GLOBAL ACADEMY OF TECHNOLOGY

Department of Artificial Intelligence and Machine Learning

Affiliated to VTU, Accredited by NAAC with 'A' grade

RR Nagar, Bengaluru – 560 098

LAB MANUAL

Big Data Analytics Lab Manual

V Semester

Course Code: 22AML54

Version 1.0

w.e.f 15.11.2023

Editorial Committee

Big Data Analytics Laboratory Faculty,

Dept. of AIML

Approved by

Dr. Roopa

Professor & Head,
Department of AIML

GLOBAL ACADEMY OF TECHNOLOGY

Department of Artificial Intelligence and Machine Learning

Affiliated to VTU, Accredited by NAAC with 'A' grade

RR Nagar, Bengaluru – 560 098

LABORATORY CERTIFICATE

This is to certify that Mr./ Ms .... bearing USN.. of Artificial Intelligence and Machine Learning Department has satisfactorily completed the course of Big Data Analytics Laboratory prescribed by Global Academy of Technology (Autonomous Institute, under VTU) for the Academic Year 2023-2024

Marks	
Maximum	**Obtained**
50	

Signature of the Faculty

Signature of the HOD

Date:

DOCUMENT LOG

Name of the document	Big Data Analytics Laboratory Manual
Syllabus Scheme	2022
Current version number and date	V1 / 15.11.2023
Subject code	22AML54
Editorial Committee	Big Data Analytics Laboratory Faculty
Approved by	HOD, Dept. of AIML
Lab Faculty	Prof. Srinath K S Mrs. Ramya

Table of Contents

Sl. No.	Particulars.		Page No.
1	**Vision and Mission of The Department**		1
2	**PEOs, PSOs, POs**		2
3	Course Details • Course Objectives		3
4	Syllabus • Course Outcomes • Conduction of Practical Examination • CO-PO-PSO Mapping		5
5	Lab Evaluation Process		7
6	Marks Distribution		7
7	Rubrics		8
8	**PROGRAMS**		
	Program 1	Implement a map reduce for word count from a given input text file.	9
	Program 2	Execute a Map reduce python program for printing maximum salary for a given input file.	13
	Program 3	Execute a Map reduce python program for printing maximum salary for a given input file.	15
	Program 4	Execute a python program to implement map reduce concepts for printing year wise sales from a given csv file.	17
	Program 5	Execute a python program to implement map reduce for inverted index of a given data set.	19
	Program 6	Execute a python program to implement word count using spark cell	21
	Program 7	Develop a program to Agglomerative Hierarchical clustering	26
	Program 8	Develop a program to implement OPTICS algorithms	28
	Program 9	Implement DBSCAN algorithm using appropriate Data sets.	30
	Program 10	Implement multiple data visualization method using plotly	32

9	**Installation steps of Hadoop and Spark**	36
10	**Viva Questions**	65

Vision of the Department

To provide progressive education and create transformative professionals and leaders to harness the power of technology and make a positive impact on the society.

Mission of the Department

Quality Education: To adopt a student centric curriculum delivery process with emphasis on problem solving and programming skills.

Innovation: To collaborate with industries and professional bodies and make the students industry ready. To drive innovation through multi-disciplinary research and development activities.

Skill Development & Ethics: To endorse additional skill development through student forums and experiential learning. Inculcate human values for a smarter and ethical world

Program Educational Objectives (PEOs) of Department

After the course completion, CSE graduates will be able to:

- Able to practice and implement their success skills like problem solving, communication and collaboration for providing innovative engineering solutions.
- Contribute their AI & ML expertise grounded in computer science as members and leaders of professional engineering teams in multidisciplinary applications.
- Demonstrate lifelong learning through continued professional development and higher education in top graduate technical, research and management programs.
- Demonstrate a commitment to society by applying the skills and knowledge for a smarter and ethical world.

Program Specific Outcomes (PSOs)

- Graduates will be Proficient in programming and problem-solving skills for developing, managing software and distributed systems.
- Graduates will be able to identify, formulate, predict and solve real world problems by applying principles of Artificial Intelligence & Machine learning.

Program Outcomes (POs)

Engineering Graduates will be able to:

Engineering Knowledge
Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

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.

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.

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.

Modern Tool Usage
Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

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.

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.

Ethics
Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

Individual and Team Work
Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

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.

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.

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.

Course Details

Course Name: Big data Analytics Laboratory

Course Code: 20AML54

Course prerequisite: Engineering Mathematics, Matrix, and Linear Algebra, Multivariate Statistics

Course Objectives

1. Install Hadoop and configure Single node cluster mentioning name node and data node
2. Implement the basic knowledge of Big Data Analytics using Hadoop-Map reduce concepts in python
3. Implement the Spark concepts in python
4. Implementing the machine learning algorithms using appropriate data set
5. Demonstrate Data visualization concepts using portly

SYLLABUS

Subject Code	**22AML54**	**CIE Marks**	**50**
Hours/Week (L: T: P)	**0:0:2**	**SEE Marks**	**50**
Credits	**01**	**Examination Hours**	**03**

Description (If any):

1. The programs can be implemented in Python
2. Data sets can be taken from standard repositories https://archive.ics.uci.edu/ml/datasets.html or constructed by the students.

Lab Programs:

1. Implement map reduce for word count from a given input text file.
2. Execute a Map-reduce python program for printing average salary for a given input file.
3. Execute a Map-reduce python program for printing maximum salary for a given input file.
4. Execute a python program to implement map reduce concepts for printing year wise sales from a given csv file.
5. Execute a python program to implement Inverted index
6. Write and Execute word count program using spark shell
7. Develop a program to Agglomerative Hierarchical clustering.
8. Develop a program to implement OPTICS algorithms.
9. Implement DBSCAN algorithm using appropriate Data sets.
10. Implement multiple data visualization methods using Polly

Course Outcomes

Upon successful completion of this course, students are able to

22AML54.1	Implement Hadoop-Map reduce concepts using python
22AML54.2	Implement spark concepts using python
22AML54.3	Apply Machine Learning algorithms to solve real-world problems.
22AML54.4	Implement Data visualization concepts using plotly

CO-PO Mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	3	1	1	3	-	-	-	-	1	-	1	-	2
CO2	3	3	1	1	3	-	-	-	-	1	-	1	-	2
CO3	3	3	1	1	3	-	-	-	-	1	-	1	-	2
CO4	3	3	1	1	3	-	-	-	-	1	-	1	-	2
AVG	**3**	**3**	**1**	**1**	**3**	**-**	**-**	**-**	**-**	**1**	**-**	**1**	**-**	**2**

Assessment Details (both CIE and SEE):

The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is50%. The student has to obtain a minimum of 40% marks individually both in CIE and SEE to pass. Practical Semester End Exam (SEE) is conducted for 100 marks (3 hours' duration).Based on this, grading will be awarded.

Continuous Internal Evaluation (CIE):

60% CIE marks awarded in case of practical shall be based on the weekly evaluation of laboratory journals/ reports after the conduction of every experiment and 40% marks for one practice test for practical-based learning.

Conduct of Practical Examination (SEE):

For **Laboratories** having only **one part:** Students are allowed to pick one experiment from the lot with equal opportunity. **Change of experiment** is allowed only once and marks allottedfor procedure to be **made zero** of the changed part only.

MARKS DISTRIBUTION

	a) Continuous Internal Evaluation (CIE)
1	**Observation + Record (Max. Marks 30) - (A)** Write-up + Execution + Viva Voce = 5+20+5 = 30 M
2	**Internal Assessment (Max. Marks 20) - (B)** Write-up + Execution + Viva Voce = 10+30+10 = 50 (scaled down to 20) **Total Marks = A + B = 50 Marks**
	b)Semester End Examination (SEE):
1	**Part A:** Write-up + Execution + Viva Voce = 15 + 70 + 15 = 100 Marks Total = 100 (scaled down to 50) **Total Marks = 50 Marks**

LAB EVALUATION PROCESS

Continuous Internal Evaluation (CIE)

Evaluation of CIE		
S. No	**Activity**	**Marks**
1	Average of Weekly Entries	30
2	Internal Assessment reduced to	20
TOTAL		**50**

Semester End Examination (SEE)

External Assessment Evaluation (End of Semester)		
S. No	**Activity**	**Marks**
1	Write-up + Execution + Viva Voce	100
TOTAL		**100**

LAB RUBRICS

Rubrics for Evaluation of Record and Observation

Attribute	Max. Marks	Good	Satisfactory	Poor
		4-5	2-3	0-1
Write-up	**05**	Writes the program without errors.	Writes the program with few mistakes.	Unable to write the program.
	Max. Marks	**10-20**	**5-9**	**0-4**
Execution of program	**20**	• Debugs the program independently. • Executed the program for all possible inputs.	• Works with little help from faculty. • All possible input cases not covered.	• Unable to complete the execution of the program within the lab session.
	Max. Marks	**3-5**	**1-2**	**0**
Viva Voce	**05**	• Able to explain the logic of the program. • Answered all questions.	• Partially understood the logic of the program. • Answered few questions.	• Not understood the logic of the program. • Not answering any questions

Rubrics for Evaluation of Internal

Test Part A

Attribute	Max Marks	Good	Satisfactory	Poor
		4-5	2-3	0-1
Write-up	**10**	• Completes source code. • No syntax and logical errors. • All possible inputs listed with expected output.	• Completes source code. • Syntax and logical errors exist. • All possible inputs listed with expected output.	• Incomplete source code. • Change of program.
	Max Marks	**21-30**	**6-20**	**1-5**
Execution	**30**	Able to debug the program and get the right set o outputs.	• Program, executed for only few input cases.	Not executed.
	Max Marks	**3-5**	**1-2**	**0**
Viva Voce	**10**	Answering all questions.	Answering few questions.	Not Able to answer basic questions.

Program 1

Implement map reduce for word count from a given input text file.

Step1. Type mapper.py and reducer.py using any editor

```
$sudo Nano mapper.py
```

mapper.py

```
#! /user/bin/envy
python
"""mapper.py"""

import sys
# input comes from STDIN (standard input)
for line in sys.stdin:
# remove leading and trailing whitespace
        line = line.strip()
# split the line into words
        words = line.split()
# increase counters
        for word in words:
# write the results to STDOUT (standard output);
# what we output here will be the input for the
# Reduce step, i.e. the input for reducer.py
#
# tab-delimited; the trivial word count is 1
                print('%s\t%s' % (word, 1))
```

reducer.py

```
#!/usr/bin/env python
"""reducer.py"""
from operator import itemgetter
import sys
current_word = None
current_count = 0
word = None
# input comes from STDIN
for line in sys.stdin:
# remove leading and trailing whitespace
        line = line.strip()
# parse the input we got from mapper.py
        word, count = line.split('\t', 1)
# convert count (currently a string) to int
        try:
                count = int(count)
        except ValueError:
# count was not a number, so silently
# ignore/discard this line
        continue
# this IF-switch only works because Hadoop sorts map output
# by key (here: word) before it is passed to the reducer
    if current_word == word:
        current_count += count
    else:
        if current_word:
# write result to STDOUT
            print('%s\t%s' % (current_word, current_count))
        current_count = count
        current_word = word
# do not forget to output the last word if needed!
if current_word == word:
    print('%s\t%s' % (current_word, current_count))
```

Step 2. Create input.txt file

```
$sudo nano input.txt
```

input.txt

Global Academy of Technology (GAT), established in the year 2001, is one of the most sought-after engineering and management colleges in Bengaluru, Karnataka. Located in a sprawling campus of 10-acre land, GAT is a campus ideal for students to hone their academics in an atmosphere of optimism. The institute is enabled with :

GAT provides ample opportunities for various co-curricular and extra-curricular activities for the students. The campus brims with more than 3500 students and 300 experienced staff involved in effective Teaching and Learning Process. Academics is supplemented with mentoring, peer learning and counselling to ensure holistic development of students. GAT has academic alliances with various institutions, industries, and research organizations to provide industry perspective to the students.

