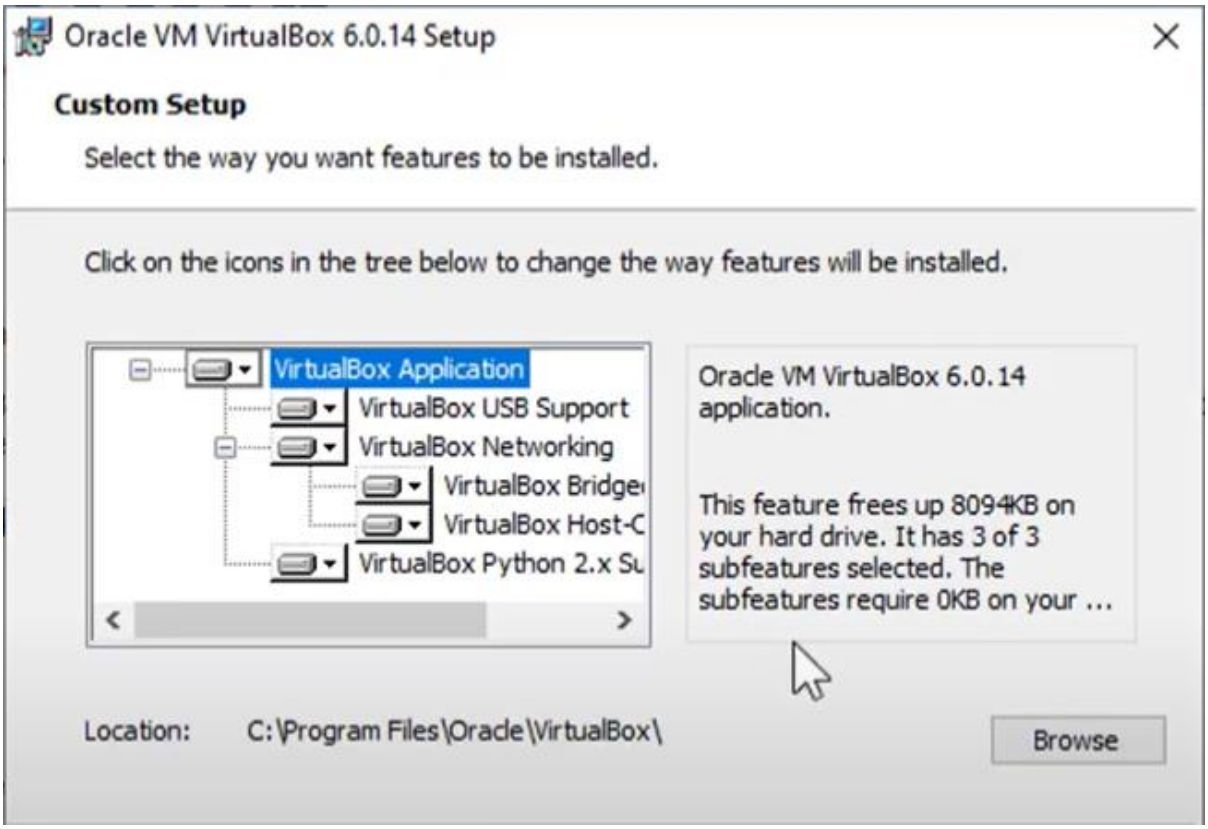
INSTALLATION OF VIRTUALBOX:

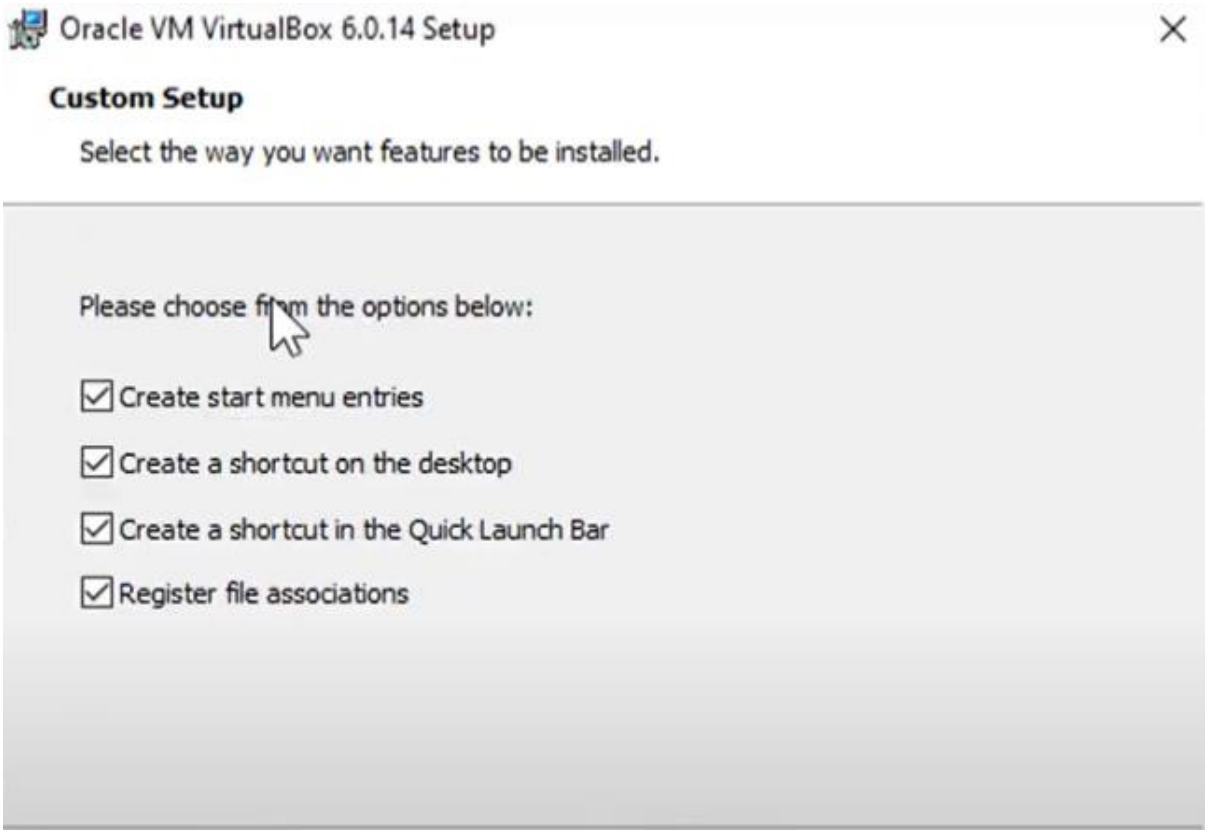
Open web browser and type https:// virtualbox.org/wiki/Downloads

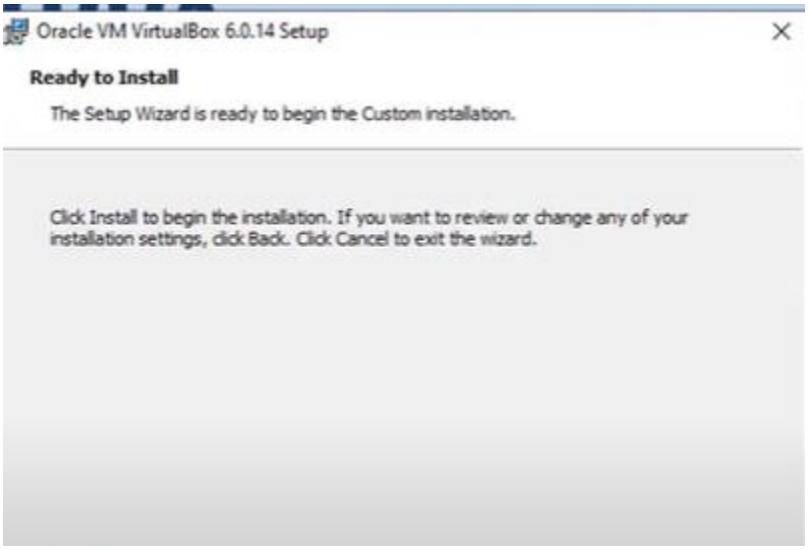
Virtual Box web page go on to windows hosts and you will see that an exe file download will start

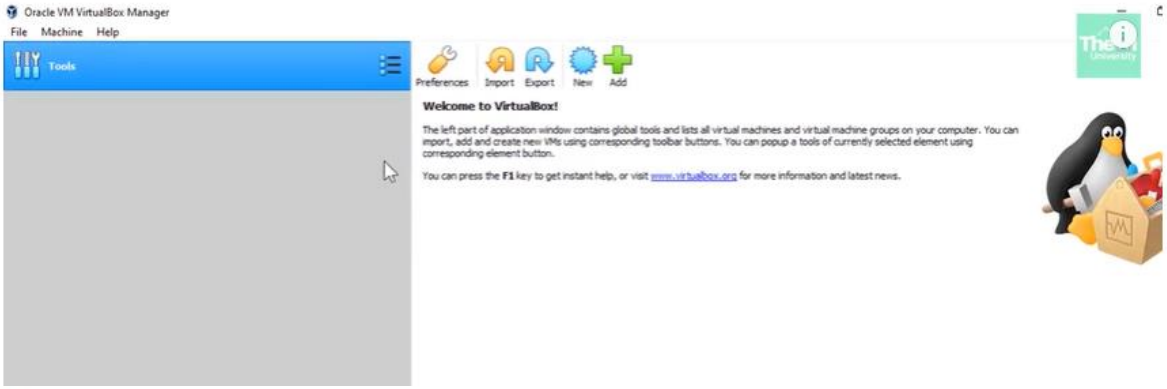
To follow these instruction

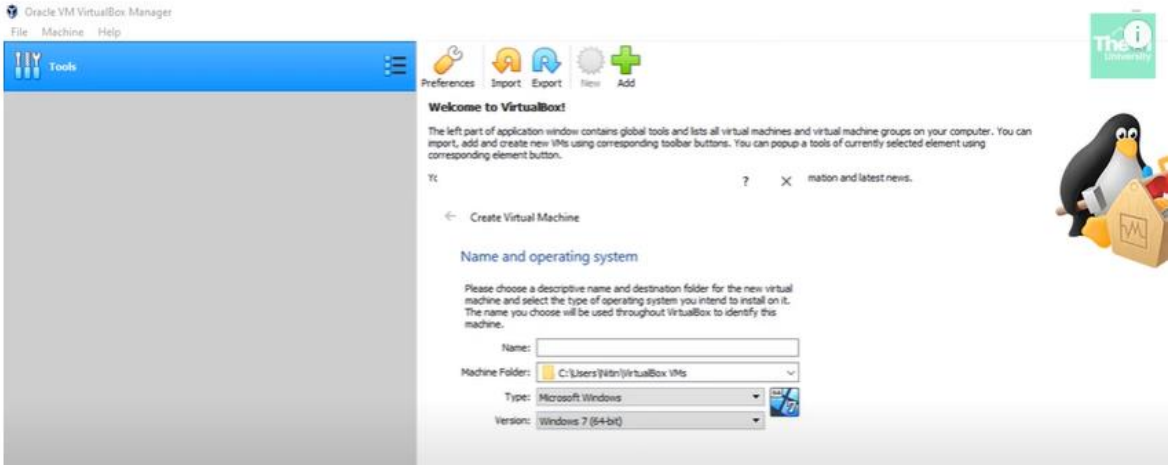
VritualBox 6.0.1 for setup so click on next


Oracle VM VirtualBox 6.0.14 Setup
Custom Setup
Select the way you want features to be installed.
Click on the icons in the tree below to change the way features will be installed.
VirtualBox Application
VirtualBox USB Support
VirtualBox Networking
VirtualBox Bridge
VirtualBox Host-C
VirtualBox Python 2.x Su
Oracle VM VirtualBox 6.0.14 application.
This feature frees up 8094KB on your hard drive. It has 3 of 3 subfeatures selected. The subfeatures require 0KB on your ...
Location:
C:\Program Files\Oracle\VirtualBox\
Browse

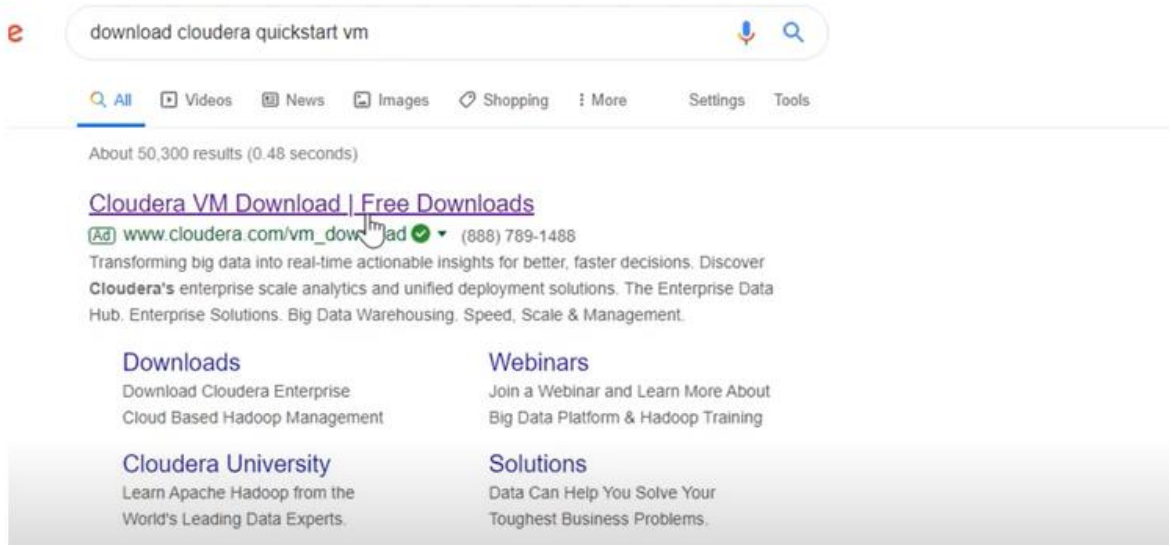

Oracle VM VirtualBox 6.0.14 Setup
Custom Setup
Select the way you want features to be installed.
Please choose from the options below:
Create start menu entries
Create a shortcut on the desktop
Create a shortcut in the Quick Launch Bar
Register file associations





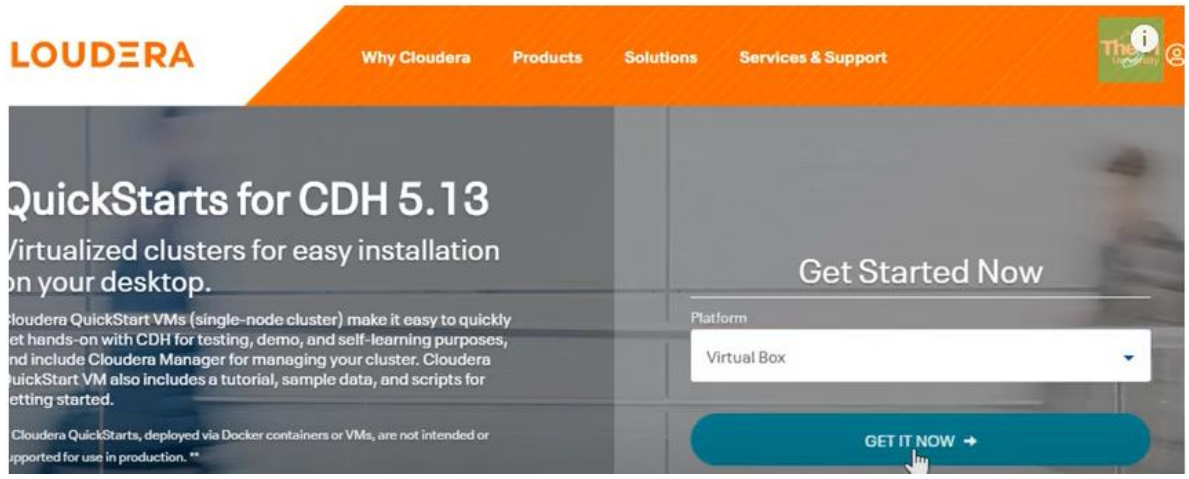


Type in web browser cloudera VM

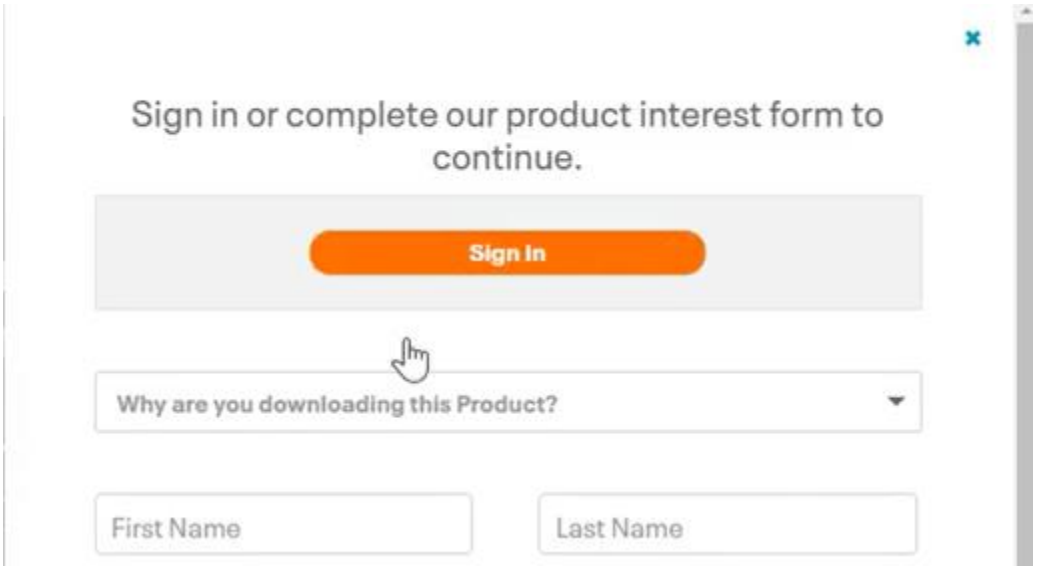


Select cloudera QuickStarts

In the Get Strated Now DropBox select Virtual Box and Click GET IT Now->

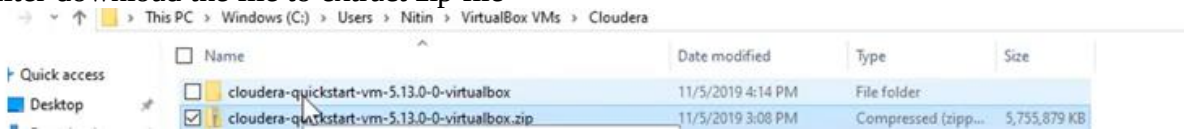


To fill Require Data

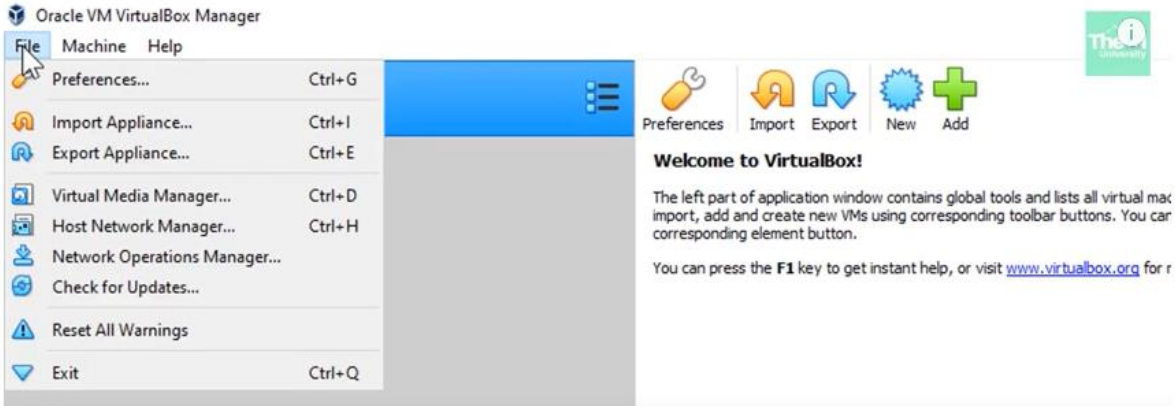


After fill these data the following window is appear

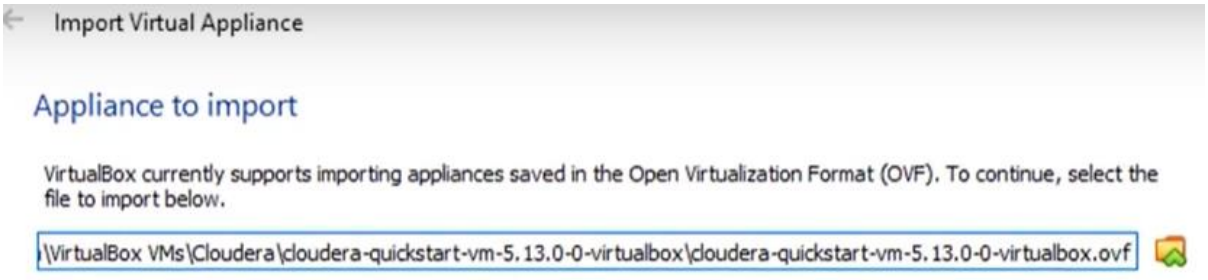
After download the file to extract zip file



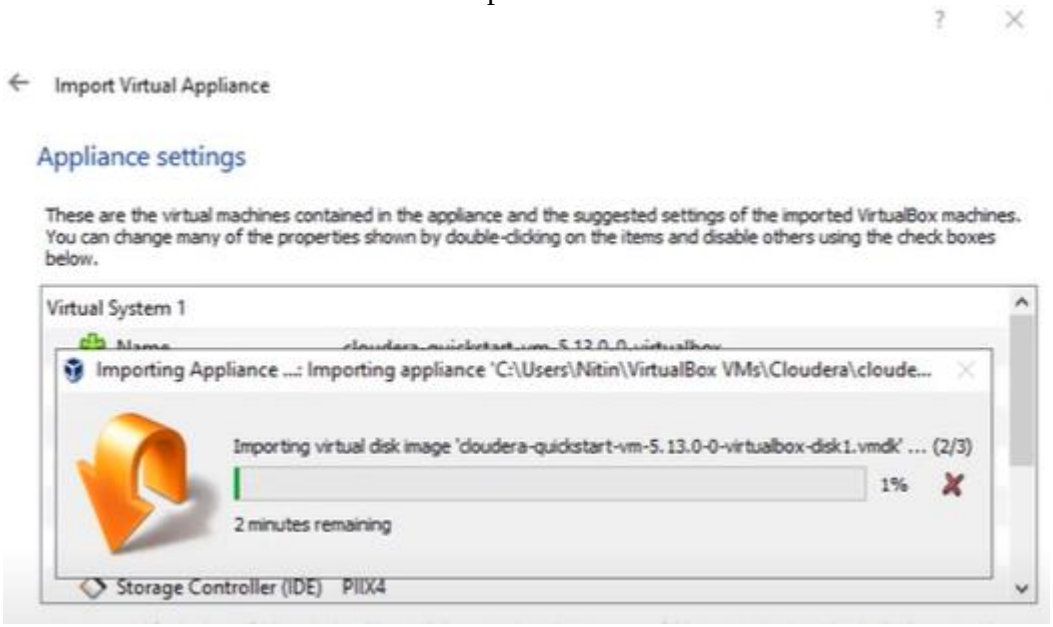
Go to Virtual Box go to file menu click Import Appliance



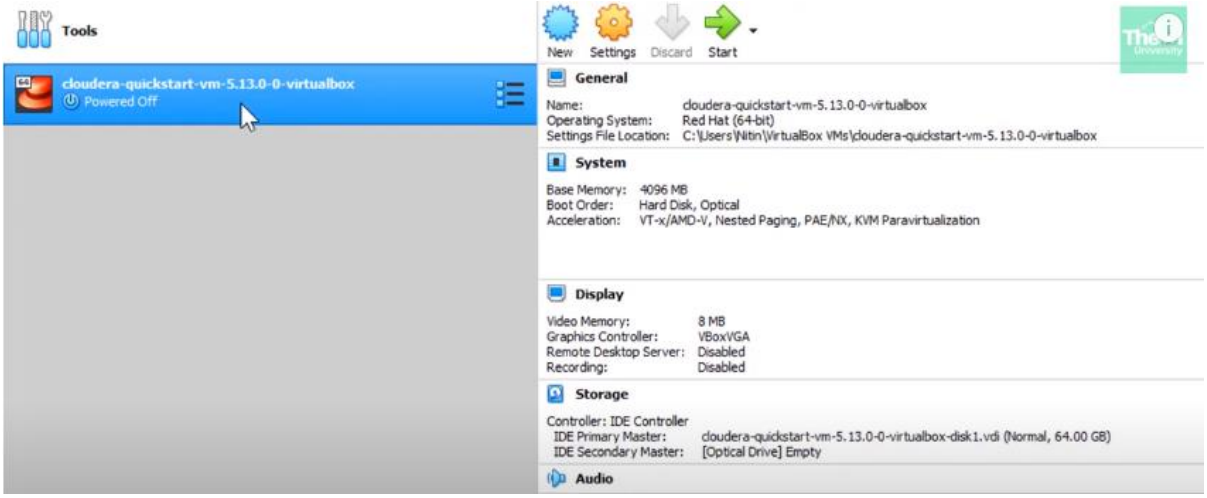
Click Browser folder to specify path



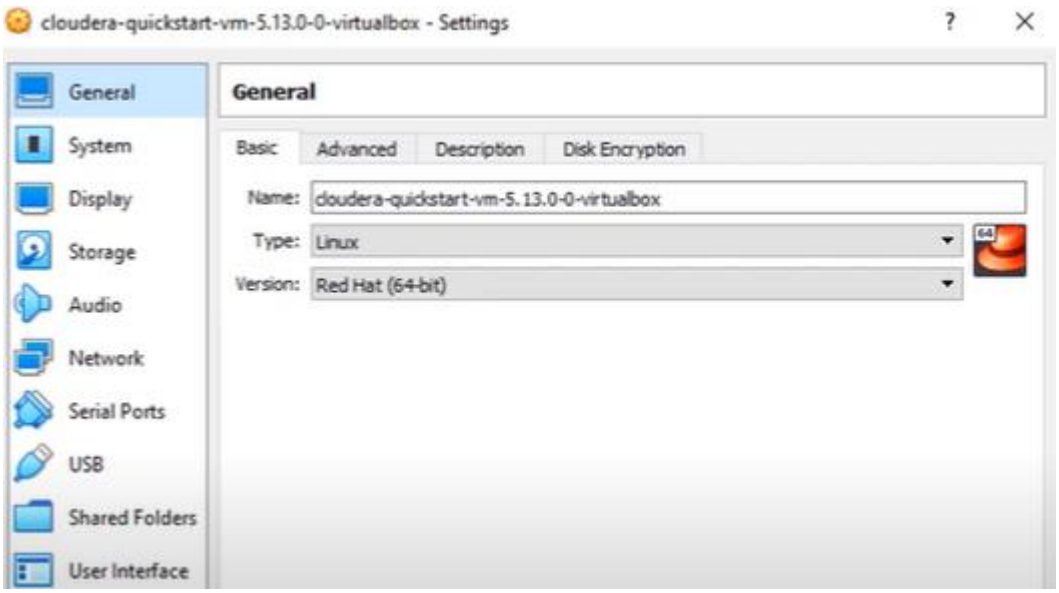
Click next to fill the data and click import



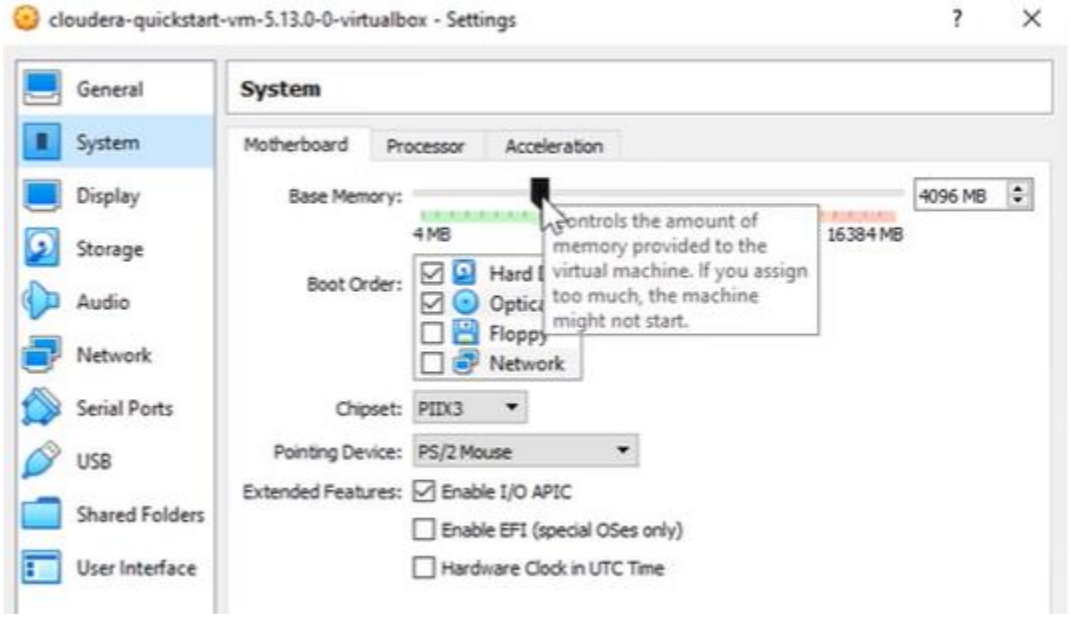
Once complete the import process click QuickStart VM



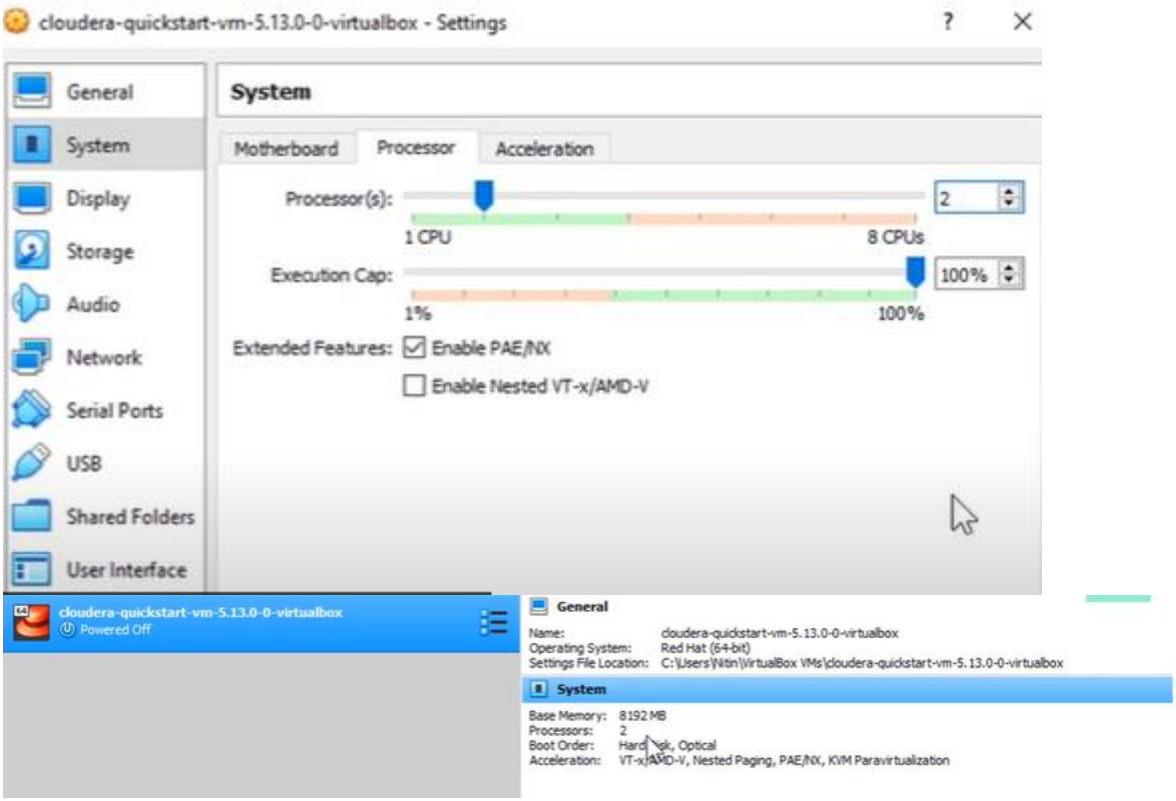
Now click on Setting increasing the RAM as well as number of course



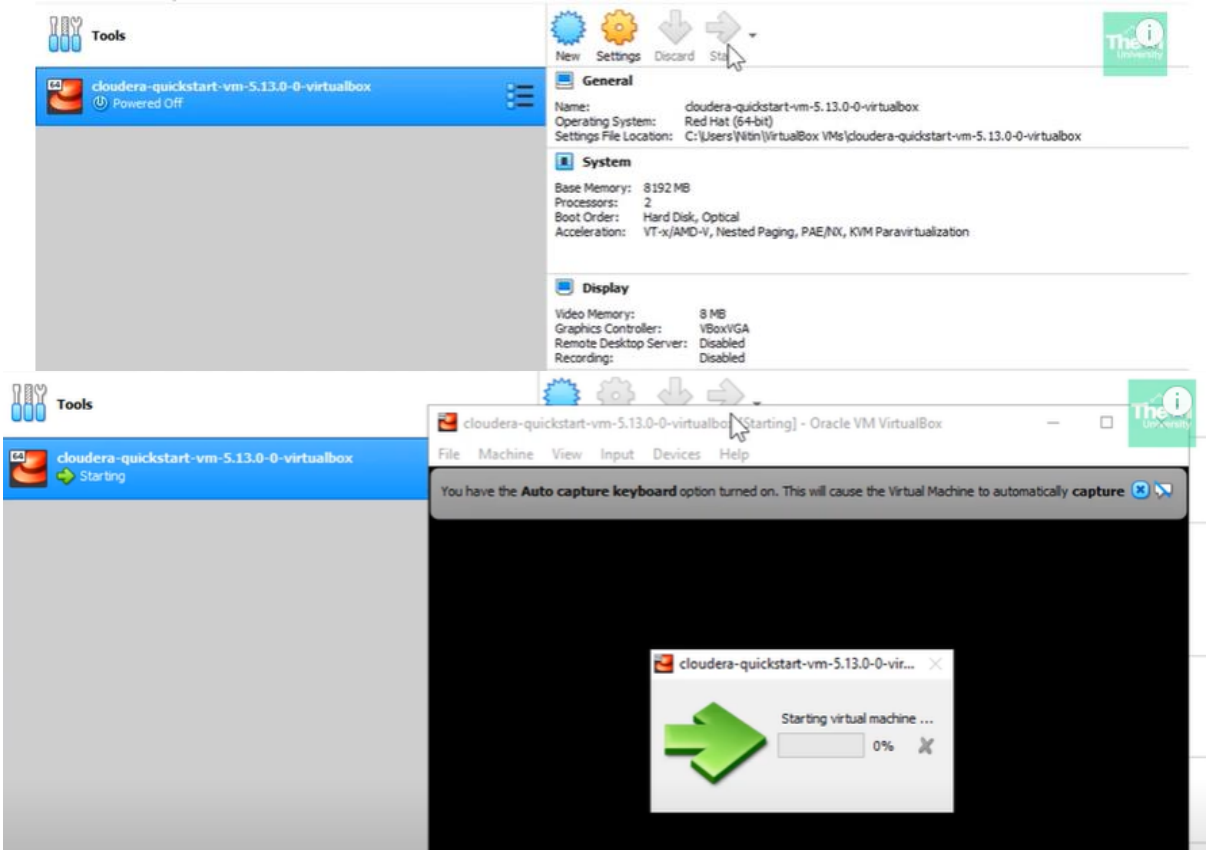
Click in the System appear one window select MotherBoard to increase the Base Memory size



Goto Processor Menu and increase processors(2)



Now click Start Button



Now Your Machine is Running following window is appear

Now Click on Terminal top here so that we can make the admin console accessible

Experiment 1:
Aim: Practice on basic Linux commands.

Basic Linux Commands

pwd

pwd (print working directory) – The pwd command is used to display the name of the current working directory in the Linux system using the terminal.

syntax

pwd [-LP]

Sr.No.	Option & Description
1	**-L (logical)**Display the value of $pwd if it names the current working directory
2	**-P (physical)**Display the physical directory, without any soft link
3	**--help**Displays a help message and then exits.