Step 3. Run the map-reduce word count program on UNIX platform

```
$cat input.txt | python3 mapper.py | sort | python3 reducer.py
```

Step 4. Run the map-reduce word count program on Hadoop

1. $sudo su hduser
2. $ start-dfs.sh
3. $start-yarn.sh
4. $jps
5. $hafs dfs -put /input.txt /
6. hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming 3.3.4.jar -files ./mapper.py,./reducer.py -mapper "python3 mapper.py" -reducer "python3 reducer.py" -input /5000-8.txt -output /0000009
7. Observe the result in localhost:port_number
8. Download outputfile and view it for wordcount

Output

```
300     1
3500    1
Academics       1
GAT     2
Learning        1
Process.        1
Teaching        1
The     1
academic        1
activities      1
alliances       1
ample   1
and     5
brims   1
campus          1
co-curricular   1
counselling     1
development     1
effective       1
ensure  1
experienced     1
extra-curricular        1
for     2
has     1
holistic1
in      1
industries,     1
industry        1
institutions,   1
involved        1
is      1
learning        1
mentoring,      1
more    1
of      1
opportunities   1
```

```
organizations   1
peer    1
perspective     1
provide         1
provides        1
research        1
staff   1
students        1
students.       3
supplemented    1
than    1
the     2
to      3
various2
with    3
```

Program 2

Execute a python program to implement map reduce concepts for printing average salary

mapper.py

```
#!/usr/bin/env python
import sys
# Input format: employee_id, name, salary
for line in sys.stdin:
# Split the input line into fields
            fields = line.strip().split(",")
# Check that the input line has the correct number of fields
            if len(fields) != 3:
                        continue
# Extract the salary from the input line
            salary = int(fields[2])
# Emit the salary as the output key and a count of 1 as the output value
            print("{0}\t{1}".format(salary, 1))
```

reducer.py

```
#!/usr/bin/env python
import sys
# Initialize variables to hold the sum of salaries and the count of employees
total_salary = 0
employee_count = 0
# Process input from mapper
for line in sys.stdin:
# Split the input line into key and value
            salary, count = line.strip().split('\t')
# Update the sum of salaries and the count of employees
            total_salary += int(salary) * int(count)
            employee_count += int(count)
# Calculate the average salary
 if employee_count > 0:
    average_salary = total_salary / employee_count # Output the result
    print("Average salary: {0}".format(average_salary))
```

input.txt

1	John Doe	50000
2	Jane Smith	60000
3	Bob Johnson	70000
4	Susan Williams	55000
5	Tom Davis	65000
6	Emily Brown	75000
7	Michael Lee	45000
8	Kelly Green	55000
9	David Kim	80000
10	Michelle Wong	90000

Output

```
Average salary: 64500.0
```

Program 3

Execute a Map reduce program for printing maximum salary for a given input file.

mapper.py

```
#!/usr/bin/env python
  import sys

  for line in sys.stdin:
     # Remove leading and trailing whitespace
     line = line.strip()
     # Split the line into columns
     columns = line.split()
     # Extract the salary column
     if len(columns)==3:
        salary = columns[2]
     # Output the salary with a null key
        print(f"None\t{salary}")
```

reducer.py

```
import sys

# Initialize the maximum salary to 0
max_salary = 0

# Iterate over each line in the input
for line in sys.stdin:
   # Split the line into key and value
   key, value = line.strip().split('\t', 1)

   # Convert the value to an integer
   try:
      salary = int(value)
   except ValueError:
      continue

   # Update the maximum salary if this salary is higher
   if salary > max_salary:
      max_salary = salary

# Output the final maximum salary
print(f"Max Salary: {max_salary}")
```

input.txt

```
Alice   Engineering    75000
Bob     Sales          60000
Charlie Engineering    85000
David Sales            45000
Eve     Marketing      90000
Frank Engineering      100000
```

Output

```
Max Salary: 100000
```

Program 4

Execute a Map reduce program for printing sales

mapper.py

```
#!/usr/bin/env python
import sys
import csv

for line in csv.reader(iter(sys.stdin.readline, '')):
    if len(line) >= 3:  # Make sure we have enough columns
        year = line[0]
        sales = line[2]  # Get just the sales amount
        try:
            # Verify it's a valid number
            sales_value = int(sales)
            print(f"{year}\t{sales_value}")
```

reducer.py

```
#!/usr/bin/env python
import sys

current_year = None
total_sales = 0

for line in sys.stdin:
    # Split the input into year and sales
    year, sales = line.strip().split('\t')

    if current_year is None:
        current_year = year

    if year == current_year:
        total_sales += int(sales)
    else:
        # Print total for the year and reset for next year
        print(f"{current_year}\t{total_sales}")
        current_year = year
        total_sales = int(sales)

# Print the last year's total
if current_year is not None:
    print(f"{current_year}\t{total_sales}")
         +
```

input.txt

2018	Jan	10000
2018	Feb	12000
2018	Mar	15000
2019	Jan	20000
2019	Feb	22000
2019	Mar	25000
2020	Jan	30000
2020	Feb	32000
2020	Mar	35000
2018	gyu	500000
2019	uyt	750000

Output

```
2018  37000

2019  67000

2020  97000
```

Program 5

5. Execute a python program to implement inverted index

Mapper.py

```
import sys

def mapper():
    for line in sys.stdin:
        line = line.strip()
        document, words = line.split(" ", 1)
        word_list = words.split()
        for word in word_list:
            print(f"{word}\t{document}")

if _name___== "_main_":
    mapper()
```

Reducer.py

```
import sys

def reducer():
    current_word = None
    doc_list = []

    for line in sys.stdin:
        line = line.strip()
        word, document = line.split("\t", 1)

        if current_word == word:
            doc_list.append(document)
        else:
            if current_word:
                print(f"{current_word}\t{', '.join(set(doc_list))}")
            doc_list = [document]
            current_word = word

    if current_word:
        print(f"{current_word}\t{', '.join(set(doc_list))}")

if _name___== "_main_":
    reducer()
```

Input.txt

```
doc1 apple banana
doc2 banana orange
doc3 apple mango
doc4 banana apple
```

Output

```
apple: doc1, doc3, doc4
banana: doc1, doc2, doc4
orange: doc2
mango: doc3
```

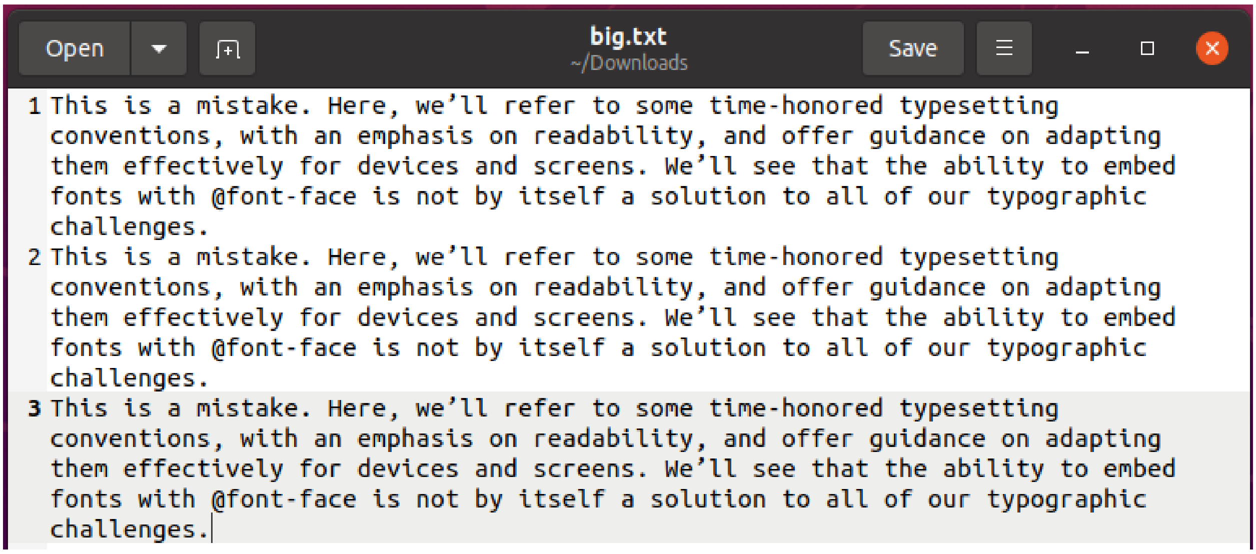
Program 6

Execute python program to implement word count program using spark shell.

Steps are used to learn how to perform wordcount using spark.

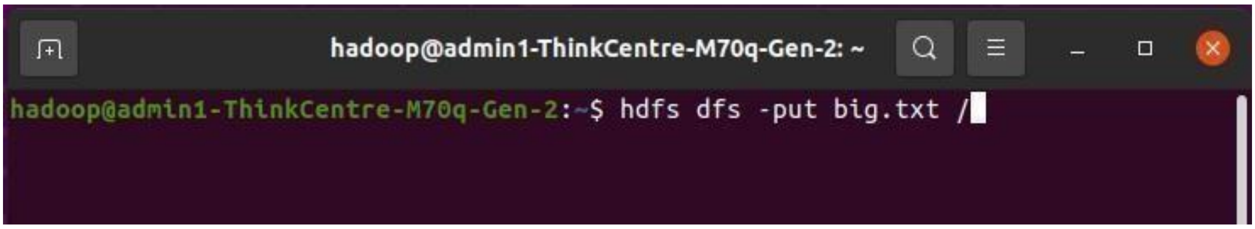
Step 1: Create a Text File and move dfs

```
gedit big.txt
```



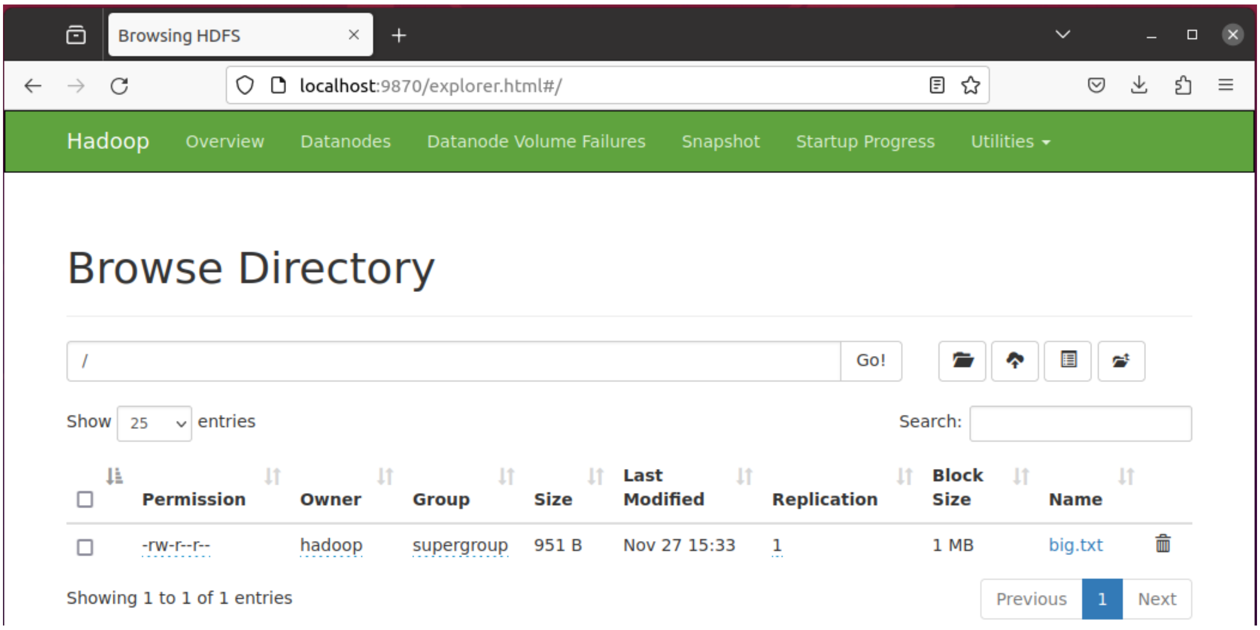
```
1 This is a mistake. Here, we'll refer to some time-honored typesetting
conventions, with an emphasis on readability, and offer guidance on adapting
them effectively for devices and screens. We'll see that the ability to embed
fonts with @font-face is not by itself a solution to all of our typographic
challenges.
2 This is a mistake. Here, we'll refer to some time-honored typesetting
conventions, with an emphasis on readability, and offer guidance on adapting
them effectively for devices and screens. We'll see that the ability to embed
fonts with @font-face is not by itself a solution to all of our typographic
challenges.
3 This is a mistake. Here, we'll refer to some time-honored typesetting
conventions, with an emphasis on readability, and offer guidance on adapting
them effectively for devices and screens. We'll see that the ability to embed
fonts with @font-face is not by itself a solution to all of our typographic
challenges.
```

```
hdfs dfs -put big.txt
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -put big.txt /
```

```
localhost:9870
```



Browse Directory

/

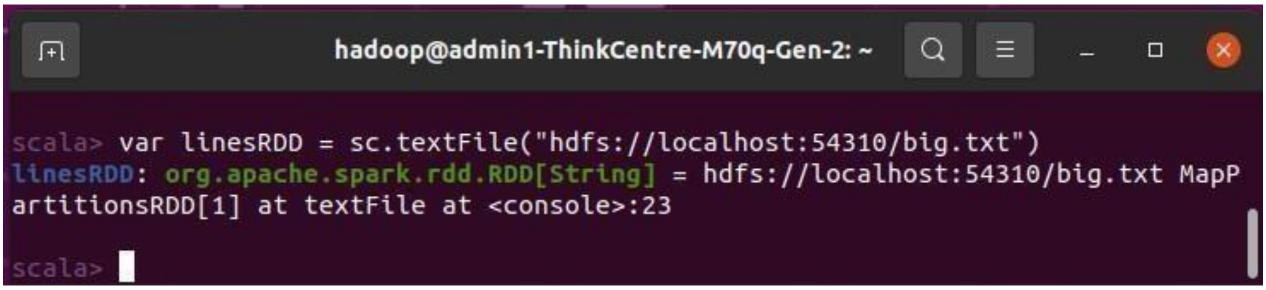
Show 25 entries   Search:

☐	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
☐	-rw-r--r--	hadoop	supergroup	951 B	Nov 27 15:33	1	1 MB	big.txt

Showing 1 to 1 of 1 entries

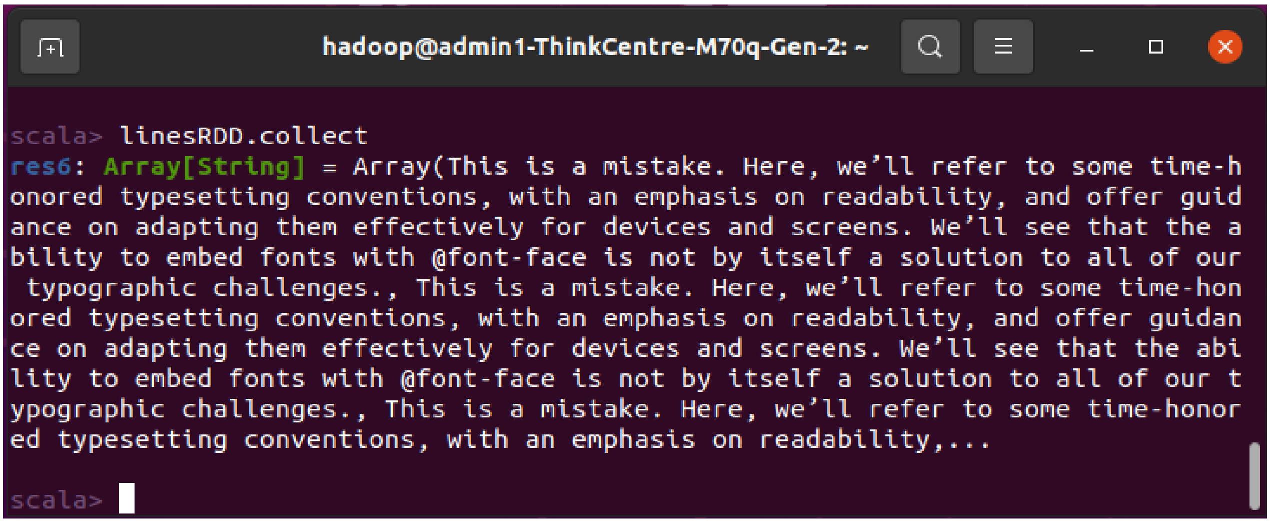
Step 2: Create RDD from a file in HDFS, type the following on spark-shell, and press enter:

```
var linesRDD = sc.textFile("hdfs://localhost:54310/big.txt")
```



```
scala> var linesRDD = sc.textFile("hdfs://localhost:54310/big.txt")
linesRDD: org.apache.spark.rdd.RDD[String] = hdfs://localhost:54310/big.txt MapP
artitionsRDD[1] at textFile at <console>:23

scala> 
```

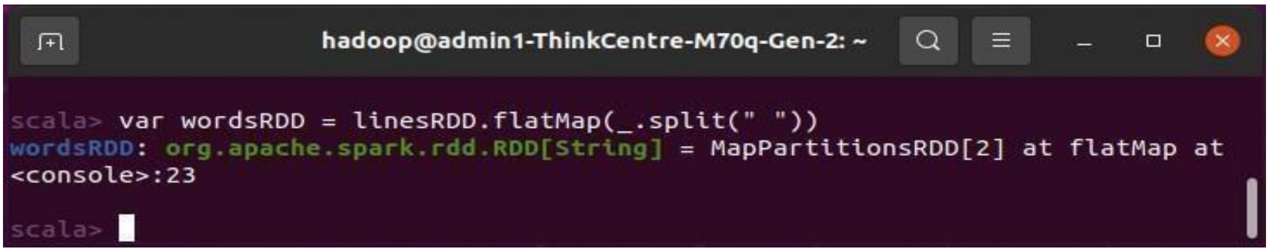
```
linesRDD.collect
```



```
scala> linesRDD.collect
res6: Array[String] = Array(This is a mistake. Here, we’ll refer to some time-h
onored typesetting conventions, with an emphasis on readability, and offer guid
ance on adapting them effectively for devices and screens. We’ll see that the a
bility to embed fonts with @font-face is not by itself a solution to all of our
 typographic challenges., This is a mistake. Here, we’ll refer to some time-hon
ored typesetting conventions, with an emphasis on readability, and offer guidan
ce on adapting them effectively for devices and screens. We’ll see that the abi
lity to embed fonts with @font-face is not by itself a solution to all of our t
ypographic challenges., This is a mistake. Here, we’ll refer to some time-honor
ed typesetting conventions, with an emphasis on readability,...

scala> 
```

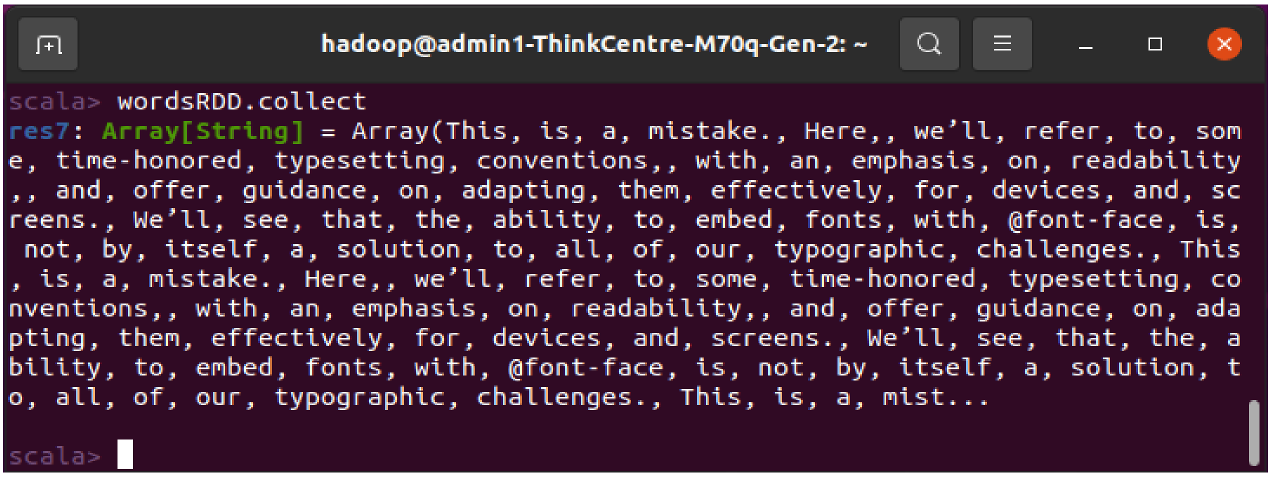
Step 3: Convert each record into word

```
var wordsRDD = linesRDD.flatMap(_.split(" "))
```



```
scala> var wordsRDD = linesRDD.flatMap(_.split(" "))
wordsRDD: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[2] at flatMap at
<console>:23

scala> 
```

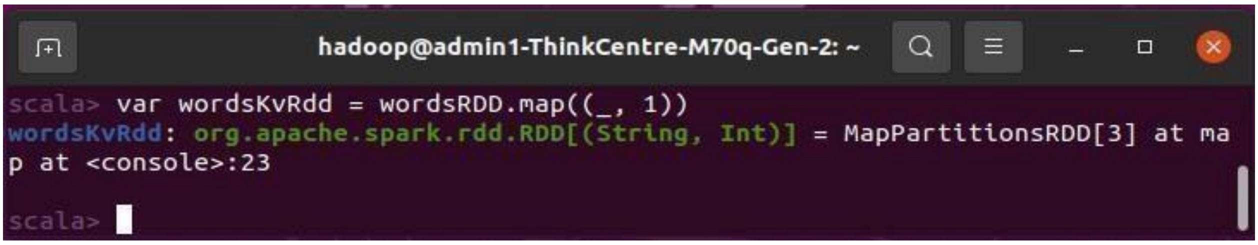
wordsRDD.collect



```
scala> wordsRDD.collect
res7: Array[String] = Array(This, is, a, mistake., Here,, we'll, refer, to, som
e, time-honored, typesetting, conventions,, with, an, emphasis, on, readability
,, and, offer, guidance, on, adapting, them, effectively, for, devices, and, sc
reens., We'll, see, that, the, ability, to, embed, fonts, with, @font-face, is,
 not, by, itself, a, solution, to, all, of, our, typographic, challenges., This
, is, a, mistake., Here,, we'll, refer, to, some, time-honored, typesetting, co
nventions,, with, an, emphasis, on, readability,, and, offer, guidance, on, ada
pting, them, effectively, for, devices, and, screens., We'll, see, that, the, a
bility, to, embed, fonts, with, @font-face, is, not, by, itself, a, solution, t
o, all, of, our, typographic, challenges., This, is, a, mist...

scala> 
```

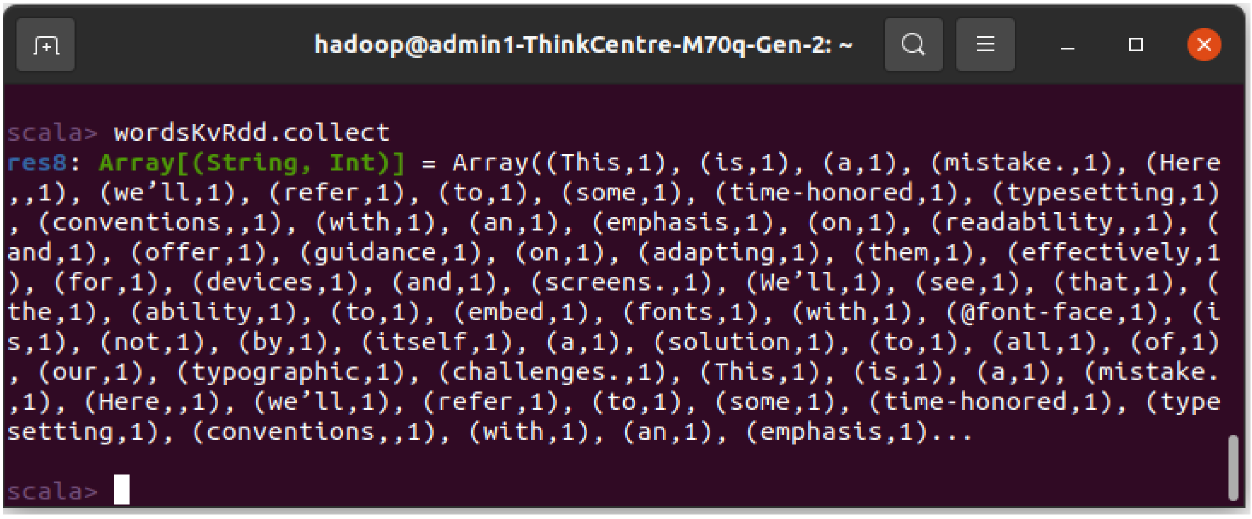
Step 4: Convert each word into key-value pair.

var wordsKvRdd = wordsRDD.map((_, 1))



```
scala> var wordsKvRdd = wordsRDD.map((_, 1))
wordsKvRdd: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[3] at ma
p at <console>:23

scala> 
```

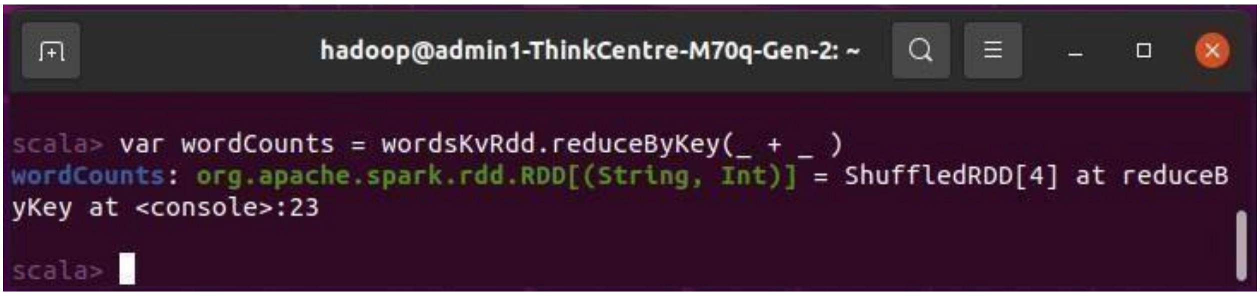
wordsKvRdd.collect



```
scala> wordsKvRdd.collect
res8: Array[(String, Int)] = Array((This,1), (is,1), (a,1), (mistake.,1), (Here
,,1), (we'll,1), (refer,1), (to,1), (some,1), (time-honored,1), (typesetting,1)
, (conventions,,1), (with,1), (an,1), (emphasis,1), (on,1), (readability,,1), (
and,1), (offer,1), (guidance,1), (on,1), (adapting,1), (them,1), (effectively,1
), (for,1), (devices,1), (and,1), (screens.,1), (We'll,1), (see,1), (that,1), (
the,1), (ability,1), (to,1), (embed,1), (fonts,1), (with,1), (@font-face,1), (i
s,1), (not,1), (by,1), (itself,1), (a,1), (solution,1), (to,1), (all,1), (of,1)
, (our,1), (typographic,1), (challenges.,1), (This,1), (is,1), (a,1), (mistake.
,1), (Here,,1), (we'll,1), (refer,1), (to,1), (some,1), (time-honored,1), (type
setting,1), (conventions,,1), (with,1), (an,1), (emphasis,1)...

scala> 
```

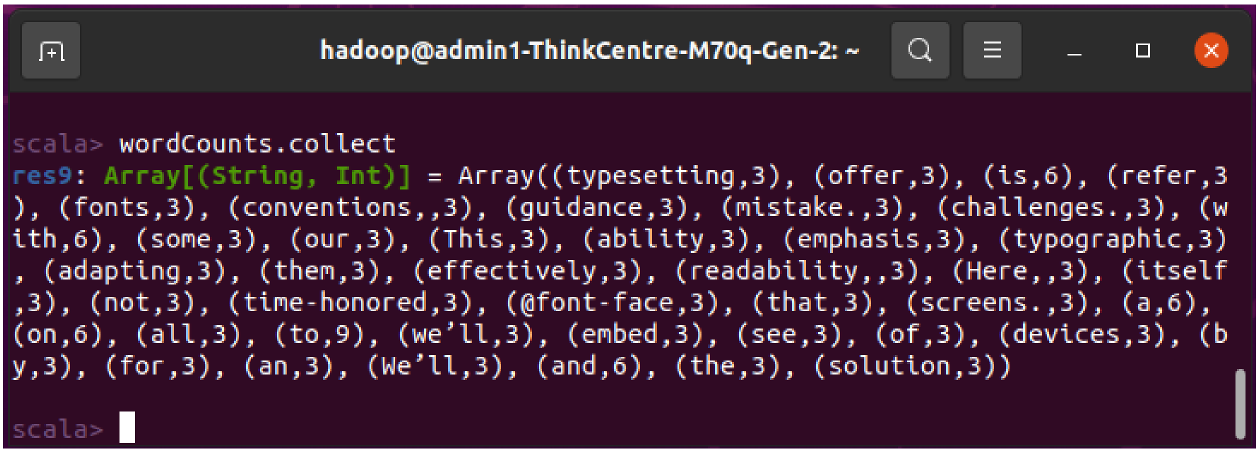
Step 5: Group By key and perform aggregation on each key:

```
var wordCounts = wordsKvRdd.reduceByKey(_ + _ )
```



```
scala> var wordCounts = wordsKvRdd.reduceByKey(_ + _ )
wordCounts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceB
yKey at <console>:23

scala> 
```