Output:

```
cloudera@localhost ~]$ pwd
'home/cloudera
cloudera@localhost ~]$
```

Create File using touch

touch command: It is used to create a file without any content. The file created using touch command is empty. This command can be used when the user doesn't have data to store at the time of file creation.

Syntax:

```
touch file_name
```

output

```
[cloudera@localhost ~]$ touch sample
[cloudera@localhost ~]$
```

To Add Content Existing File using vi

The vi editor tool is an interactive tool as it displays changes made in the file on the screen while you edit the file.

In vi editor you can insert, edit or remove a word as cursor moves throughout the file.

Commands are specified for each function like to delete it's x or dd.

The vi editor is case-sensitive. For example, **p** allows you to paste after the current line while **P** allows you to paste before the current line.

vi syntax:

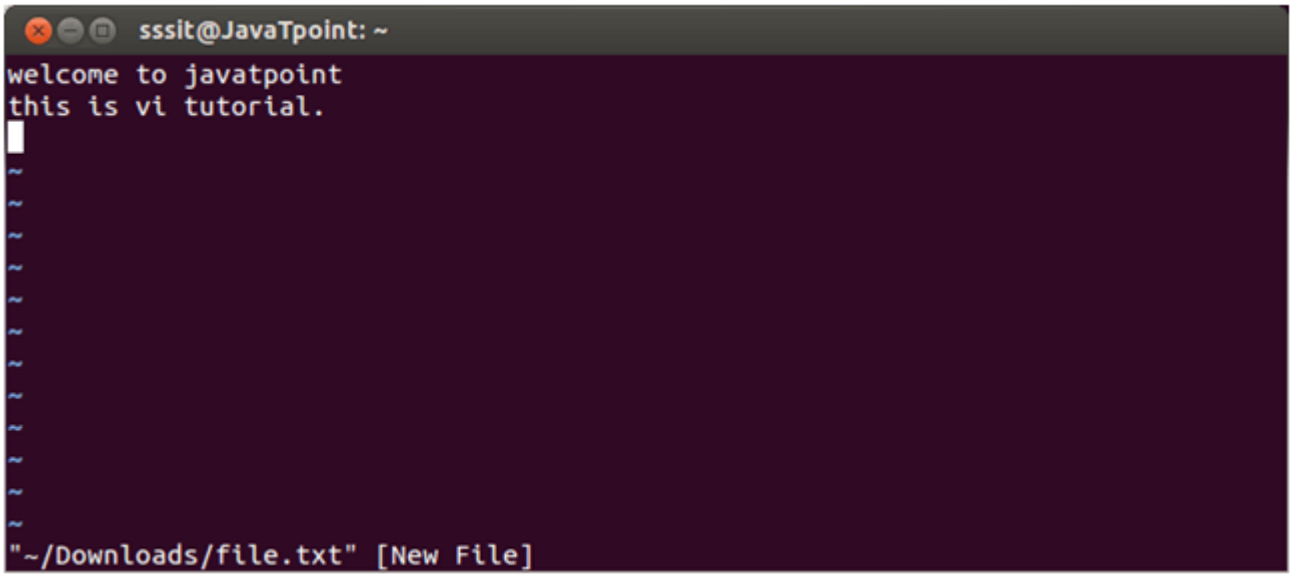
vi **<fileName>**

Command mode

This is what you'll see when you'll press enter after the above command. If you'll start typing, nothing will appear as you are in command mode. By default vi opens in command mode.

Insert mode

To move to the insert mode press **i.** Although, there are other commands also to move to insert mode which we'll study in next page.



Look at the above snapshot, after pressing **i** we have entered into insert mode. Now we can write anything. To move to the next line press enter.

Once you have done with your typing, press **esc** key to return to the command mode.

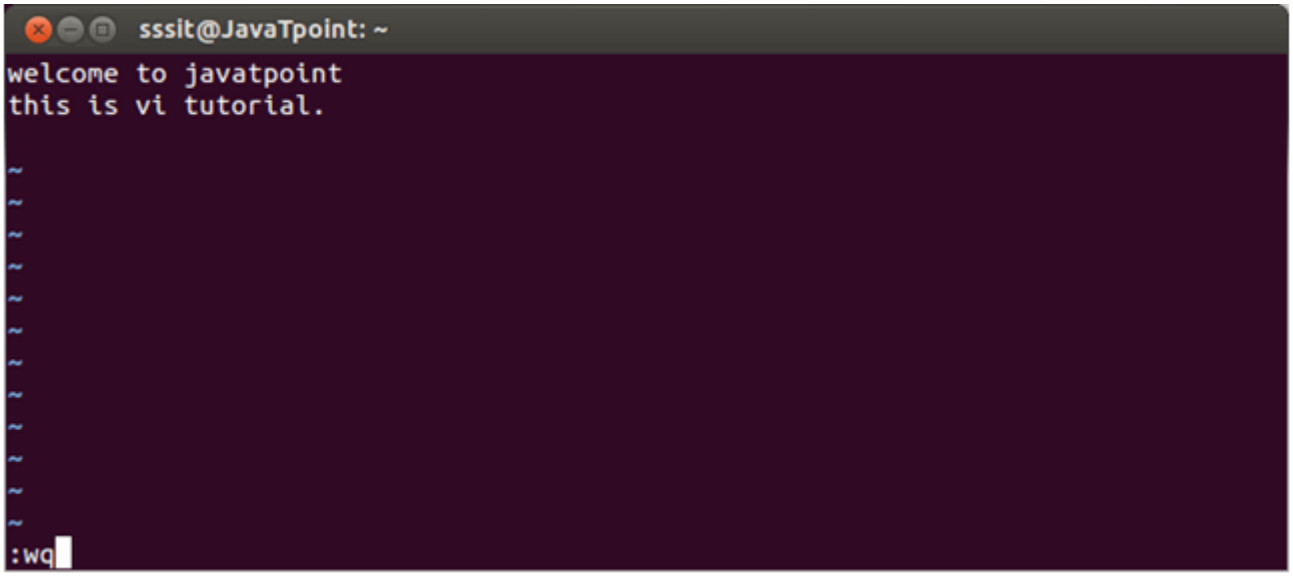
To save and quit

You can save and quit vi editor from command mode. Before writing save or quit command you have to press colon **(:).** Colon allows you to give instructions to vi.

exit vi table:

Commands	Action
:wq	Save and quit
:w	Save
:q	Quit
:w fname	Save as fname
ZZ	Save and quit
:q!	Quit discarding changes made
:w!	Save (and write to non-writable file)

Type **:wq** to save and exit the file.



Look at the above snapshot, command :wq will save and quit the vi editor. When you'll type it in command mode, it will automatically come at bottom left corner.

Output

```
cloudera@localhost ~]$ vi sample
cloudera@localhost ~]$
```

To see Content of file using cat command

Cat(concatenate) command is very frequently used in Linux. It reads data from the file and gives its content as output. It helps us to create, view, and concatenate files. So let us see some frequently used cat commands.

Syntax

```
cat file_name
```

output

```
[cloudera@localhost ~]$ cat sample
Welcome to Big Data Analytics Lab
[cloudera@localhost ~]$
```

To see List of file using ls command

In Linux, the command **"ls"** is one of the most commonly used. It's used to display a list of files and sub-directories in the current directory. If you're new to using the command line, the first command you should learn is probably **ls**.This command can be used by both regular users as well as system administrators.

syntax:

```
ls [ Options ] [File]
```

Options Description

ls –a list all files including hidden file starting with '.'.
ls –d list directories - with ' */'.
ls –l list with long format - show permissions.
ls –F Append indicator (one of */=>@|) to entries.
ls –lh This command will show you the file sizes in human readable format.
ls –r list in reverse order.ls -ilist file's inode(index) number.
ls –ltr View Reverse Output Order by Date.
ls –t sort by time & date
.ls –n It is used to print group ID and owner ID instead of their names.
ls –m A list of entries separated by commas should fill the width.
ls –g This allows you to exclude the owner and group information columns.

```
[cloudera@localhost ~]$ ls
532         add.sh    Documents  hello.c  Music     sample     workspace
add.class   a.out     Downloads  keer     Pictures  Templates
add.java    datasets  eclipse    lib      Public    text1
add.py      Desktop   exp.txt    mul.c    r1        Videos
[cloudera@localhost ~]$
```

To see hidden Files

```
[cloudera@localhost ~]$ ls -al
total 232
drwx------. 29 cloudera cloudera 4096 Aug  2 01:59 .
drwxr-xr-x.  3 root     root     4096 Jun  1  2014 ..
drwxrwxr-x   2 cloudera cloudera 4096 Aug  1 23:42 532
-rw-rw-r--   1 cloudera cloudera  626 Aug  1 22:36 add.class
-rw-rw-r--   1 cloudera cloudera  125 Aug  1 22:36 add.java
-rw-rw-r--   1 cloudera cloudera   78 Aug  1 22:03 add.py
-rw-rw-r--   1 cloudera cloudera   81 Aug  1 23:00 add.sh
-rwxrwxr-x   1 cloudera cloudera 7618 Aug  1 22:28 a.out
-rw-------   1 cloudera cloudera  957 Aug  1 23:29 .bash_history
-rw-r--r--.  1 cloudera cloudera   18 Jun  1  2014 .bash_logout
-rw-r--r--.  1 cloudera cloudera  176 Jun  1  2014 .bash_profile
-rw-r--r--.  1 cloudera cloudera  176 Jun  1  2014 .bashrc
drwxr-xr-x   3 cloudera cloudera 4096 Aug  1 02:48 .cache
drwxr-xr-x.  4 cloudera cloudera 4096 Aug  1 02:48 .config
drwxr-xr-x.  2 cloudera cloudera 4096 Jun  1  2014 datasets
drwx------   3 cloudera cloudera 4096 Aug  1 02:48 .dbus
drwxr-xr-x.  2 cloudera cloudera 4096 Jun  1  2014 Desktop
drwxr-xr-x.  3 cloudera cloudera 4096 Jun  1  2014 Documents
drwxr-xr-x   2 cloudera cloudera 4096 Aug  1 02:48 Downloads
drwxrwsr-x.  9 cloudera cloudera 4096 Aug  1 23:00 eclipse
-rw-r--r--.  1 cloudera cloudera  500 May  7  2013 .emacs
-rw-------   1 cloudera cloudera   16 Aug  1 02:48 .esd_auth
```

To see sorting of files

```
[cloudera@localhost ~]$ ls -lt
total 104
-rw-rw-r--  1 cloudera cloudera   34 Aug  2 01:59 sample
drwxrwxr-x  2 cloudera cloudera 4096 Aug  2 01:42 keer
-rw-rw-r--  1 cloudera cloudera   36 Aug  2 01:39 r1
-rw-rw-r--  1 cloudera cloudera   10 Aug  2 01:23 text1
drwxrwxr-x  2 cloudera cloudera 4096 Aug  1 23:42 532
-rw-rw-r--  1 cloudera cloudera   51 Aug  1 23:34 hello.c
drwxr-xr-x. 5 cloudera cloudera 4096 Aug  1 23:03 workspace
drwxrwsr-x. 9 cloudera cloudera 4096 Aug  1 23:00 eclipse
-rw-rw-r--  1 cloudera cloudera   81 Aug  1 23:00 add.sh
-rw-rw-r--  1 cloudera cloudera  626 Aug  1 22:36 add.class
-rw-rw-r--  1 cloudera cloudera  125 Aug  1 22:36 add.java
-rw-rw-r--  1 cloudera cloudera  564 Aug  1 22:32 mul.c
-rwxrwxr-x  1 cloudera cloudera 7618 Aug  1 22:28 a.out
-rw-rw-r--  1 cloudera cloudera   78 Aug  1 22:03 add.py
-rw-rw-r--  1 cloudera cloudera   85 Aug  1 03:00 exp.txt
drwxr-xr-x  2 cloudera cloudera 4096 Aug  1 02:48 Downloads
drwxr-xr-x  2 cloudera cloudera 4096 Aug  1 02:48 Music
drwxr-xr-x  2 cloudera cloudera 4096 Aug  1 02:48 Pictures
drwxr-xr-x  2 cloudera cloudera 4096 Aug  1 02:48 Public
drwxr-xr-x  2 cloudera cloudera 4096 Aug  1 02:48 Templates
drwxr-xr-x  2 cloudera cloudera 4096 Aug  1 02:48 Videos
```

Create Directory

mkdir command in Linux allows the user to create directories (also referred to as folders in some operating systems). This command can create multiple directories at once as well as set the permissions for the directories. It is important to note that the user executing

this command must have enough permission to create a directory in the parent directory, or he/she may receive a 'permission denied' error.

Syntax

mkdir directory-name

```
[cloudera@localhost ~]$ mkdir CSE-C
[cloudera@localhost ~]$
```

```
[cloudera@localhost ~]$ ls
532         add.sh    Desktop    exp.txt  mul.c     r1         Videos
add.class   a.out     Documents  hello.c  Music     sample     workspace
add.java    CSE-C     Downloads  keer     Pictures  Templates
add.py      datasets  eclipse    lib      Public    text1
[cloudera@localhost ~]$
```

Change Directory

cd command in Linux known as the change directory command. It is used to move efficiently from the current working directory to different directories in our System.

Syntax

cd directory-name

```
[cloudera@localhost ~]$ cd CSE-C
[cloudera@localhost CSE-C]$
```

Copy file

cp stands for a **copy**. This command is used to copy files or groups of files or directories. It creates an exact image of a file on a disk with a different file name. *cp* command requires at least two filenames in its arguments.

Syntax:

```
cp [OPTION] Source Destination
cp [OPTION] Source Directory
cp [OPTION] Source-1 Source-2 Source-3 Source-n Directory
```

output:

```
[cloudera@localhost ~]$ ls
532         add.sh    Desktop    exp.txt  mul.c     r1         Videos
add.class   a.out     Documents  hello.c  Music     sample     workspace
add.java    CSE-C     Downloads  keer     Pictures  Templates
add.py      datasets  eclipse    lib      Public    text1
[cloudera@localhost ~]$ cp add.java CSE-C
[cloudera@localhost ~]$ cd CSE-C
[cloudera@localhost CSE-C]$ ls
add.java
[cloudera@localhost CSE-C]$
```

Move file

In UNIX-based operating systems like Linux and macOS, `**mv**` stands for "move". But in this article, we will be talking about the "mv command in Linux". As its name suggests this

command is used to rename file directories and move files from one location to another within a file system.

Two Distinct Functions of `mv` Command

- 1) Renaming a file or directory.
- 2) Moving a file or directory to another location

Syntax

mv [options(s)] [source_file_name(s)] [Destination_file_name]

output:

```
[cloudera@localhost CSE-C]$ cd
[cloudera@localhost ~]$ ls
532         add.sh    Desktop    exp.txt  mul.c     r1         Videos
add.class   a.out     Documents  hello.c  Music     sample     workspace
add.java    CSE-C     Downloads  keer     Pictures  Templates
add.py      datasets  eclipse    lib      Public    text1
[cloudera@localhost ~]$ mv hello.c CSE-C
[cloudera@localhost ~]$ ls
532         add.sh    Desktop    exp.txt  Music     sample     workspace
add.class   a.out     Documents  keer     Pictures  Templates
add.java    CSE-C     Downloads  lib      Public    text1
add.py      datasets  eclipse    mul.c    r1        Videos
[cloudera@localhost ~]$ cd CSE-C
[cloudera@localhost CSE-C]$ ls
add.java  hello.c
[cloudera@localhost CSE-C]$
```

Remove file

rm stands for **remove** here. rm command is used to remove objects such as files, directories, symbolic links and so on from the file system like UNIX.

Syntax

`rm [OPTION]... FILE...`

```
[cloudera@localhost CSE-C]$ ls
add.java  hello.c
[cloudera@localhost CSE-C]$ rm hello.c
rm: remove regular file `hello.c'? y
[cloudera@localhost CSE-C]$ ls
add.java
[cloudera@localhost CSE-C]$
```

Clear the Screen

clear is a standard Unix computer operating system command that is used to clear the terminal screen.

Syntax

Clear

```
[cloudera@localhost CSE-C]$ ls
add.java  hello.c
[cloudera@localhost CSE-C]$ rm hello.c
rm: remove regular file `hello.c'? y
[cloudera@localhost CSE-C]$ ls
add.java
[cloudera@localhost CSE-C]$ clear
```

After

```
[cloudera@localhost CSE-C]$
```

System Info-commands

Date

date command is used to display the system date and time. date command is also used to set date and time of the system.

Syntax:

```
date [OPTION]... [+FORMAT]
date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]
```

```
[cloudera@localhost CSE-C]$ date
Wed Aug  2 02:25:19 PDT 2023
[cloudera@localhost CSE-C]$
```

cal command is a calendar command in Linux which is used to see the calendar of a specific month or a whole year.

Syntax:

```
cal [ [ month ] year]
```

```
[cloudera@localhost CSE-C]$ cal
     August 2023
Su Mo Tu We Th Fr Sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31

[cloudera@localhost CSE-C]$
```

W command

The **'w'** command in Linux gives us important information about who is currently using the computer, how much the computer is being used, and what programs are running. It's a handy tool for people who take care of computer systems, as it helps them keep an eye on what users are doing,

Syntax of `w` command in Linux

```
w [options] user [...]
```

```
cloudera@localhost CSE-C]$ w
02:27:32 up 40 min,  2 users,  load average: 0.00, 0.00, 0.00
SER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
cloudera tty1     :0               01:48   40:17   2.83s  0.00s pam: gdm-autolo
cloudera pts/0    :0.0             01:49    0.00s  0.03s  0.00s w
cloudera@localhost CSE-C]$
```

Whoami

whoami command is used both in *Unix Operating System* and as well as in *Windows Operating System*.

- It is basically the concatenation of the strings **"who","am","i"** as **whoami**.
- It displays the username of the current user when this command is invoked.
- It is similar as running the id command with the options **-un**.

The earliest versions were created in 2.9 BSD as a convenience form for who am i, the Berkeley Unix who command's way of printing just the logged in user's identity. The GNU version was written by Richard Mlynarik and is part of the GNU Core Utilities (coreutils).

Syntax:

```
geekforgeeks@HP~: whoami
```

```
[cloudera@localhost CSE-C]$ whoami
cloudera
[cloudera@localhost CSE-C]$
```

Experiment 2:

Aim : Implement the following file management tasks in Hadoop

a. Adding files and Directories
b. Retrieving files
c. Deleting files
d. Copying files from local file system to HDFS and vice versa.
e. Moving files.

Basic Hadoop Commands:

Version

Syntax: hadoop version

```
[cloudera@localhost ~]$ hadoop version
Hadoop 2.0.0-cdh4.7.0
Subversion git://rhel64-6-0-mk3.jenkins.cloudera.com.121.29.172.in-addr.arpa/dat
a/1/jenkins/workspace/generic-package-rhel64-6-0/topdir/BUILD/hadoop-2.0.0-cdh4.
7.0/src/hadoop-common-project/hadoop-common -r 8e266e052e423af592871e2dfe09d54c0
3f6a0e8
Compiled by jenkins on Wed May 28 10:11:59 PDT 2014
From source with checksum f60207d0daa9f943f253cc8932d598c8
This command was run using /usr/lib/hadoop/hadoop-common-2.0.0-cdh4.7.0.jar
[cloudera@localhost ~]$
```

Create Directory

```
-mkdir [-p] <path> ...: Create a directory in specified location.
                  -p  Do not fail if the directory already exists
```

Example:

```
[cloudera@localhost ~]$ hadoop fs -mkdir /user/cloudera/5h2
[cloudera@localhost ~]$
```

Create Sub Directory:

Example

```
[cloudera@localhost ~]$ hadoop fs -mkdir /user/cloudera/5h2/exp2
[cloudera@localhost ~]$
```

Checking

```
[cloudera@localhost ~]$ hadoop fs -ls
Found 6 items
drwxr-xr-x   - cloudera cloudera          0 2023-08-10 21:49 5h2
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:34 A3
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:37 DIR
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 03:20 SUBDIR
drwxr-xr-x   - cloudera cloudera          0 2023-08-10 21:48 cse-c
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:49 user
[cloudera@localhost ~]$
```

To See list of Files

```
-ls [-d] [-h] [-R] [<path> ...]:       List the contents that match the specified file pattern. If
               path is not specified, the contents of /user/<currentUser>
               will be listed. Directory entries are of the form
                       dirName (full path) <dir>
               and file entries are of the form
                       fileName(full path) <r n> size
               where n is the number of replicas specified for the file
               and size is the size of the file, in bytes.
                 -d  Directories are listed as plain files.
                 -h  Formats the sizes of files in a human-readable fashion
                     rather than a number of bytes.
                 -R  Recursively list the contents of directories.
```

Example

1. **hadoop fs -ls**

```
[cloudera@localhost ~]$ hadoop fs -ls
Found 6 items
drwxr-xr-x   - cloudera cloudera          0 2023-08-10 21:49 5h2
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:34 A3
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:37 DIR
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 03:20 SUBDIR
drwxr-xr-x   - cloudera cloudera          0 2023-08-10 21:48 cse-c
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:49 user
[cloudera@localhost ~]$
```

hadoop fs –ls /

```
[cloudera@localhost ~]$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - hbase hbase               0 2014-06-01 16:16 /hbase
drwxr-xr-x   - solr  solr                0 2014-06-01 16:16 /solr
drwxrwxrwx   - hdfs  supergroup          0 2014-06-01 16:16 /tmp
drwxr-xr-x   - hdfs  supergroup          0 2014-06-01 16:17 /user
drwxr-xr-x   - hdfs  supergroup          0 2014-06-01 16:15 /var
[cloudera@localhost ~]$
```

hadoop fs –ls –R or hadoop fs –lsr

```
[cloudera@localhost ~]$ hadoop fs -lsr
lsr: DEPRECATED: Please use 'ls -R' instead.
drwxr-xr-x   - cloudera cloudera          0 2023-08-10 21:49 5h2
drwxr-xr-x   - cloudera cloudera          0 2023-08-10 21:49 5h2/exp2
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:34 A3
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:37 DIR
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 03:20 SUBDIR
-rw-r--r--   3 cloudera cloudera         33 2023-08-08 03:18 SUBDIR/sample
-rw-r--r--   3 cloudera cloudera         33 2023-08-08 03:20 SUBDIR/sample1
drwxr-xr-x   - cloudera cloudera          0 2023-08-10 21:48 cse-c
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:49 user
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 03:06 user/cloudera
drwxr-xr-x   - cloudera cloudera          0 2023-08-08 02:49 user/cloudera/SUBDI
R1
-rw-r--r--   3 cloudera cloudera         39 2023-08-08 02:54 user/cloudera/file1
-rw-r--r--   3 cloudera cloudera         39 2023-08-08 03:06 user/cloudera/file2
[cloudera@localhost ~]$
```

put

```
-put <localsrc> ... <dst>:      Copy files from the local file system
                 into fs.
```

Example

```
[cloudera@localhost ~]$ touch h2.txt
[cloudera@localhost ~]$ vi h2.txt
[cloudera@localhost ~]$ cat h2.txt
welcome to lab
[cloudera@localhost ~]$ hadoop fs -put h1.txt  /user/cloudera/5h2
```

copyFromLocal

```
-copyFromLocal <localsrc> ... <dst>:    Identical to the -put command.
```

Example

```
[cloudera@localhost ~]$ touch two.txt
[cloudera@localhost ~]$ vi two.txt
[cloudera@localhost ~]$ cat two.txt
Welcome to Basic Hadoop Commands

[cloudera@localhost ~]$ ls
1.txt  balu      Desktop    Downloads  lib    manoj2  one.txt   Public     two.txt  workspace
2.txt  datasets  Documents  eclipse    manoj  Music   Pictures  Templates  Videos
[cloudera@localhost ~]$ hadoop fs -lsr user/cloduera/CSE-C
lsr: DEPRECATED: Please use 'ls -R' instead.
lsr: `user/cloduera/CSE-C': No such file or directory
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C
lsr: DEPRECATED: Please use 'ls -R' instead.
drwxr-xr-x   - cloudera cloudera          0 2023-08-09 01:37 user/cloudera/CSE-C/exp2
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 01:46 user/cloudera/CSE-C/one.txt
[cloudera@localhost ~]$ hadoop fs -copyFromLocal two.txt user/cloudera/CSE-C
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C
lsr: DEPRECATED: Please use 'ls -R' instead.
drwxr-xr-x   - cloudera cloudera          0 2023-08-09 01:37 user/cloudera/CSE-C/exp2
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 01:46 user/cloudera/CSE-C/one.txt
-rw-r--r--   3 cloudera cloudera         34 2023-08-09 01:50 user/cloudera/CSE-C/two.txt
[cloudera@localhost ~]$
```

GET

```
-get [-ignoreCrc] [-crc] <src> ... <localdst>:  Copy files that match the file pattern <src>
                to the local name.  <src> is kept.  When copying multiple,
                files, the destination must be a directory.
```

Example:

```
[cloudera@localhost ~]$ ls
h2      add.sh    Desktop    eclipse  lib              Pictures   Videos
3       add.ty    divitha    file     m.c              Public     workspace
3-1     Afile     Documents  h2       multiplication.c  r1
dd.py   datasets  Downloads  h2.txt   Music            Templates
[cloudera@localhost ~]$
```

```
[cloudera@localhost ~]$ hadoop fs -get user/cloudera/CSE-C/one.txt
[cloudera@localhost ~]$ ls
1.txt  balu      Desktop    Downloads  lib    manoj2  one.txt   Public     Videos
2.txt  datasets  Documents  eclipse    manoj  Music   Pictures  Templates  workspace
[cloudera@localhost ~]$
```

copyToLocal

```
-copyToLocal [-ignoreCrc] [-crc] <src> ... <localdst>:  Identical to the -get command.
```

Example

```
[cloudera@localhost ~]$ hadoop fs -copyToLocal user/cloudera/CSE-C/two.txt
[cloudera@localhost ~]$ ls
1.txt  balu      Desktop    Downloads  lib    manoj2  one.txt   Public     two.txt  workspace
2.txt  datasets  Documents  eclipse    manoj  Music   Pictures  Templates  Videos
[cloudera@localhost ~]$
```

Remove

```
-rm [-f] [-r|-R] [-skipTrash] <src> ...:       Delete all files that match the specified file pattern.
               Equivalent to the Unix command "rm <src>"
               -skipTrash option bypasses trash, if enabled, and immediately
               deletes <src>
                 -f     If the file does not exist, do not display a diagnostic
                        message or modify the exit status to reflect an error.
                 -[rR]  Recursively deletes directories
```

LookUp

Help:

```
-help [cmd ...]:        Displays help for given command or all commands if none
                is specified.
```

Example

```
[cloudera@localhost ~]$ hadoop fs -help help
-help [cmd ...]:        Displays help for given command or all commands if none
                is specified.
[cloudera@localhost ~]$
```

cp: copy file from source to destination

```
-cp <src> ... <dst>:    Copy files that match the file pattern <src> to a
                destination.  When copying multiple files, the destination
                must be a directory.
```

Example

```
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C
lsr: DEPRECATED: Please use 'ls -R' instead.
drwxr-xr-x   - cloudera cloudera          0 2023-08-09 01:37 user/cloudera/CSE-C/exp2
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 01:46 user/cloudera/CSE-C/one.txt
-rw-r--r--   3 cloudera cloudera         34 2023-08-09 01:50 user/cloudera/CSE-C/two.txt
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C/exp2
lsr: DEPRECATED: Please use 'ls -R' instead.
[cloudera@localhost ~]$ hadoop fs -cp user/cloudera/CSE-C/one.txt user/cloudera/CSE-C/exp2
[cloudera@localhost ~]$ hadoop fs -lsr  user/cloudera/CSE-C/exp2
lsr: DEPRECATED: Please use 'ls -R' instead.
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 02:12 user/cloudera/CSE-C/exp2/one.txt
[cloudera@localhost ~]$
```

mv:Move File From Source to Destination

```
-mv <src> ... <dst>:    Move files that match the specified file pattern <src>
                to a destination <dst>.  When moving multiple files, the
                destination must be a directory.
```

Example

```
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C
lsr: DEPRECATED: Please use 'ls -R' instead.
drwxr-xr-x   - cloudera cloudera          0 2023-08-09 02:12 user/cloudera/CSE-C/exp2
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 02:12 user/cloudera/CSE-C/exp2/one.txt
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 01:46 user/cloudera/CSE-C/one.txt
-rw-r--r--   3 cloudera cloudera         34 2023-08-09 01:50 user/cloudera/CSE-C/two.txt
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-D/exp2
lsr: DEPRECATED: Please use 'ls -R' instead.
lsr: `user/cloudera/CSE-D/exp2': No such file or directory
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C/exp2
lsr: DEPRECATED: Please use 'ls -R' instead.
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 02:12 user/cloudera/CSE-C/exp2/one.txt
[cloudera@localhost ~]$ hadoop fs -mv user/cloudera/CSE-C/two.txt user/cloudera/CSE-C/exp2
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C
lsr: DEPRECATED: Please use 'ls -R' instead.
drwxr-xr-x   - cloudera cloudera          0 2023-08-09 02:17 user/cloudera/CSE-C/exp2
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 02:12 user/cloudera/CSE-C/exp2/one.txt
-rw-r--r--   3 cloudera cloudera         34 2023-08-09 01:50 user/cloudera/CSE-C/exp2/two.txt
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 01:46 user/cloudera/CSE-C/one.txt
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C/exp2
lsr: DEPRECATED: Please use 'ls -R' instead.
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 02:12 user/cloudera/CSE-C/exp2/one.txt
-rw-r--r--   3 cloudera cloudera         34 2023-08-09 01:50 user/cloudera/CSE-C/exp2/two.txt
[cloudera@localhost ~]$
```

du:Show Disk Usage

```
-du [-s] [-h] <path> ...:       Show the amount of space, in bytes, used by the files that
                match the specified file pattern. The following flags are optional:
                  -s   Rather than showing the size of each individual file that
                       matches the pattern, shows the total (summary) size.
                  -h   Formats the sizes of files in a human-readable fashion
                       rather than a number of bytes.

                Note that, even without the -s option, this only shows size summaries
                one level deep into a directory.
                The output is in the form
                        size    name(full path)
```

Example

```
[cloudera@localhost ~]$ hadoop fs -du
39  .Trash
0   balu
0   sample.txt
92  user
[cloudera@localhost ~]$ hadoop fs -du user/cloudera/CSE-C
53  user/cloudera/CSE-C/exp2
19  user/cloudera/CSE-C/one.txt
[cloudera@localhost ~]$
```

dus:-summary of Disk Usage

```
dus:    (DEPRECATED) Same as 'du -s'
```

```
[cloudera@localhost ~]$ hadoop fs -dus user/cloudera/CSE-C
dus: DEPRECATED: Please use 'du -s' instead.
72  user/cloudera/CSE-C
[cloudera@localhost ~]$ hadoop fs -dus user/cloudera/CSE-C/exp2
dus: DEPRECATED: Please use 'du -s' instead.
53  user/cloudera/CSE-C/exp2
[cloudera@localhost ~]$
```

moveFromLocal

```
-moveFromLocal <localsrc> ... <dst>:    Same as -put, except that the source is
                deleted after it's copied.
```

Example

```
[cloudera@localhost ~]$ ls
1.txt  balu      Desktop    Downloads  lib    manoj2  one.txt   Public     three.txt  Videos
2.txt  datasets  Documents  eclipse    manoj  Music   Pictures  Templates  two.txt    workspace
[cloudera@localhost ~]$ hadoop fs -moveFromLocal three.txt user/cloudera/CSE-C
[cloudera@localhost ~]$ ls
1.txt  balu      Desktop    Downloads  lib    manoj2  one.txt   Public     two.txt  workspace
2.txt  datasets  Documents  eclipse    manoj  Music   Pictures  Templates  Videos
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C
lsr: DEPRECATED: Please use 'ls -R' instead.
drwxr-xr-x   - cloudera cloudera          0 2023-08-09 02:17 user/cloudera/CSE-C/exp2
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 02:12 user/cloudera/CSE-C/exp2/one.txt
-rw-r--r--   3 cloudera cloudera         34 2023-08-09 01:50 user/cloudera/CSE-C/exp2/two.txt
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 01:46 user/cloudera/CSE-C/one.txt
-rw-r--r--   3 cloudera cloudera         42 2023-08-09 02:25 user/cloudera/CSE-C/three.txt
[cloudera@localhost ~]$
```

moveToLocal

```
-moveToLocal <src> <localdst>:  Not implemented yet
```

Example

```
[cloudera@localhost ~]$ hadoop fs -moveToLocal user/cloudera/CSE-C/three.txt balu
moveToLocal: Option '-moveToLocal' is not implemented yet.
```

touchZ

```
-touchz <path> ...:     Creates a file of zero length
                at <path> with current time as the timestamp of that <path>.
                An error is returned if the file exists with non-zero length
```

Example

```
[cloudera@localhost ~]$ hadoop fs -touchz user/cloudera/CSE-C/exp2/four.txt
[cloudera@localhost ~]$ hadoop fs -lsr user/cloudera/CSE-C/exp2
lsr: DEPRECATED: Please use 'ls -R' instead.
-rw-r--r--   3 cloudera cloudera          0 2023-08-09 02:33 user/cloudera/CSE-C/exp2/four.txt
-rw-r--r--   3 cloudera cloudera         19 2023-08-09 02:12 user/cloudera/CSE-C/exp2/one.txt
-rw-r--r--   3 cloudera cloudera         34 2023-08-09 01:50 user/cloudera/CSE-C/exp2/two.txt
[cloudera@localhost ~]$
```

Expunge

```
-expunge:       Empty the Trash
[cloudera@localhost ~]$ hadoop fs -help expunge
-expunge:       Empty the Trash
[cloudera@localhost ~]$ hadoop fs -expunge user/cloudera/CSE-C/exp2
-expunge: Too many arguments: expected 0 but got 1
Usage: hadoop fs [generic options] -expunge
[cloudera@localhost ~]$
```

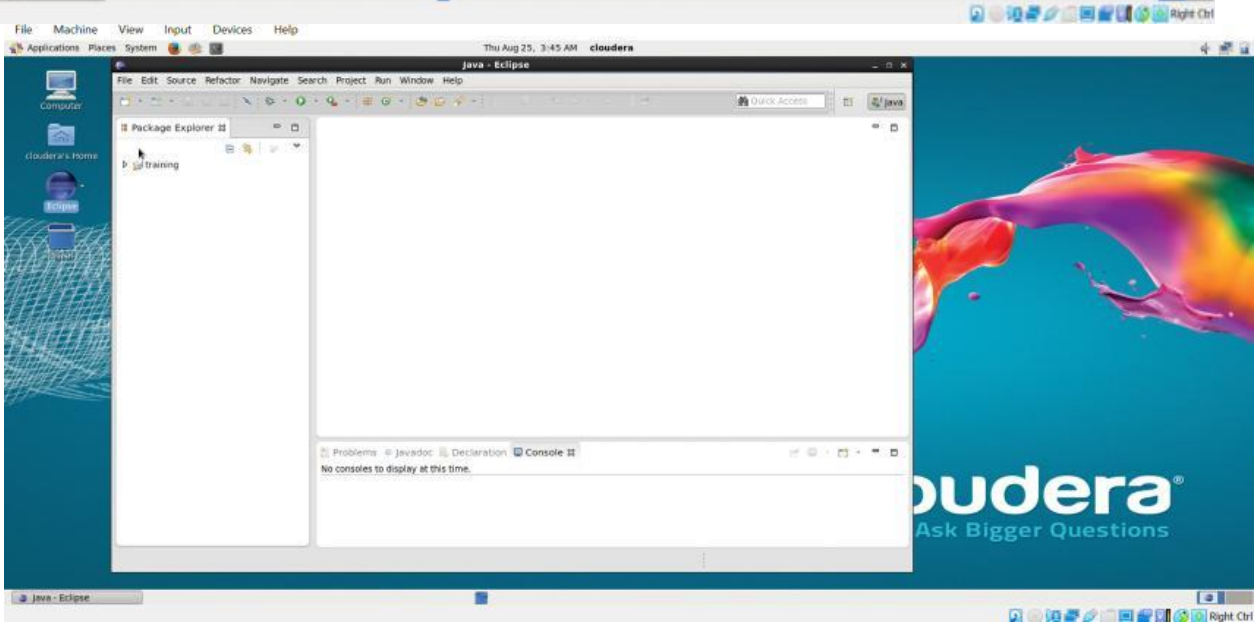
Getmerge

```
-getmerge [-nl] <src> <localdst>:       Get all the files in the directories that
                match the source file pattern and merge and sort them to only
                one file on local fs. <src> is kept.
                  -nl   Add a newline character at the end of each file.
```

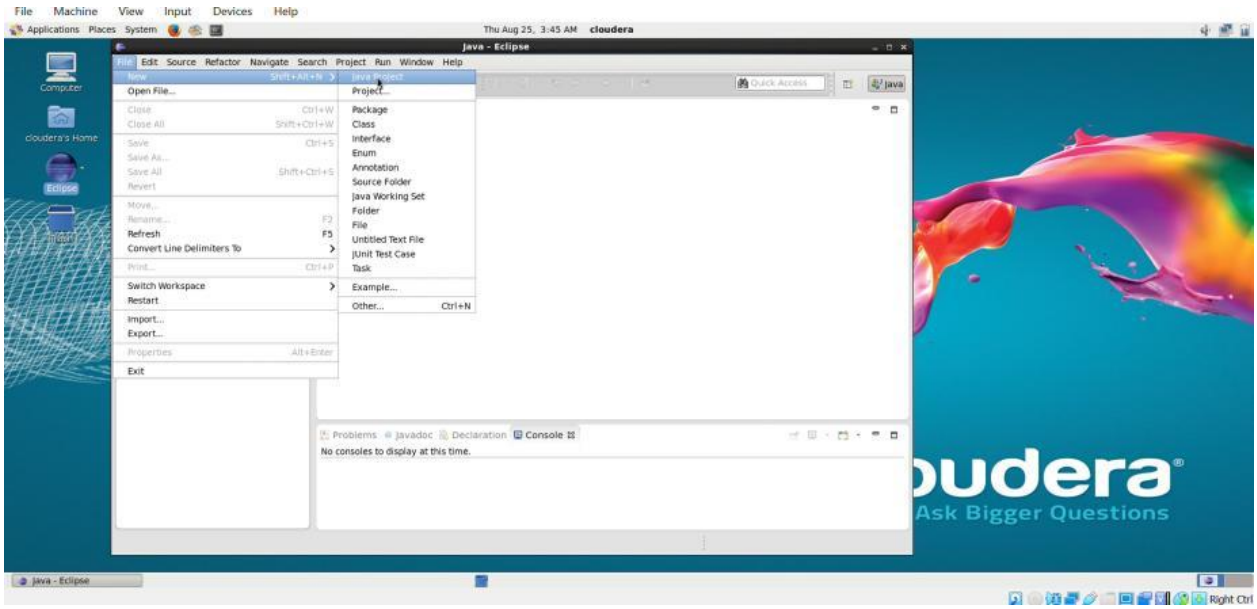
Experiment 3:

Aim : Write Driver Code, Mapper code, Reducer Code to count number of words in a given file.