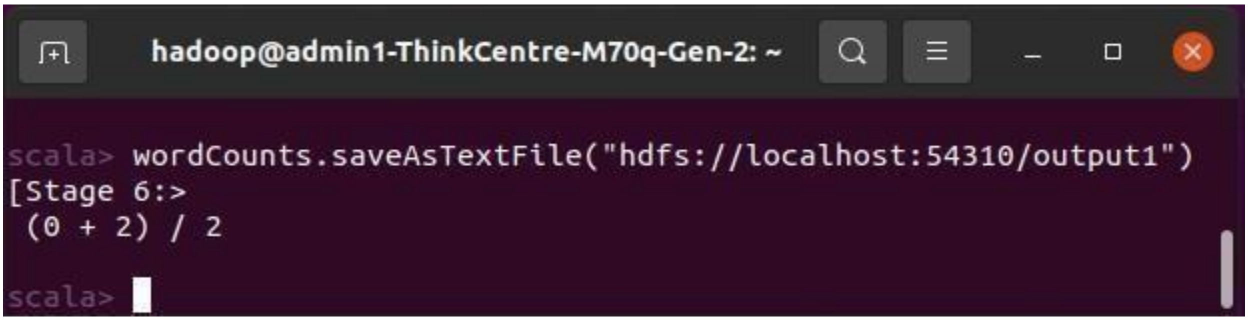
```
wordCounts.collect
```



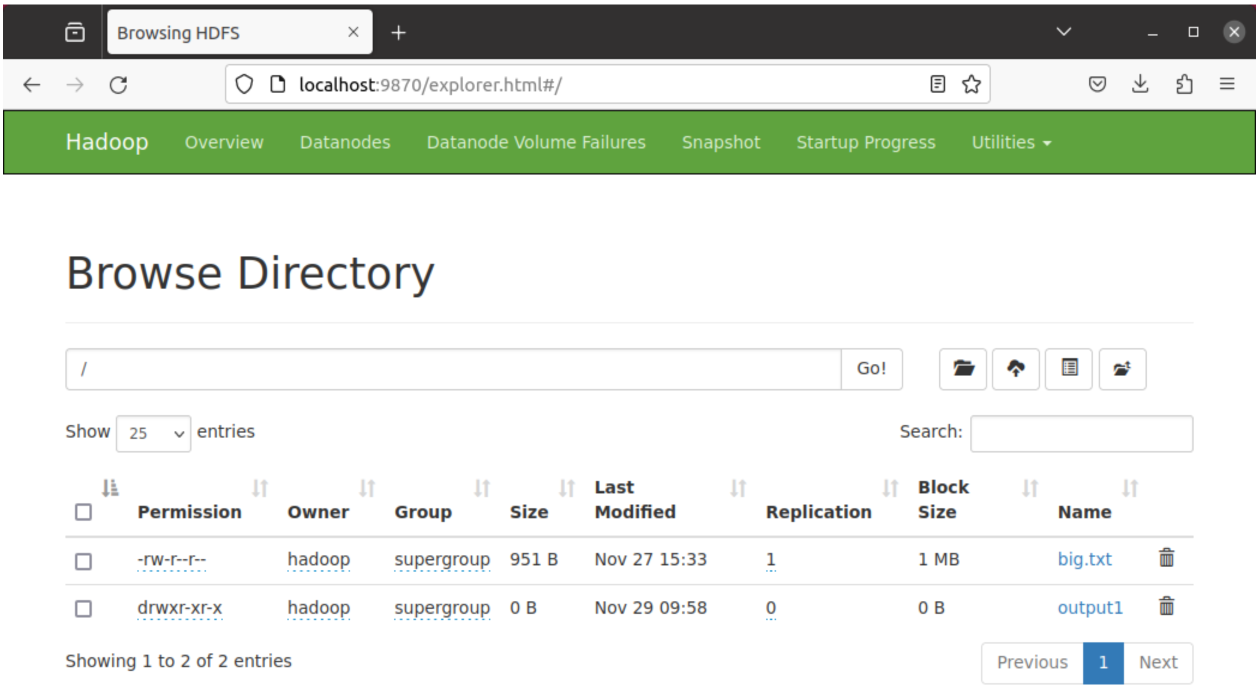
```
scala> wordCounts.collect
res9: Array[(String, Int)] = Array((typesetting,3), (offer,3), (is,6), (refer,3
), (fonts,3), (conventions,,3), (guidance,3), (mistake.,3), (challenges.,3), (w
ith,6), (some,3), (our,3), (This,3), (ability,3), (emphasis,3), (typographic,3)
, (adapting,3), (them,3), (effectively,3), (readability,,3), (Here,,3), (itself
,3), (not,3), (time-honored,3), (@font-face,3), (that,3), (screens.,3), (a,6),
(on,6), (all,3), (to,9), (we’ll,3), (embed,3), (see,3), (of,3), (devices,3), (b
y,3), (for,3), (an,3), (We’ll,3), (and,6), (the,3), (solution,3))

scala> 
```

Step 6: Save the results into HDFS:

```
wordCounts.saveAsTextFile("hdfs://localhost:54310/output1")
```



```
scala> wordCounts.saveAsTextFile("hdfs://localhost:54310/output1")
[Stage 6:>
 (0 + 2) / 2

scala> 
```



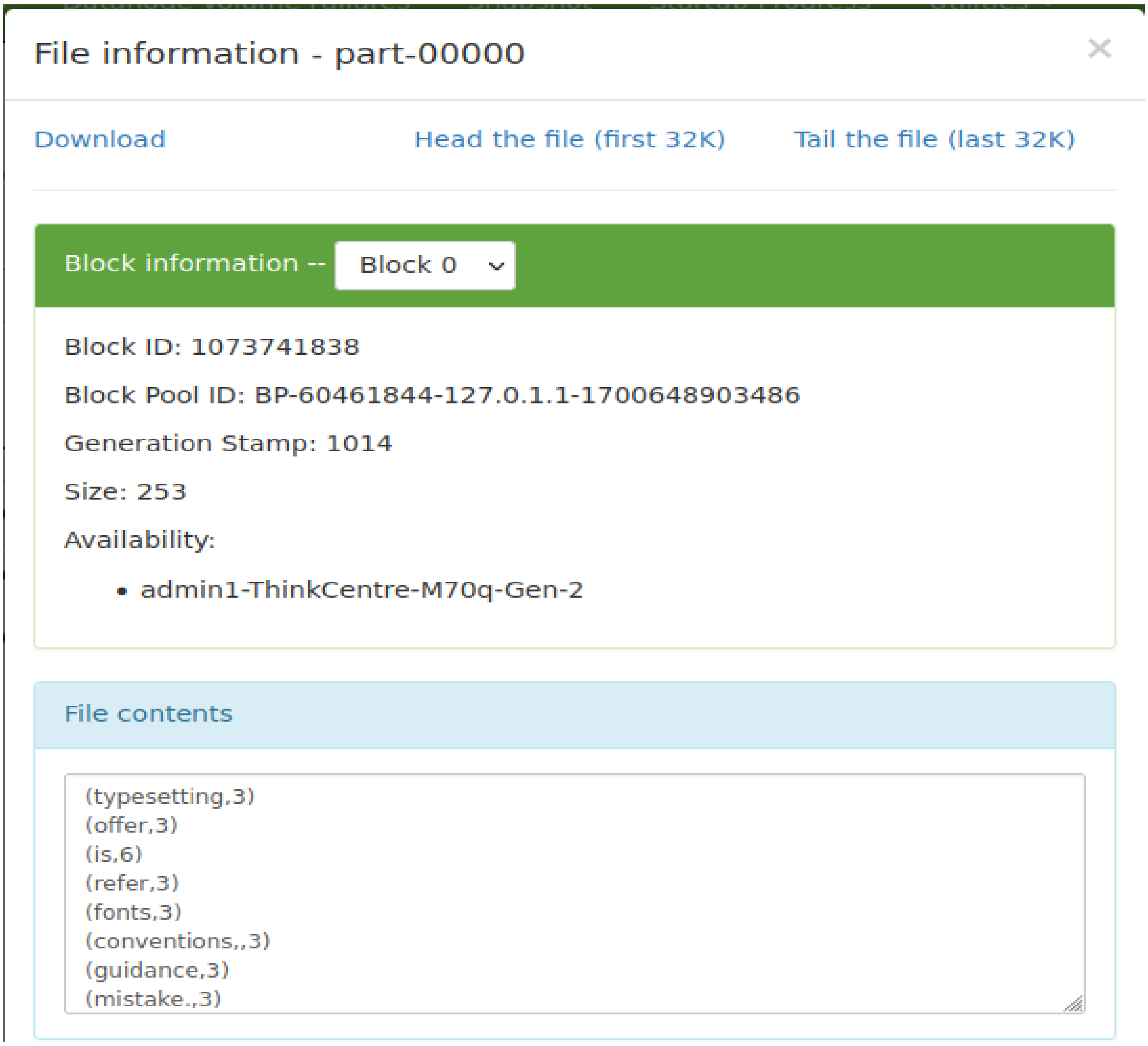
Browse Directory

/ Go!

Show 25 entries Search:

□	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	-rw-r--r--	hadoop	supergroup	951 B	Nov 27 15:33	1	1 MB	big.txt
□	drwxr-xr-x	hadoop	supergroup	0 B	Nov 29 09:58	0	0 B	output1

Showing 1 to 2 of 2 entries



File information - part-00000

Block information -- Block 0

Block ID: 1073741838

Block Pool ID: BP-60461844-127.0.1.1-1700648903486

Generation Stamp: 1014

Size: 253

Availability:

- admin1-ThinkCentre-M70q-Gen-2

File contents

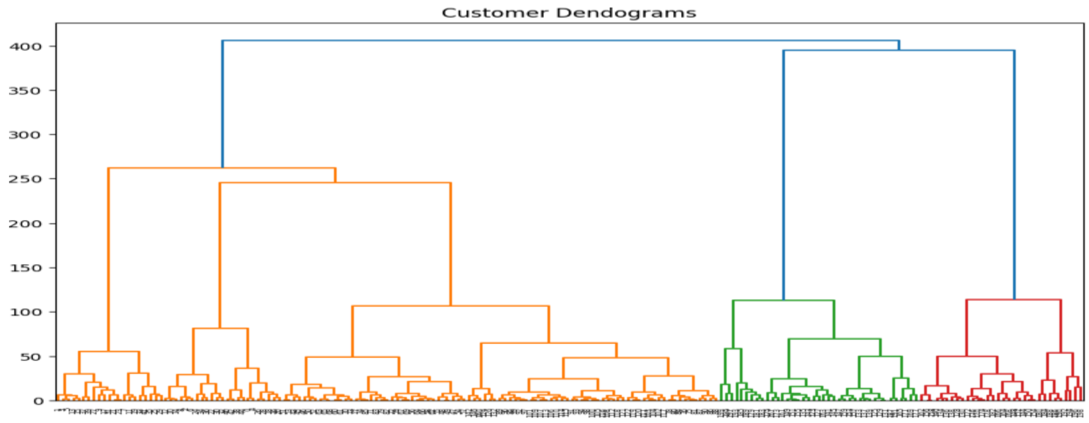
```
(typesetting,3)
(offer,3)
(is,6)
(refer,3)
(fonts,3)
(conventions,,3)
(guidance,3)
(mistake.,3)
```