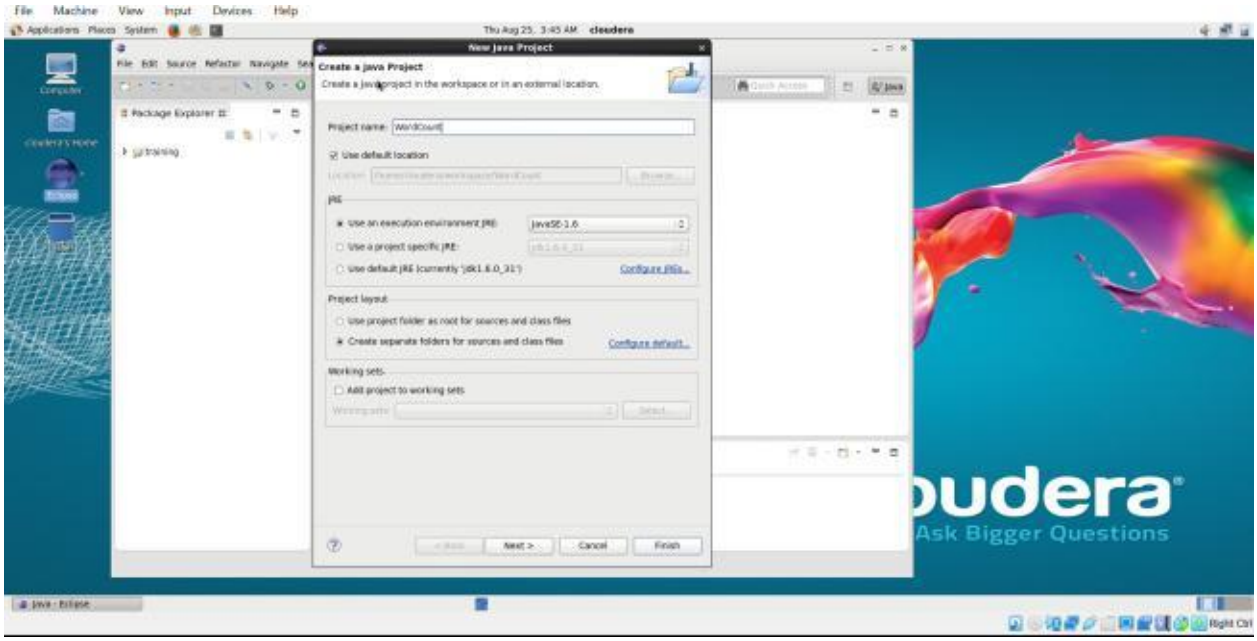
1. Open Oracle VM Virtual box -> click start -> open cloudera -> open eclipse.



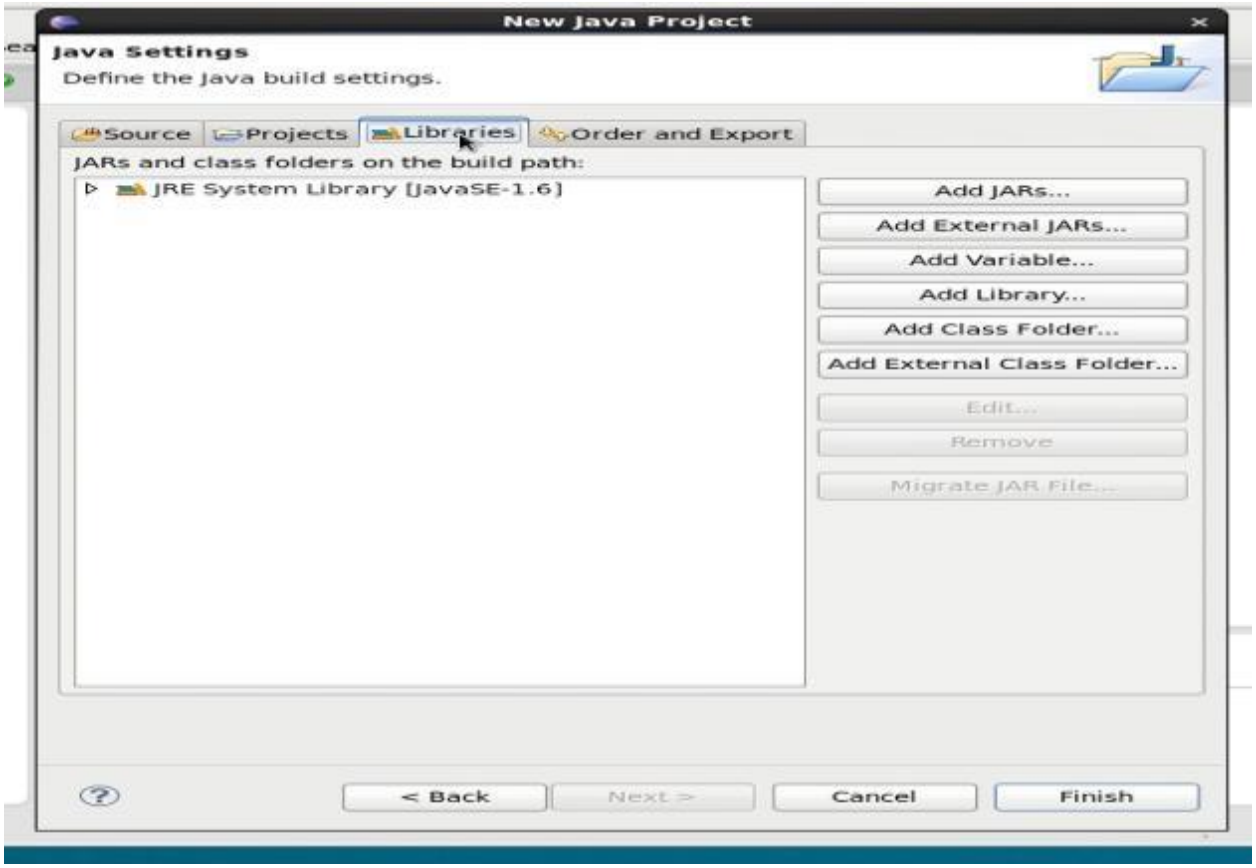
2. Click on file->New->Java Project



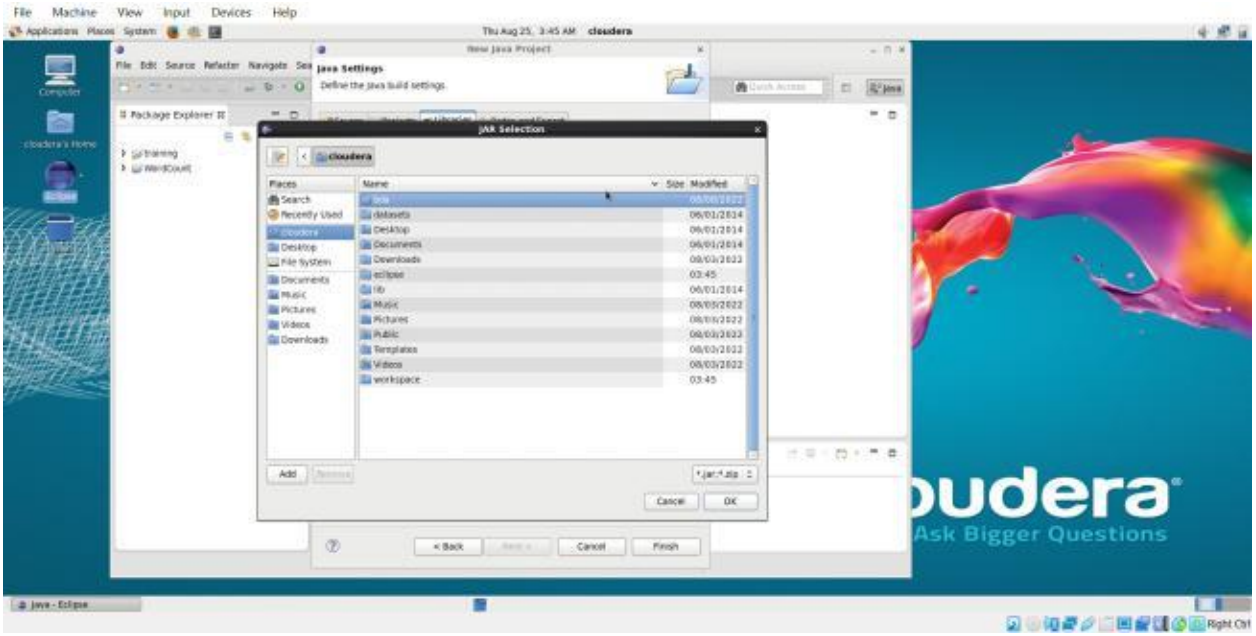
3. Give your project name as "**WordCount**" and click on next



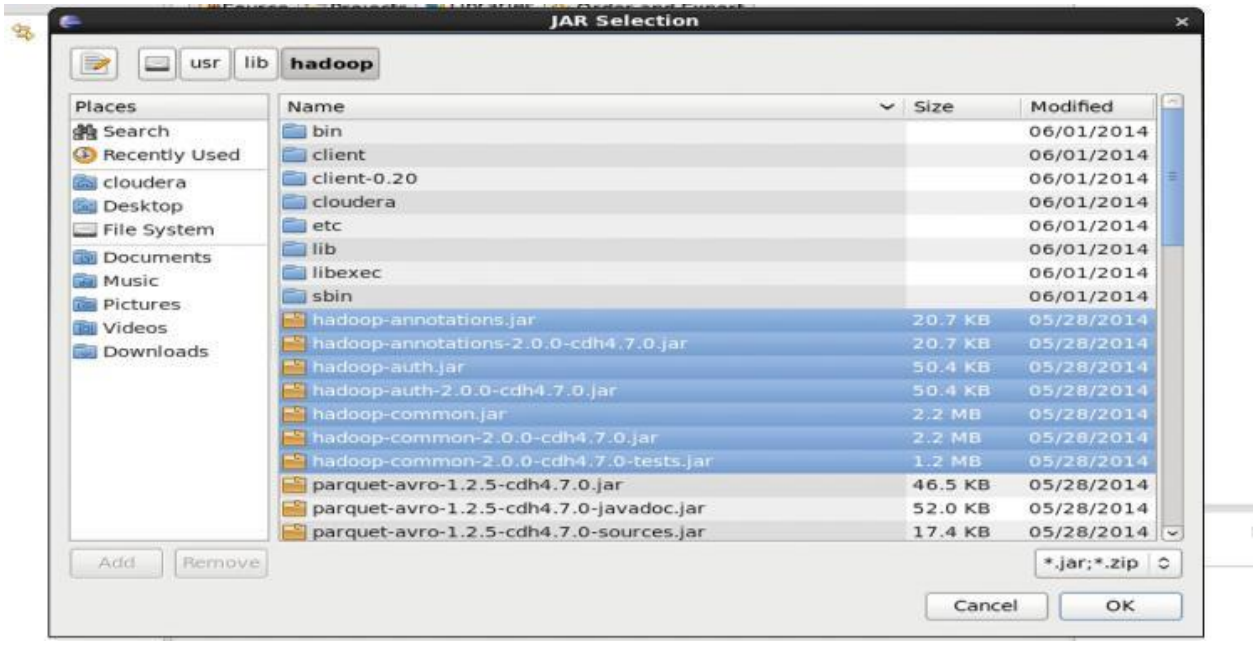
4. After that, click on libraries on the top and click on add external jars on the right side



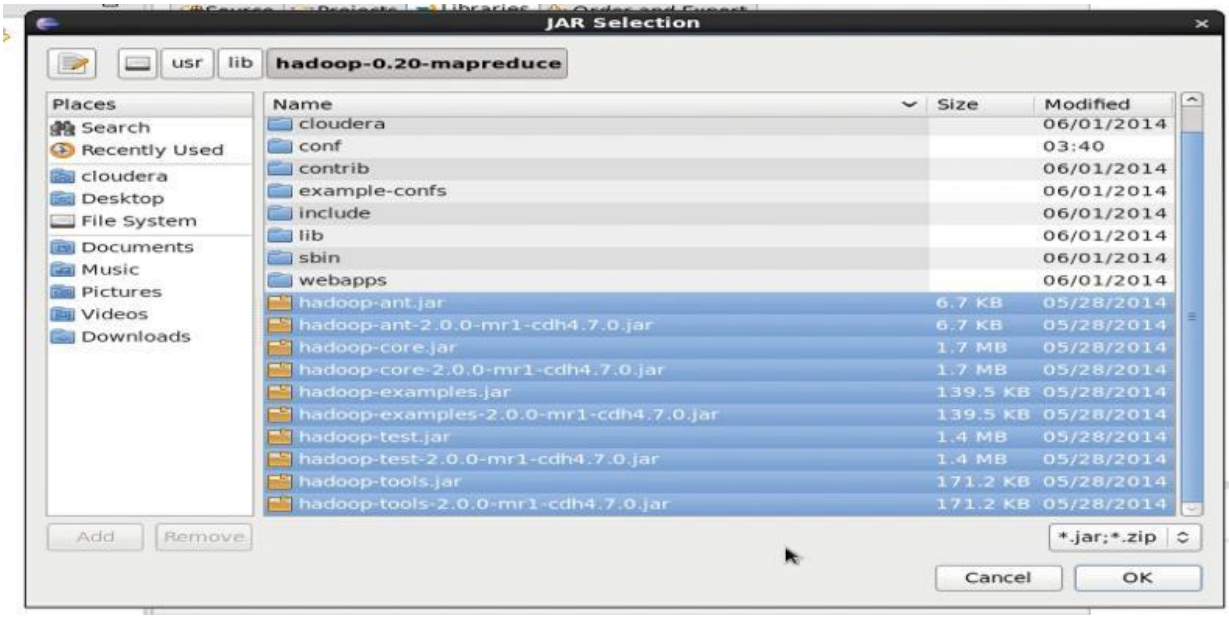
5. A dialog box will appear



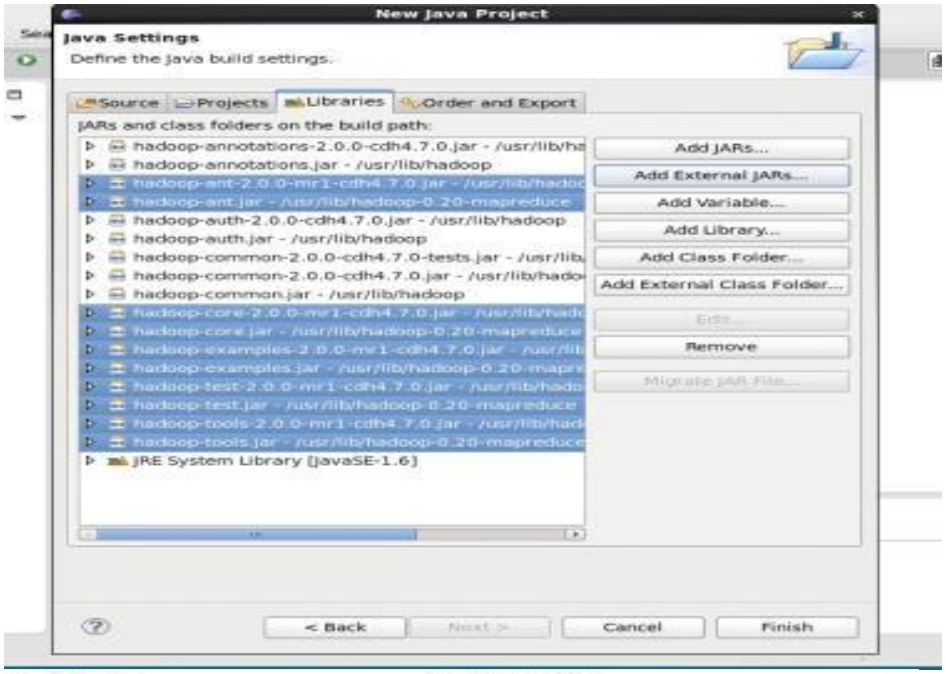
6. Click on file system on the left, and then usr->lib ->hadoop.
   Select all the files named with hadoop and click ok.

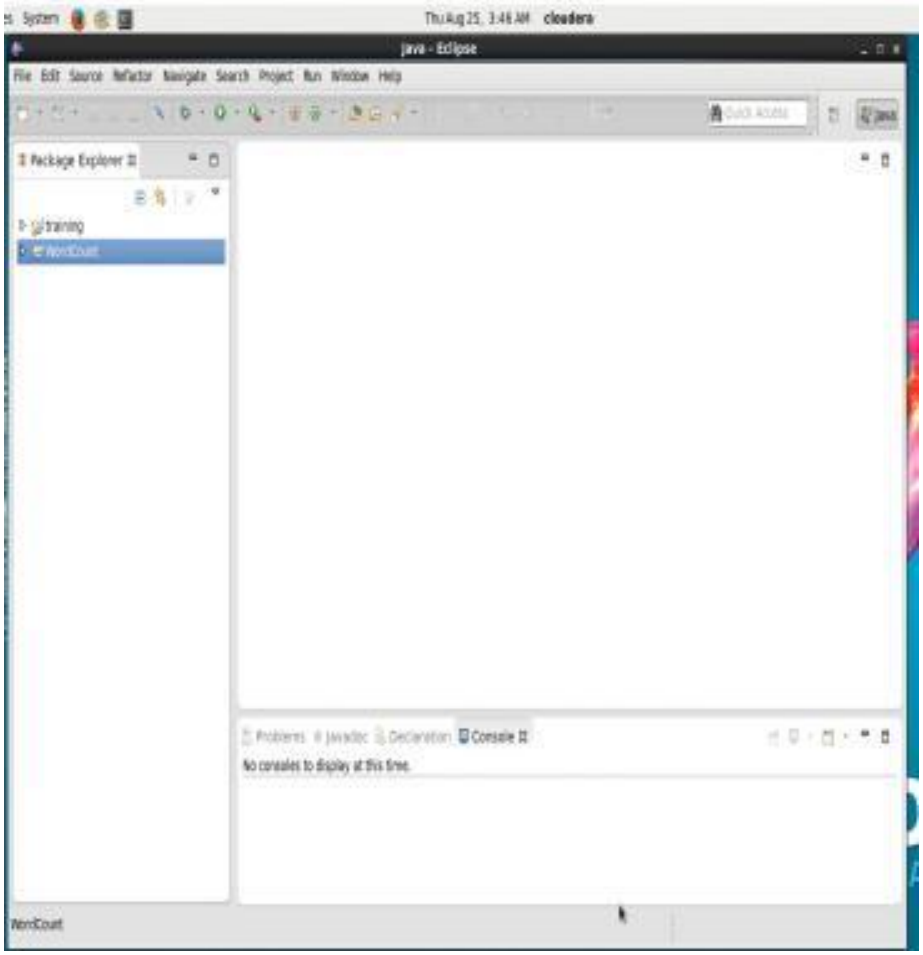


7. Go back to lib and click **hadoop-0.20-mapreduce** and select all the files named with hadoop and click ok

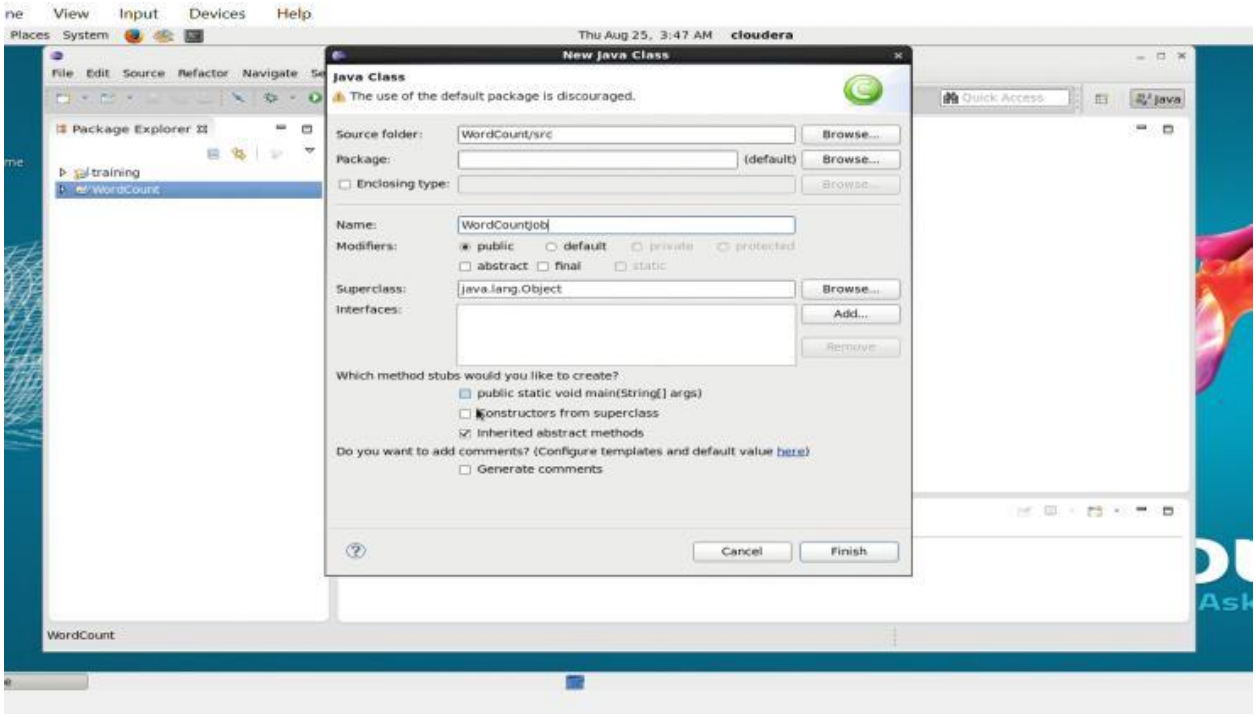


8. All the external files are added and now click on finish. The project is created successfully.

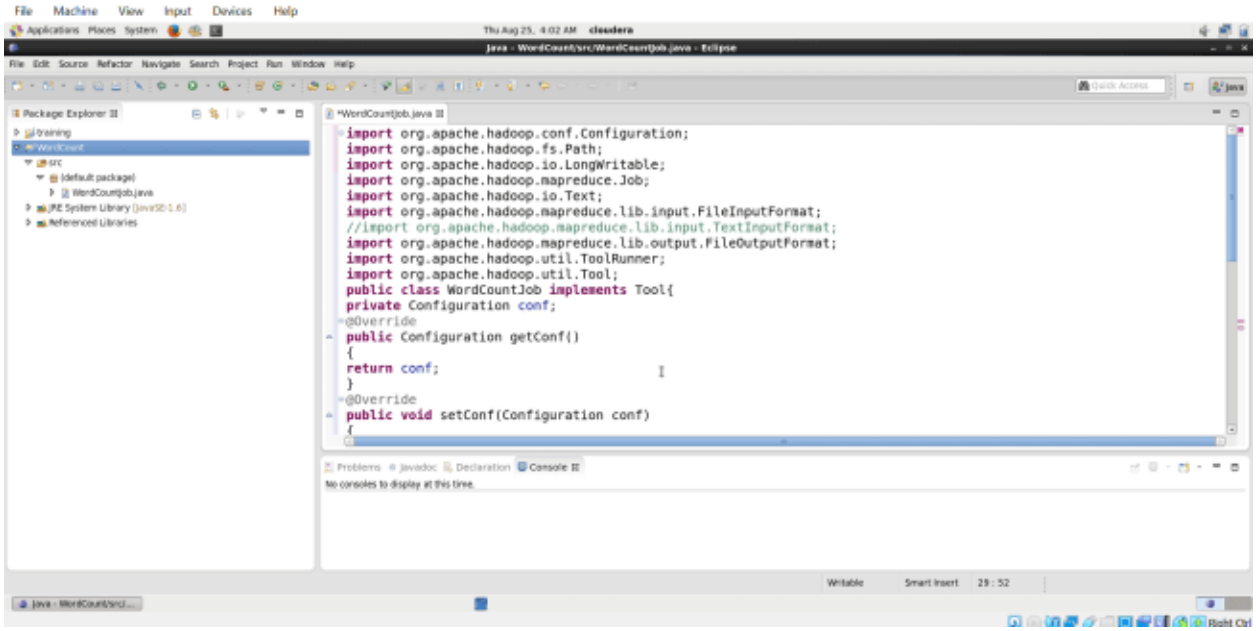




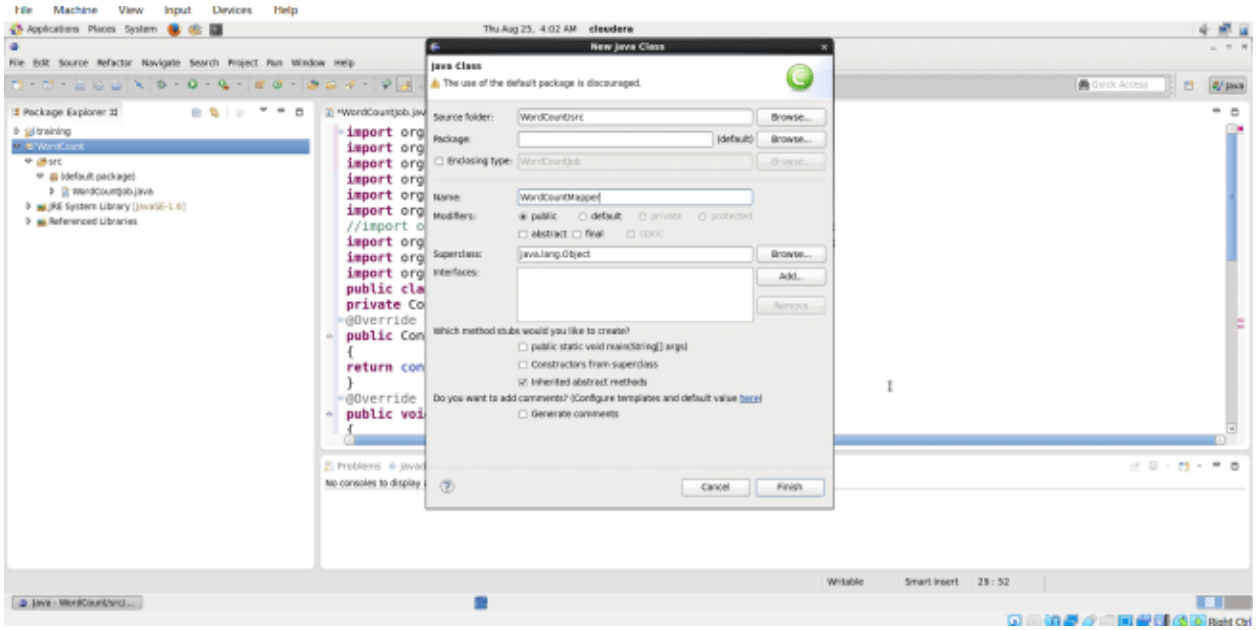
9. Create a class with name "**WordCountJob**"



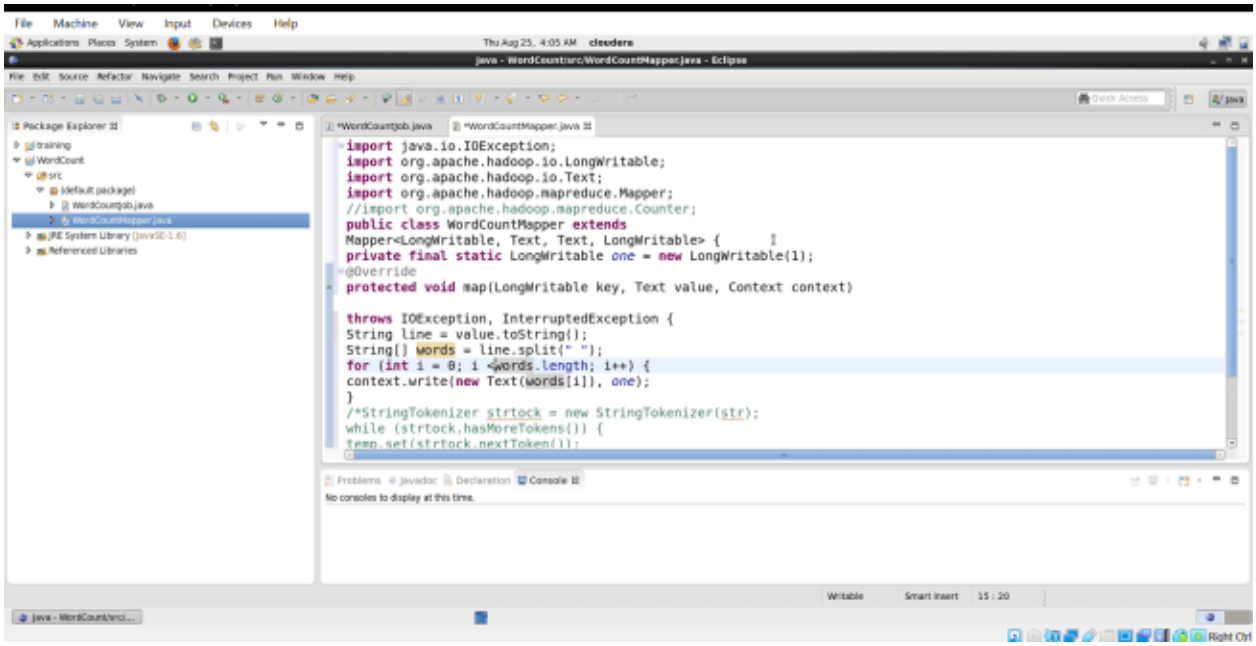
10. Copy and paste the program 4 lines at a time-If copy and paste doesn't work then goto device in cloudera and set both shared clipboard and drag and drop as **bidirectional.**



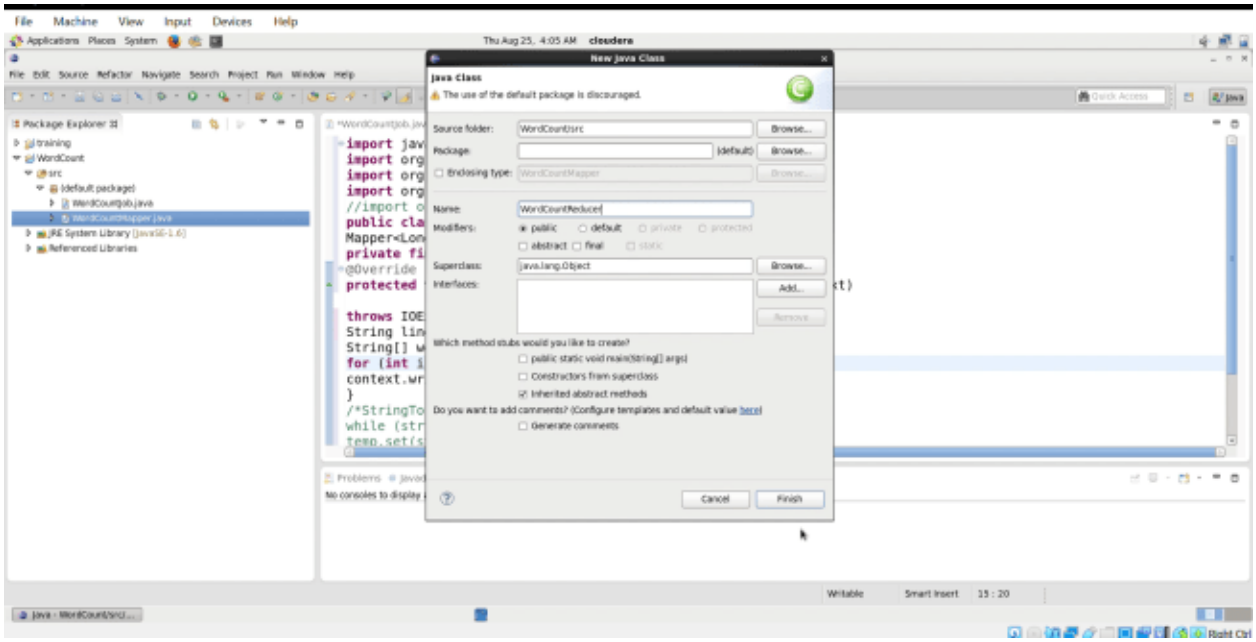
11. Create another class "**WordCountMapper**"



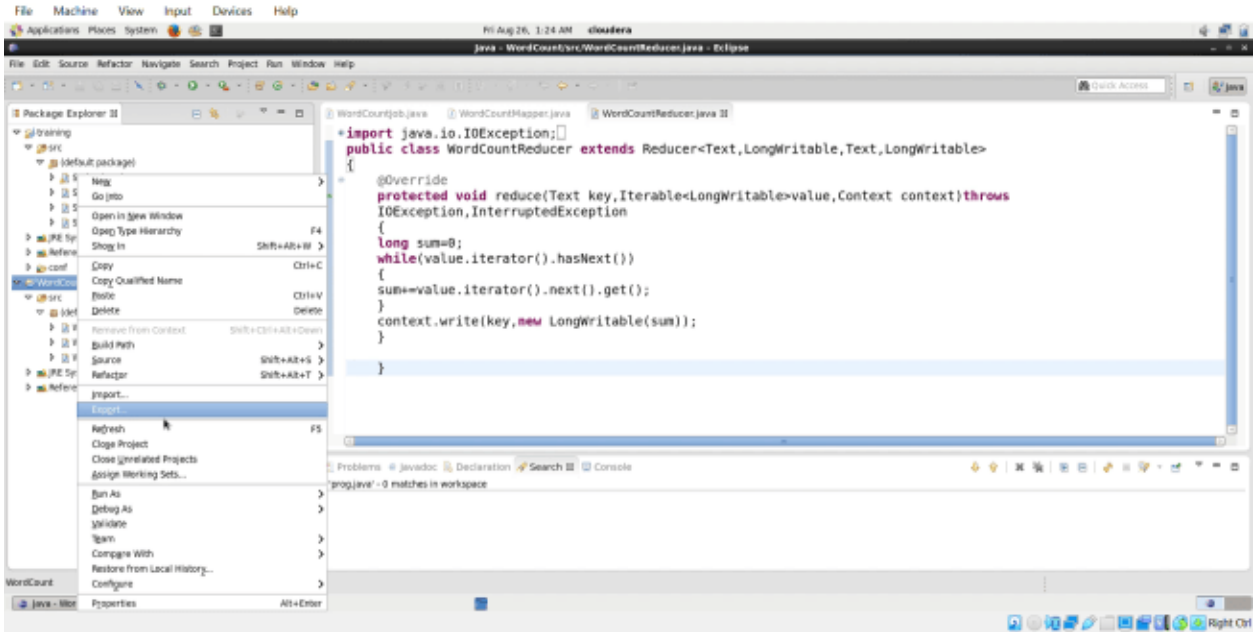
12. Copy and paste the program



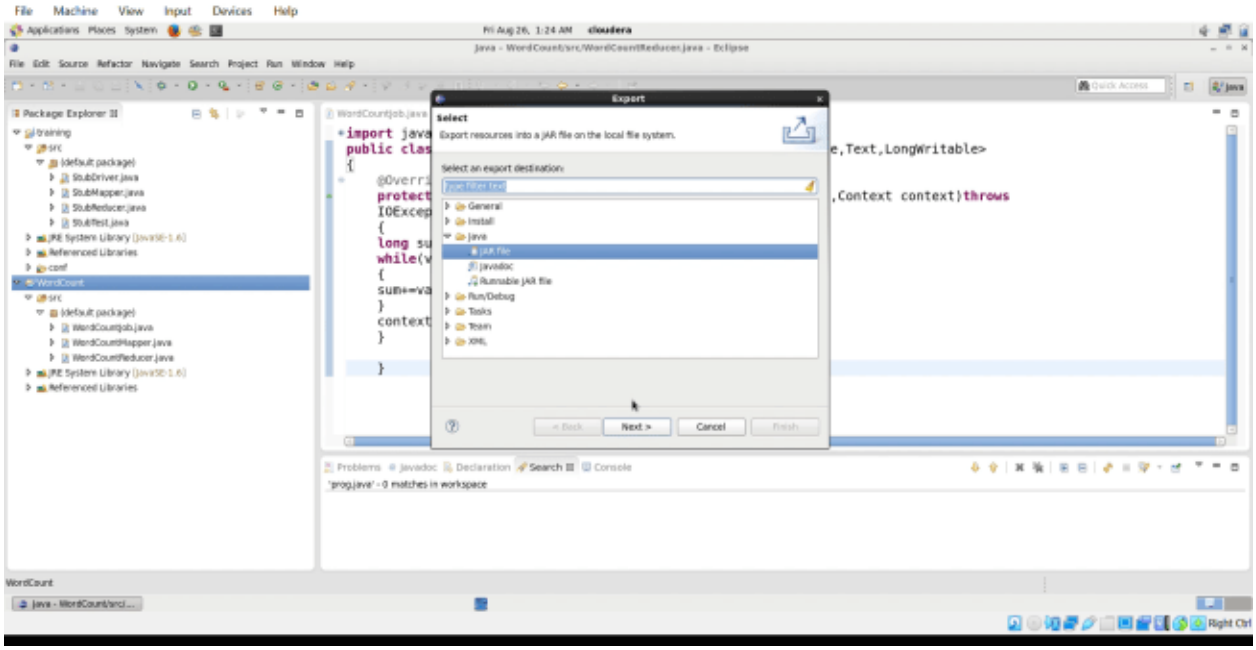
13. Create another class "**WordCountReducer**"



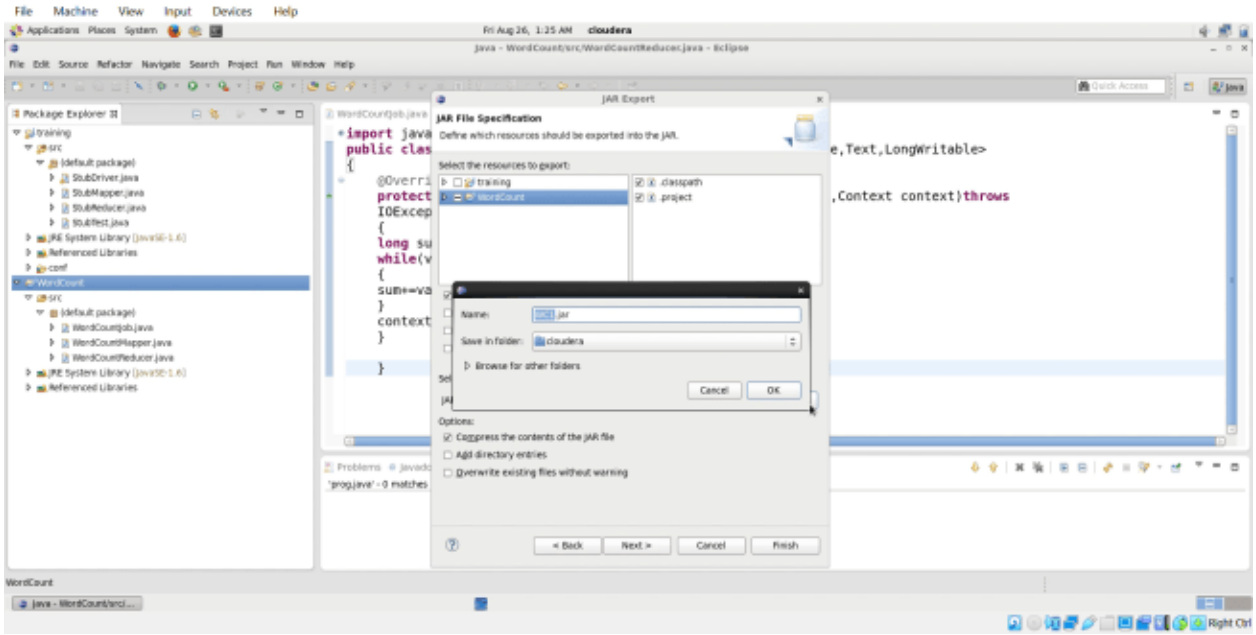
14. After finishing all the programs, right click on your project and select export.



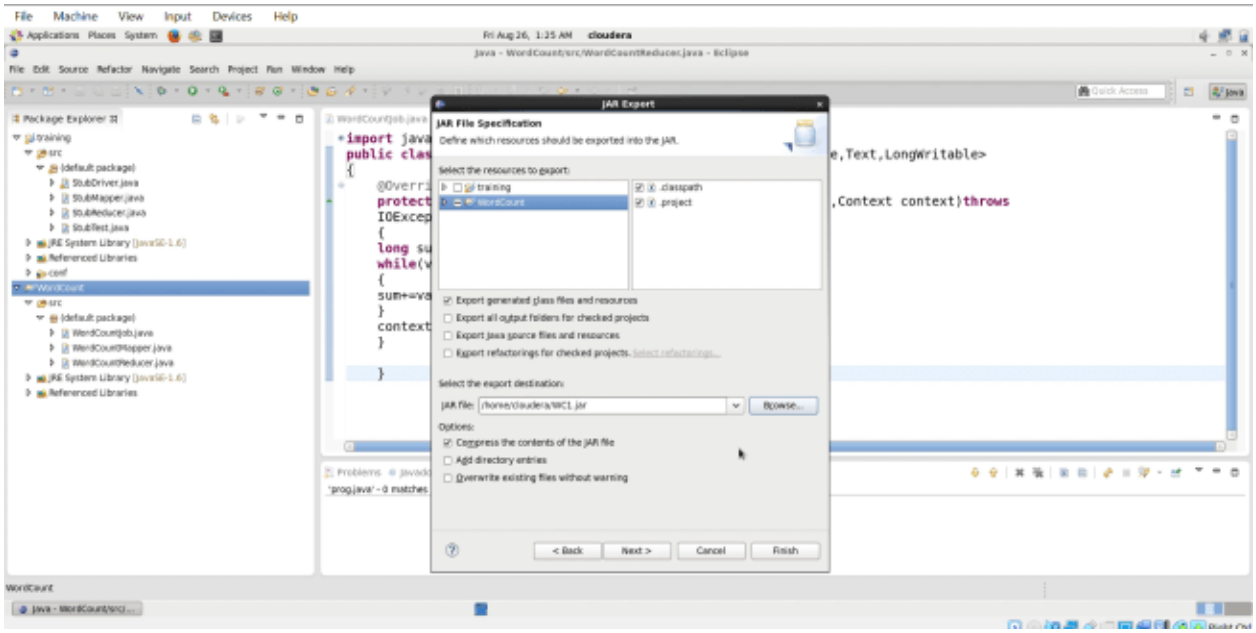
15. Select java-> JAR file->next



16. Click on browse and a dialog box will appear, give any name for the jar file and click ok.

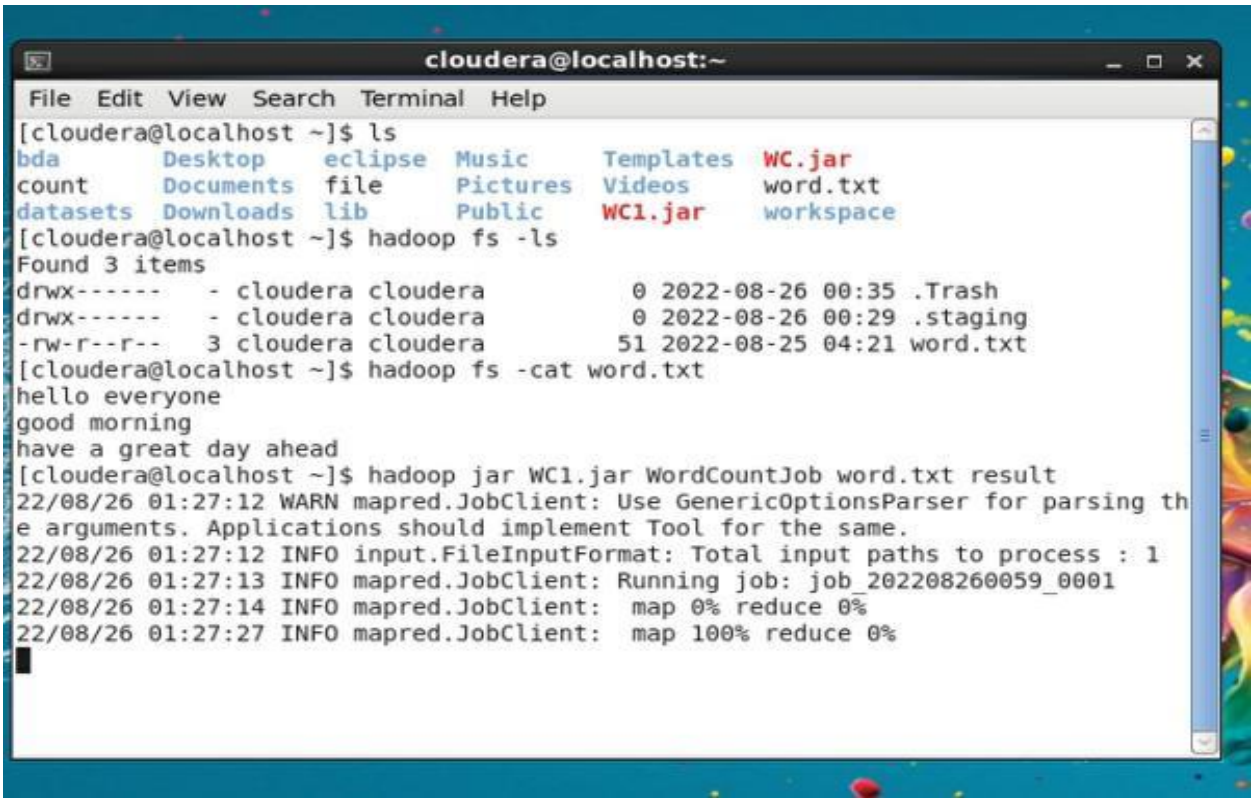


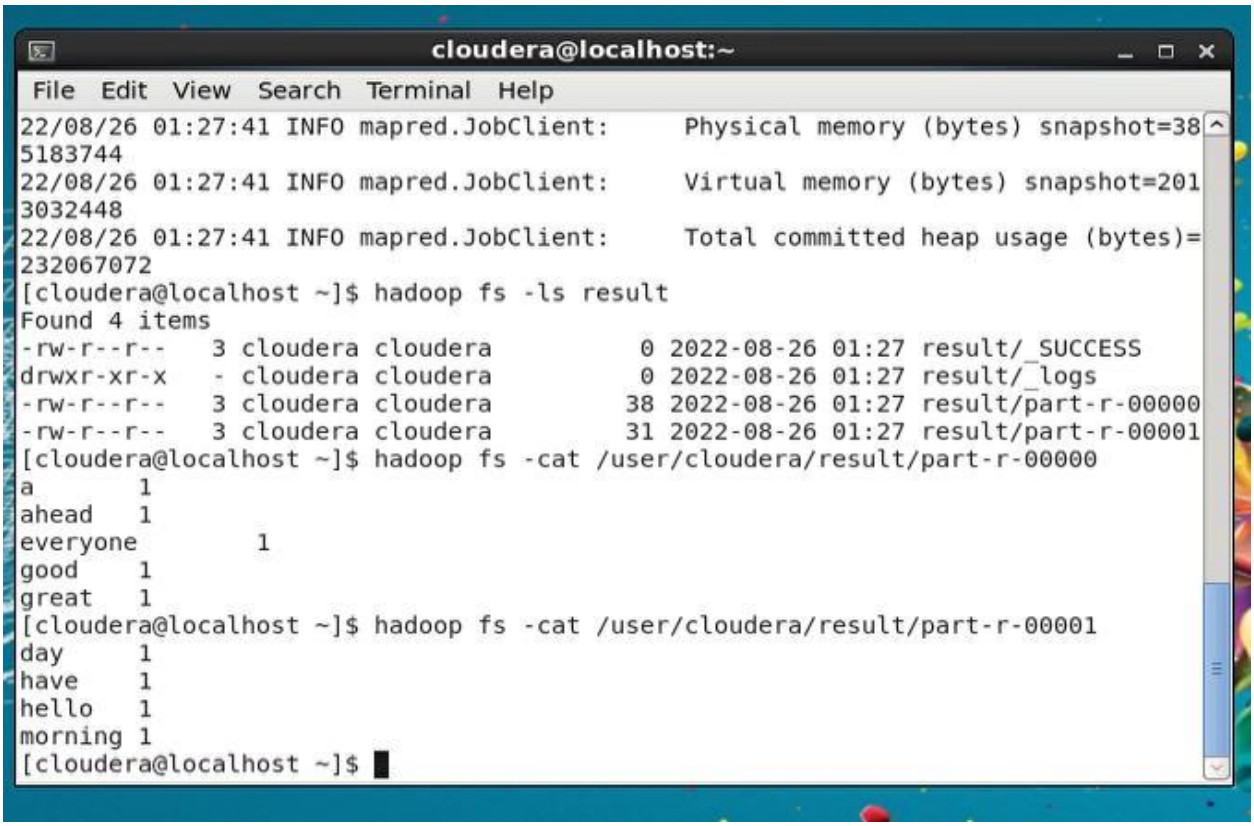
17. Click on finish



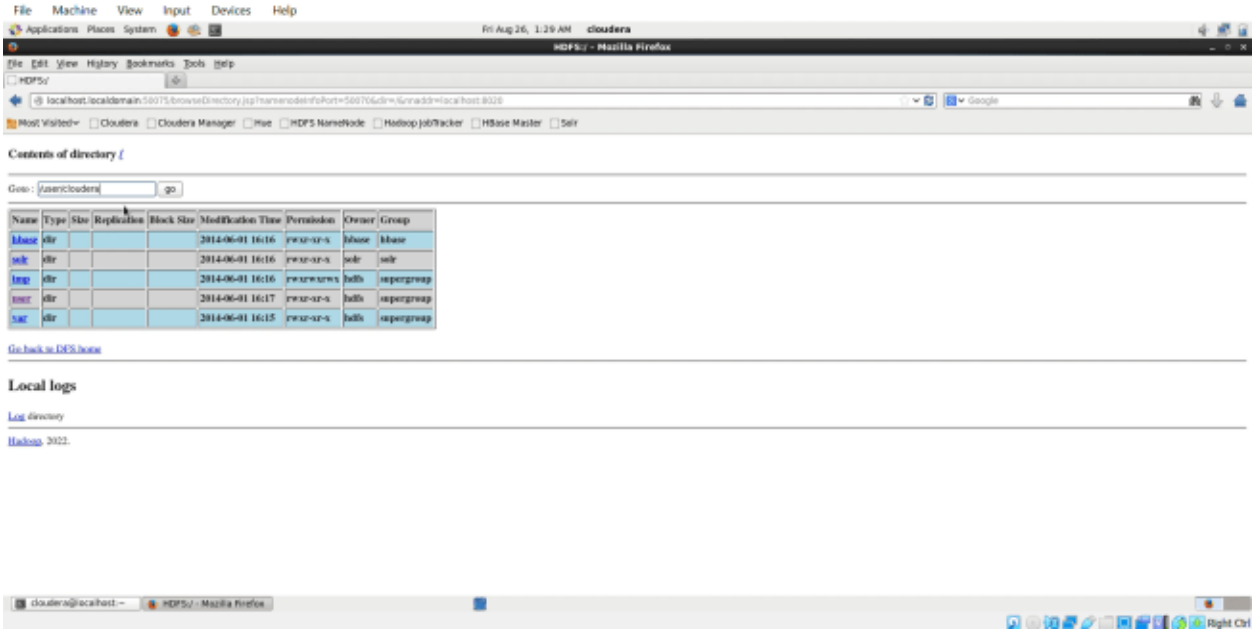
18. Now, open the terminal and create a text file “word”.

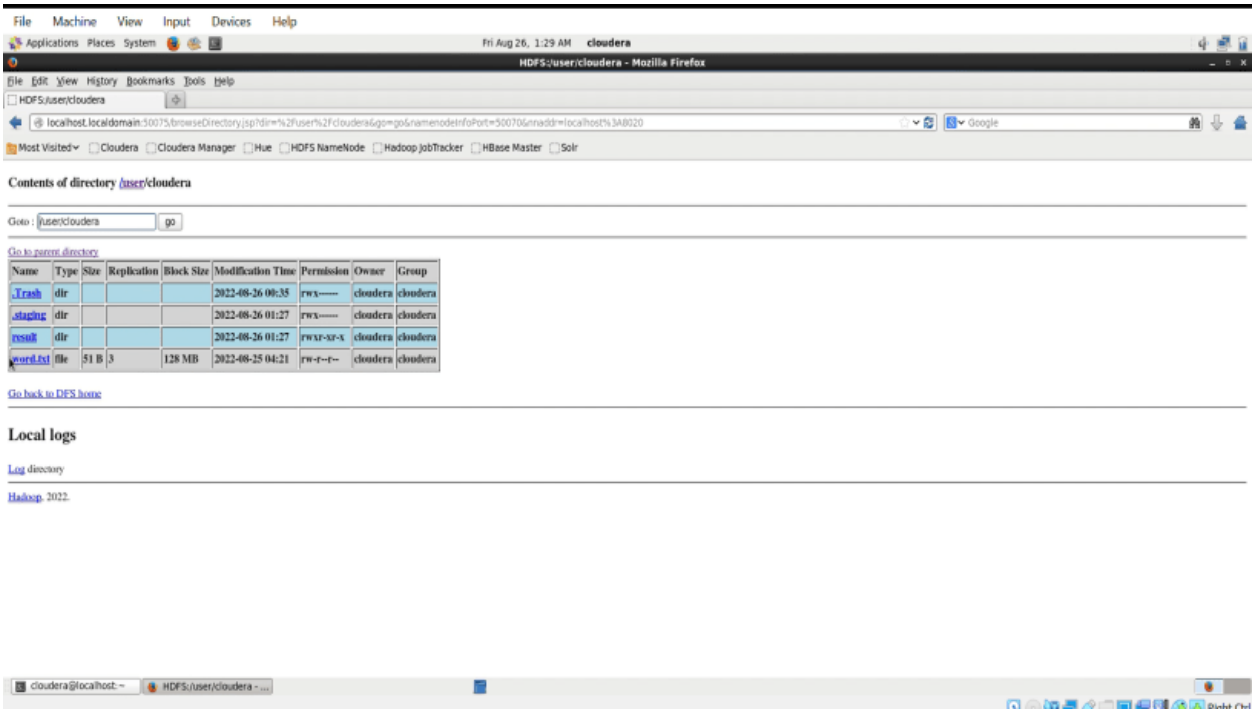
19. Insert few sentences in the file.

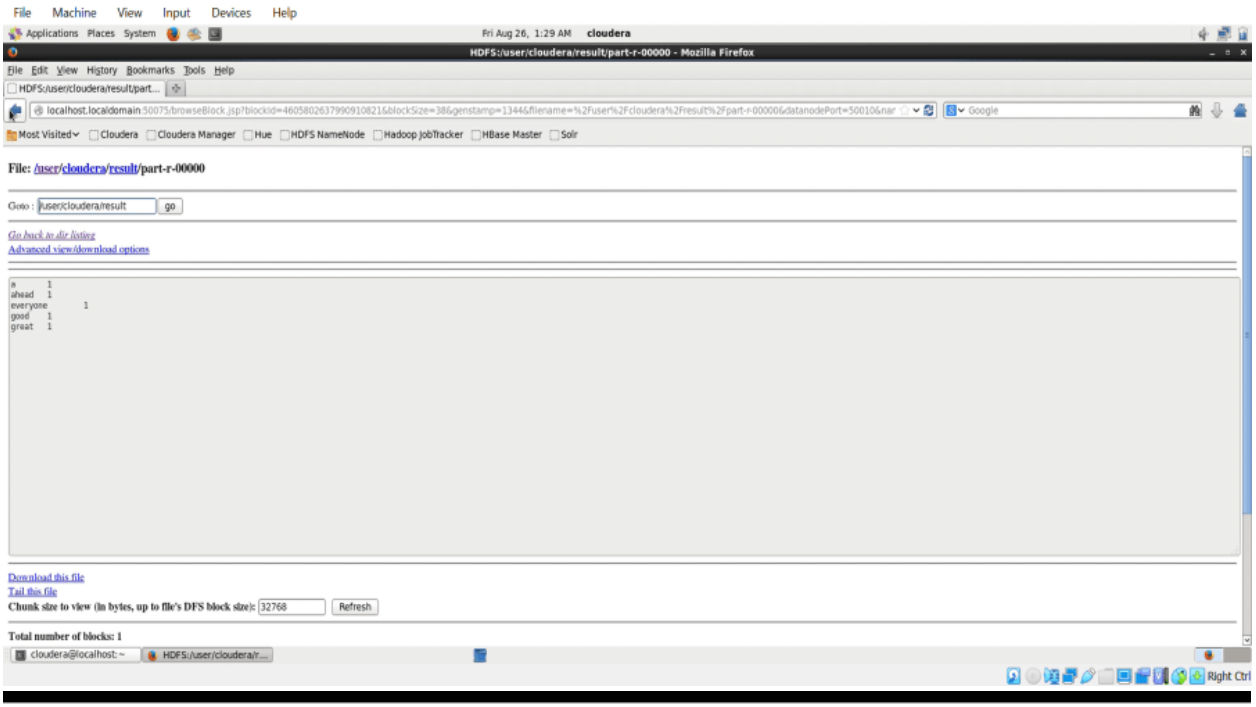

cloudera@localhost:~
File Edit View Search Terminal Help
[cloudera@localhost ~]$ ls
bda Desktop eclipse Music Templates WC.jar
count Documents file Pictures Videos word.txt
datasets Downloads lib Public WC1.jar workspace
[cloudera@localhost ~]$ hadoop fs -ls
Found 3 items
drwx------ - cloudera cloudera 0 2022-08-26 00:35 .Trash
drwx------ - cloudera cloudera 0 2022-08-26 00:29 .staging
-rw-r--r-- 3 cloudera cloudera 51 2022-08-25 04:21 word.txt
[cloudera@localhost ~]$ hadoop fs -cat word.txt
hello everyone
good morning
have a great day ahead
[cloudera@localhost ~]$ hadoop jar WC1.jar WordCountJob word.txt result
22/08/26 01:27:12 WARN mapred.JobClient: Use GenericOptionsParser for parsing th
e arguments. Applications should implement Tool for the same.
22/08/26 01:27:12 INFO input.FileInputFormat: Total input paths to process : 1
22/08/26 01:27:13 INFO mapred.JobClient: Running job: job_202208260059_0001
22/08/26 01:27:14 INFO mapred.JobClient: map 0% reduce 0%
22/08/26 01:27:27 INFO mapred.JobClient: map 100% reduce 0%

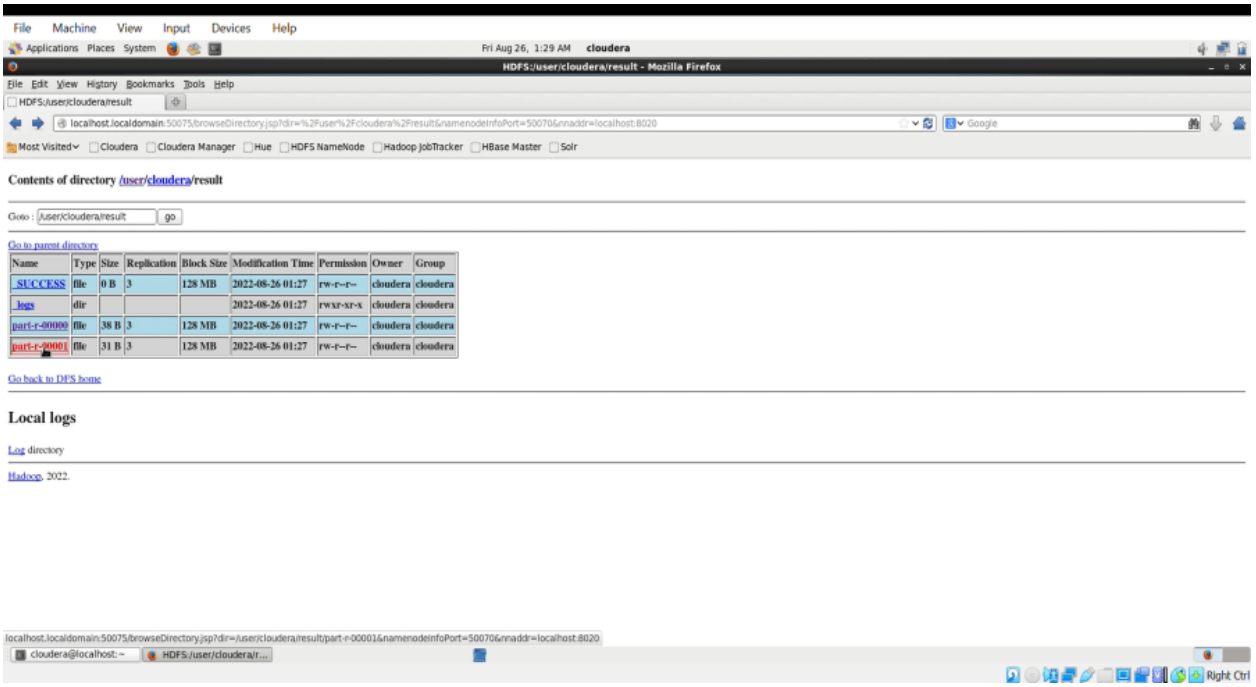

cloudera@localhost:~
File Edit View Search Terminal Help
22/08/26 01:27:41 INFO mapred.JobClient: Physical memory (bytes) snapshot=38
5183744
22/08/26 01:27:41 INFO mapred.JobClient: Virtual memory (bytes) snapshot=201
3032448
22/08/26 01:27:41 INFO mapred.JobClient: Total committed heap usage (bytes)=
232067072
[cloudera@localhost ~]$ hadoop fs -ls result
Found 4 items
-rw-r--r-- 3 cloudera cloudera 0 2022-08-26 01:27 result/_SUCCESS
drwxr-xr-x - cloudera cloudera 0 2022-08-26 01:27 result/_logs
-rw-r--r-- 3 cloudera cloudera 38 2022-08-26 01:27 result/part-r-00000
-rw-r--r-- 3 cloudera cloudera 31 2022-08-26 01:27 result/part-r-00001
[cloudera@localhost ~]$ hadoop fs -cat /user/cloudera/result/part-r-00000
a 1
ahead 1
everyone 1
good 1
great 1
[cloudera@localhost ~]$ hadoop fs -cat /user/cloudera/result/part-r-00001
day 1
have 1
hello 1
morning 1
[cloudera@localhost ~]$

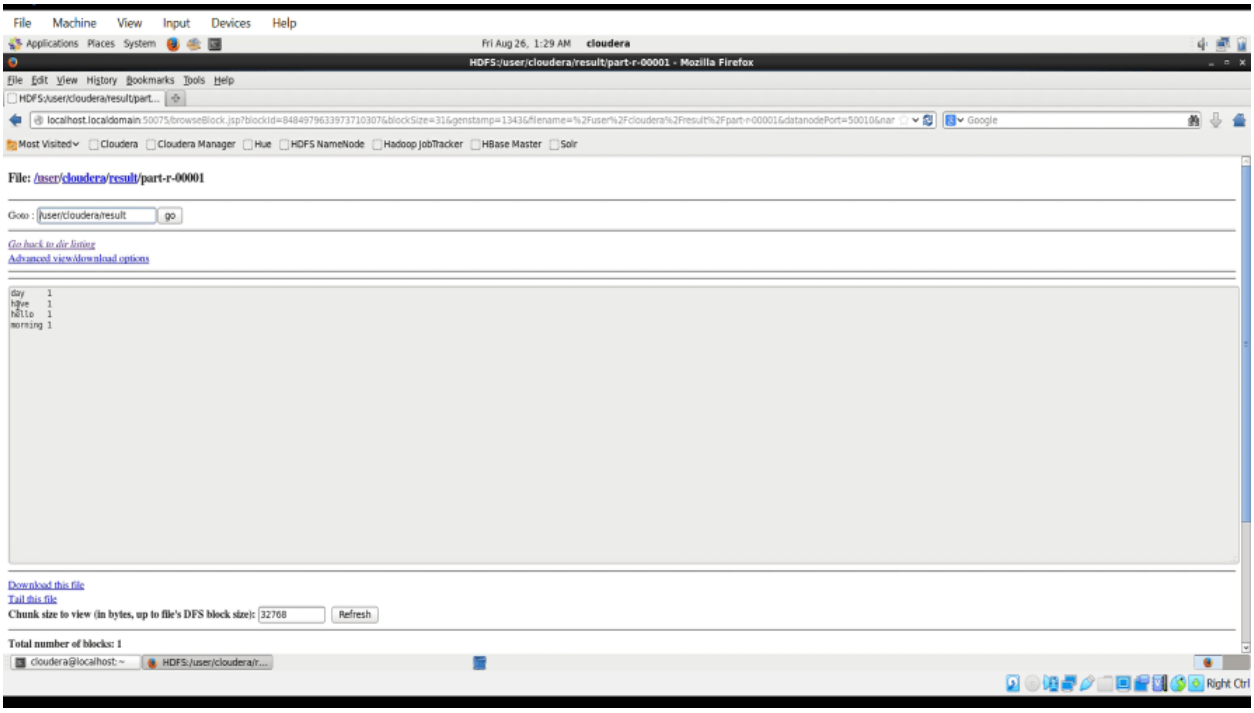
OUTPUT:


File Machine View Input Devices Help
Applications Places System
Fri Aug 26, 1:29 AM cloudera
HDFS:/ - Mozilla Firefox
File Edit View History Bookmarks Tools Help
HDFS:/
localhost.localdomain:50075/browseDirectory.jsp?namenodeInfoPort=50070&dir=/&nnaddr=localhost:8020
Most Visited Cloudera Cloudera Manager Hue HDFS NameNode Hadoop JobTracker HBase Master Solr
Contents of directory /
Goto : /user/cloudera go
Name Type Size Replication Block Size Modification Time Permission Owner Group
hbase dir 2014-06-01 16:16 rwxr-xr-x hbase hbase
solr dir 2014-06-01 16:16 rwxr-xr-x solr solr
tmp dir 2014-06-01 16:16 rwxrwxrwx hdfs supergroup
user dir 2014-06-01 16:17 rwxr-xr-x hdfs supergroup
var dir 2014-06-01 16:15 rwxr-xr-x hdfs supergroup
Go back to DFS home
Local logs
Log directory
Hadoop, 2022.
cloudera@localhost:~
HDFS:/ - Mozilla Firefox
Right Ctrl

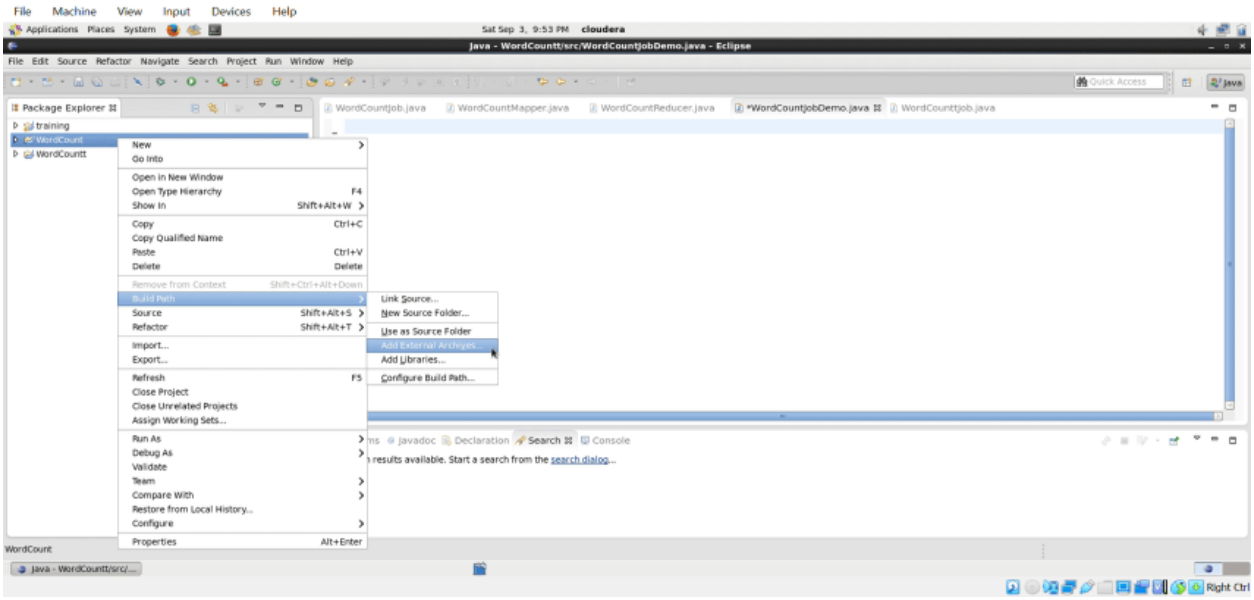

File Machine View Input Devices Help
Applications Places System
Fri Aug 26, 1:29 AM cloudera
HDFS:/user/cloudera - Mozilla Firefox
File Edit View History Bookmarks Tools Help
HDFS:/user/cloudera
localhost.localdomain:50075/browseDirectory.jsp?dir=%2Fuser%2Fcloudera&go=go&namenodeInfoPort=50070&nnaddr=localhost%3A8020
Most Visited Cloudera Cloudera Manager Hue HDFS NameNode Hadoop JobTracker HBase Master Solr
Contents of directory /user/cloudera
Goto : /user/cloudera go
Go to parent directory
Name Type Size Replication Block Size Modification Time Permission Owner Group
.Trash dir 2022-08-26 00:35 rwx------ cloudera cloudera
.staging dir 2022-08-26 01:27 rwx------ cloudera cloudera
result dir 2022-08-26 01:27 rwxr-xr-x cloudera cloudera
word.txt file 51 B 3 128 MB 2022-08-25 04:21 rw-r--r-- cloudera cloudera
Go back to DFS home
Local logs
Log directory
Hadoop, 2022.
cloudera@localhost:~
HDFS:/user/cloudera - ...