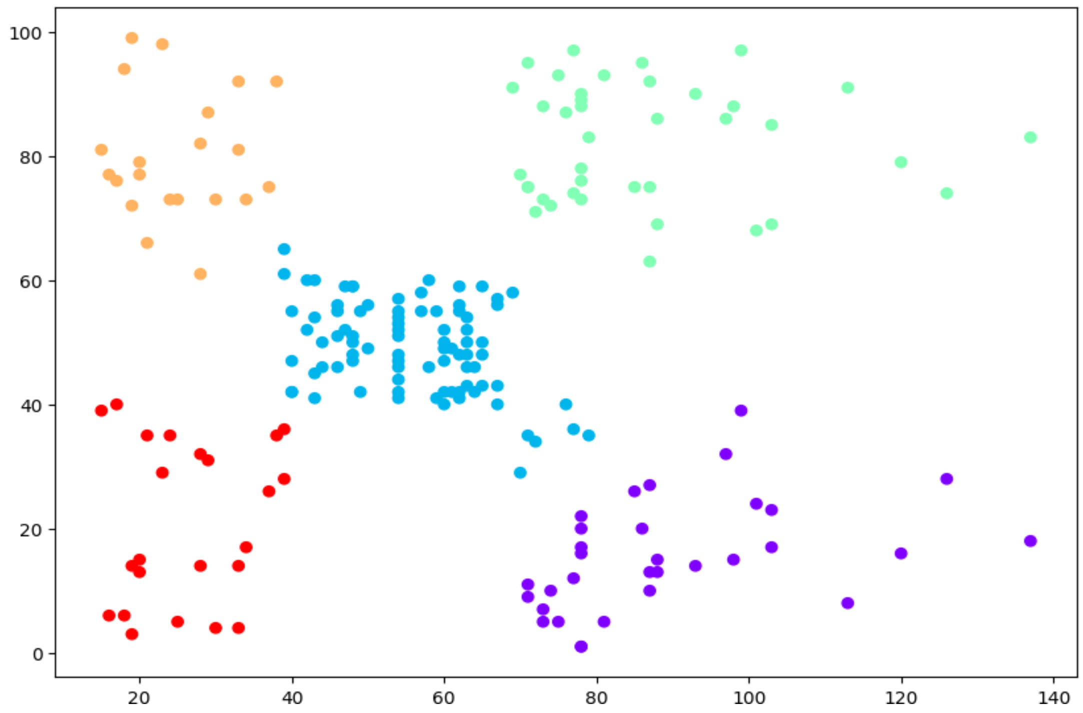
Program 7

Develop a program to Agglomerative Hierarchical clustering.

```
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline
import numpy as np

customer_data = pd.read_csv('hierarchical-clustering-with-python-and-scikit-learn-shopping-
data.csv')

customer_data.shape
customer_data.head()

data = customer_data.iloc[:, 3:5].values

data

import scipy.cluster.hierarchy as shc

plt.figure(figsize=(10, 7))
plt.title("Customer Dendograms")
dend = shc.dendrogram(shc.linkage(data, method='ward'))

from sklearn.cluster import AgglomerativeClustering

cluster = AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='ward')
labels_=cluster.fit_predict(data)

labels_

plt.figure(figsize=(10, 7))
plt.scatter(data[:,0], data[:,1], c=cluster.labels_, cmap='rainbow')
```

Output



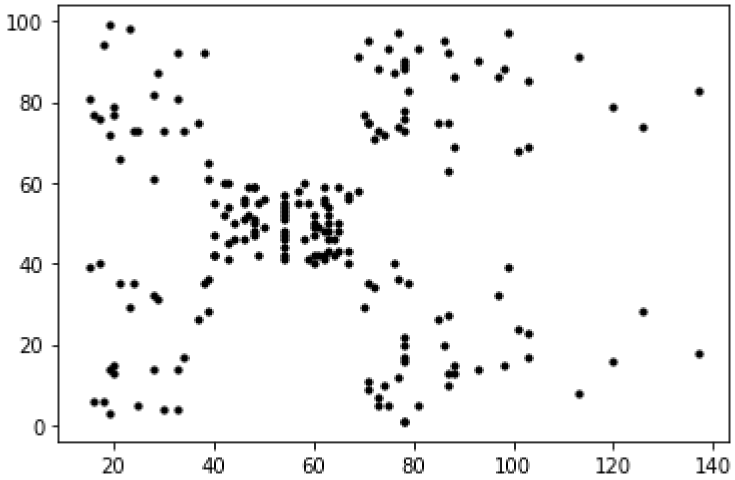

0
20
40
60
80
100
20
40
60
80
100
120
140

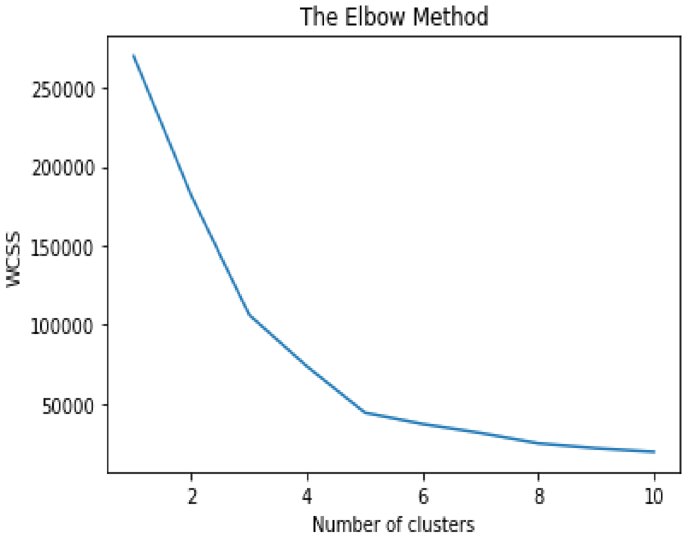
Program 8

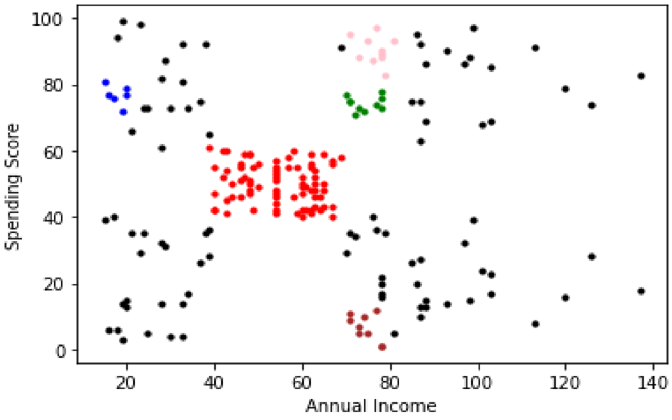
Implement DBSCAN algorithm using appropriate Data sets.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("Mall_customers.csv")

df.head()

df.tail()

df.shape

df = df.iloc[:, [3,4]].values

df

plt.scatter(df[:,0], df[:,1], s=10, c= "black")

from sklearn.cluster import KMeans

wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters= i,
    init = 'k-means++', max_iter= 300, n_init= 10)
    kmeans.fit(df)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11), wcss)
plt.title("The Elbow Method")
plt.xlabel("Number of clusters")
plt.ylabel("WCSS")
plt.show()

from sklearn.cluster import DBSCAN

dbscan = DBSCAN(eps=5, min_samples=5)

labels = dbscan.fit_predict(df)

np.unique(labels)

# Visualising the clusters
plt.scatter(df[labels == -1, 0], df[labels == -1, 1], s = 10, c = 'black')
plt.scatter(df[labels == 0, 0], df[labels == 0, 1], s = 10, c = 'blue')
plt.scatter(df[labels == 1, 0], df[labels == 1, 1], s = 10, c = 'red')
plt.scatter(df[labels == 2, 0], df[labels == 2, 1], s = 10, c = 'green')
plt.scatter(df[labels == 3, 0], df[labels == 3, 1], s = 10, c = 'brown')
plt.scatter(df[labels == 4, 0], df[labels == 4, 1], s = 10, c = 'pink')
plt.scatter(df[labels == 5, 0], df[labels == 5, 1], s = 10, c = 'yellow')
plt.scatter(df[labels == 6, 0], df[labels == 6, 1], s = 10, c = 'silver')
plt.xlabel('Annual Income')
plt.ylabel('Spending Score')
plt.show()
```

Output:







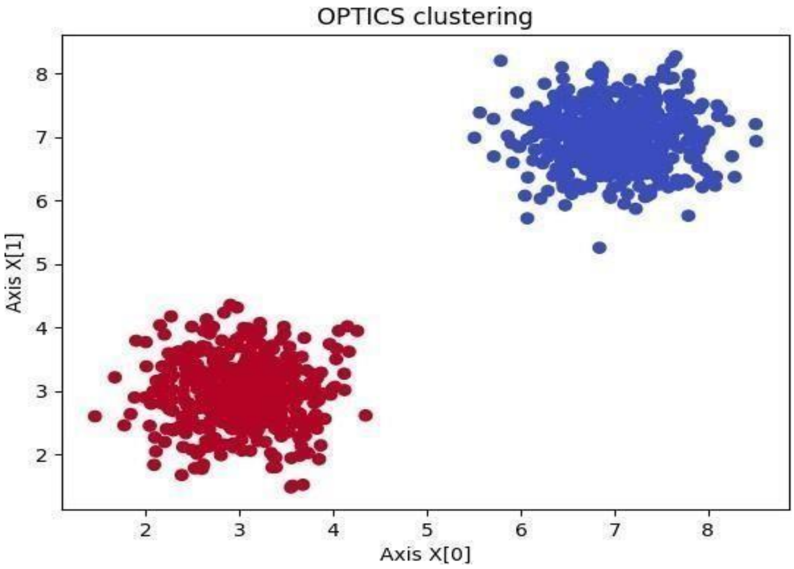
Program 9

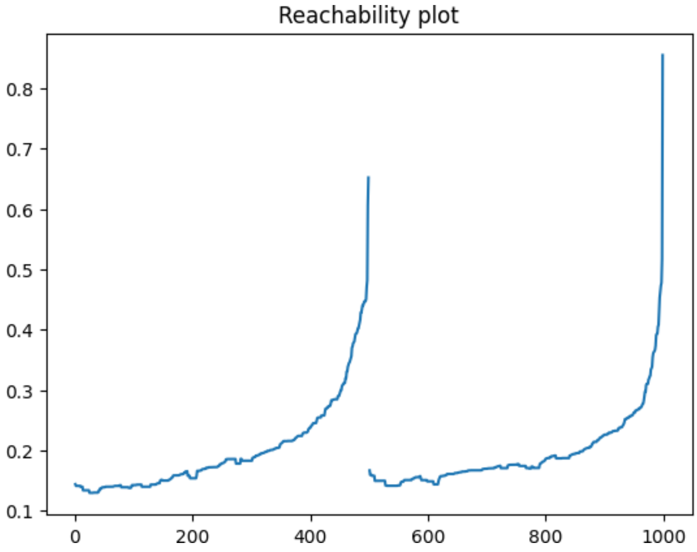
Implement OPTICS algorithm using appropriate Data sets.

```
from sklearn.datasets import make_blobsfrom
sklearn.cluster import OPTICS import numpy
as np
import matplotlib.pyplot as plt

# Configuration options num_samples_total
= 1000
cluster_centers = [(3,3), (7,7)] num_classes
= len(cluster_centers)epsilon = 2.0
min_samples = 22
cluster_method = 'xi'
metric = 'minkowski'

# Generate data
X, y = make_blobs(n_samples = num_samples_total, centers = cluster_centers,n_features =
num_classes, center_box=(0, 1), cluster_std = 0.5)

# Compute OPTICS
db = OPTICS(max_eps=epsilon, min_samples=min_samples,
cluster_method=cluster_method, metric=metric).fit(X)
labels = db.labels_

no_clusters = len(np.unique(labels) )
no_noise = np.sum(np.array(labels) == -1, axis=0)

print('Estimated no. of clusters: %d' % no_clusters)
print('Estimated no. of noise points: %d' % no_noise)

# Generate scatter plot for training data
colors = list(map(lambda x: '#3b4cc0' if x == 1 else '#b40426', labels))
plt.scatter(X[:,0], X[:,1], c=colors, marker="o", picker=True) plt.title(f'OPTICS
clustering')
plt.xlabel('Axis X[0]')
plt.ylabel('Axis X[1]')
plt.show()


#Generate reachability plot
reachability = db.reachability_[db.ordering_]
plt.plot(reachability)
plt.title('Reachability plot')
plt.show()
```

Output:

Estimated no. of clusters: 3
Estimated no. of noise points: 30





Program 10

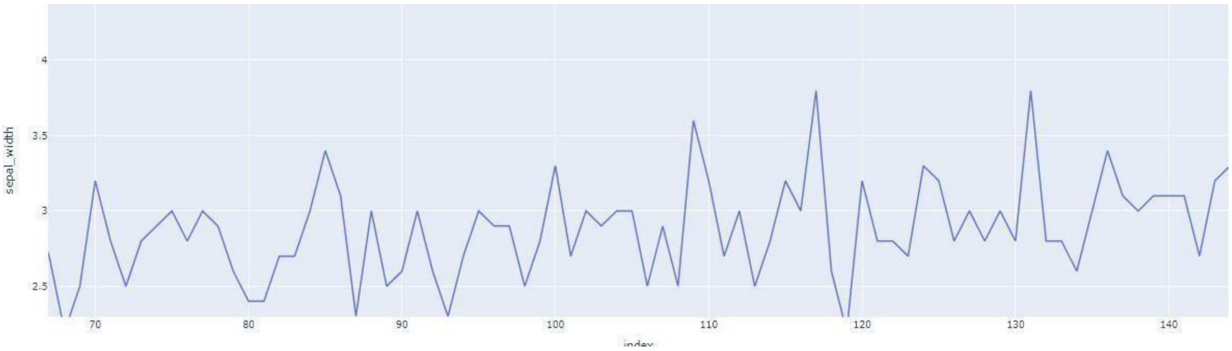
Implement data visualization using Plotly.

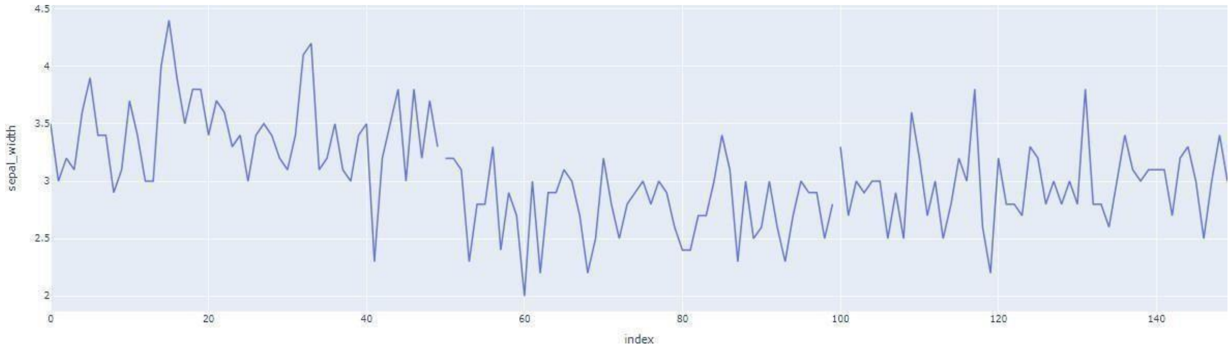
Plotly is an open-source module of Python which is used for data visualization and supports various graphs like line charts, scatter plots, bar charts, histograms, area plot, etc. In this article, we will see how to plot a basic chart with plotly and also how to make a plot interactive. But beforestarting you might be wondering why there is a need to learn plotly, so let’s have a look atit.

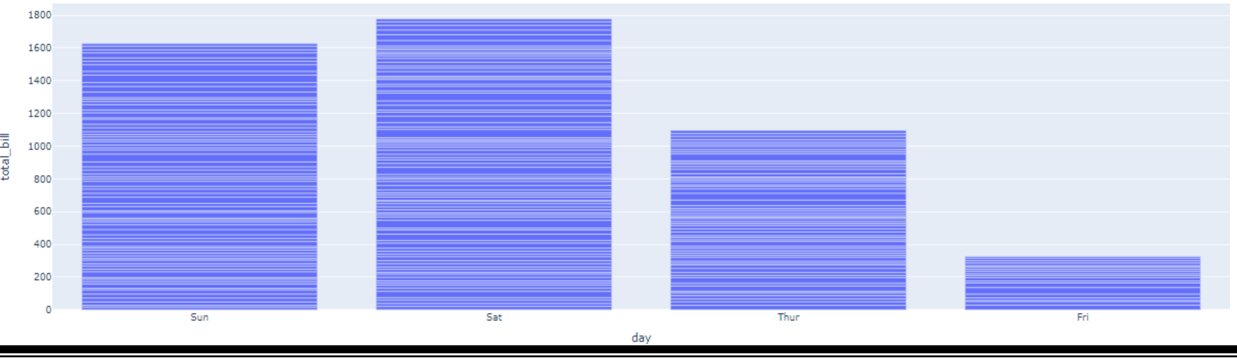
```
import plotly.express as px
# using the iris dataset
df = px.data.iris()