File Machine View Input Devices Help
Applications Places System
Fri Aug 26, 1:29 AM cloudera
HDFS:/user/cloudera/result/part-r-00000 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
HDFS:/user/cloudera/result/part...
localhost.localdomain:50075/browseBlock.jsp?blockId=4605802637990910821&blockSize=38&genstamp=1344&filename=%2Fuser%2Fcloudera%2Fresult%2Fpart-r-00000&datanodePort=50010&nar
Google
Most Visited Cloudera Cloudera Manager Hue HDFS NameNode Hadoop JobTracker HBase Master Solr
File: /user/cloudera/result/part-r-00000
Goto : /user/cloudera/result go
Go back to dir listing
Advanced view/download options
a 1
ahead 1
everyone 1
good 1
great 1
Download this file
Tail this file
Chunk size to view (in bytes, up to file's DFS block size): 32768 Refresh
Total number of blocks: 1
cloudera@localhost: ~
HDFS:/user/cloudera/r...
Right Ctrl

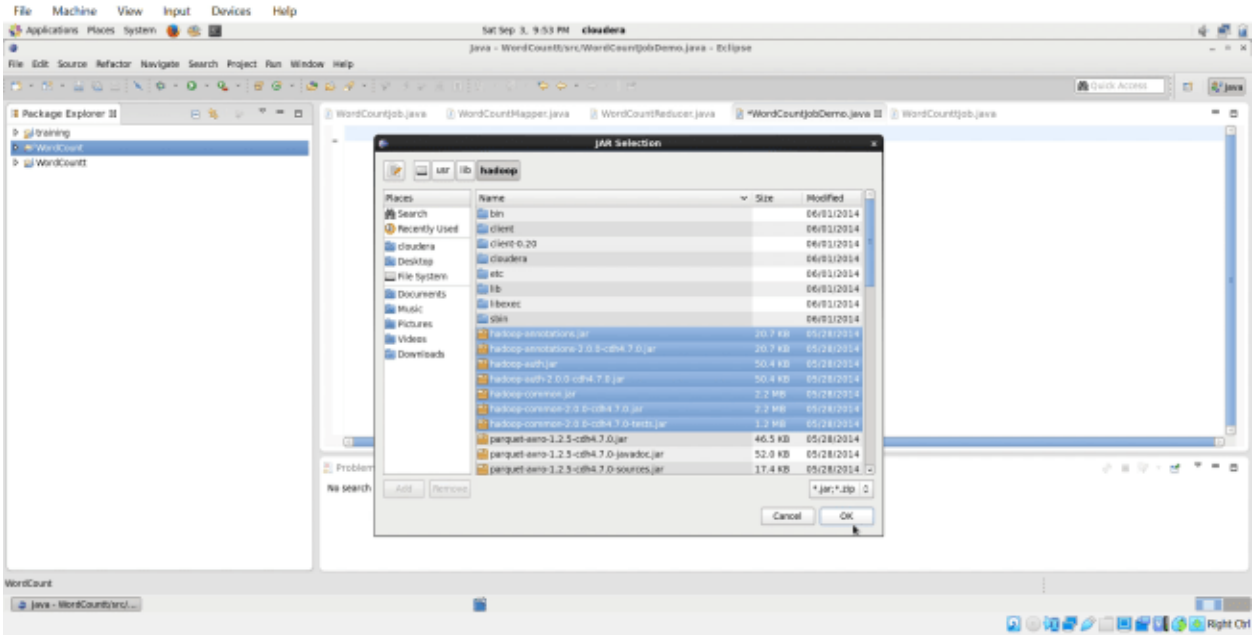

File Machine View Input Devices Help
Applications Places System
Fri Aug 26, 1:29 AM cloudera
HDFS:/user/cloudera/result - Mozilla Firefox
File Edit View History Bookmarks Tools Help
HDFS:/user/cloudera/result
localhost.localdomain:50075/browseDirectory.jsp?dir=%2Fuser%2Fcloudera%2Fresult&namenodeInfoPort=50070&nnaddr=localhost:8020
Google
Most Visited Cloudera Cloudera Manager Hue HDFS NameNode Hadoop JobTracker HBase Master Solr
Contents of directory /user/cloudera/result
Goto : /user/cloudera/result go
Go to parent directory
Name Type Size Replication Block Size Modification Time Permission Owner Group
_SUCCESS file 0 B 3 128 MB 2022-08-26 01:27 rw-r--r-- cloudera cloudera
_logs dir 2022-08-26 01:27 rwxr-xr-x cloudera cloudera
part-r-00000 file 38 B 3 128 MB 2022-08-26 01:27 rw-r--r-- cloudera cloudera
part-r-00001 file 31 B 3 128 MB 2022-08-26 01:27 rw-r--r-- cloudera cloudera
Go back to DFS home
Local logs
Log directory
Hadoop, 2022.
localhost.localdomain:50075/browseDirectory.jsp?dir=/user/cloudera/result/part-r-00001&namenodeInfoPort=50070&nnaddr=localhost:8020
cloudera@localhost: ~
HDFS:/user/cloudera/r...
Right Ctrl

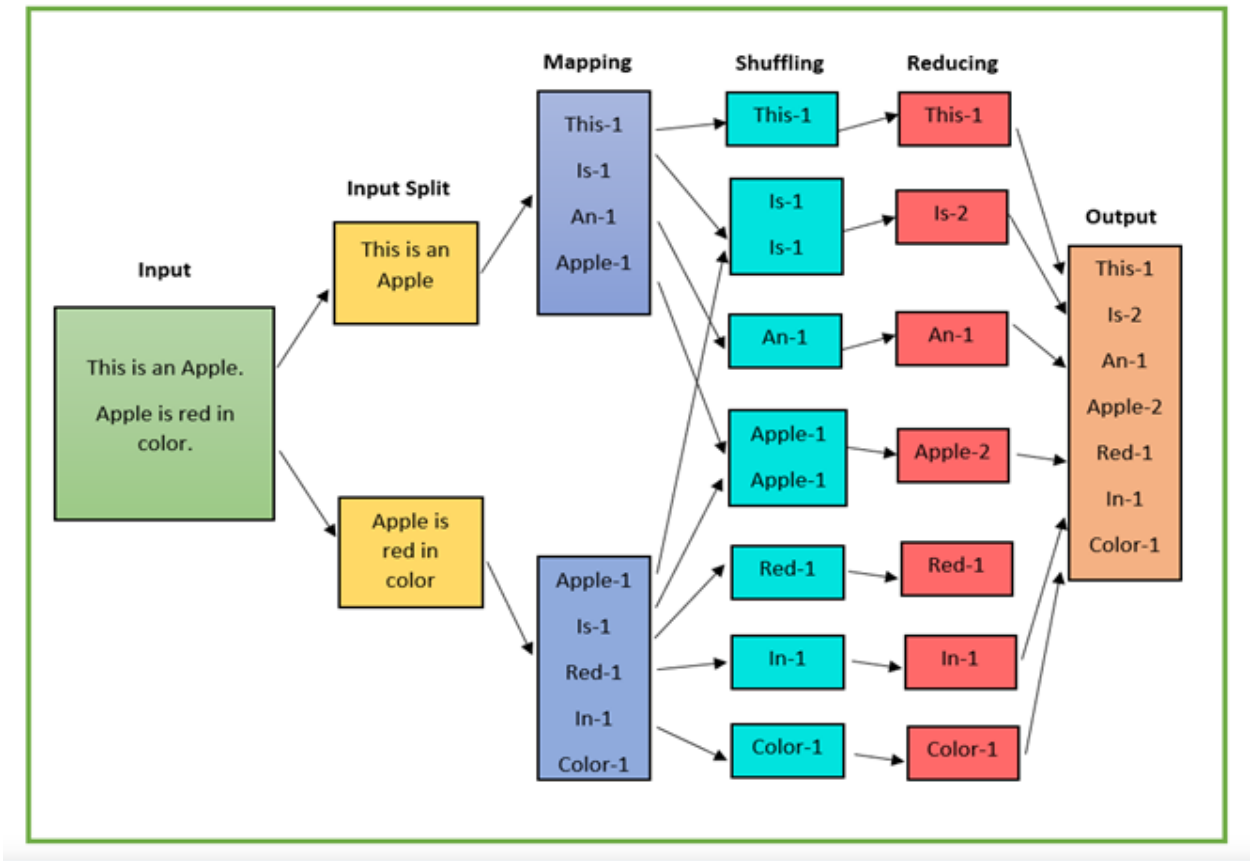


If there is any error in executing then set the path- right click on your project->build path->add external archives.



Add all the hadoop and mapreduce external jar files and click ok.





Experiment 4:

Aim: Write a MapReduce program that mines weather data. Weather sensors collecting data every hour at many locations across the globe gather a large volume of log data, which is a good candidate for analysis with MapReduce, since it is semi structured and record-oriented.

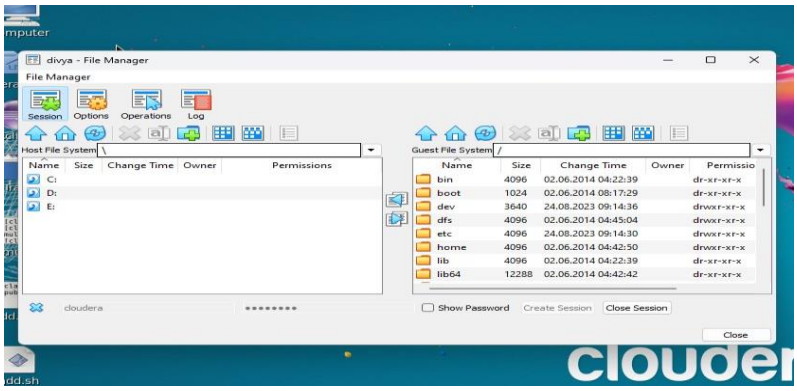
Programs:

Click Oracle VM box and click start button.

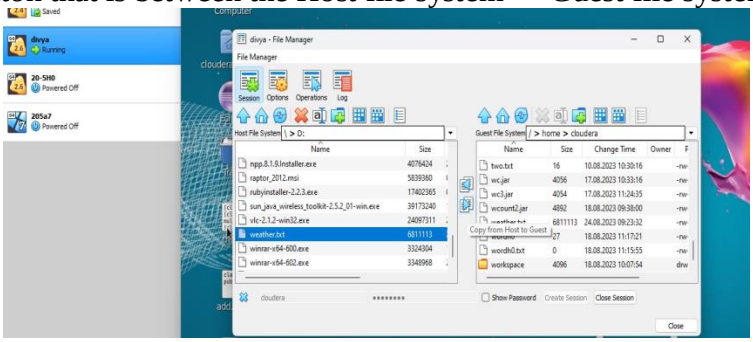
Transform Weather.txt from windows OS to hadoop file system.

Click Machine->File Manager

First create a session with cloudera as both username and password and open the drive where the dataset is placed

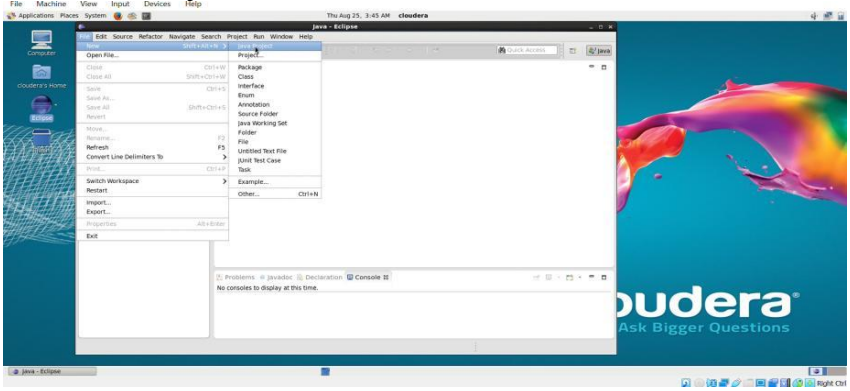


Click the arrow button that is between the Host file system -> Guest file system

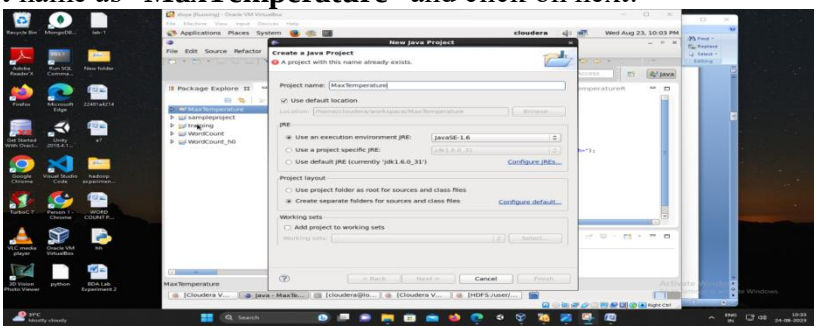


Open Oracle VM Virtual box -> click start -> open cloudera -> open eclipse.

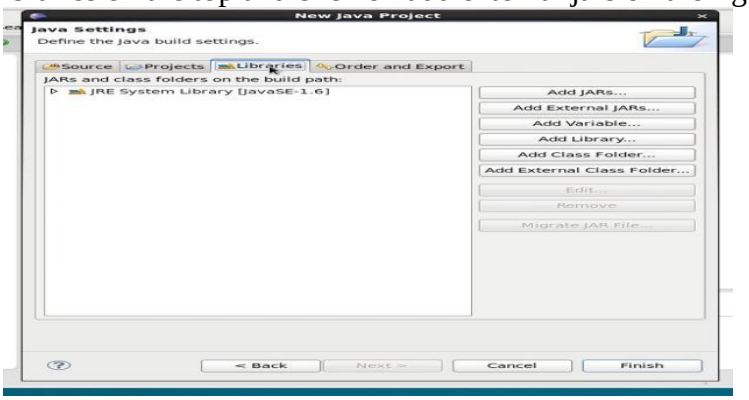
Click on file->New->Java Project



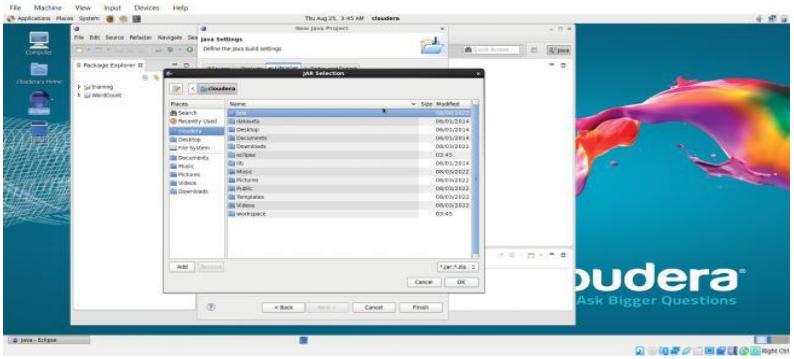
Give your project name as "**MaxTemperature**" and click on next.



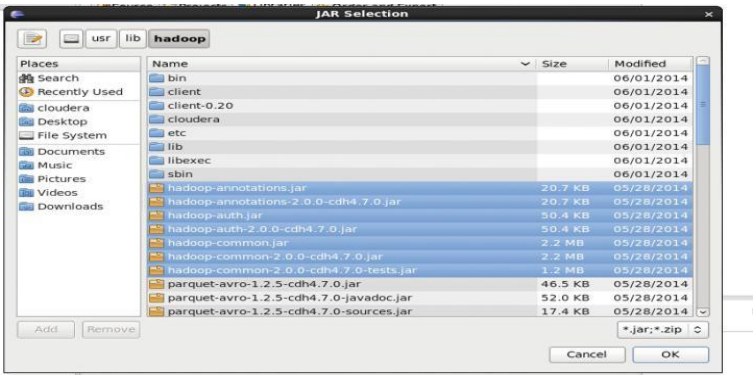
After that, click on libraries on the top and click on add external jars on the right side



A dialog box will appear.

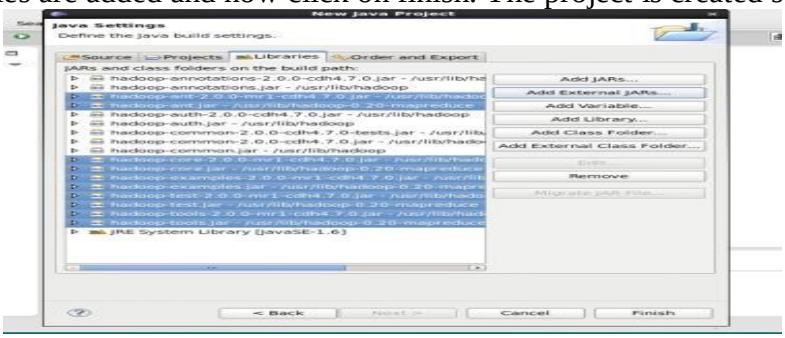


Click on file system on the left, and then usr->lib ->hadoop. Select all the files named with hadoop and click ok.

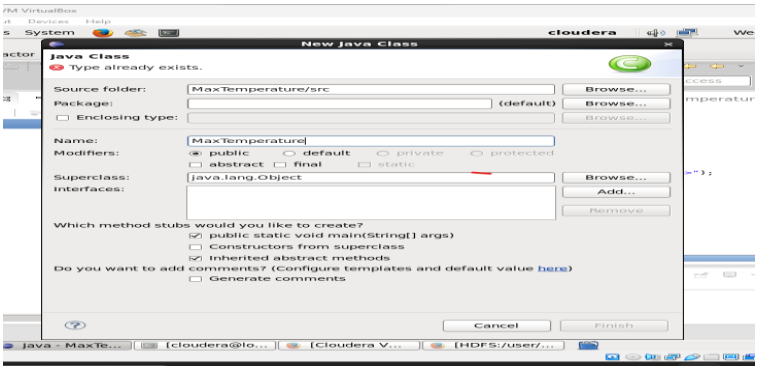


Go back to lib and click **hadoop-0.20-mapreduce** and select all the files named with hadoop and click ok.

All the external files are added and now click on finish. The project is created successfully.



Create a class with name "**MaxTemperature**"

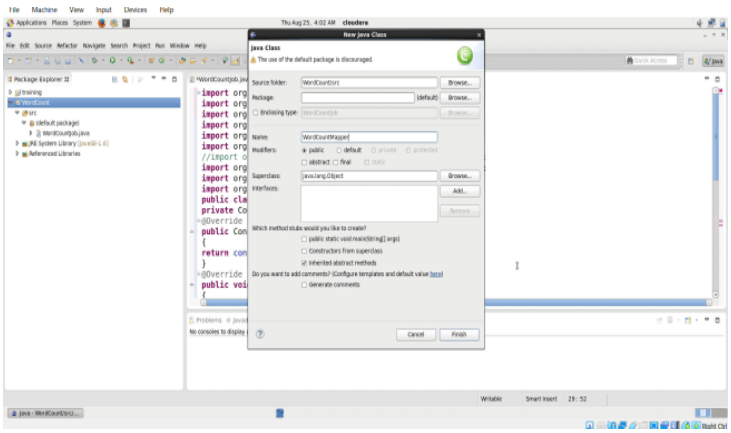


Copy and paste the program 4 lines at a time- If copy and paste doesn't work then goto device in cloudera and set both shared clipboard and drag and drop as **bidirectional.**

MaxTemperature:

```
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.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
//import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
public class MaxTemperature {
public static void main(String[] args) throws Exception {
if (args.length != 2) {
System.err.println("Usage: MaxTemperature <input path> <output path>");
System.exit(-1);
}
Job job = new Job();
job.setJarByClass(MaxTemperature.class);
job.setJobName("Max temperature");
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
job.setMapperClass(MaxTemperatureMapper.class);
job.setReducerClass(MaxTemperatureReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```

Create another class "**MaxTemperatureMapper**"

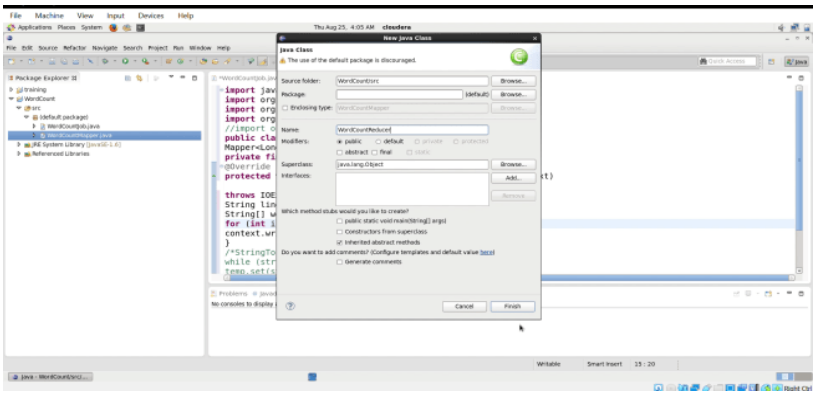


Copy and paste the program:

Max Tempeature MAPPER:

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class MaxTemperatureMapper
extends Mapper<LongWritable, Text, Text, IntWritable> {
private static final int MISSING = 9999;
@Override
public void map(LongWritable key, Text value, Context context)
throws IOException, InterruptedException {
String line = value.toString();
String year = line.substring(15, 19);
int airTemperature;
if (line.charAt(87) == '+') { // parseInt doesn't like leading plus signs
airTemperature = Integer.parseInt(line.substring(88, 92));
} else {
airTemperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);
if (airTemperature != MISSING && quality.matches("[01459]")) {
context.write(new Text(year), new IntWritable(airTemperature));
}
}
}
```

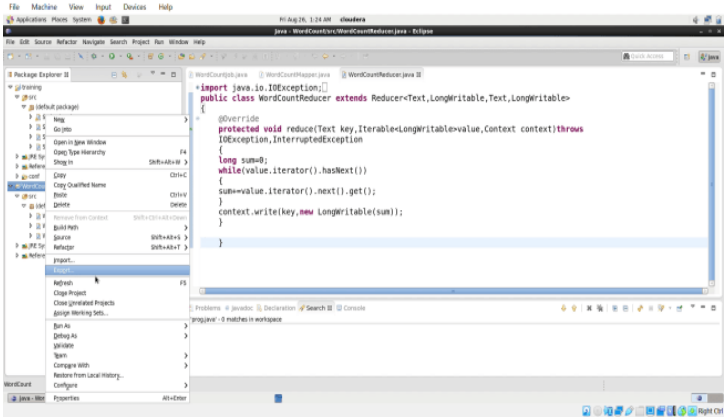
Create another class "**MaxTemperatureReducer**"



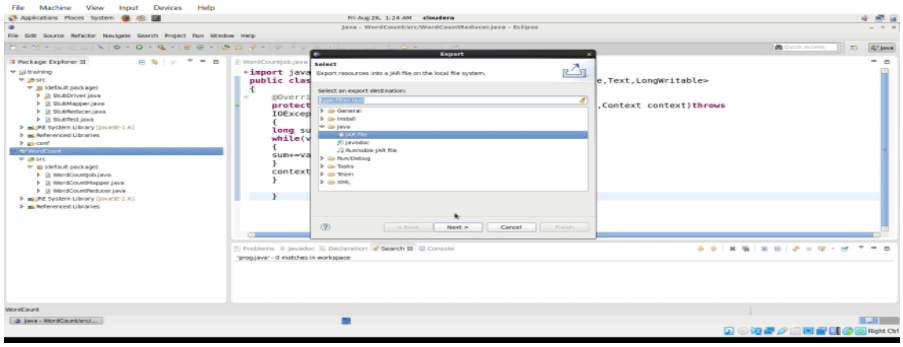
MAX TEMPERATURE REDUCER:

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class MaxTemperatureReducer
extends Reducer<Text, IntWritable, Text, IntWritable> {
@Override
public void reduce(Text key, Iterable<IntWritable> values,
Context context)
throws IOException, InterruptedException {
int maxValue = Integer.MIN_VALUE;
for (IntWritable value : values) {
maxValue = Math.max(maxValue, value.get());
}
context.write(key, new IntWritable(maxValue));
}
}
```

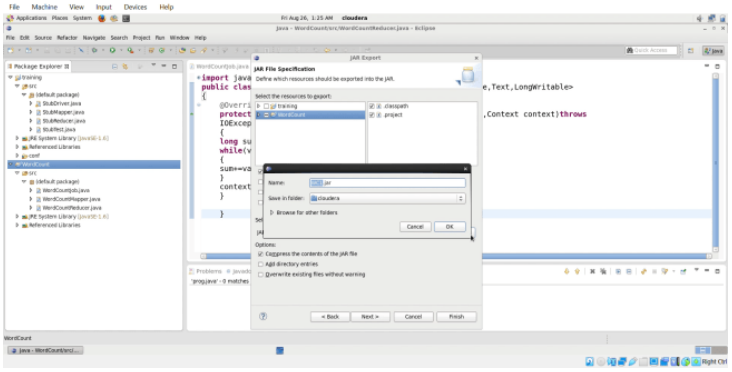
After finishing all the programs, right click on your project and select export.



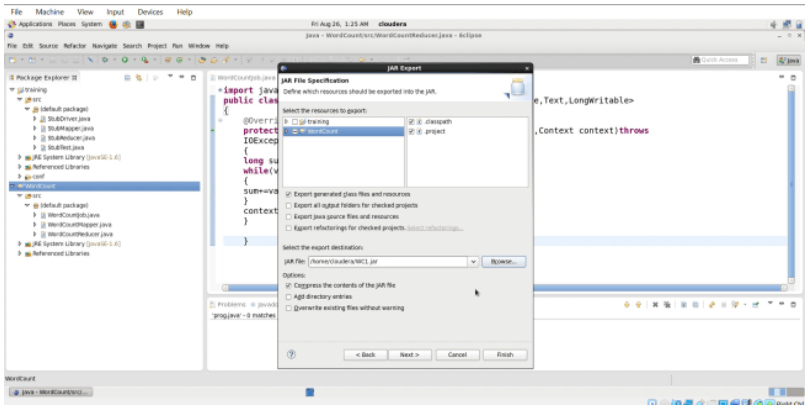
Select java-> JAR file->next



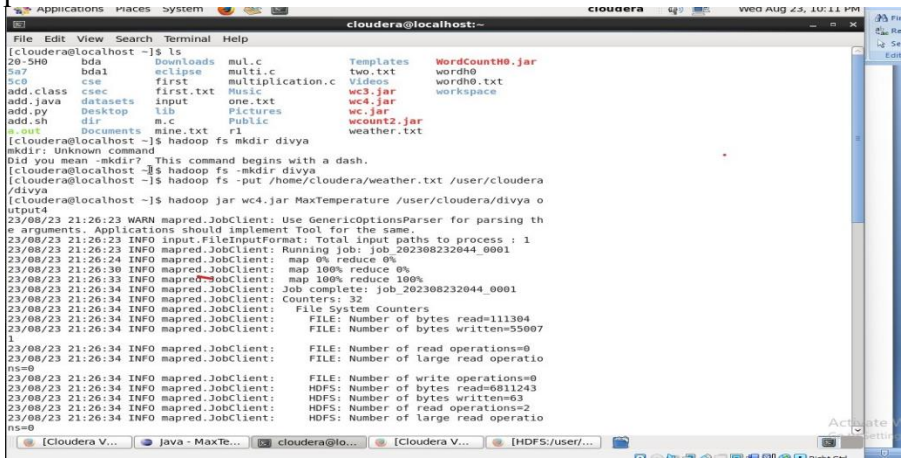
Click on browse and a dialog box will appear, give any name for the jar file and click ok.

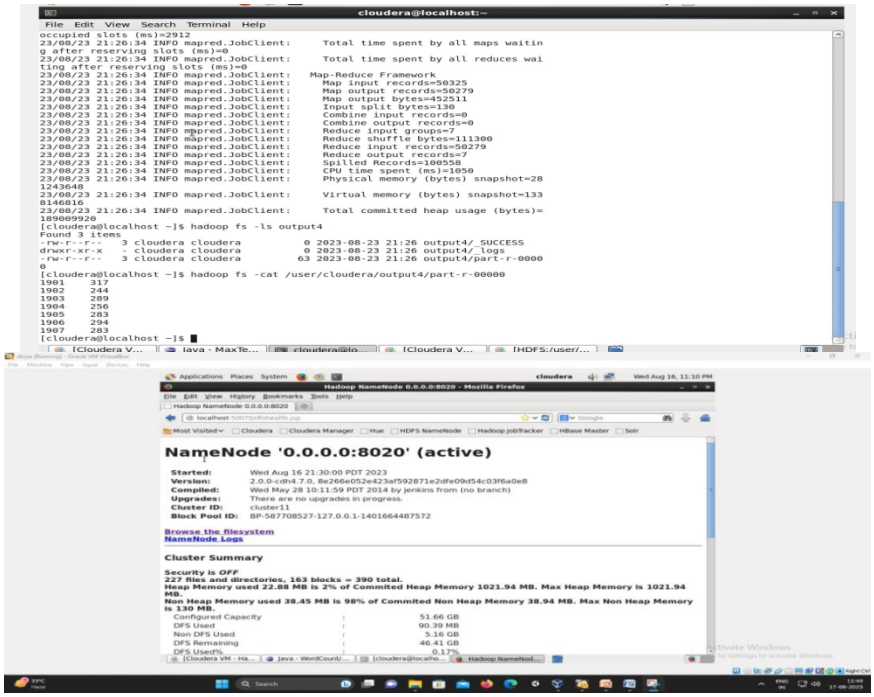


Click on finish.

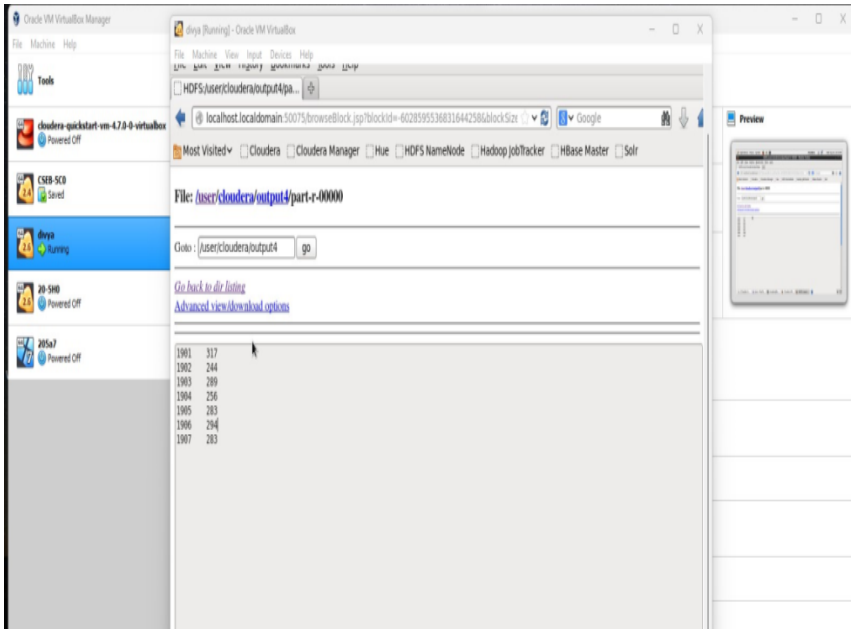


Now, open the terminal and create a text file “word”.




cloudera@localhost:~
File Edit View Search Terminal Help
occupied slots (ms)=2912
23/08/23 21:26:34 INFO mapred.JobClient: Total time spent by all maps waitin
g after reserving slots (ms)=0
23/08/23 21:26:34 INFO mapred.JobClient: Total time spent by all reduces wai
ting after reserving slots (ms)=0
23/08/23 21:26:34 INFO mapred.JobClient: Map-Reduce Framework
23/08/23 21:26:34 INFO mapred.JobClient: Map input records=50325
23/08/23 21:26:34 INFO mapred.JobClient: Map output records=50279
23/08/23 21:26:34 INFO mapred.JobClient: Map output bytes=452511
23/08/23 21:26:34 INFO mapred.JobClient: Input split bytes=130
23/08/23 21:26:34 INFO mapred.JobClient: Combine input records=0
23/08/23 21:26:34 INFO mapred.JobClient: Combine output records=0
23/08/23 21:26:34 INFO mapred.JobClient: Reduce input groups=7
23/08/23 21:26:34 INFO mapred.JobClient: Reduce shuffle bytes=111300
23/08/23 21:26:34 INFO mapred.JobClient: Reduce input records=50279
23/08/23 21:26:34 INFO mapred.JobClient: Reduce output records=7
23/08/23 21:26:34 INFO mapred.JobClient: Spilled Records=100558
23/08/23 21:26:34 INFO mapred.JobClient: CPU time spent (ms)=1850
23/08/23 21:26:34 INFO mapred.JobClient: Physical memory (bytes) snapshot=28
1243648
23/08/23 21:26:34 INFO mapred.JobClient: Virtual memory (bytes) snapshot=133
8146816
23/08/23 21:26:34 INFO mapred.JobClient: Total committed heap usage (bytes)=
189009920
[cloudera@localhost ~]$ hadoop fs -ls output4
Found 3 items
-rw-r--r-- 3 cloudera cloudera 0 2023-08-23 21:26 output4/_SUCCESS
drwxr-xr-x - cloudera cloudera 0 2023-08-23 21:26 output4/_logs
-rw-r--r-- 3 cloudera cloudera 63 2023-08-23 21:26 output4/part-r-0000
0
[cloudera@localhost ~]$ hadoop fs -cat /user/cloudera/output4/part-r-00000
1901 317
1902 244
1903 289
1904 256
1905 283
1906 294
1907 283
[cloudera@localhost ~]$
Hadoop NameNode 0.0.0.0:8020 - Mozilla Firefox
NameNode '0.0.0.0:8020' (active)
Started: Wed Aug 16 21:30:00 PDT 2023
Version: 2.0.0-cdh4.7.0, 8e266e052e423af592871e2dfe09d54c03f6a0e8
Compiled: Wed May 28 10:11:59 PDT 2014 by jenkins from (no branch)
Upgrades: There are no upgrades in progress.
Cluster ID: cluster11
Block Pool ID: BP-587708527-127.0.0.1-1401664487572
Browse the filesystem
NameNode Logs
Cluster Summary
Security is OFF
227 files and directories, 163 blocks = 390 total.
Heap Memory used 22.88 MB is 2% of Commited Heap Memory 1021.94 MB. Max Heap Memory is 1021.94 MB.
Non Heap Memory used 38.45 MB is 98% of Commited Non Heap Memory 38.94 MB. Max Non Heap Memory is 130 MB.
Configured Capacity : 51.66 GB
DFS Used : 90.39 MB
Non DFS Used : 5.16 GB
DFS Remaining : 46.41 GB
DFS Used% : 0.17%

Output:


Oracle VM VirtualBox Manager
Tools
cloudera-quickstart-vm-4.7.0-0-virtualbox
Powered Off
CSEB-SCO
Saved
divya
Running
20-SHO
Powered Off
205a7
Powered Off
divya [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
HDFS://user/cloudera/output4/pa...
localhost.localdomain:50075/browseBlock.jsp?blockId=-6028595536831644258&blockSize
Most Visited Cloudera Cloudera Manager Hue HDFS NameNode Hadoop JobTracker HBase Master Solr
Preview
File: /user/cloudera/output4/part-r-00000
Goto : /user/cloudera/output4
go
Go back to dir listing
Advanced view/download options
1901 317
1902 244
1903 289
1904 256
1905 283
1906 294
1907 283

Experiment No:5

Aim: Implement Matrix Multiplication with Hadoop Map Reduce.

Programs:

Click Oracle VM box and click start button.

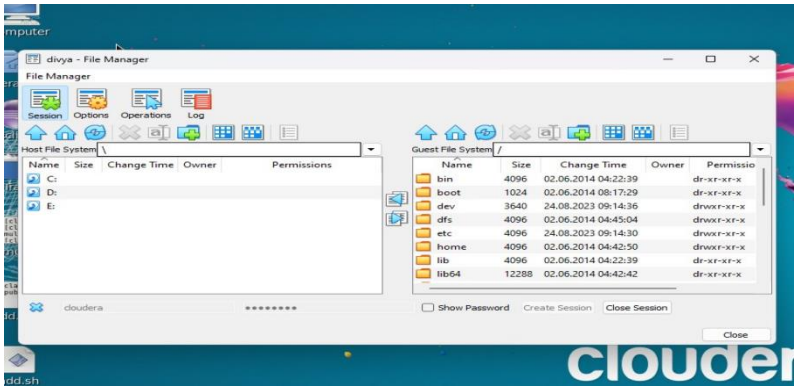
Transform mmdata.txt from windows OS to hadoop file system.

mmdata.txt:

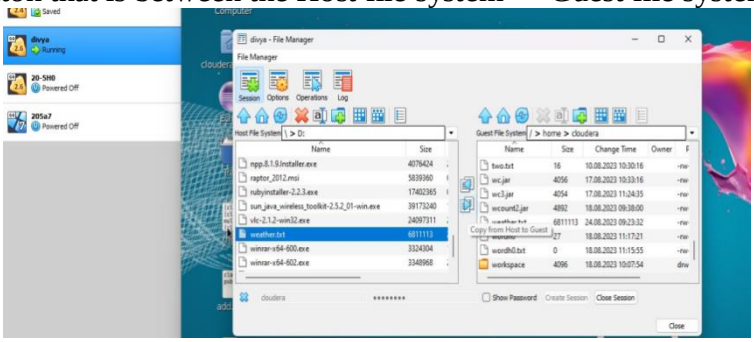
A,0,0,4
A,0,1,5
A,0,2,6
A,0,3,7
A,0,4,8
A,1,0,1
A,1,1,3
A,1,2,4
A,1,3,5
A,1,4,6
B,0,0,6
B,0,1,3
B,0,2,4
B,1,0,1
B,1,1,2
B,1,2,7
B,2,0,8
B,2,1,6
B,2,2,5
B,3,0,2
B,3,1,3
B,3,2,1
B,4,0,7
B,4,1,4
B,4,2,2

Click Machine->File Manager

First create a session with cloudera as both username and password and open the drive where the dataset is placed

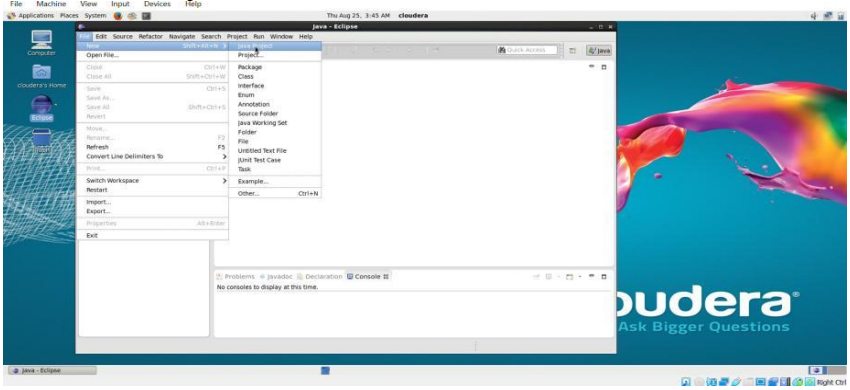


Click the arrow button that is between the Host file system -> Guest file system

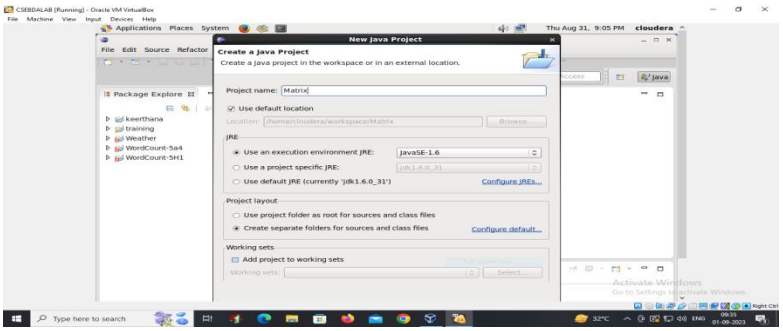


Open Oracle VM Virtual box -> click start -> open cloudera -> open eclipse.

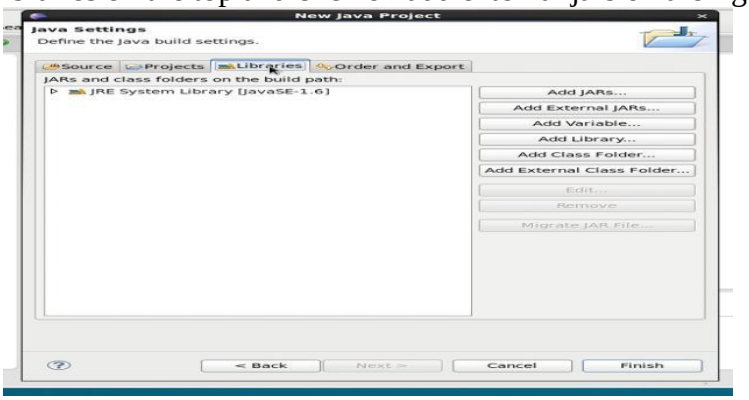
Click on file->New->Java Project



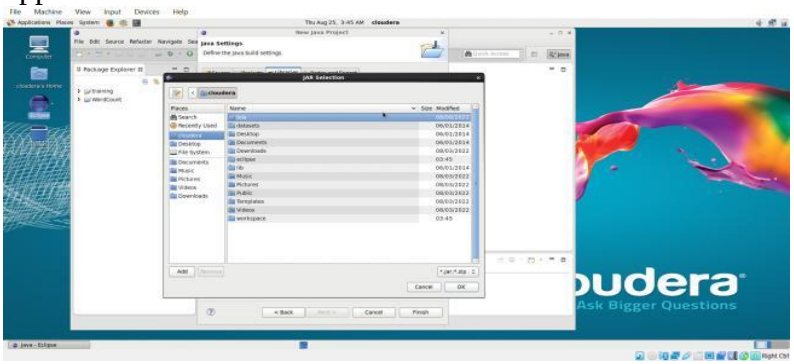
Give your project name as "**Matrix**" and click on next.



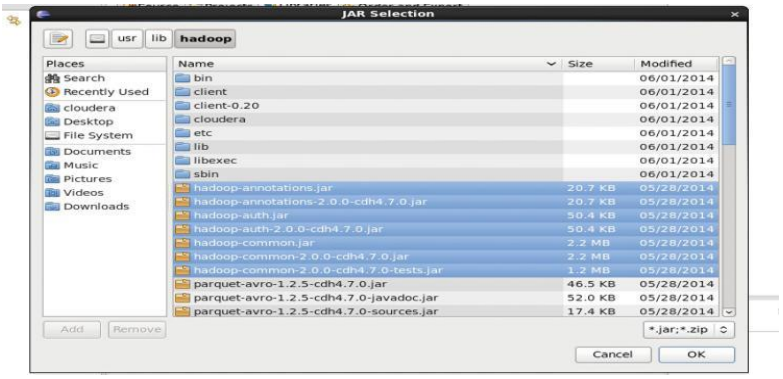
After that, click on libraries on the top and click on add external jars on the right side



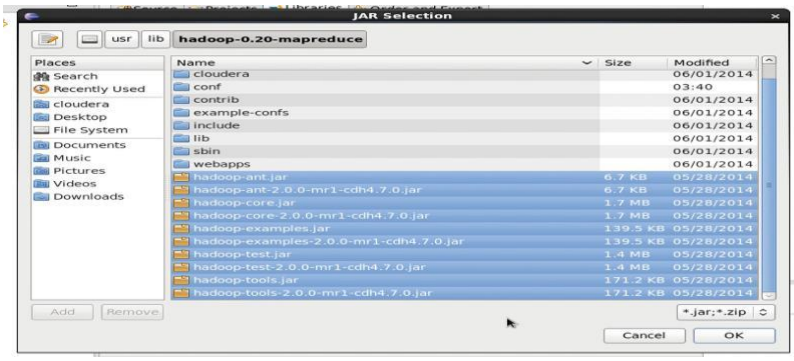
A dialog box will appear.



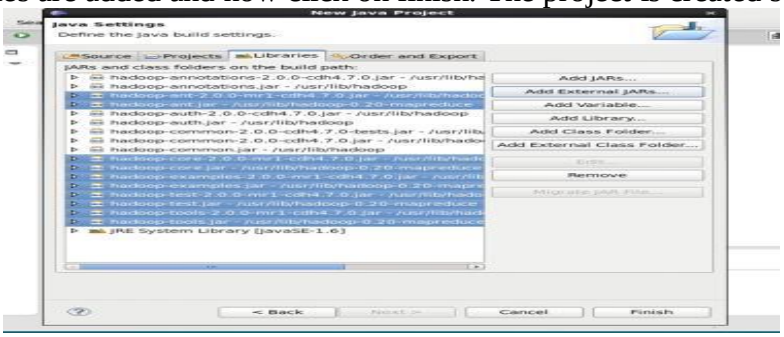
Click on file system on the left, and then usr->lib ->hadoop. Select all the files named with hadoop and click ok.



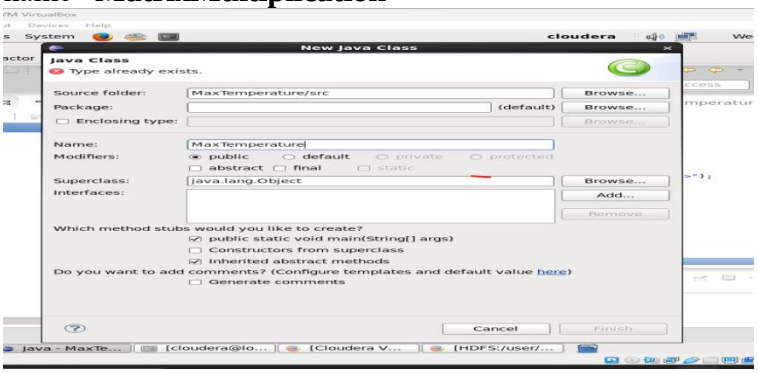
Go back to lib and click **hadoop-0.20-mapreduce** and select all the files named with hadoop and click ok.



All the external files are added and now click on finish. The project is created successfully.



Create a class with name "**MatrixMultiplication**"



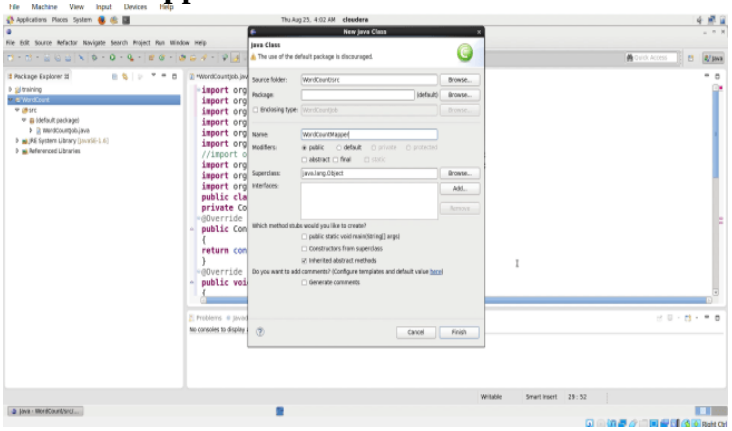
Copy and paste the program 4 lines at a time- If copy and paste doesn't work then goto device in cloudera and set both shared clipboard and drag and drop as **bidirectional.**

MatrixMultiplication:

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
public class MatrixMultiplication {
public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
// A is an m-by-n matrix; B is an n-by-p matrix.
conf.set("m", "2");
```

```
conf.set("n", "5");
conf.set("p", "3");
Job job = new Job(conf, "MatrixMultiplication");
job.setJarByClass(MatrixMultiplication.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(Text.class);
job.setMapperClass(MatrixMapper.class);
job.setReducerClass(MatrixReducer.class);
job.setInputFormatClass(TextInputFormat.class);
job.setOutputFormatClass(TextOutputFormat.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
job.waitForCompletion(true);
}
}
```