# plotting the line chart
fig = px.line(df, y="sepal_width",)
# showing the plot
fig.show()
# plotting the line chart
fig = px.line(df, y="sepal_width", line_group='species')

# showing the plot
fig.show()

df = px.data.tips()
# Creating the bar chart
fig = px.bar(df, x='day', y="total_bill")
fig.show()
# using the dataset
df = px.data.tips()
 # plotting the histogram
fig = px.histogram(df, x="total_bill")
# showing the plot
fig.show()
df = px.data.tips()
# plotting the histogram
fig = px.histogram(df, x="total_bill", color='sex', nbins=50, histnorm='percent',barmode='overlay')

# showing the plot
fig.show()
df = px.data.tips()
fig = px.pie(df, values="total_bill", names="day")
fig.show()
df = px.data.tips()
# plotting the boxplot
```

```
fig = px.box(df, x="day", y="tip")
# showing the plot
fig.show()
# using the dataset
df = px.data.tips()
# plotting the violin plot
fig = px.violin(df, x="day", y="tip")
# showing the plot
fig.show()
# data to be plotted
df =  px.data.tips()#
plotting the figure
fig = px.scatter_3d(df, x="total_bill", y="sex", z="tip")
fig.show()
```

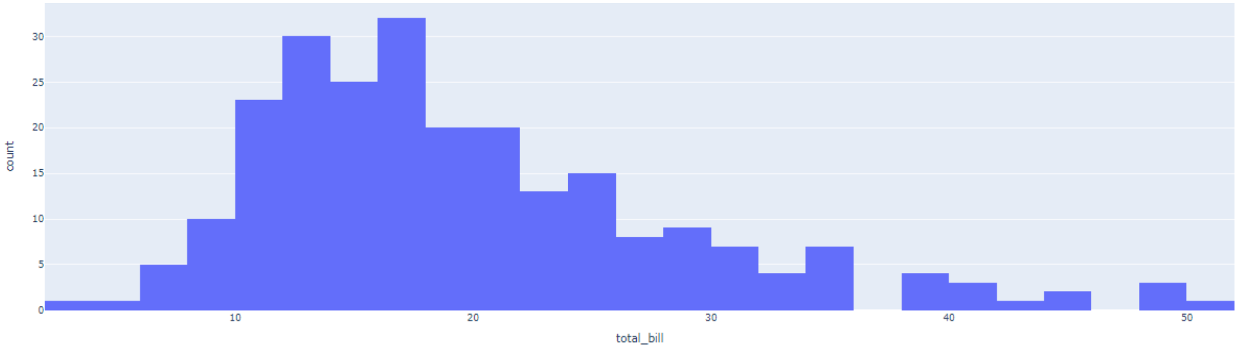
Output:

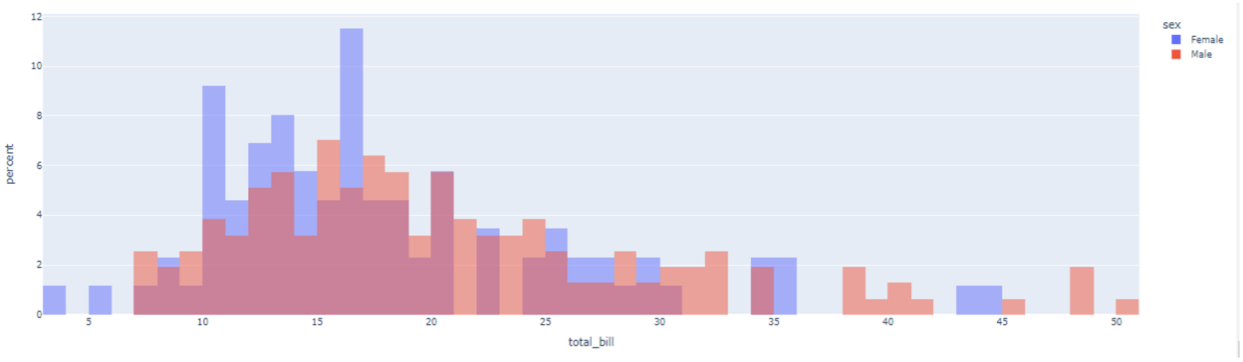


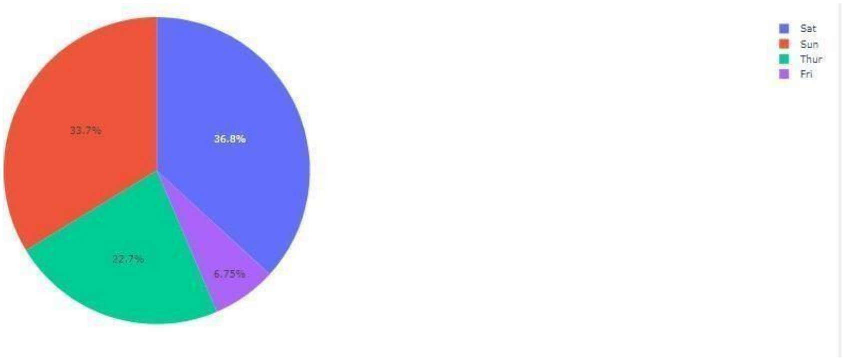


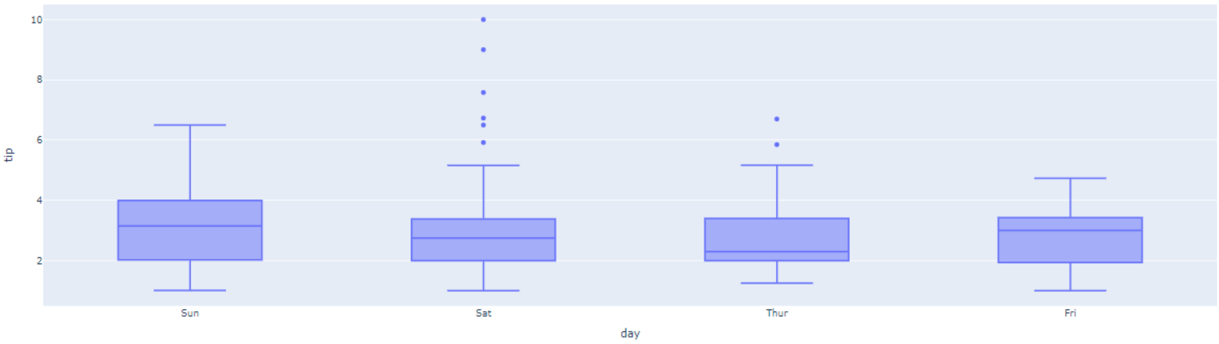


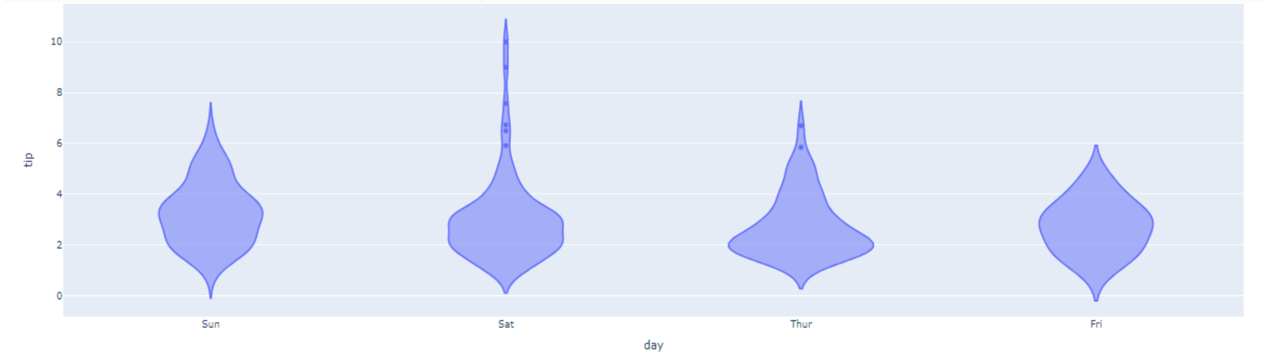
I

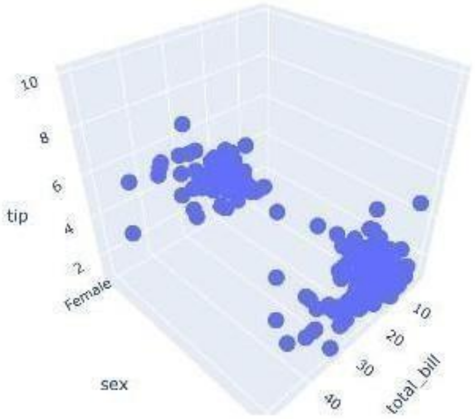

count
0
5
10
15
20
25
30
10
20
30
40
50
total_bill


percent
0
2
4
6
8
10
12
5
10
15
20
25
30
35
40
45
50
total_bill
sex
Female
Male


33.7%
36.8%
22.7%
6.75%
Sat
Sun
Thur
Fri


tip
2
4
6
8
10
Sun
Sat
Thur
Fri
day


10
8
6
4
2
0
tip
Sun
Sat
Thur
Fri
day


10
8
6
tip
4
2
Female
sex
10
20
30
40
total_bill

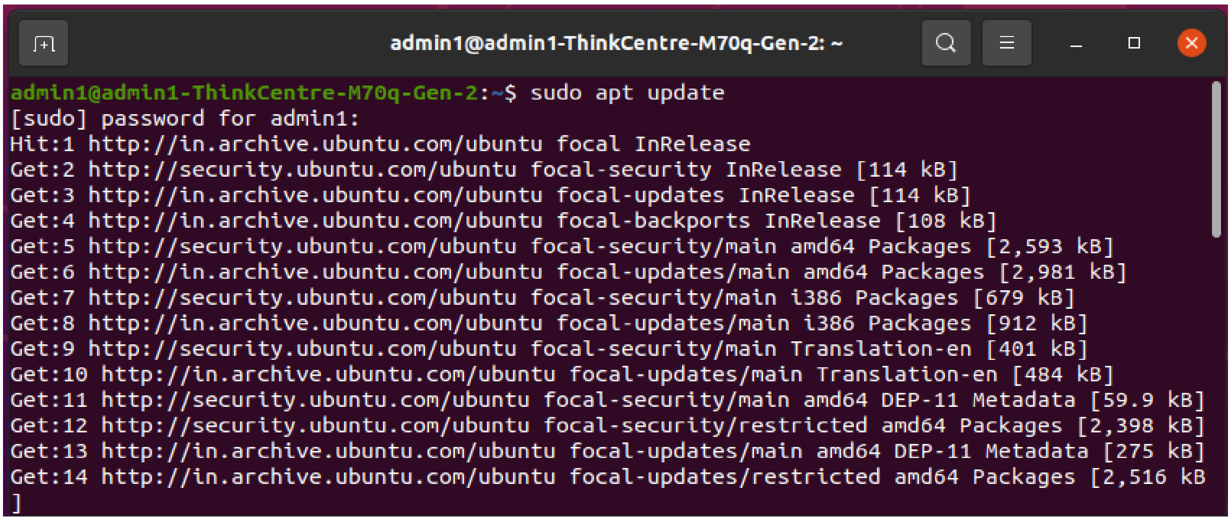
Installation steps of Hadoop and Spark

Demo 1 – Hadoop Installation

StepstoInstallHadoop3and Configureasinglenodecluster on Ubuntu 18 0r 20

1. Use the following command to update your system before initiating a new installation.

```
sudo apt update
```

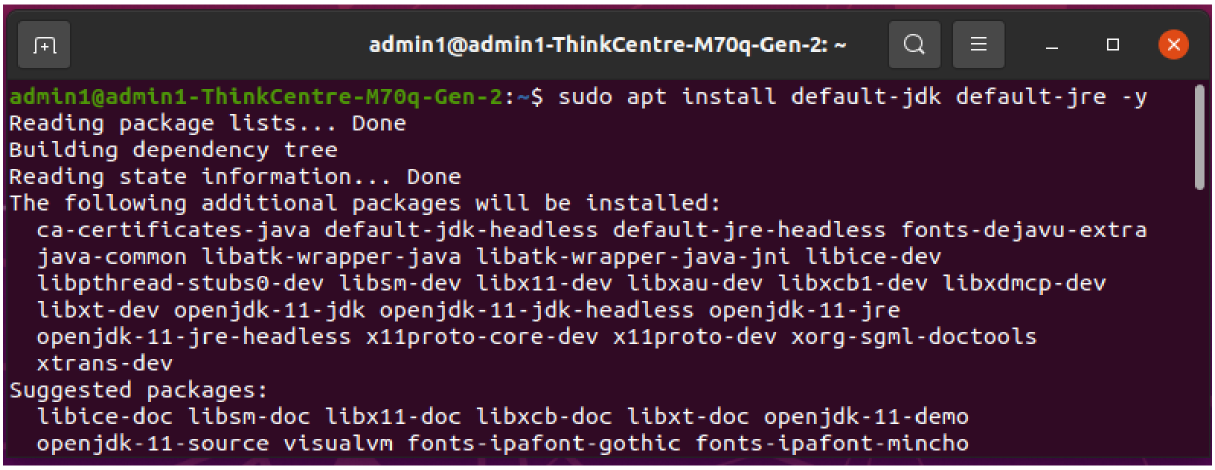


```
admin1@admin1-ThinkCentre-M70q-Gen-2:~$ sudo apt update
[sudo] password for admin1:
Hit:1 http://in.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [2,593 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [2,981 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/main i386 Packages [679 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [912 kB]
Get:9 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [401 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [484 kB]
Get:11 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [59.9 kB]
Get:12 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [2,398 kB]
Get:13 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metadata [275 kB]
Get:14 http://in.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [2,516 kB
]
```

Install and configure the Oracle JDK

2. The Hadoop framework is written in Java and its services require a compatible Java Runtime Environment (JRE) and Java Development Kit (JDK).

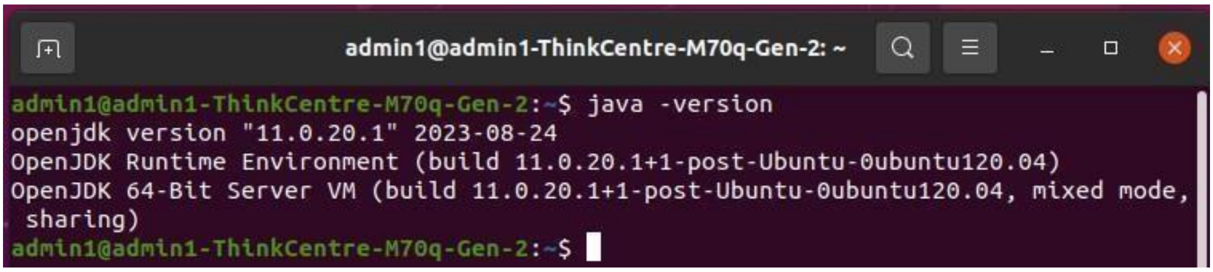
```
sudo apt install default -jdk default -jre -y
```



```
admin1@admin1-ThinkCentre-M70q-Gen-2:~$ sudo apt install default-jdk default-jre -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java default-jdk-headless default-jre-headless fonts-dejavu-extra
  java-common libatk-wrapper-java libatk-wrapper-java-jni libice-dev
  libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev libxdmcp-dev
  libxt-dev openjdk-11-jdk openjdk-11-jdk-headless openjdk-11-jre
  openjdk-11-jre-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools
  xtrans-dev
Suggested packages:
  libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-11-demo
  openjdk-11-source visualvm fonts-ipafont-gothic fonts-ipafont-mincho
```

3. Once the installation process is complete, verify the current Java version .

```
java -version
```



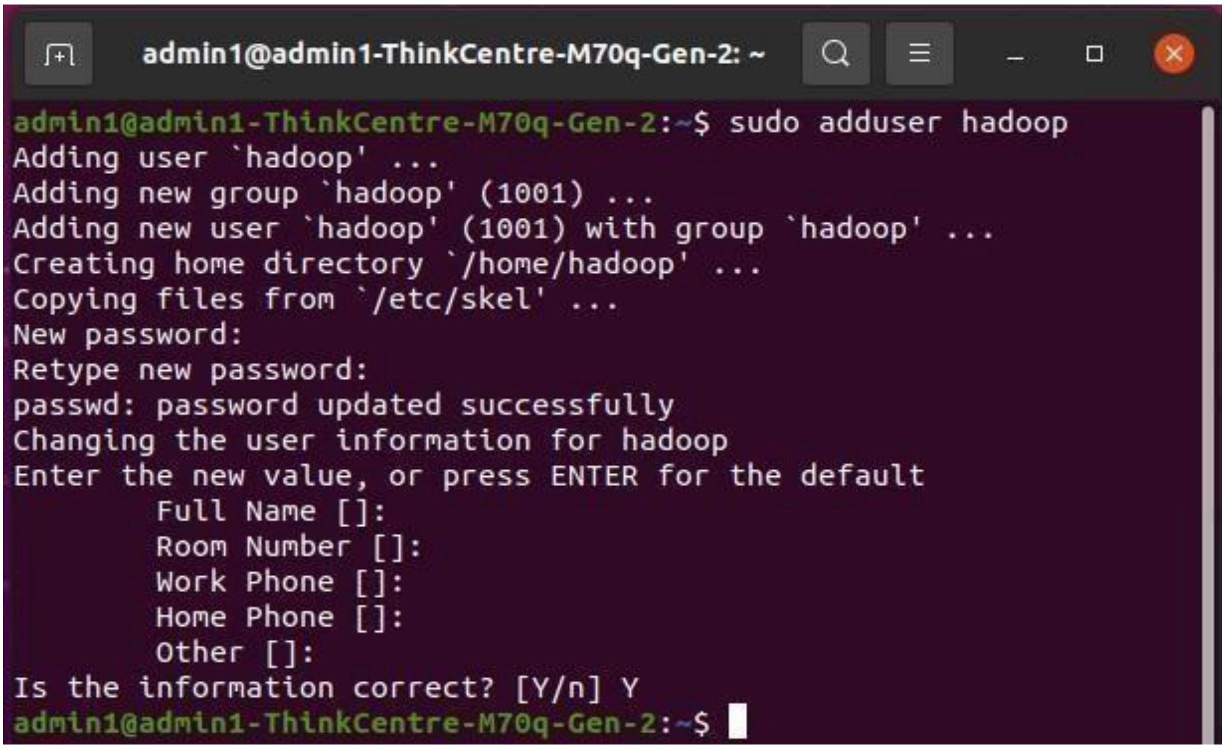
admin1@admin1-ThinkCentre-M70q-Gen-2: ~

```
admin1@admin1-ThinkCentre-M70q-Gen-2:~$ java -version
openjdk version "11.0.20.1" 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu120.04, mixed mode,
 sharing)
admin1@admin1-ThinkCentre-M70q-Gen-2:~$
```

Create Hadoop User (Optional)

4. Create a new user called hadoop .

```
sudo adduser hadoop
```

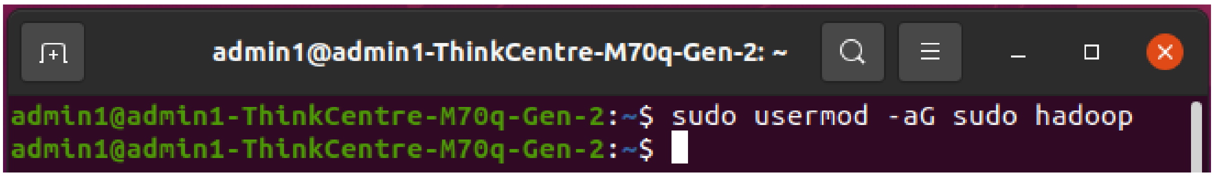


admin1@admin1-ThinkCentre-M70q-Gen-2: ~

```
admin1@admin1-ThinkCentre-M70q-Gen-2:~$ sudo adduser hadoop
Adding user `hadoop' ...
Adding new group `hadoop' (1001) ...
Adding new user `hadoop' (1001) with group `hadoop' ...
Creating home directory `/home/hadoop' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for hadoop
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
admin1@admin1-ThinkCentre-M70q-Gen-2:~$
```

5. To enable superuser privileges to the new user, add it to the sudo group .

```
sudo usermod -aG sudo hadoop
```

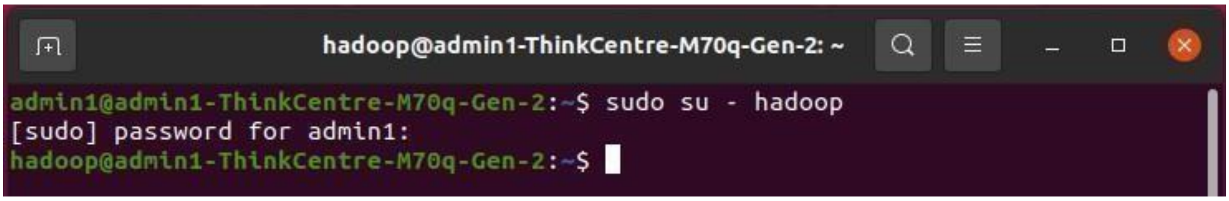


admin1@admin1-ThinkCentre-M70q-Gen-2: ~

```
admin1@admin1-ThinkCentre-M70q-Gen-2:~$ sudo usermod -aG sudo hadoop
admin1@admin1-ThinkCentre-M70q-Gen-2:~$
```

6. Change to the Hadoop user now.

```
sudo su - hadoop
```



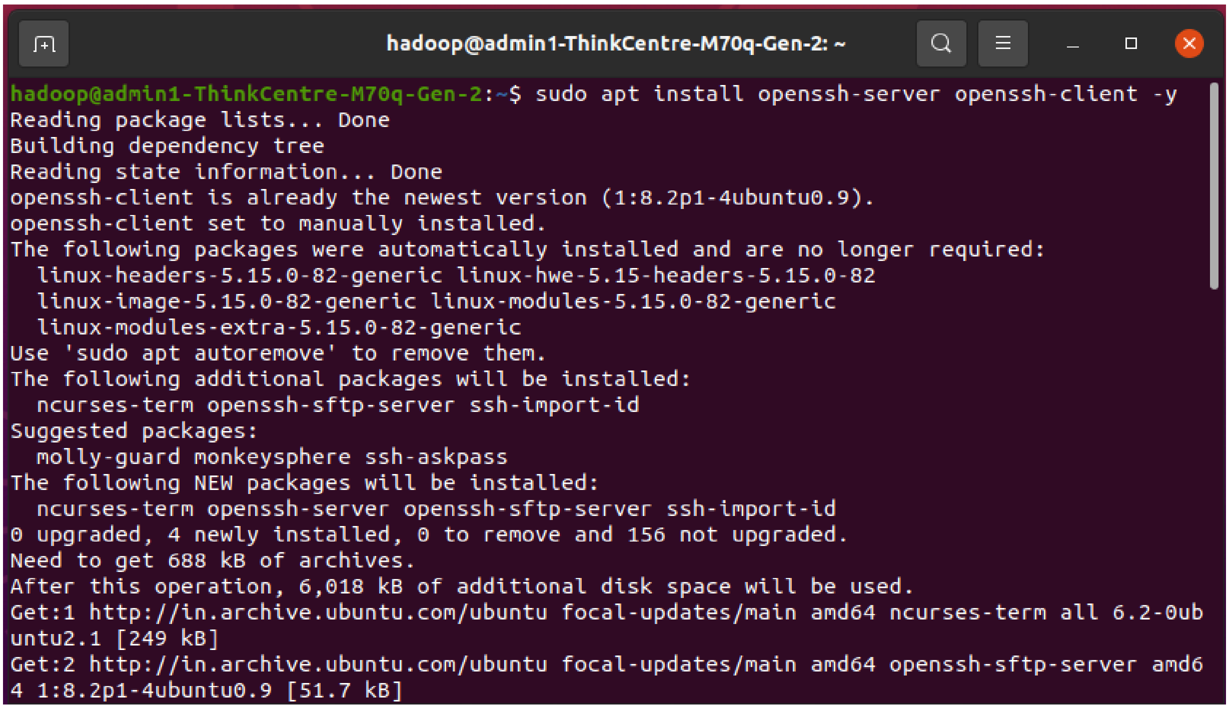
```
admin1@admin1-ThinkCentre-M70q-Gen-2:~$ sudo su - hadoop
[sudo] password for admin1:
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

Configure Password-less SSH

In order to manage nodes in a cluster, Hadoop requires SSH access

7. Install OpenSSH server and client

```
sudo apt install openssh-server openssh-client -y
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo apt install openssh-server openssh-client -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-client is already the newest version (1:8.2p1-4ubuntu0.9).
openssh-client set to manually installed.
The following packages were automatically installed and are no longer required:
  linux-headers-5.15.0-82-generic linux-hwe-5.15-headers-5.15.0-82
  linux-image-5.15.0-82-generic linux-modules-5.15.0-82-generic
  linux-modules-extra-5.15.0-82-generic
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 4 newly installed, 0 to remove and 156 not upgraded.
Need to get 688 kB of archives.
After this operation, 6,018 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 ncurses-term all 6.2-0ub
untu2.1 [249 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 openssh-sftp-server amd6
4 1:8.2p1-4ubuntu0.9 [51.7 kB]
```