Create another class "**MatrixMapper**"



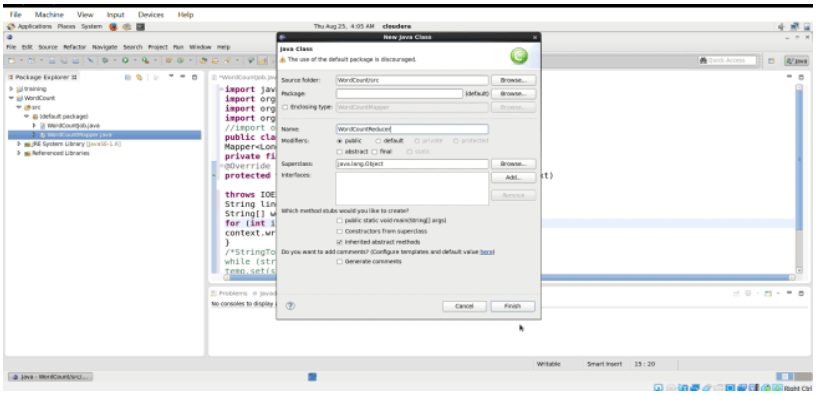
Copy and paste the program:

MatrixMapper:

```
import java.io.IOException;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class MatrixMapper extends Mapper<LongWritable, Text,
Text, Text> {
public void map(LongWritable key, Text value, Context
context) throws IOException, InterruptedException {
Configuration conf = context.getConfiguration();
int m = Integer.parseInt(conf.get("m"));
int p = Integer.parseInt(conf.get("p"));
String line = value.toString();
String[] indicesAndValue = line.split(",");
Text outputKey = new Text();
Text outputValue = new Text();
```

```
if (indicesAndValue[0].equals("A")) {
for (int k = 0; k < p; k++) {
outputKey.set(indicesAndValue[1] + "," + k);
outputValue.set("A," + indicesAndValue[2] + "," +
indicesAndValue[3]);
context.write(outputKey, outputValue);
}
} else {
for (int i = 0; i < m; i++) {
outputKey.set(i + "," + indicesAndValue[2]);
outputValue.set("B," + indicesAndValue[1] + "," +
indicesAndValue[3]);
context.write(outputKey, outputValue);
}
}
}
```

}Create another class "**MatrixReducer**"

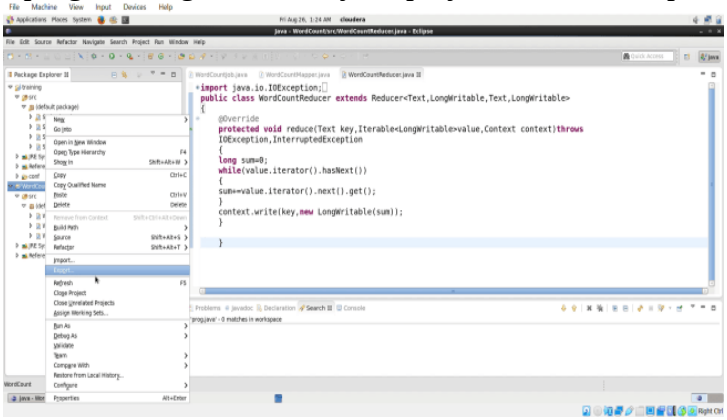


MatrixReducer:

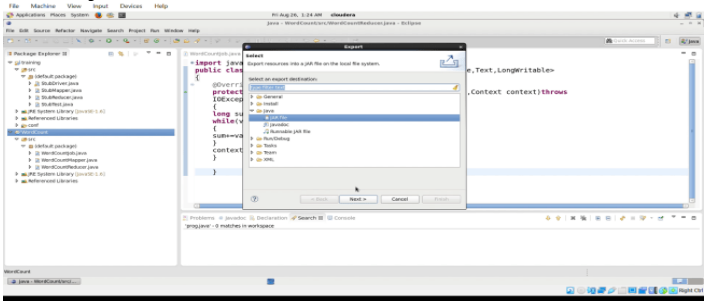
```
import java.io.IOException;
import java.util.HashMap;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class MatrixReducer extends Reducer<Text, Text, Text, Text> {
public void reduce(Text key, Iterable<Text> values, Context
context) throws IOException, InterruptedException {
String[] value;
HashMap<Integer, Float> hashA = new HashMap<Integer,
Float>();
HashMap<Integer, Float> hashB = new HashMap<Integer,
Float>();
for (Text val : values) {
value = val.toString().split(",");
if (value[0].equals("A")) {
hashA.put(Integer.parseInt(value[1]),
Float.parseFloat(value[2]));
```

```
} else {
hashB.put(Integer.parseInt(value[1]),
Float.parseFloat(value[2]));
}
}
int n =
Integer.parseInt(context.getConfiguration().get("n"));
float result = 0.0f;
float a_ij;
float b_jk;
for (int j = 0; j < n; j++) {
a_ij = hashA.containsKey(j) ? hashA.get(j) : 0.0f;
b_jk = hashB.containsKey(j) ? hashB.get(j) : 0.0f;
result += a_ij * b_jk;
}
if (result != 0.0f) {
context.write(null, new Text(key.toString() + "," +
Float.toString(result)));
}
}
}
```

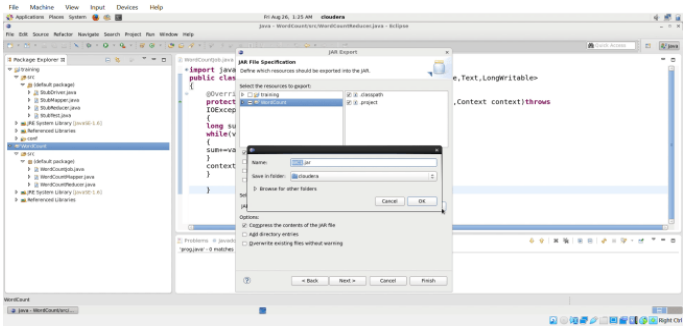
After finishing all the programs, right click on your project and select export.



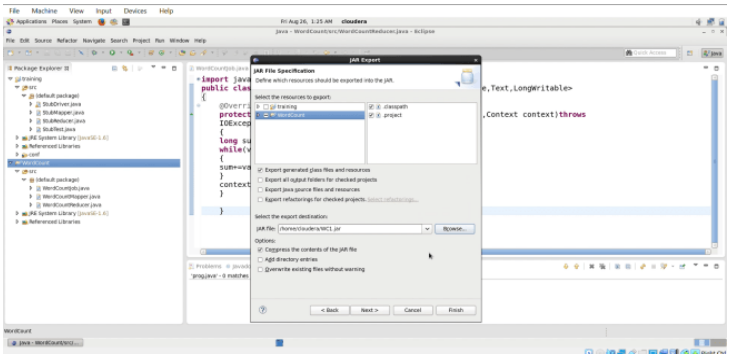
Select java-> JAR file->next



Click on browse and a dialog box will appear, give any name for the jar file and click ok.



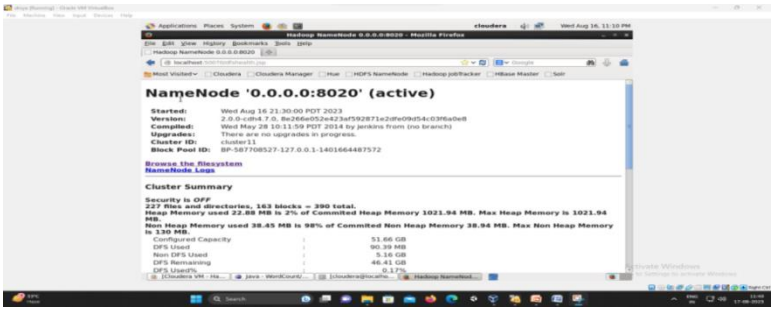
Click on finish.



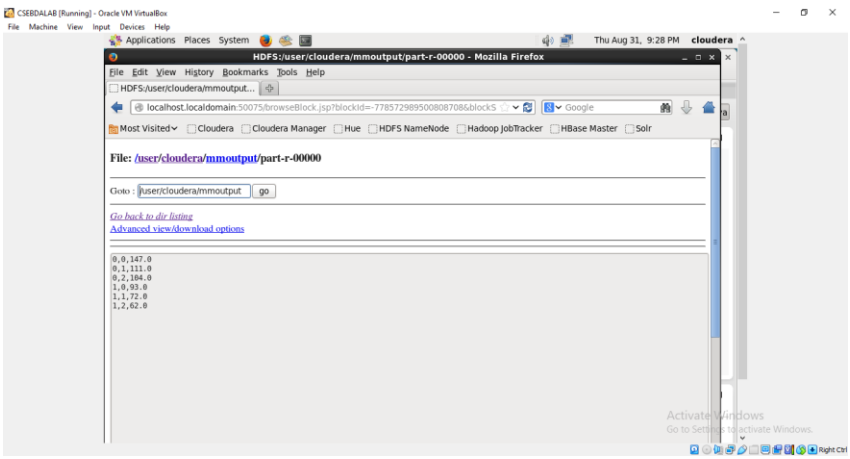
Now, open the terminal and create a directory file “mm”.

```
cloudera@localhost:~
File  Edit  View  Search  Terminal  Help
[cloudera@localhost ~]$ ls
20_5H1.jar  Desktop     exp3    lib          Pictures   WC.jar          workspace
CSEBDALAB   Documents   h1.txt  Matrix.jar   Public     Weather.jar
~CSEBDALAB  Downloads   h2.txt  mmdata.txt   Templates  weather.txt
datasets    eclipse     jhan    Music        Videos     WordPadma.txt
[cloudera@localhost ~]$ hadoop fs mkdir mm
mkdir: Unknown command
Did you mean -mkdir?  This command begins with a dash.
[cloudera@localhost ~]$ hadoop fs -mkdir mm
[cloudera@localhost ~]$ hadoop fs -put home/cloudera/mmdata.txt user/cloudera/mm
put: `home/cloudera/mmdata.txt': No such file or directory
[cloudera@localhost ~]$ hadoop fs -put /home/cloudera/mmdata.txt /user/cloudera/
mm
```

```
cloudera@localhost:~
File  Edit  View  Search  Terminal  Help
23/08/31 21:27:00 INFO mapred.JobClient:     Total time spent by all maps waitin
g after reserving slots (ms)=0
23/08/31 21:27:00 INFO mapred.JobClient:     Total time spent by all reduces wai
ting after reserving slots (ms)=0
23/08/31 21:27:00 INFO mapred.JobClient:   Map-Reduce Framework
23/08/31 21:27:00 INFO mapred.JobClient:     Map input records=25
23/08/31 21:27:00 INFO mapred.JobClient:     Map output records=60
23/08/31 21:27:00 INFO mapred.JobClient:     Map output bytes=600
23/08/31 21:27:00 INFO mapred.JobClient:     Input split bytes=126
23/08/31 21:27:00 INFO mapred.JobClient:     Combine input records=0
23/08/31 21:27:00 INFO mapred.JobClient:     Combine output records=0
23/08/31 21:27:00 INFO mapred.JobClient:     Reduce input groups=6
23/08/31 21:27:00 INFO mapred.JobClient:     Reduce shuffle bytes=341
23/08/31 21:27:00 INFO mapred.JobClient:     Reduce input records=60
23/08/31 21:27:00 INFO mapred.JobClient:     Reduce output records=6
23/08/31 21:27:00 INFO mapred.JobClient:     Spilled Records=120
23/08/31 21:27:00 INFO mapred.JobClient:     CPU time spent (ms)=670
23/08/31 21:27:00 INFO mapred.JobClient:     Physical memory (bytes) snapshot=29
1561472
23/08/31 21:27:00 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=133
8150912
23/08/31 21:27:00 INFO mapred.JobClient:     Total committed heap usage (bytes)=
171315200
[cloudera@localhost ~]$
```


NameNode '0.0.0.0:8020' (active)
Started: Wed Aug 16 21:30:00 PDT 2023
Version: 2.0.0-cdh4.7.0, 8e266e052e423af592871e2dfe09d54c03f6a0e8
Compiled: Wed May 28 10:11:59 PDT 2014 by jenkins from (no branch)
Upgrades: There are no upgrades in progress.
Cluster ID: cluster11
Block Pool ID: BP-587708527-127.0.0.1-1401664487572
Browse the filesystem
NameNode Logs
Cluster Summary
Security is OFF
227 files and directories, 163 blocks = 390 total.
Heap Memory used 22.88 MB is 2% of Commited Heap Memory 1021.94 MB. Max Heap Memory is 1021.94 MB.
Non Heap Memory used 38.45 MB is 98% of Commited Non Heap Memory 38.94 MB. Max Non Heap Memory is 130 MB.
Configured Capacity : 51.66 GB
DFS Used : 90.39 MB
Non DFS Used : 5.16 GB
DFS Remaining : 46.41 GB
DFS Used% : 0.17%

Output:

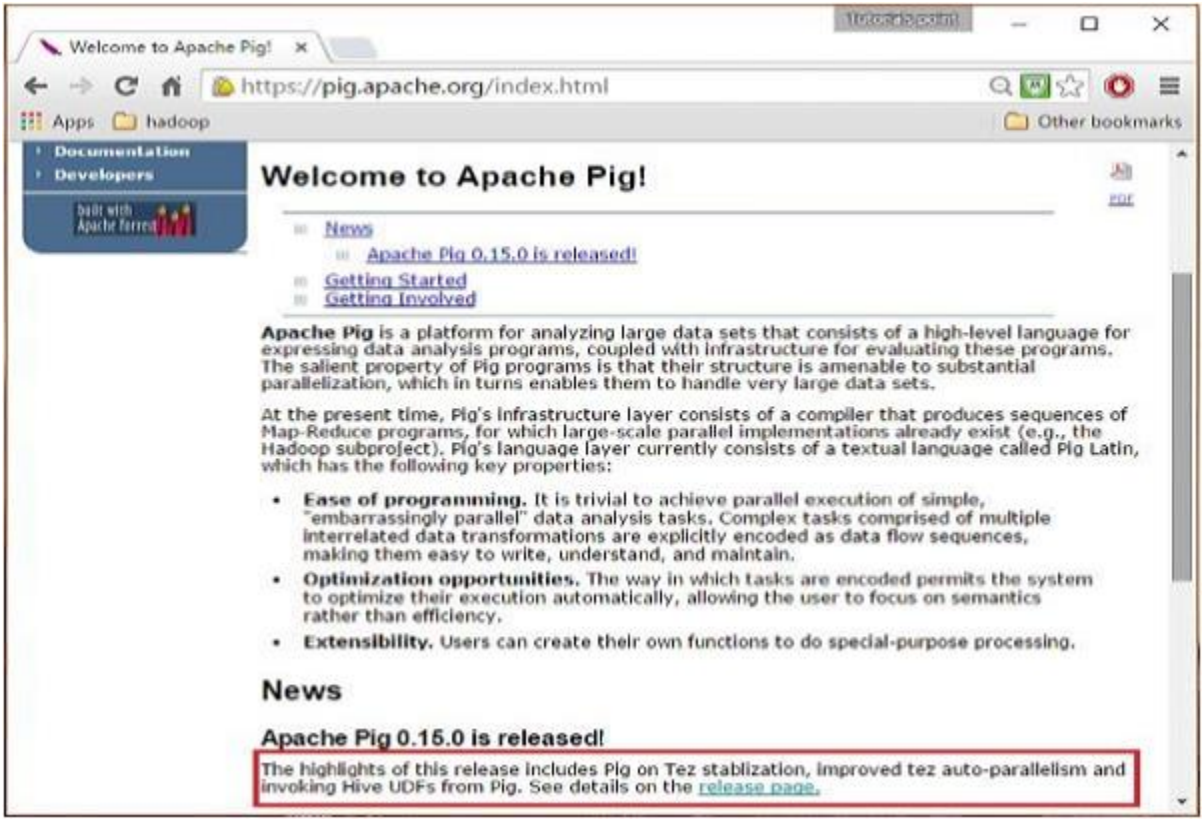

CSERDALAB [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Applications Places System
Thu Aug 31, 9:28 PM cloudera
HDFS:/user/cloudera/mmoutput/part-r-00000 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
HDFS:/user/cloudera/mmoutput...
localhost.localdomain:50075/browseBlock.jsp?blockId=-7785729895008087086blockS
Google
Most Visited Cloudera Cloudera Manager Hue HDFS NameNode Hadoop JobTracker HBase Master Solr
File: /user/cloudera/mmoutput/part-r-00000
Goto : /user/cloudera/mmoutput go
Go back to dir listing
Advanced view/download options
0,0,147.0
0,1,111.0
0,2,104.0
1,0,93.0
1,1,72.0
1,2,62.0
Activate Windows
Go to Settings to activate Windows.
Right Ctrl

Installation of PIG:
Download Apache Pig

First of all, download the latest version of Apache Pig from the following website https://pig.apache.org/.

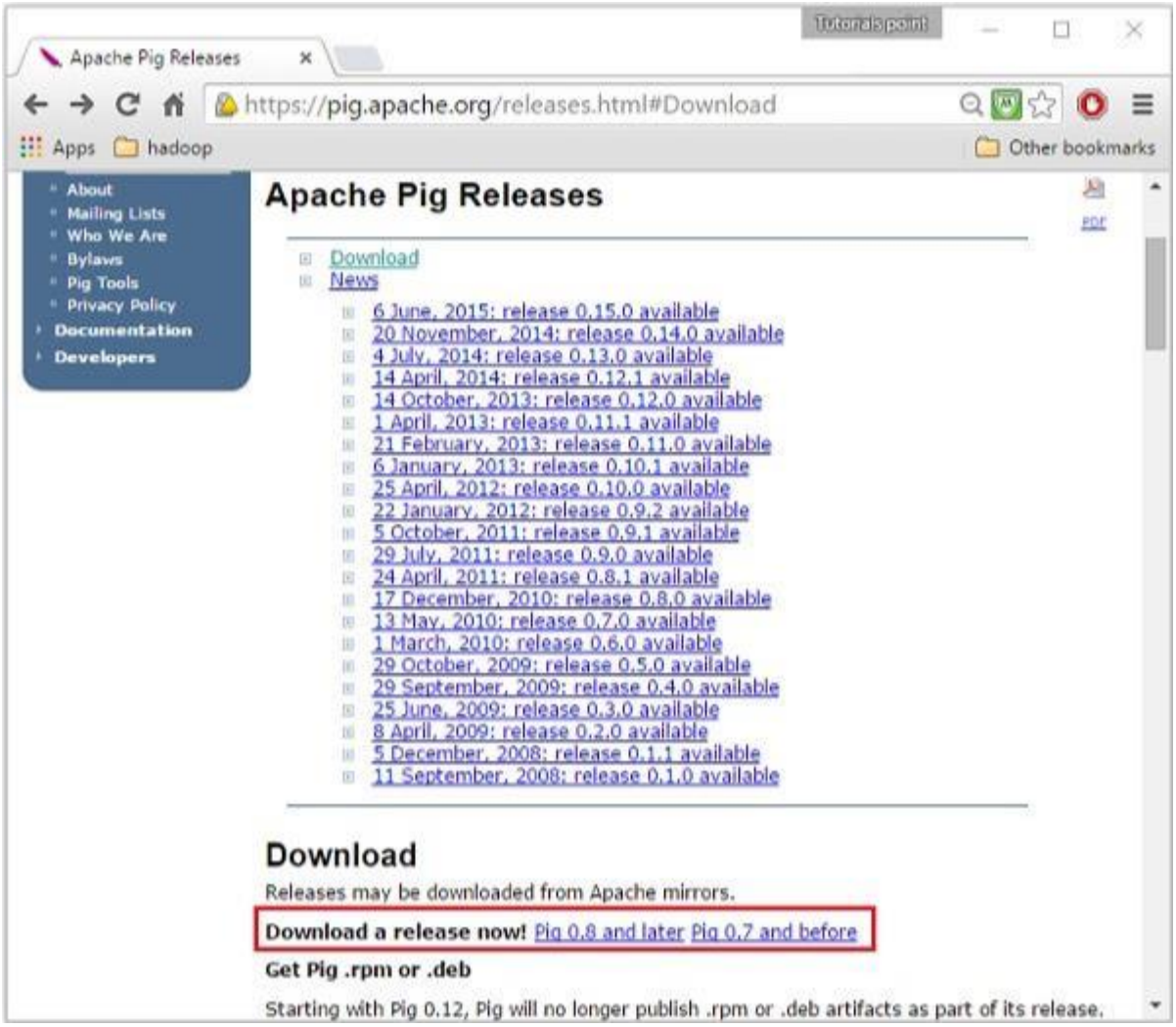
Step 1

Open the homepage of Apache Pig website. Under the section **News,** click on the link **release page** as shown in the following snapshot.



Step 2

On clicking the specified link, you will be redirected to the **Apache Pig Releases** page. On this page, under the **Download** section, you will have two links, namely, **Pig 0.8 and later** and **Pig 0.7 and before**. Click on the link **Pig 0.8 and later**, then you will be redirected to the page having a set of mirrors.

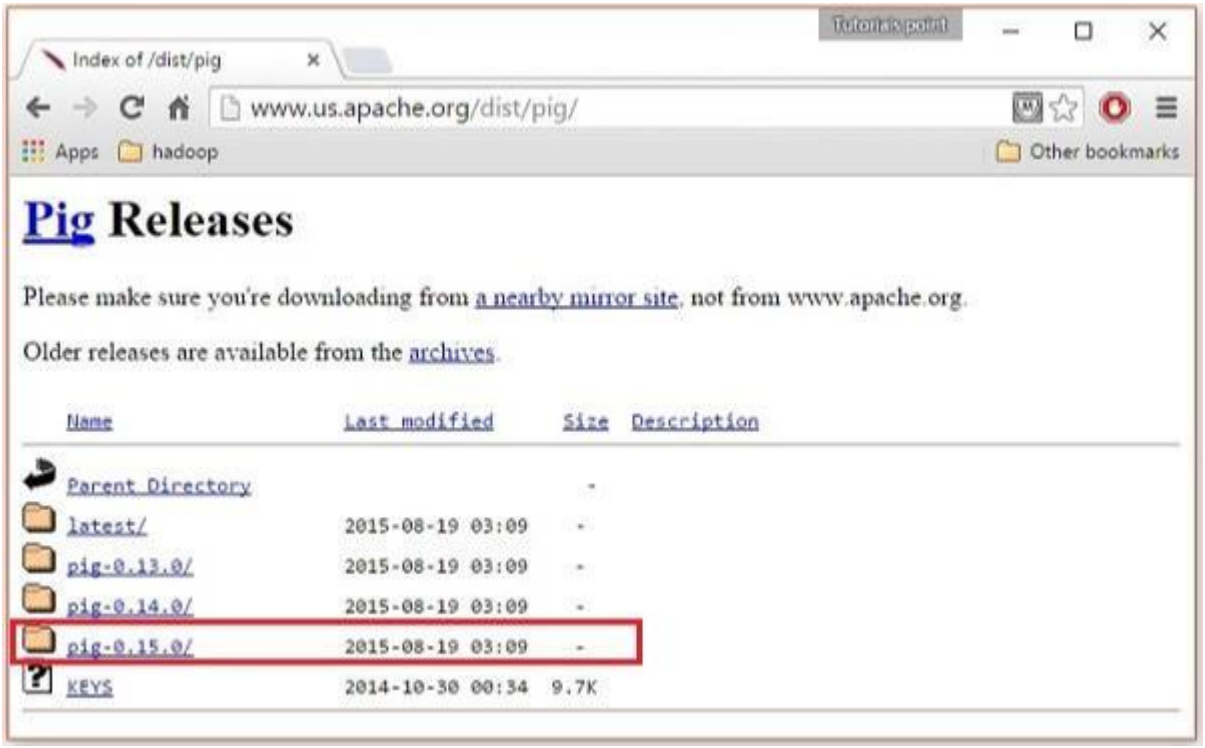


Step 3

Choose and click any one of these mirrors as shown below.

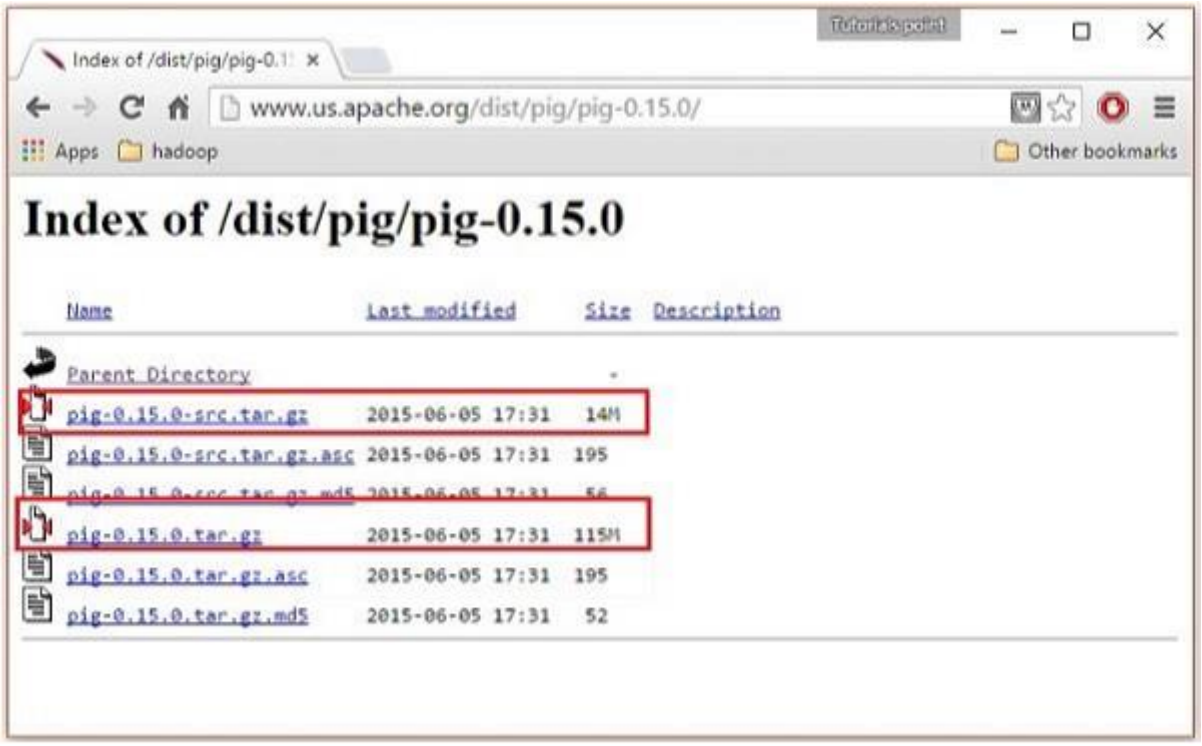
Step 4

These mirrors will take you to the **Pig Releases** page. This page contains various versions of Apache Pig. Click the latest version among them.



Step 5

Within these folders, you will have the source and binary files of Apache Pig in various distributions. Download the tar files of the source and binary files of Apache Pig 0.15, **pig0.15.0-src.tar.gz** and **pig-0.15.0.tar.gz.**



Install Apache Pig

After downloading the Apache Pig software, install it in your Linux environment by following the steps given below.

Step 1

Create a directory with the name Pig in the same directory where the installation directories of **Hadoop, Java,** and other software were installed. (In our tutorial, we have created the Pig directory in the user named Hadoop).

```
$ mkdir Pig
```

Step 2

Extract the downloaded tar files as shown below.

```
$ cd Downloads/
$ tar zxvf pig-0.15.0-src.tar.gz
```

```
$ tar zxvf pig-0.15.0.tar.gz
```

Step 3

Move the content of **pig-0.15.0-src.tar.gz** file to the **Pig** directory created earlier as shown below.

```
$ mv pig-0.15.0-src.tar.gz/* /home/Hadoop/Pig/
```

Configure Apache Pig

After installing Apache Pig, we have to configure it. To configure, we need to edit two files − **bashrc and pig.properties**.

.bashrc file

In the **.bashrc** file, set the following variables −

- **PIG_HOME** folder to the Apache Pig's installation folder,
- **PATH** environment variable to the bin folder, and
- **PIG_CLASSPATH** environment variable to the etc (configuration) folder of your Hadoop installations (the directory that contains the core-site.xml, hdfs-site.xml and mapred-site.xml files).

```
export PIG_HOME = /home/Hadoop/Pig
export PATH  = $PATH:/home/Hadoop/pig/bin
export PIG_CLASSPATH = $HADOOP_HOME/conf
```

pig.properties file

In the **conf** folder of Pig, we have a file named **pig.properties**. In the pig.properties file, you can set various parameters as given below.

```
pig -h properties
```

Verifying the Installation

Verify the installation of Apache Pig by typing the version command. If the installation is successful, you will get the version of Apache Pig as shown below.

```
$ pig –version
```

Apache Pig Execution Modes

Local Mode

In this mode, all the files are installed and run from your local host and local file system. There is no need of Hadoop or HDFS. This mode is generally used for testing purpose.

MapReduce Mode

MapReduce mode is where we load or process the data that exists in the Hadoop File System (HDFS) using Apache Pig. In this mode, whenever we execute the Pig Latin statements to process the data, a MapReduce job is invoked in the back-end to perform a particular operation on the data that exists in the HDFS.

Apache Pig Execution Mechanisms

Apache Pig scripts can be executed in three ways, namely, interactive mode, batch mode, and embedded mode.

- **Interactive Mode** (Grunt shell) − You can run Apache Pig in interactive mode using the Grunt shell. In this shell, you can enter the Pig Latin statements and get the output (using Dump operator).
- **Batch Mode** (Script) − You can run Apache Pig in Batch mode by writing the Pig Latin script in a single file with **.pig** extension.
- **Embedded Mode** (UDF) − Apache Pig provides the provision of defining our own functions (**U**ser **D**efined **F**unctions) in programming languages such as Java, and using them in our script.

Local mode	MapReduce mode
Command − $ ./pig –x local	**Command −** $ ./pig -x mapreduce

https://www.tutorialspoint.com/apache_pig/index.htm

Experiment No:6

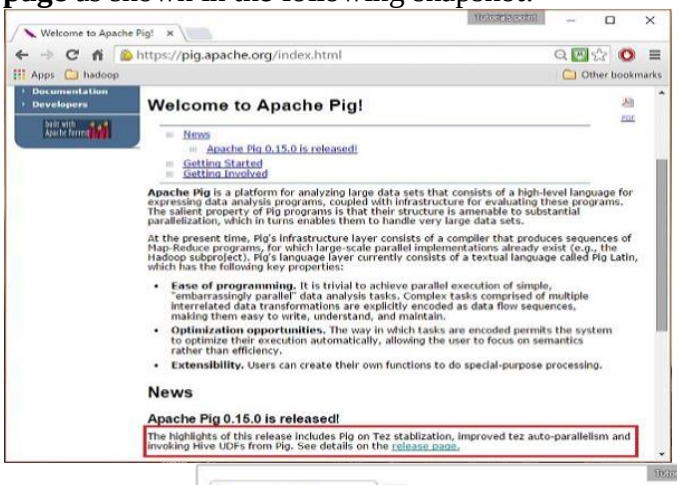
Aim: Install Pig and write Pig latin scripts to load, store and filter data.

Download Apache Pig

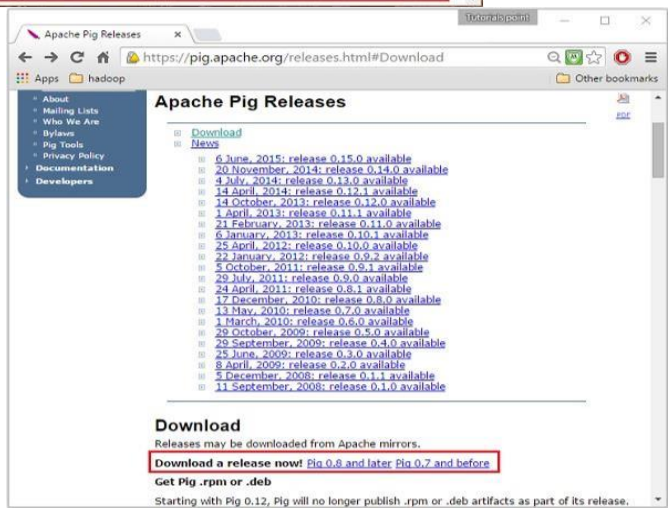
First of all, download the latest version of Apache Pig from the following website https://pig.apache.org/.

Step 1

Open the homepage of Apache Pig website. Under the section **News,** click on the link **release page** as shown in the following snapshot.



Step 2: **On the page, you will see a set of mirrors.**



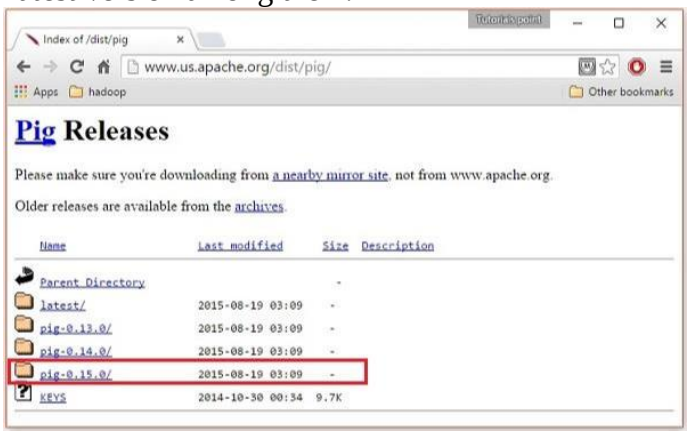
clicking the specified link, you will be redirected to the Apache Pig Releases **page. On this page, under the** Download **section, you have two links, namely,** Pig 0.8 and later **and** Pig 0.7 and before**. Click on the link** Pig 0.8 and later**, then you will be redirected to the page having a set of mirrors.**

Step 3

Choose and click any one of these mirrors as shown below.

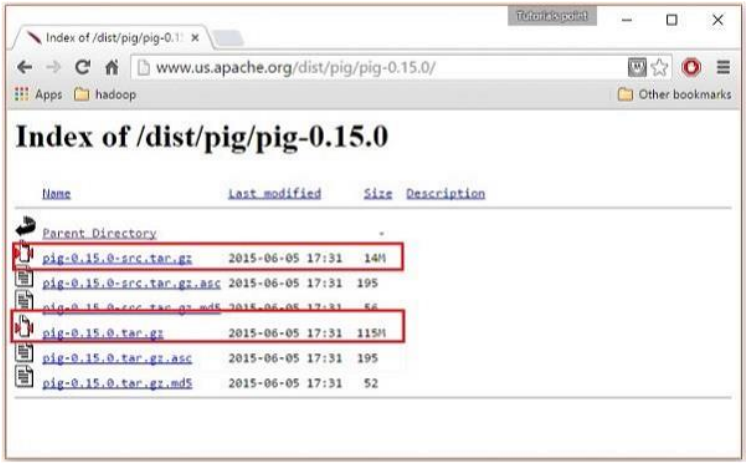
Step 4

These mirrors will take you to the **Pig Releases** page. This page contains various versions of Apache Pig. Click the latest version among them.



Step 5

Within these folders, you will have the source and binary files of Apache Pig in various distributions. Download the tar files of the source and binary files of Apache Pig 0.15, **pig0.15.0-src.tar.gz** and **pig-0.15.0.tar.gz.**



Install Apache Pig

After downloading the Apache Pig software, install it in your Linux environment by following the steps given below.

Step 1

Create a directory with the name Pig in the same directory where the installation directories of **Hadoop, Java,** and other software were installed. (In our tutorial, we have created the Pig directory in the user named Hadoop).

```
$ mkdir Pig
```

Step 2

Extract the downloaded tar files as shown below.

```
$ cd Downloads/
$ tar zxvf pig-0.15.0-src.tar.gz
$ tar zxvf pig-0.15.0.tar.gz
```

Step 3

Move the content of **pig-0.15.0-src.tar.gz** file to the **Pig** directory created earlier as shown below.

```
$ mv pig-0.15.0-src.tar.gz/* /home/Hadoop/Pig/
```

Configure Apache Pig

After installing Apache Pig, we have to configure it. To configure, we need to edit two files − **bashrc and pig.properties**.

.bashrc file

In the **.bashrc** file, set the following variables −

- **PIG_HOME** folder to the Apache Pig's installation folder,
- **PATH** environment variable to the bin folder, and
- **PIG_CLASSPATH** environment variable to the etc (configuration) folder of your Hadoop installations (the directory that contains the core-site.xml, hdfs-site.xml and mapred-site.xml files).

```
export PIG_HOME = /home/Hadoop/Pig
export PATH  = $PATH:/home/Hadoop/pig/bin
export PIG_CLASSPATH = $HADOOP_HOME/conf
```

pig.properties file

In the **conf** folder of Pig, we have a file named **pig.properties**. In the pig.properties file, you can set various parameters as given below.

```
pig -h properties
```

Verifying the Installation

Verify the installation of Apache Pig by typing the version command. If the installation is successful, you will get the version of Apache Pig as shown below.

```
$ pig –version
```

Apache Pig Execution Modes

Local Mode

In this mode, all the files are installed and run from your local host and local file system. There is no need of Hadoop or HDFS. This mode is generally used for testing purpose.

MapReduce Mode

MapReduce mode is where we load or process the data that exists in the Hadoop File System (HDFS) using Apache Pig. In this mode, whenever we execute the Pig Latin statements to process the data, a MapReduce job is invoked in the back-end to perform a particular operation on the data that exists in the HDFS.

Apache Pig Execution Mechanisms

Apache Pig scripts can be executed in three ways, namely, interactive mode, batch mode, and embedded mode.

- **Interactive Mode** (Grunt shell) − You can run Apache Pig in interactive mode using the Grunt shell. In this shell, you can enter the Pig Latin statements and get the output (using Dump operator).

- **Batch Mode** (Script) − You can run Apache Pig in Batch mode by writing the Pig Latin script in a single file with **.pig** extension.
- **Embedded Mode** (UDF) − Apache Pig provides the provision of defining our own functions (**U**ser **D**efined **F**unctions) in programming languages such as Java, and using them in our script.

Local mode	MapReduce mode
Command − $ ./pig –x local	**Command −** $ ./pig -x mapreduce

https://www.tutorialspoint.com/apache_pig/index.htm

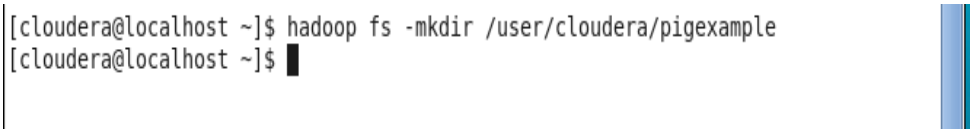
Procedure:

Click Oracle VM box and click start button.

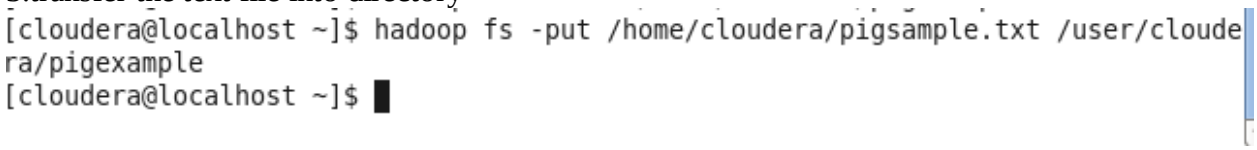
Steps: using single file system

1.open terminal and create a text file using touch command and insert data.

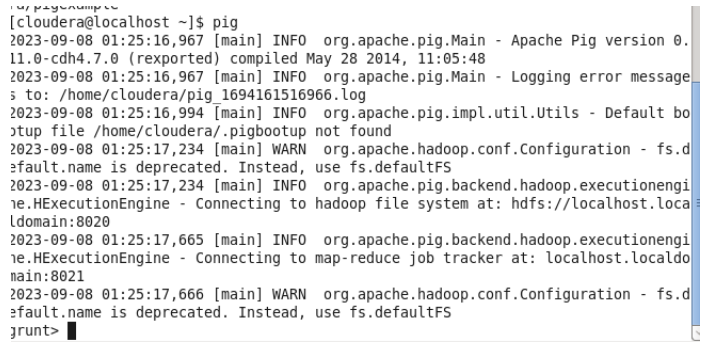
2.Create a directory “pigexample”



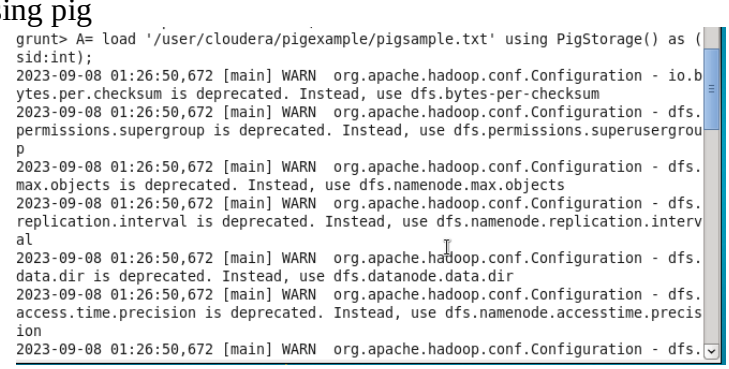
3.transfer the text file into directory



4.Open pig



5.Load the text file using pig



6.To see output

```
grunt> dump A;
2023-09-08 01:28:09,699 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: UNKNOWN
2023-09-08 01:28:09,793 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
2023-09-08 01:28:09,807 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2023-09-08 01:28:09,807 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
2023-09-08 01:28:09,910 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
```

7.Output:

```
2023-09-08 01:28:22,184 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2023-09-08 01:28:22,185 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2023-09-08 01:28:22,200 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2023-09-08 01:28:22,200 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(1)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)
(10)
grunt>
```

8.Quit:

```
(10)
grunt> quit
[cloudera@localhost ~]$
```

Steps: using csv format

1.open terminal and create a text file using touch command and insert data.

```
[cloudera@localhost ~]$ touch pigsample1.txt
[cloudera@localhost ~]$ vi pigsample1.txt
[cloudera@localhost ~]$ cat pigsample1.txt
501,padma,cse
502,divya,it
503,mounika,cse
504,nuthana,it
[cloudera@localhost ~]$
```

2.transfer the text file into directory

```
[cloudera@localhost ~]$ hadoop fs -put /home/cloudera/pigsample1.txt /user/cloud
era/pigexample
```

3.Open pig

```
[cloudera@localhost ~]$ pig
2023-09-08 01:33:48,010 [main] INFO  org.apache.pig.Main - Apache Pig version 0.
11.0-cdh4.7.0 (rexported) compiled May 28 2014, 11:05:48
2023-09-08 01:33:48,010 [main] INFO  org.apache.pig.Main - Logging error message
s to: /home/cloudera/pig_1694162028006.log
2023-09-08 01:33:48,028 [main] INFO  org.apache.pig.impl.util.Utils - Default bo
otup file /home/cloudera/.pigbootup not found
2023-09-08 01:33:48,285 [main] WARN  org.apache.hadoop.conf.Configuration - fs.d
efault.name is deprecated. Instead, use fs.defaultFS
2023-09-08 01:33:48,285 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to hadoop file system at: hdfs://localhost.loca
ldomain:8020
2023-09-08 01:33:48,706 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to map-reduce job tracker at: localhost.localdo
main:8021
2023-09-08 01:33:48,710 [main] WARN  org.apache.hadoop.conf.Configuration - fs.d
efault.name is deprecated. Instead, use fs.defaultFS
grunt>
```

4. .Load the text file using pig

```
grunt> A= load '/user/cloudera/pigexample/pigsample1.txt' using PigStorage(',')
as (sid:int,sname:chararray,dept:chararray);
2023-09-08 01:37:40,924 [main] WARN  org.apache.hadoop.conf.Configuration - io.b
ytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2023-09-08 01:37:40,924 [main] WARN  org.apache.hadoop.conf.Configuration - dfs.
permissions.supergroup is deprecated. Instead, use dfs.permissions.superusergrou
p
2023-09-08 01:37:40,924 [main] WARN  org.apache.hadoop.conf.Configuration - dfs.
max.objects is deprecated. Instead, use dfs.namenode.max.objects
2023-09-08 01:37:40,926 [main] WARN  org.apache.hadoop.conf.Configuration - dfs.
replication.interval is deprecated. Instead, use dfs.namenode.replication.interv
al
2023-09-08 01:37:40,926 [main] WARN  org.apache.hadoop.conf.Configuration - dfs.
data.dir is deprecated. Instead, use dfs.datanode.data.dir
2023-09-08 01:37:40,926 [main] WARN  org.apache.hadoop.conf.Configuration - dfs.
access.time.precision is deprecated. Instead, use dfs.namenode.accesstime.precis
```

5.To see output

```
grunt> dump A;
2023-09-08 01:38:10,249 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: UNKNOWN
2023-09-08 01:38:10,316 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
2023-09-08 01:38:10,329 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2023-09-08 01:38:10,329 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
```

6.Output:

```
2023-09-08 01:38:22,601 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2023-09-08 01:38:22,602 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2023-09-08 01:38:22,616 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2023-09-08 01:38:22,616 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(501,padma,cse)
(502,divya,it)
(503,mounika,cse)
(504,nuthana,it)
grunt>
```

Steps: using tab space format

1.open terminal and create a text file using touch command and insert data

```
[cloudera@localhost ~]$ touch pigsample2.txt
[cloudera@localhost ~]$ vi pigsample2.txt
[cloudera@localhost ~]$ cat pigsample2.txt
501 raju cse
502 goutham it
503 sai cse
504 nikil it
[cloudera@localhost ~]$
```

2.Transfer the text file into directory

```
[cloudera@localhost ~]$ hadoop fs -put /home/cloudera/pigsample2.txt /user/cloud
era/pigexample
[cloudera@localhost ~]$
```

3.Open pig and load the text file

```
grunt> A= load '/user/cloudera/pigexample/pigsample2.txt' using ('\t') as (sid:i
nt,sname:chararray,dept:chararray);
```

4.To see Output

```
grunt> dump A;
2023-09-08 01:38:10,249 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: UNKNOWN
2023-09-08 01:38:10,316 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
2023-09-08 01:38:10,329 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2023-09-08 01:38:10,329 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
```

5.Output:

```
2023-09-08 01:38:22,601 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2023-09-08 01:38:22,602 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2023-09-08 01:38:22,616 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2023-09-08 01:38:22,616 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(501,padma,cse)
(502,divya,it)
(503,mounika,cse)
(504,nuthana,it)
grunt>
```

Describe:

```
grunt> describe A;
A: {sid: int,sname: chararray,dept: chararray}
grunt>
```

Filter:

```
grunt> f1= filter A by dept=='cse';
grunt> dump f1;
2023-09-08 01:54:00,172 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: FILTER
2023-09-08 01:54:00,289 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
2023-09-08 01:54:00,301 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
```

Output:

```
2023-09-08 01:54:12,632 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2023-09-08 01:54:12,633 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2023-09-08 01:54:12,644 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2023-09-08 01:54:12,644 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(501,padma,cse)
(503,mounika,cse)
grunt>
```

Foreach:

```
grunt> f3= foreach A generate sid,dept;
grunt> dump f3;
2023-09-08 01:55:50,497 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
```

Output:

```
2023-09-08 01:56:02,402 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2023-09-08 01:56:02,402 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2023-09-08 01:56:02,406 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2023-09-08 01:56:02,406 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(501,cse)
(502,it)
(503,cse)
(504,it)
grunt>
```

Sort:

```
grunt> s= order A by dept ASC;
grunt> dump s;
```

Output:

Experiment No:7

Aim: Write Pig Latin scripts to perform data processing operations.

a)Grouping and joining data.

b)Sorting data

c)Combining and Splitting data.

Procedure:

Grouping data:

Syntax: variablename= group filename by Columnname;

1st-method : Create file in local file system

[cloudera@localhost ~]$ cat > a.txt

10
20
30
40
50
10
20
30
40
60
10
20
30
40
50

Transform the file from local system to hadoop environment

cloudera@localhost ~]$hadoop fs –put /home/cloudera/a.txt /user/cloudera/pigexample

Open the pig environment

[cloudera@localhost ~]$pig

Load the data into PigStorage

grunt>A= load 'user/cloudera/pigexample/a.txt' using PigStorage() as (age:int);

grunt>dump A;

```
cess : 1
(10)
(20)
(30)
(40)
(50)
(10)
(20)
(30)
(40)
(60)
(10)
(20)
(30)
(40)
(50)
grunt>
```

grunt>gr= group A by age;

grunt>dump gr;

```
cess : 1
(10,{(10),(10),(10)})
(20,{(20),(20),(20)})
(30,{(30),(30),(30)})
(40,{(40),(40),(40)})
(50,{(50),(50)})
(60,{(60)})
grunt>
```

2nd method:

[cloudera@localhost ~]$ cat > b.txt

```
[cloudera@localhost ~]$ vi h1.txt
[cloudera@localhost ~]$ cat h1.txt
5D0,mahi,CSE
5G4,Divya,CSE
5H1,Padma,CSE
5H7,Mounika,CSE
484,Ramya,ECE
4A5,Raju,ECE
1234,Siri,IT
12B6,Charan,IT
4201,Pavani,AIDS
[cloudera@localhost ~]$
```

cloudera@localhost ~]$hadoop fs –put /home/cloudera/b.txt /user/cloudera/pigexample

[cloudera@localhost ~]$pig

grunt>A= load ‘user/cloudera/pigexample/b.txt’ using PigStorage(‘,’) as (rno:chararray,sname:chararray,branch:chararray);

grunt>dump A;

```
2023-10-05 23:13:08,476 [M
cess : 1
(5D0,mahi,CSE)
(5G4,Divya,CSE)
(5H1,Padma,CSE)
(5H7,Mounika,CSE)
(484,Ramya,ECE)
(4A5,Raju,ECE)
(1234,Siri,IT)
(12B6,Charan,IT)
(4201,Pavani,AIDS)
grunt>
```

g=group A by branch;

dump g;

```
cess : 1
(IT,{(1234,Siri,IT),(12B6,Charan,IT)})
(CSE,{(5D0,mahi,CSE),(5G4,Divya,CSE),(5H1,Padma,CSE),(5H7,Mounika,CSE)})
(ECE,{(484,Ramya,ECE),(4A5,Raju,ECE)})
(AIDS,{(4201,Pavani,AIDS)})
grunt>
```

Sort:

Syntax: Variable= order data by attributename ASC/DESC;

Sort by Ascending order

grunt> sort1 = order data by age ASC;

grunt> dump sort1;

(12)
(19)
(24)
(24)
(25)
(27)
(35)
(35)
(45)
(55)
(65)

Sort by Descending order

grunt> sort2 = order data by age DESC;

grunt> dump sort2;

(65)
(55)
(45)
(35)
(35)
(27)

(25)
(24)
(24)
(19)
(12)

JOIN:

Joins can be of the following types −

- Self-join
- Inner-join
- Outer-join − left join, right join, and full join

```
cloudera@localhost ~]$ cat>a.txt
1,2,3
4,2,1
8,3,4
4,3,3
7,2,5
8,4,3
[cloudera@localhost ~]$ cat>b.txt
2,4
8,9
1,3
2,7
2,9
4,6
4,9
[cloudera@localhost ~]$ hadoop fs -put a.txt
[cloudera@localhost ~]$ hadoop fs -put b.txt
```

Self join

```
grunt> ONE= load 'a.txt' using PigStorage(',') as (a1:int,a2:int,a3:int);
grunt> TWO = load 'a.txt' using PigStorage(',') as (a1:int,a2:int,a3:int);
SELFJ = JOIN ONE by a1 , TWO BY a1;
grunt> describe SELFJ;
SELFJ: {ONE::a1: int,ONE::a2: int,ONE::a3: int,TWO::a1: int,TWO::a2: int,TWO::a3: int}
```

Equi-join.

```
grunt> A = load 'a.txt' using PigStorage(',') as (a1:int,a2:int,a3:int);
grunt> B = load 'b.txt' using PigStorage(',') as (b1:int,b2:int,b3:int);
grunt> X = Join A by a1, B by b1;
grunt> Dump X;
(1,2,3,1,3,)
(4,2,1,4,6,)
(4,2,1,4,9,)
(4,3,3,4,6,)
(4,3,3,4,9,)
(8,3,4,8,9,)
(8,4,3,8,9,)
```

Left outer join

```
A = LOAD  ‘A.txt' using PigStorage(',') AS (a1:int,a2:int,a3:int);
B = LOAD, ‘B.txt' using PigStorage(',') AS (b1:int,b2:int);
LEFTJ = JOIN A by a1 LEFT OUTER, B BY b1;
DUMP LEFTJ;
(1,2,3,1,3)
(4,3,3,4,9)
(4,3,3,4,6)
(4,2,1,4,9)
(4,2,1,4,6)
(7,2,5,,)
(8,4,3,8,9)
(8,3,4,8,9)
```

Right outer join

```
A = LOAD  ‘A.txt' using PigStorage(',') AS (a1:int,a2:int,a3:int);
B = LOAD, ‘B.txt' using PigStorage(',') AS (b1:int,b2:int);
RIGHTJ = JOIN A by a1 RIGHT OUTER, B BY b1;
DUMP RIGHTJ;
(1,2,3,1,3)
(,,,2,4)
(,,,2,7)
(,,,2,9)
(4,2,1,4,6)
(4,2,1,4,9)
(4,3,3,4,6)
(4,3,3,4,9)
(8,3,4,8,9)
(8,4,3,8,9)
```

Full join

```
A = LOAD  ‘A.txt' using PigStorage(',') AS (a1:int,a2:int,a3:int);
B = LOAD, ‘B.txt' using PigStorage(',') AS (b1:int,b2:int);
FULLJ = JOIN A by a1 FULL, B BY b1;
DUMP FULLJ;
(1,2,3,1,3)
(,,,2,4)
(,,,2,7)
(,,,2,9)
(4,2,1,4,6)
(4,2,1,4,9)
(4,3,3,4,6)
(4,3,3,4,9)
(7,2,5,,)
(8,3,4,8,9)
(8,4,3,8,9)
```

UNION & SPLIT

UNION combines multiple relations together whereas SPLIT partitions a relation in to multiple ones.

```
grunt> cat a.txt
1,2,3
4,2,1
8,3,4
grunt> cat b.txt
4,3,3
7,2,5
8,4,3
grunt> a = load 'a.txt' using PigStorage(',') as (a1:int, a2:int, a3:int);
grunt> b = load 'b.txt' using PigStorage(',') as (b1:int, b2:int, b3:int);
grunt> dump a;
(1,2,3)
(4,2,1)
(8,3,4)
grunt> dump b;
(4,3,3)
(7,2,5)
(8,4,3)
grunt> c = UNION a, b;
(1,2,3)
(4,2,1)
(8,3,4)
(4,3,3)
(7,2,5)
(8,4,3)
```

SPLIT:

```
grunt> SPLIT c into sp1 if $0 == 4, sp2 if $0 == 8;
```

Split operation on ‘c’ sends a tuple to sp1 if its first field ($0) is 0 , and to sp2 if it’s 1

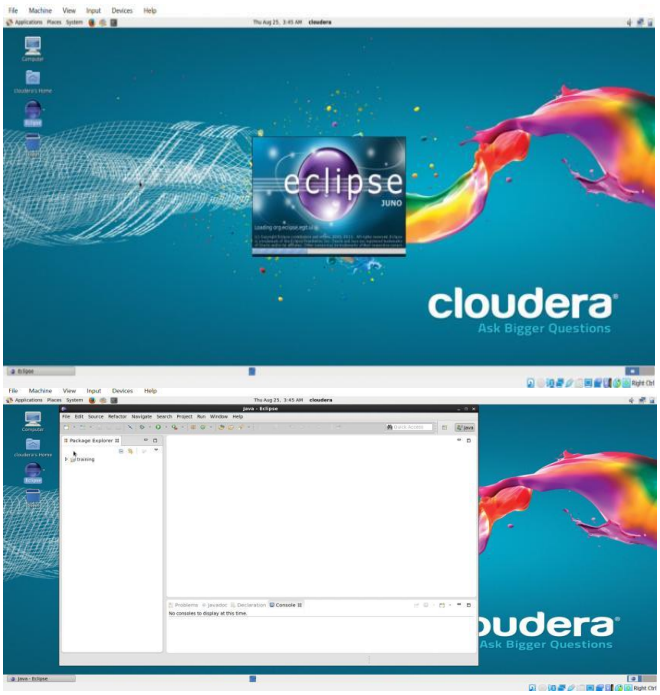
```
grunt> dump sp1;
(4,3,3)
(4,2,1)
grunt > dump sp2;
(8,4,3)
(8,3,4)
```

Experiment No:8

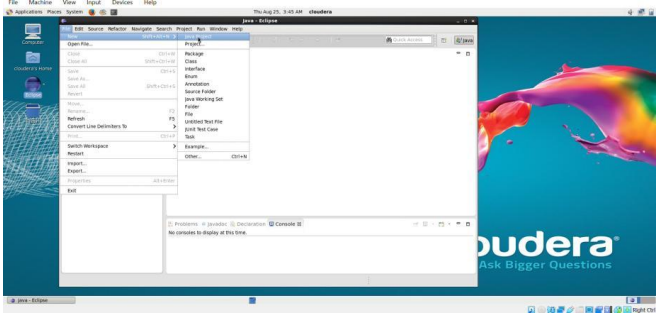
Aim: Implement User Defined functions in PIG.

Procedure:

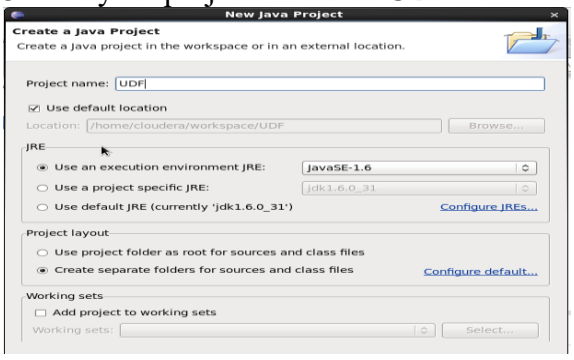
20. Open Oracle VM Virtual box -> click start -> open cloudera -> open eclipse.



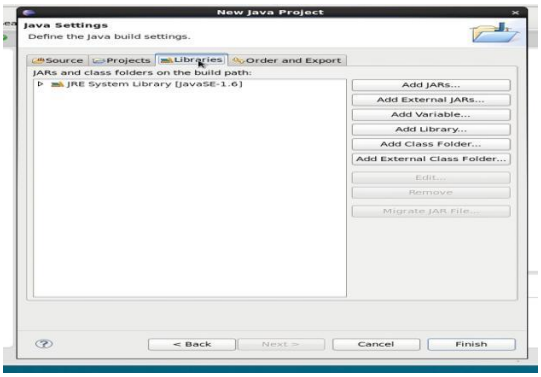
2.Click on file->New->Java Project.



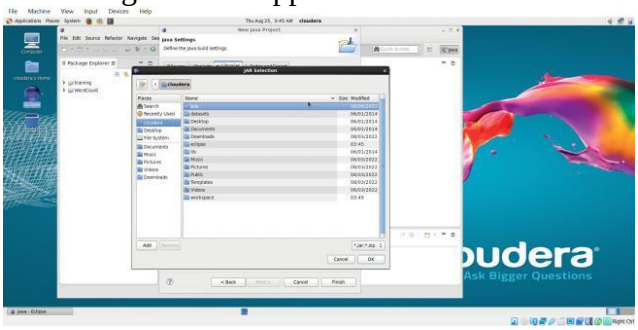
3.Give your project name as "**UDF**" and click on next.



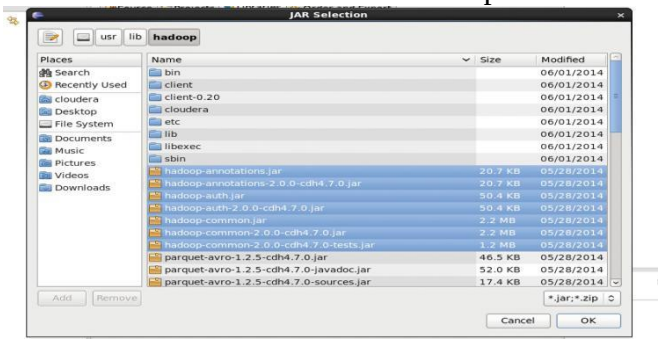
4.After that, click on libraries on the top and click on add external jars on the right side.



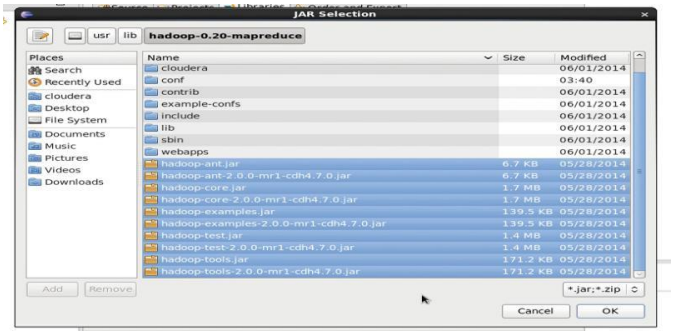
5.A dialog box will appear.



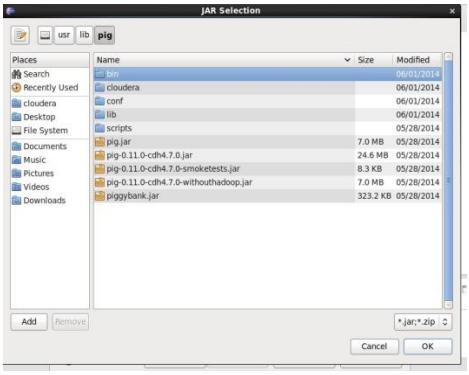
6.Click on file system on the left, and then usr->lib->hadoop.
Select all the files named with hadoop and click ok.



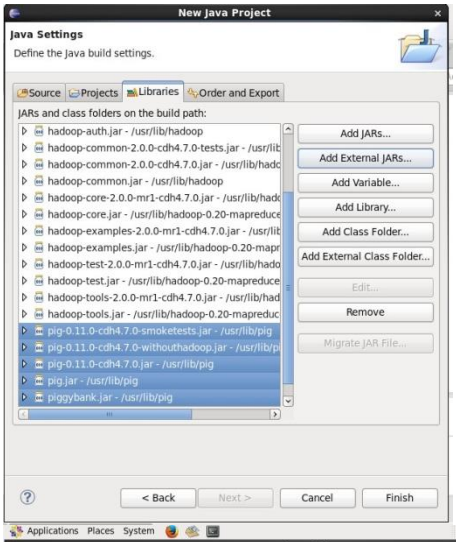
7.Go back to lib and click **hadoop-0.20-mapreduce** and select all the files named with hadoop and click ok.

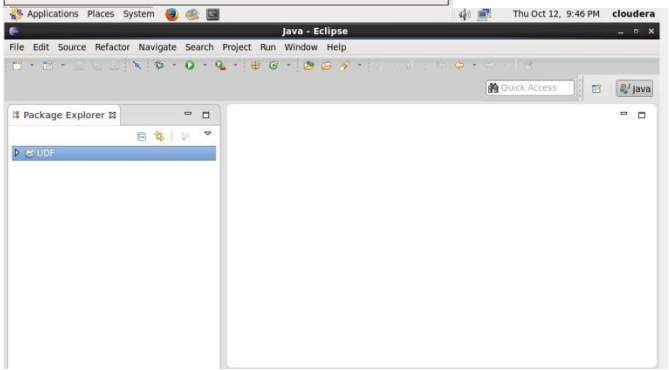


8.Go back to lib and click **pig** and select all the files named with PIG and click ok.

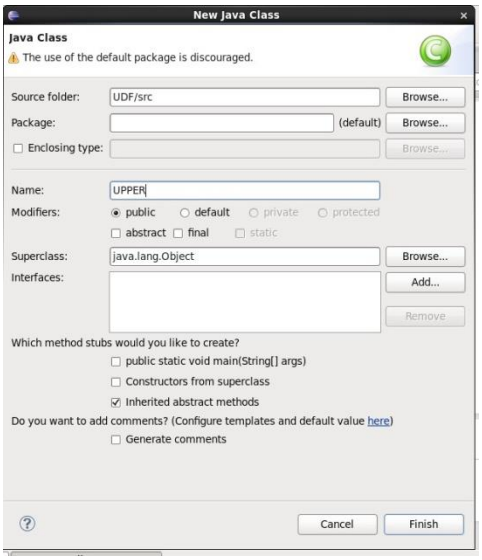


9.All the external files are added and now click on finish. The project is created successfully.





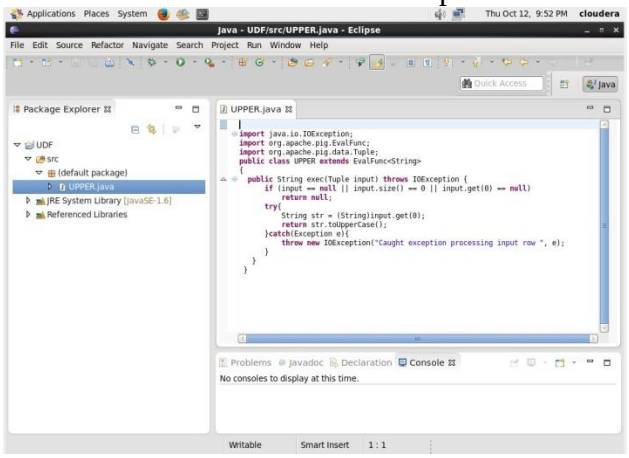
10.Create a class with name "**UPPER**" and click on finish.



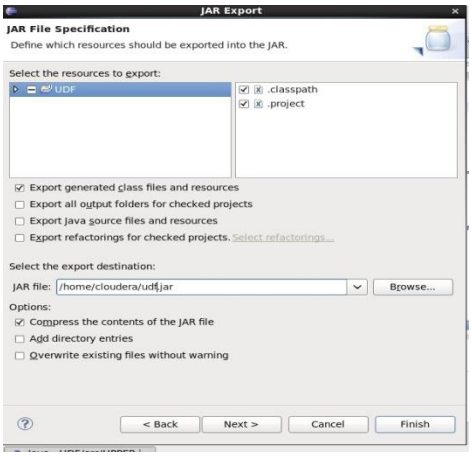
11.Open chrome and search for PIG UDF in Apache pig and copy the program into eclipse.

```
import java.io.IOException;
import org.apache.pig.EvalFunc;
import org.apache.pig.data.Tuple;
public class UPPER extends EvalFunc<String>
{
  public String exec(Tuple input) throws IOException {
      if (input == null || input.size() == 0 || input.get(0) == null)
          return null;
      try{
          String str = (String)input.get(0);
        return str.toUpperCase();
     }catch(Exception e){
        throw new IOException("Caught exception processing input row ", e);
      }
  }
}
```

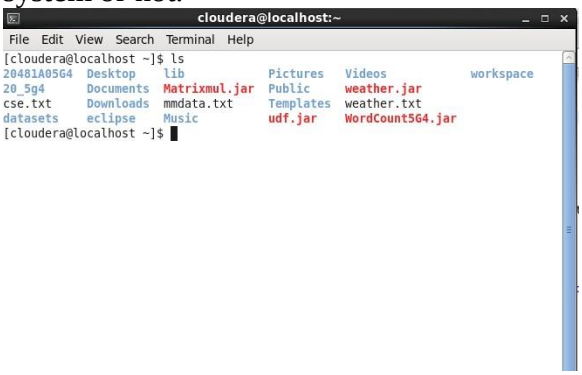
12.Copy and paste the program 4 lines at a time-If copy and paste doesn't work then goto device in cloudera and set both shared clipboard and drag and drop as **bidirectional.**



13. After finishing all the programs, right click on your project and select export Select java-> JAR file->next, Click on browse and a dialog box will appear, give any name for the jar file and click ok and click on finish.



14.Now go to cloud environment →click on terminal.Check whether the jar file is added on local system or not.



15.Create a directory in the local system named as **"pigexample"** .

16.Create a text file in local system named as **"pigsample2.txt"** and put the text file into directory.

```
[cloudera@localhost ~]$ touch pigsample2.txt
[cloudera@localhost ~]$ vi pigsample2.txt
[cloudera@localhost ~]$ cat pigsample2.txt
1,Divya,CSE
2,Padhu,CSE
3,Mouni,CSE
4,Nuthana,IT
5,Suma,IT
6,Swathi,AIDS
7,Raju,ECE
8,Manoj,ECE
9,Rakesh,EEE
10,Suresh,AIML
[cloudera@localhost ~]$
```

```
[cloudera@localhost ~]$ hadoop fs -mkdir pigexample
[cloudera@localhost ~]$ hadoop fs -put /home/cloudera/pigsample2.txt /user/cloud
era/pigexample
```

17.Open Pig Environment.

```
[cloudera@localhost ~]$ pig
2023-10-12 21:59:21,918 [main] INFO  org.apache.pig.Main - Apache Pig version 0.
11.0-cdh4.7.0 (rexported) compiled May 28 2014, 11:05:48
2023-10-12 21:59:21,918 [main] INFO  org.apache.pig.Main - Logging error message
s to: /home/cloudera/pig_1697173161917.log
2023-10-12 21:59:21,940 [main] INFO  org.apache.pig.impl.util.Utils - Default bo
otup file /home/cloudera/.pigbootup not found
2023-10-12 21:59:22,137 [main] WARN  org.apache.hadoop.conf.Configuration - fs.d
efault.name is deprecated. Instead, use fs.defaultFS
2023-10-12 21:59:22,137 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to hadoop file system at: hdfs://localhost.loca
ldomain:8020
2023-10-12 21:59:22,551 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.HExecutionEngine - Connecting to map-reduce job tracker at: localhost.localdo
main:8021
2023-10-12 21:59:22,552 [main] WARN  org.apache.hadoop.conf.Configuration - fs.d
efault.name is deprecated. Instead, use fs.defaultFS
grunt>
```

18.Register the jar file.

Syntax:REGISTER jarfile location

```
grunt> REGISTER /home/cloudera/udf.jar
2023-10-12 22:02:57,762 [main] WARN  org.apache.hadoop.conf.Configuration - dfs.
df.interval is deprecated. Instead, use fs.df.interval
2023-10-12 22:02:57,762 [main] WARN  org.apache.hadoop.conf.Configuration - dfs.
max.objects is deprecated. Instead, use dfs.namenode.max.objects
2023-10-12 22:02:57,762 [main] WARN  org.apache.hadoop.conf.Configuration - hado
op.native.lib is deprecated. Instead, use io.native.lib.available
```

19.Define UDF and load text file.

Syntax:DEFINE function-name class-name();

```
grunt> DEFINE myfun UPPER();
grunt> stu= load '/user/cloudera/pigexample/pigsample2.txt' using PigStorage(','
) as (sid:int,sname:chararray,dept:chararray);
grunt> dump stu;
2023-10-12 22:05:24,009 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: UNKNOWN
2023-10-12 22:05:24,072 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
2023-10-12 22:05:24,084 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
```

20.Output:

```
2023-10-12 22:05:36,649 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MapReduceLauncher - Success!
2023-10-12 22:05:36,651 [main] INFO  org.apache.pig.data.SchemaTupleBackend - Ke
y [pig.schematuple] was not set... will not generate code.
2023-10-12 22:05:36,667 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileI
nputFormat - Total input paths to process : 1
2023-10-12 22:05:36,667 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.util.MapRedUtil - Total input paths to process : 1
(1,Divya,CSE)
(2,Padhu,CSE)
(3,Mouni,CSE)
(4,Nuthana,IT)
(5,Suma,IT)
(6,Swathi,AIDS)
(7,Raju,ECE)
(8,Manoj,ECE)
(9,Rakesh,EEE)
(10,Suresh,AIML)
grunt>
```

21.Use FOREACH to generate your function.

Syntax:variable= FOREACH data generate function-name(attribute);

```
grunt> stu_UPPER= FOREACH stu GENERATE myfun(sname);
grunt> dump stu_UPPER;
2023-10-12 22:09:17,346 [main] INFO  org.apache.pig.tools.pigstats.ScriptState -
 Pig features used in the script: UNKNOWN
2023-10-12 22:09:17,348 [main] INFO  org.apache.pig.newplan.logical.rules.Column
PruneVisitor - Columns pruned for stu: $0, $2
2023-10-12 22:09:17,351 [main] INFO  org.apache.pig.backend.hadoop.executionengi
ne.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? fal
se
```

Experiment 9:

Aim: Install hive and use hive to create database and tables

a) Create and Drop databases
b) Create, Alter and Drop tables
c) Insert, Update and Delete records

What is Hive

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.

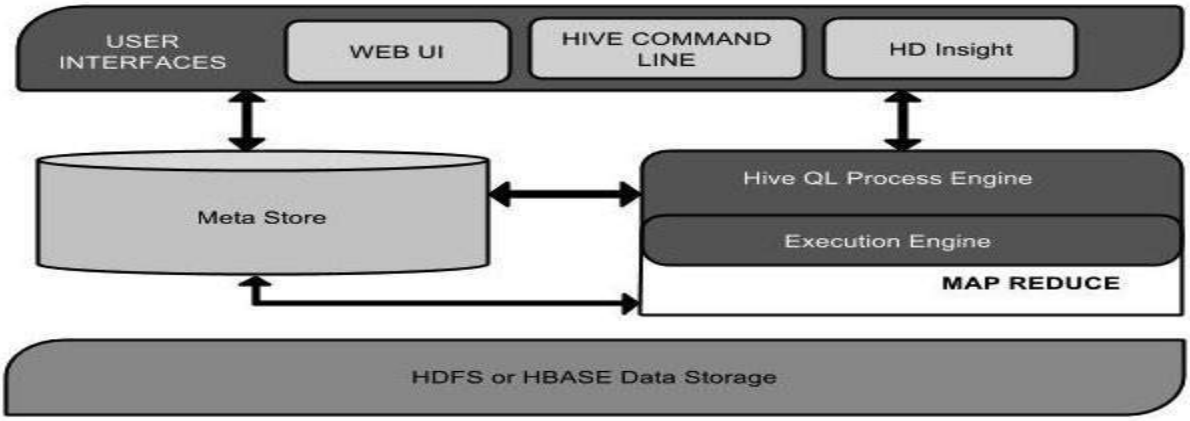
Hive is not

- A relational database
- A design for OnLine Transaction Processing (OLTP)
- A language for real-time queries and row-level updates

Features of Hive

- It stores schema in a database and processed data into HDFS.
- It is designed for OLAP.
- It provides SQL type language for querying called HiveQL or HQL.
- It is familiar, fast, scalable, and extensible.

Architecture of Hive



DATA TYPES

All the data types in Hive are classified into four types, given as follows:

- Column Types
- Literals
- Null Values
- Complex Types

Column Types

Column type are used as column data types of Hive. They are as follows:

Integral Types

Integer type data can be specified using integral data types, INT. When the data range exceeds the range of INT, you need to use BIGINT and if the data range is smaller than the INT, you use SMALLINT. TINYINT is smaller than SMALLINT.

Type	Postfix	Example
TINYINT	Y	10Y
SMALLINT	S	10S
INT	-	10
BIGINT	L	10L

String Types

String type data types can be specified using single quotes (' ') or double quotes (" "). It contains two data types: VARCHAR and CHAR. Hive follows C-types escape characters.

Data Type	Length
VARCHAR	1 to 65355
CHAR	255

Timestamp

It supports traditional UNIX timestamp with optional nanosecond precision. It supports java.sql.Timestamp format "YYYY-MM-DD HH:MM:SS.fffffffff" and format "yyyy-mm-dd hh:mm:ss.fffffffff".

Dates

DATE values are described in year/month/day format in the form {{YYYY-MM-DD}}.

Literals

The following literals are used in Hive:

Floating Point Types

Floating point types are nothing but numbers with decimal points. Generally, this type of data is composed of DOUBLE data type.

Decimal Type

Decimal type data is nothing but floating point value with higher range than DOUBLE data type. The range of decimal type is approximately -10^{-308} to 10^{308}.

Null Value

Missing values are represented by the special value NULL.

Complex Types

The Hive complex data types are as follows:

Arrays

Maps

Structs

Create Databases

Create Database is a statement used to create a database in Hive. A database in Hive is a **namespace** or a collection of tables. The **syntax** for this statement is as follows:

```
CREATE DATABASE|SCHEMA [IF NOT EXISTS] <database name>
```

Here, IF NOT EXISTS is an optional clause, which notifies the user that a database with the same name already exists. We can use SCHEMA in place of DATABASE in this command.

```
hive> CREATE DATABASE [IF NOT EXISTS] student;
```

or

```
hive> CREATE SCHEMA stident;
```

o/p:
hive> /*
OK
Time taken: 0.337 seconds
The following query is used to verify a databases list:
hive> show databases;
o/p:
OK
default
student
Time taken: 0.384 seconds

Drop Database

Drop Database is a statement that drops all the tables and deletes the database. Its syntax is as follows:

```
DROP DATABASE StatementDROP (DATABASE|SCHEMA) [IF EXISTS] database_name
[RESTRICT|CASCADE];
```

The following queries are used to drop a database. Let us assume that the database name is student

```
hive> DROP DATABASE IF EXISTS student;
```

Example
hive> drop database student;
OK
Time taken: 0.277 seconds
hive> show databases;
OK
default
Time taken: 0.056 seconds

Use database:

Syntax: use <database-name>
Hive> use student;
OK
Time taken: 0.02 seconds

Create Table

Create Table is a statement used to create a table in Hive. The syntax and example are as follows:

Example:

```
CREATE TABLE IF NOT EXISTS employee ( eid int, name String,
salary String, destination String)
COMMENT ‘Employee details’
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ‘\t’
LINES TERMINATED BY ‘\n’
STORED AS TEXTFILE;
```

```
hive> create table stu(sno int, name string, branch string);
OK
Time taken: 0.507 seconds
```

Show Schema

Syntax: describe table-name

```
hive> describe stu1;
OK
sno     int
name    string
branch string
Time taken: 0.413 seconds
```

Load Data Statement [INSERT DATA]

Generally, after creating a table in SQL, we can insert data using the Insert statement. But in Hive, we can insert data using the LOAD DATA statement.

While inserting data into Hive, it is better to use LOAD DATA to store bulk records. There are two ways to load data: one is from local file system and second is from Hadoop file system.

Syntax

The syntax for load data is as follows:

```
LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename
[PARTITION (partcol1=val1, partcol2=val2 ...)]
```

- LOCAL is identifier to specify the local path. It is optional.
- OVERWRITE is optional to overwrite the data in the table.
- PARTITION is optional.

Example
We will insert the following data into the table. It is a text file named **sample.txt** in **/home/user** directory.

```
[cloudera@localhost ~]$ vi sample.txt
[cloudera@localhost ~]$ cat sample.txt
501 Ravi CSE
502 Rani CSE
1201 Raja IT
1202 Roja IT
4201 vinay AI&ML

hive> load data local inpath '/home/cloudera/sample.txt' overwrite into table stu1;
Copying data from file:/home/cloudera/sample.txt
Copying file: file:/home/cloudera/sample.txt
Loading data to table default.stu1
rmr: DEPRECATED: Please use 'rm -r' instead.
Moved: 'hdfs://localhost.localdomain:8020/user/hive/warehouse/stu1' to trash at: hdfs://localhost.localdomain:8020/user/cloudera/.Trash/Current
chgrp: changing ownership of '/user/hive/warehouse/stu1': User does not belong to hive
Table default.stu1 stats: [num_partitions: 0, num_files: 1, num_rows: 0, total_size: 26, raw_data_size: 0]
OK
Time taken: 0.343 seconds
hive> select * from stu1;
OK
501 Ravi CSE
502 Rani CSE
1201 Raja IT
1202 Roja IT
4201 vinay AI&ML
Time taken: 0.1 seconds
hive> [cloudera@localhost ~]$
```

Alter Table Statement

It is used to alter a table in Hive.

Syntax

The statement takes any of the following syntaxes based on what attributes we wish to modify in a table.

ALTER TABLE name RENAME TO new_name
ALTER TABLE name ADD COLUMNS (col_spec[, col_spec ...])
ALTER TABLE name DROP [COLUMN] column_name
ALTER TABLE name CHANGE column_name new_name new_type
ALTER TABLE name REPLACE COLUMNS (col_spec[, col_spec ...])

Rename To… Statement

The following query renames the table from **employee** to **emp**.

EXAMPLE

```
hive> ALTER TABLE employee RENAME TO emp;
hive> alter table stu1 rename to student;
OK
Time taken: 0.558 seconds
hive> select * from student;
OK
501 Ravi CSE
502 Rani CSE
1201 Raja IT
1202 Roja IT
4201 vinay AI&ML
Time taken: 0.698 seconds
```

Add Columns

The following query adds a column named dept to the employee table.

Example

```
hive> ALTER TABLE employee ADD COLUMNS (
dept STRING COMMENT 'Department name');

hive> alter table student add columns(total string);
OK
Time taken: 0.451 seconds
hive> describe student;
OK
sno     int
name    string
branch string
total    string
```

Time taken: 0.309 seconds

Replace Statement

The following query deletes all the columns from the **employee** table and replaces it with **emp** and **name** columns:

```
hive> ALTER TABLE employee REPLACE COLUMNS (
eid INT empid Int,
ename STRING name String);
```

Drop Table Statement

The syntax is as follows:

```
DROP TABLE [IF EXISTS] table_name;
```

The following query drops a table named **employee**:

```
hive> DROP TABLE IF EXISTS employee;
```

Change Statement

Example

```
hive> ALTER TABLE employee CHANGE name ename String;
```

```
hive> alter table student change sno rno int;
OK
Time taken: 0.179 seconds
hive> describe student;
OK
rno     int
name    string
branch string
total    string
Time taken: 0.096 seconds
```

UPDATE

Use the UPDATE statement to modify data already written to Apache Hive. Depending on the condition specified in the optional WHERE clause, an UPDATE statement may affect every row in a table. You must have both the SELECT and UPDATE privileges to use this statement.

```
UPDATE tablename SET column = value [, column = value ...] [WHERE expression];
```

The UPDATE statement has the following limitations:

- The expression in the WHERE clause must be an expression supported by a Hive SELECT clause.
- Partition and bucket columns cannot be updated.
- Query vectorization is automatically disabled for UPDATE statements. However, updated tables can still be queried using vectorization.
- Subqueries are not allowed on the right side of the SET statement.

The following example demonstrates the correct usage of this statement:

```
UPDATE students SET rno = 4201 WHERE rno==501;
```

DELETE

Use the DELETE statement to delete data already written to Apache Hive.

```
DELETE FROM tablename [WHERE expression];
DELETE FROM students WHERE rno=4201;
```

Experiment 10:

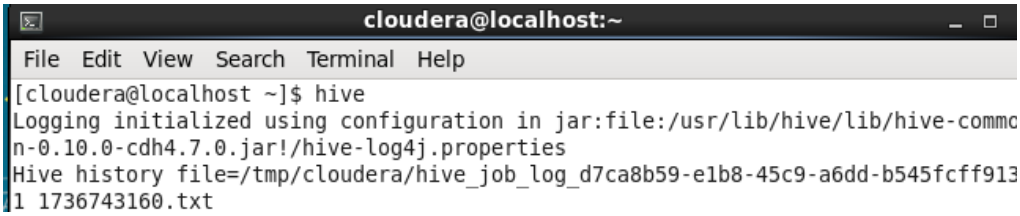
Aim: Perform data processing operations using Hive

a) **Sort and Aggregation of data**
b) **Joins**

Program:

Sorting of Data:

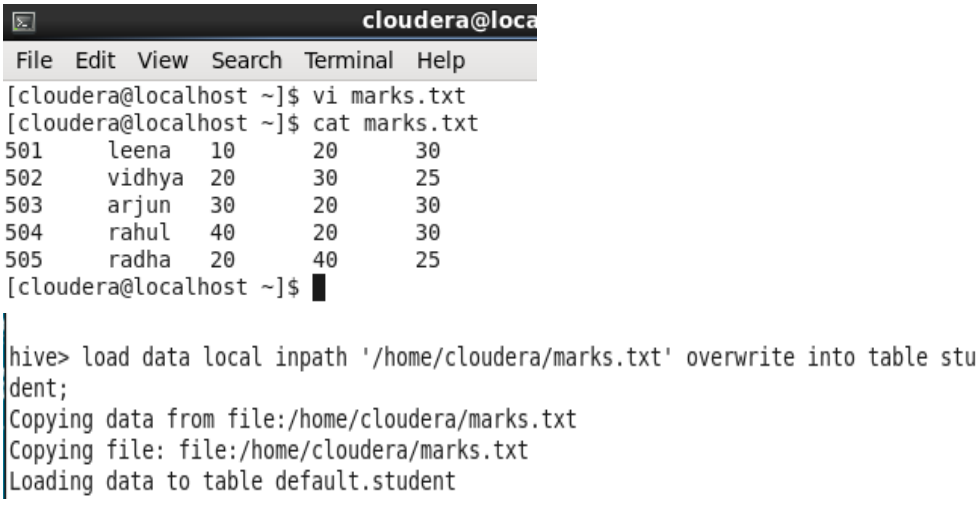
1.First enter into the hive environment in the VM and create a database named 'csebda'.



2.Create a table in the database named 'student'.

```
hive> create table if not exists student(sno int,sname string,m1 int,m2 int,m3 i
nt)
    > comment 'student'
    > row format delimited
    > fields terminated by '\t'
    > lines terminated by '\n'
    > stored as textfile;
OK
Time taken: 0.287 seconds
hive>
```

3.To insert the data into the table crate a text file in local environment and overwrite it into the table.



Order by:

```
hive> select * from student;
OK
501     leena   10      20      30
502     vidhya  20      30      25
503     arjun   30      20      30
504     rahul   40      20      30
505     radha   20      40      25
Time taken: 0.222 seconds
hive>
```

```
hive> desc student;
OK
sno     int
sname   string
m1      int
m2      int
m3      int
Time taken: 0.16 seconds
hive>
```

To arrange data in ascending order:

```
hive> select * from student order by sno;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
```

Output:

```
96 Success
Total MapReduce CPU Time Spent: 880 msec
OK
501     leena   10      20      30
502     vidhya  20      30      25
503     arjun   30      20      30
504     rahul   40      20      30
505     radha   20      40      25
Time taken: 9.791 seconds
hive>
```

To arrange data in descending order:

```
hive> select * from student order by sno desc;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
```

Output:

```
Total MapReduce CPU Time Spent: 790 msec
OK
505     radha   20      40      25
504     rahul   40      20      30
503     arjun   30      20      30
502     vidhya  20      30      25
501     leena   10      20      30
Time taken: 11.44 seconds
hive>
```

Sort by:

To arrange data in ascending order:

```
hive> select * from student sort by sname asc;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
```

Output:

```
Total MapReduce CPU Time Spent: 810 msec
OK
503     arjun   30      20      30
501     leena   10      20      30
505     radha   20      40      25
504     rahul   40      20      30
502     vidhya  20      30      25
Time taken: 10.413 seconds
```

To arrange data in descending order:

```
hive> select * from student sort by sname desc;
Total MapReduce jobs = 1
Launching Job 1 out of 1
```

Output:

```
Total MapReduce CPU Time Spent: 740 msec
OK
502     vidhya  20      30      25
504     rahul   40      20      30
505     radha   20      40      25
501     leena   10      20      30
503     arjun   30      20      30
Time taken: 9.467 seconds
```

Aggregate Functions:

min():

```
hive> select min(m1) from student;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
```

Output:

```
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulative CPU: 0.77 sec
 3 SUCCESS
Total MapReduce CPU Time Spent: 770 msec
OK
10
Time taken: 10.344 seconds
hive>
```

max():

```
hive> select max(m1) from student;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at
```

Output:

```
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulative
 3 SUCCESS
Total MapReduce CPU Time Spent: 780 m
OK
40
Time taken: 8.348 seconds
```

count(*):

```
hive> select count(*) from student;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
```

Output:

```
Job 0: Map: 1  Reduce: 1   Cumulative CPU: 0.78 sec
 2 SUCCESS
Total MapReduce CPU Time Spent: 780 msec
OK
5
Time taken: 10.3 seconds
hive>
```

sum():

```
hive> select sum(m1) from student;
Total MapReduce jobs = 1
Launching Job 1 out of 1
```

Output:

```
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulativ
 4 SUCCESS
Total MapReduce CPU Time Spent: 780
OK
120
Time taken: 10.33 seconds
hive>
```

avg():

```
hive> select avg(m1) from student;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
```

Output:

```
MapReduce Jobs Launched:
Job 0: Map: 1  Reduce: 1   Cumulative CPU:
 5 SUCCESS
Total MapReduce CPU Time Spent: 870 msec
OK
24.0
Time taken: 10.295 seconds
hive>
```

Joins:

Create two tables named sales,products and insert the data into them.

```
[cloudera@localhost ~]$ vi sales.txt
[cloudera@localhost ~]$ cat sales.txt
ramesh  10
suresh  20
haresh  0
naresh  30
rakesh  30
[cloudera@localhost ~]$ cat products.txt
10      laptop
20      printer
30      cpu
40      desktop
50      mouse
```

Sales Table:

```
[cloudera@localhost ~]$ hive
Logging initialized using configuration in jar:file:/usr/lib/hive/lib/hive-commo
n-0.10.0-cdh4.7.0.jar!/hive-log4j.properties
Hive history file=/tmp/cloudera/hive_job_log_26b7b8ac-d398-4e54-b75c-b112699a40a
e_1211167408.txt
hive> create table sale(cname string,pid int)
    > comment 'sales'
    > row format delimited
    > fields terminated by '\t'
    > lines terminated by '\n'
    > stored as textfile;
OK
Time taken: 0.285 seconds

hive> load data local inpath '/home/cloudera/sales.txt' overwrite into table sal
e;
Copying data from file:/home/cloudera/sales.txt
Copying file: file:/home/cloudera/sales.txt
Loading data to table default.sale
rmr: DEPRECATED: Please use 'rm -r' instead.
Moved: 'hdfs://localhost.localdomain:8020/user/hive/warehouse/sale' to trash at:
 hdfs://localhost.localdomain:8020/user/cloudera/.Trash/Current
chgrp: changing ownership of '/user/hive/warehouse/sale': User does not belong t
o hive
Table default.sale stats: [num_partitions: 0, num_files: 1, num_rows: 0, total_s
ize: 49, raw_data_size: 0]
OK
Time taken: 0.844 seconds

hive> select *from sale;
OK
ramesh  10
suresh  20
haresh  0
naresh  30
rakesh  30
Time taken: 0.207 seconds
```

Products Table:

```
hive> create table product(prodid int,pname string)
    > comment 'products'
    > row format delimited
    > fields terminated by '\t'
    > lines terminated by '\n'
    > stored as textfile;
OK
Time taken: 0.053 seconds

hive> load data local inpath '/home/cloudera/products.txt' overwrite into table
product;
Copying data from file:/home/cloudera/products.txt
Copying file: file:/home/cloudera/products.txt
Loading data to table default.product
rmr: DEPRECATED: Please use 'rm -r' instead.
Moved: 'hdfs://localhost.localdomain:8020/user/hive/warehouse/product' to trash
at: hdfs://localhost.localdomain:8020/user/cloudera/.Trash/Current
chgrp: changing ownership of '/user/hive/warehouse/product': User does not belon
g to hive
Table default.product stats: [num_partitions: 0, num_files: 1, num_rows: 0, tota
l_size: 48, raw_data_size: 0]
OK
Time taken: 0.193 seconds
hive> select * from product;
OK
10      laptop
20      printer
30      cpu
40      desktop
50      mouse
Time taken: 0.077 seconds
```

Inner Join:

```
hive> select sales.*,products.* from sales join products on(sales.pid=products.prodid);
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202311022151_0004, Tracking URL = http://0.0.0.0:50030/jobdetails.jsp?jobid=job_202311022151_0004
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_202311022151_0004
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2023-11-02 22:28:11,978 Stage-1 map = 0%,  reduce = 0%
2023-11-02 22:28:17,005 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.75 sec
2023-11-02 22:28:18,013 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.75 sec
2023-11-02 22:28:19,019 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.75 sec
2023-11-02 22:28:20,023 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.28 sec
2023-11-02 22:28:21,054 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.28 sec
2023-11-02 22:28:22,061 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.28 sec
MapReduce Total cumulative CPU time: 1 seconds 280 msec
Ended Job = job_202311022151_0004
MapReduce Jobs Launched:
Job 0: Map: 2  Reduce: 1   Cumulative CPU: 1.28 sec   HDFS Read: 549 HDFS Write: 58 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 280 msec
OK
ramesh  10      10      laptop
suresh  20      20      printer
naresh  30      30      cpu
Time taken: 12.737 seconds
hive>
```

Outer Join:

Left Outer Join:

```
hive> select sales.*,products.* from sales left outer join products on(sales.pid=products.prodid);
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202311022151_0006, Tracking URL = http://0.0.0.0:50030/jobdetails.jsp?jobid=job_202311022151_0006
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_202311022151_0006
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2023-11-02 22:30:52,317 Stage-1 map = 0%,  reduce = 0%
2023-11-02 22:30:57,364 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.64 sec
2023-11-02 22:30:58,370 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.64 sec
2023-11-02 22:30:59,375 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.64 sec
2023-11-02 22:31:00,378 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.64 sec
2023-11-02 22:31:01,381 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.22 sec
2023-11-02 22:31:02,392 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.22 sec
MapReduce Total cumulative CPU time: 1 seconds 220 msec
Ended Job = job_202311022151_0006
MapReduce Jobs Launched:
Job 0: Map: 2  Reduce: 1   Cumulative CPU: 1.22 sec   HDFS Read: 549 HDFS Write: 92 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 220 msec
OK
rakesh 30       NULL    NULL    NULL
haresh  0       NULL    NULL
ramesh  10      10      laptop
suresh  20      20      printer
naresh  30      30      cpu
Time taken: 12.415 seconds
```

Right Outer Join:

```
hive> select sales.*,products.* from sales right outer join products on(sales.pid=products.prodid);
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202311022151_0007, Tracking URL = http://0.0.0.0:50030/jobdetails.jsp?jobid=job_202311022151_0007
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_202311022151_0007
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2023-11-02 22:31:47,215 Stage-1 map = 0%,  reduce = 0%
2023-11-02 22:31:52,244 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.66 sec
2023-11-02 22:31:53,253 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.66 sec
2023-11-02 22:31:54,257 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.66 sec
2023-11-02 22:31:55,266 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.66 sec
2023-11-02 22:31:56,270 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.2 sec
2023-11-02 22:31:57,282 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.2 sec
MapReduce Total cumulative CPU time: 1 seconds 200 msec
Ended Job = job_202311022151_0007
MapReduce Jobs Launched:
Job 0: Map: 2  Reduce: 1   Cumulative CPU: 1.2 sec   HDFS Read: 549 HDFS Write: 90 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 200 msec
OK
ramesh  10      10      laptop
suresh  20      20      printer
naresh  30      30      cpu
NULL    NULL    40      desktop
NULL    NULL    50      mouse
Time taken: 12.432 seconds
```

Full Outer Join:

```
hive> select sales.*,products.* from sales full outer join products on(sales.pid=products.prodid);
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
Starting Job = job_202311022151_0008, Tracking URL = http://0.0.0.0:50030/jobdetails.jsp?jobid=job_202311022151_0008
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_202311022151_0008
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2023-11-02 22:33:29,895 Stage-1 map = 0%,  reduce = 0%
2023-11-02 22:33:34,926 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.69 sec
2023-11-02 22:33:35,933 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.69 sec
2023-11-02 22:33:36,936 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.69 sec
2023-11-02 22:33:37,940 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.21 sec
2023-11-02 22:33:38,944 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.21 sec
2023-11-02 22:33:39,952 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.21 sec
MapReduce Total cumulative CPU time: 1 seconds 210 msec
Ended Job = job_202311022151_0008
MapReduce Jobs Launched:
Job 0: Map: 2  Reduce: 1   Cumulative CPU: 1.21 sec   HDFS Read: 549 HDFS Write: 124 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 210 msec
OK
rakesh 30       NULL    NULL    NULL
haresh  0       NULL    NULL
ramesh  10      10      laptop
suresh  20      20      printer
naresh  30      30      cpu
NULL    NULL    40      desktop
NULL    NULL    50      mouse
Time taken: 12.307 seconds
hive>
```

Experiment 11:

Aim: Perform data processing operations using Hive

a)Views

b)Indexes

Entering into the hive enviornment and performing operations:

```
[cloudera@localhost ~]$ hive
Logging initialized using configuration in jar:file:/usr/lib/hive/lib/hive-commo
n-0.10.0-cdh4.7.0.jar!/hive-log4j.properties
Hive history file=/tmp/cloudera/hive_job_log_b965e3bf-0fee-484a-a3b3-abbe937ba2b
9_1567438058.txt
hive> desc student;
OK
sno     int
sname   string
m1      int
m2      int
m3      int
Time taken: 0.088 seconds
```

View:

Creating View:

Syntax:create view [if not exists] view_name [(c1,c2,....,cn)]
as select statement

```
hive> create view student_vw
    > as select * from student;
OK
Time taken: 0.182 seconds
```

How to execute view:

Syntax:select * from view_name;

```
hive> select * from student_vw;
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_202311091950_0001, Tracking URL = http://0.0.0.0:50030/jobdet
ails.jsp?jobid=job_202311091950_0001
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_202311091950_0001
```

Output:

```
Total MapReduce CPU Time Spent: 260 msec
OK
501     siri    10      20      30
502     giri    20      30      40
503     anu     30      40      50
510     anusha  50      10      44
506     pooja   44      20      66
517     sujatha 22      33      55
Time taken: 10.233 seconds
```

alter or change view:

Syntax:alter view[database_name] view_name as select statement;

Drop view:

Syntax:drop view [if exists] database_name;

```
hive> drop view if exists student_vw;
OK
Time taken: 0.5 seconds
```

Index:

Indexing are classified into 2 types.

1.Compact indexing

2.Bitmap indexing

Compact Indexing:

Creating:

Syntax:create index <index_name> on table <table_name><column_name>
as
'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler'
WITH DEFERRED REBUILD;

```
hive> create index index_sno on table student(sno)
    > as
    > 'org.apache.hadoop.hive.ql.index.compact.CompactIndexHandler'
    > WITH DEFERRED REBUILD;
OK
Time taken: 0.548 seconds
```

To see index details:

syntax:show index on table_name;

```
hive> show index on student;
OK
index_sno               student                 sno                     default_
_student_index_sno__    compact
Time taken: 0.641 seconds
```

Alter:

Syntax:alter index index_name on table_name REBUILD;

Drop:

Syntax:drop index index_name on table_name;

```
hive> drop index index_sno on student;
OK
Time taken: 0.081 seconds
```

BITMAP indexing:

Creating:

Syntax:create index <index_name> on table <table_name><column_name>
as
'BITMAP'
WITH DEFERRED REBUILD;

```
hive> create index index_sno on table student(sno)
    > as
    > 'BITMAP'
    > WITH DEFERRED REBUILD;
OK
Time taken: 0.09 seconds
```

To see index details:

syntax:show index on table_name;