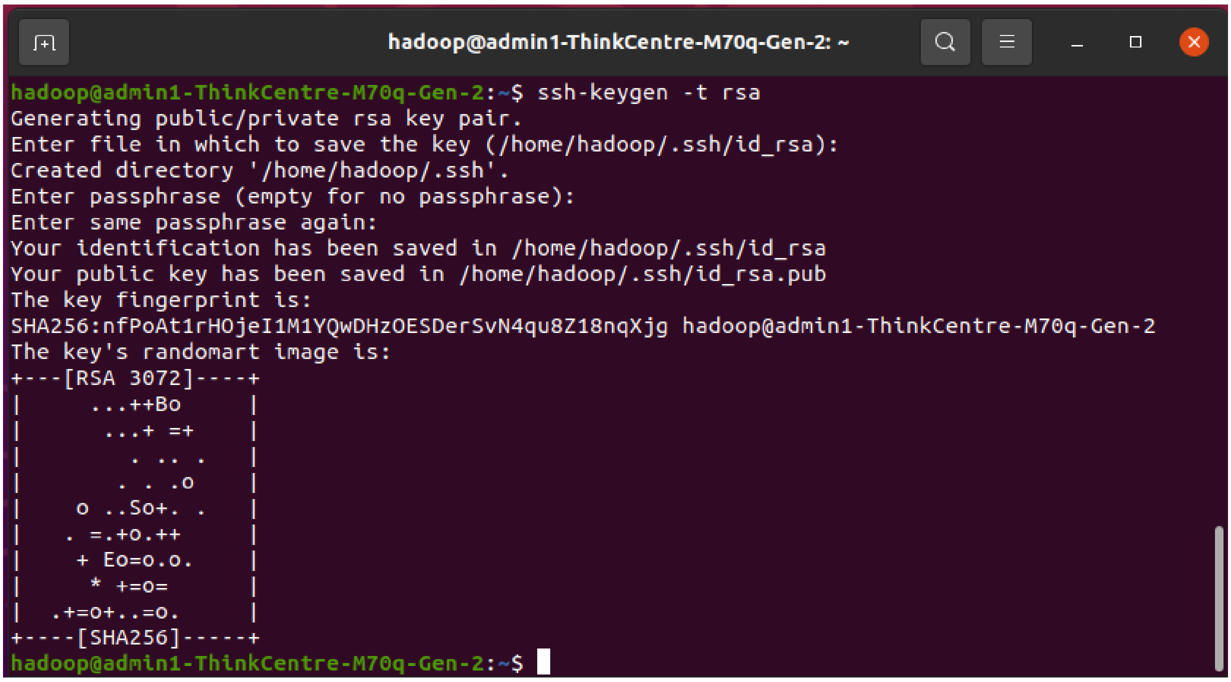
8. Generate public and private key pairs.

```
ssh-keygen -t rsa
```

Here, it will ask you:

Where to save the key (hit enter to save it inside your home directory)

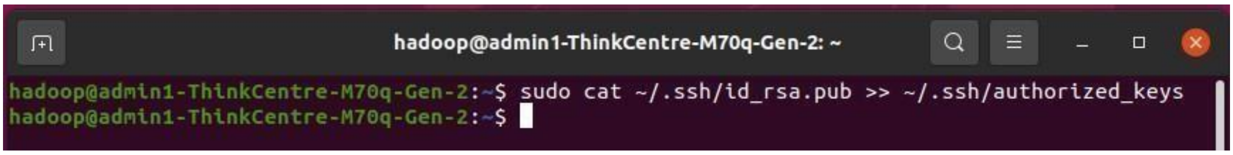
Create passphrase for keys (leave blank for no passphrase)



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
Created directory '/home/hadoop/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/hadoop/.ssh/id_rsa
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:nfPoAt1rHOjeI1M1YQwDHzOESDerSvN4qu8Z18nqXjg hadoop@admin1-ThinkCentre-M70q-Gen-2
The key's randomart image is:
+---[RSA 3072]----+
|     ...++Bo     |
|      ...+ =+    |
|        . .. .   |
|       . . .o    |
|    o ..So+. .   |
|   . =.+o.++     |
|    + Eo=o.o.    |
|     * +=o=      |
|  .+=o+..=o.     |
+----[SHA256]-----+
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

9. Add the generated public key from id_rsa.pub to authorized_keys

```
sudo cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

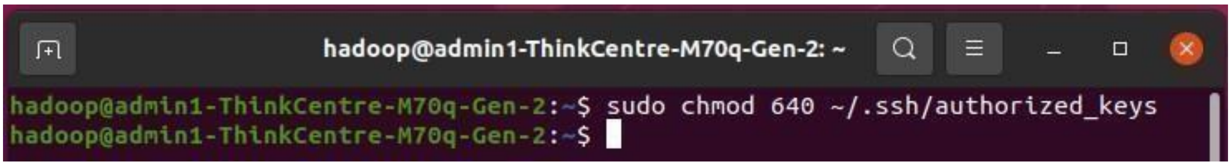


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

10. Change the file permissions for authorized_keys.

As we have stored the generated key pair in the ssh authorized key, now we will change the file permissions to “**640**” which means that only we as the “**owner**” of the file will have the read and write permissions, “**groups**” will only have the read permission. No permission will be granted to “**other users**”:

```
sudo chmod 640 ~/.ssh/authorized_keys
```



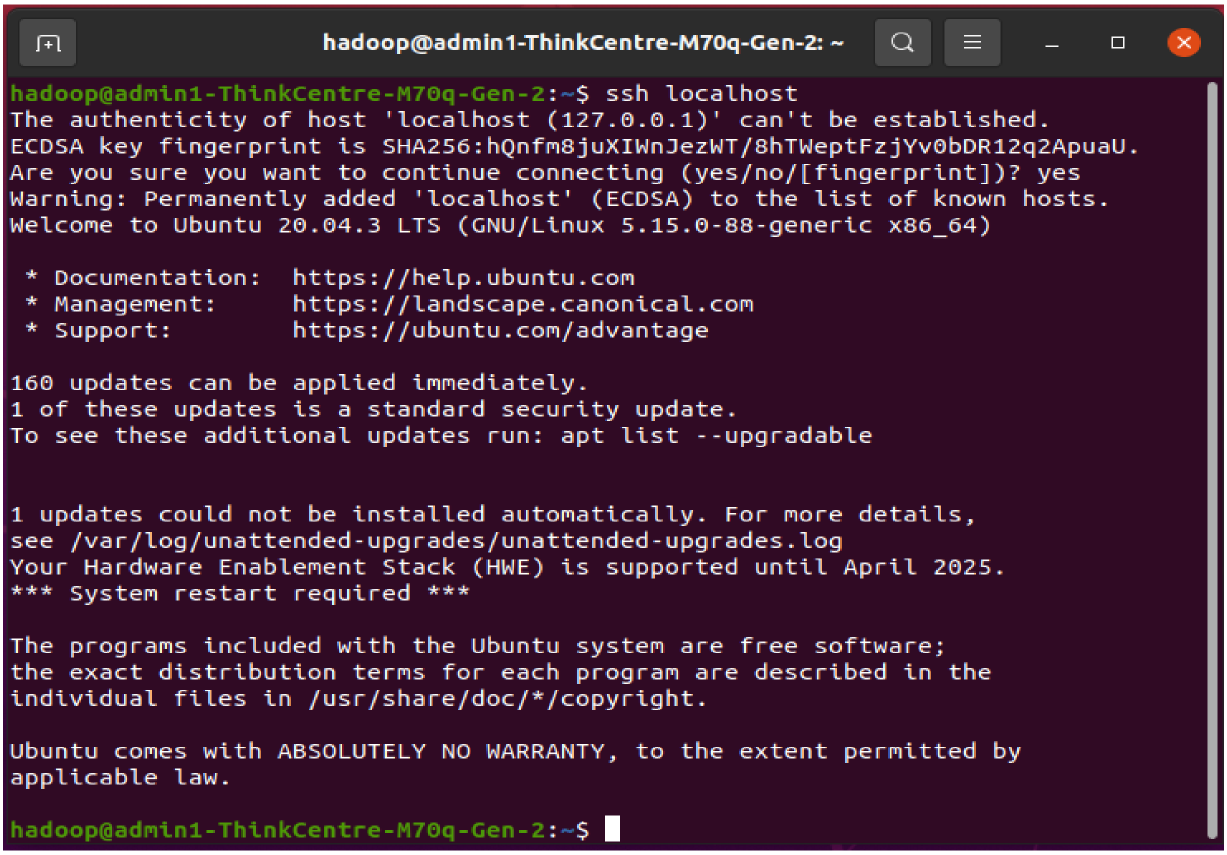
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo chmod 640 ~/.ssh/authorized_keys
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

11. Check to see if the password-less SSH is working.

You will be asked to authenticate hosts by adding RSA keys to known hosts.

Type yes and hit Enter to authenticate the localhost:

```
ssh localhost
```

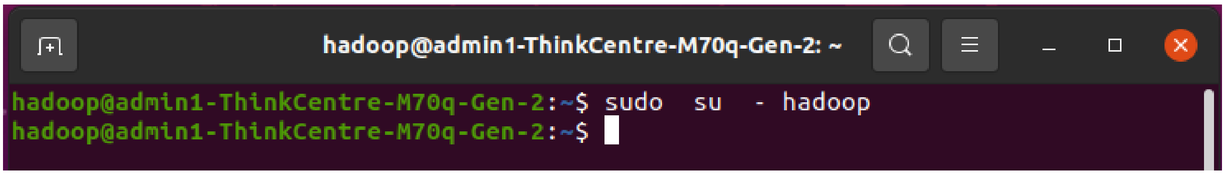


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:hQnfm8juXIWnJezWT/8hTWeptFzjYv0bDR12q2ApuaU.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.15.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

160 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable


1 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log
Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

Install and Configure Apache Hadoop in hadoop user

12. Switch to hadoop user

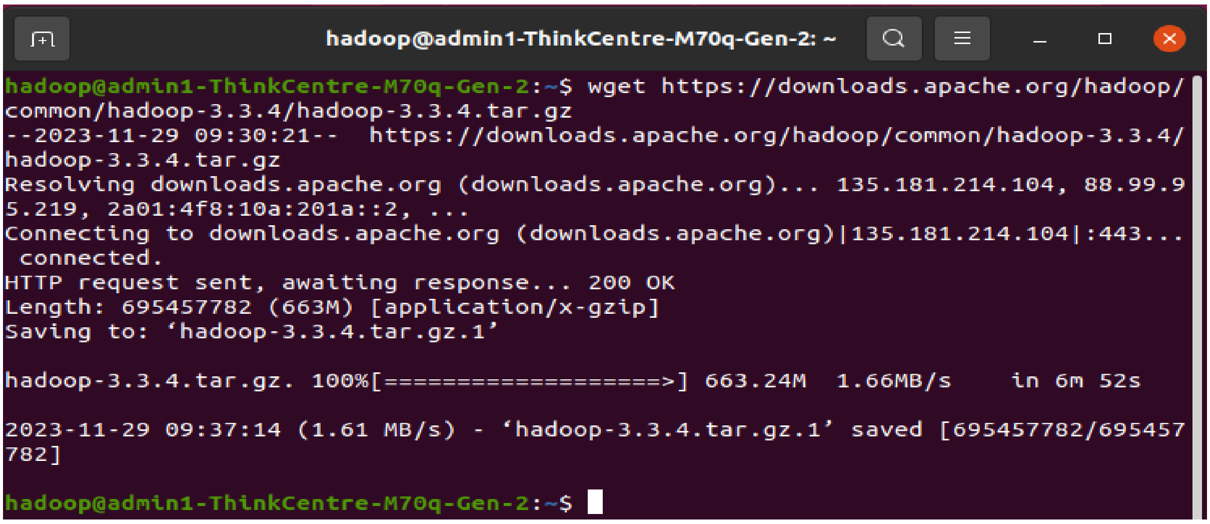
```
sudo su - hadoop
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo  su  - hadoop
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

13. Download latest stable version of Hadoop.

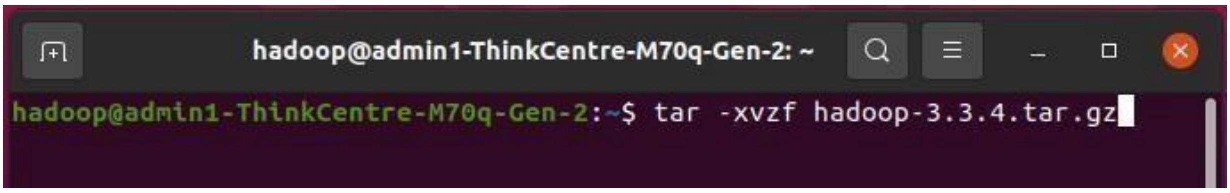
```
wget  https://downloads.apache.org/hadoop/common/hadoop-3.3.4/hadoop-3.3.4.tar.gz
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ wget https://downloads.apache.org/hadoop/
common/hadoop-3.3.4/hadoop-3.3.4.tar.gz
--2023-11-29 09:30:21--  https://downloads.apache.org/hadoop/common/hadoop-3.3.4/
hadoop-3.3.4.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 135.181.214.104, 88.99.9
5.219, 2a01:4f8:10a:201a::2, ...
Connecting to downloads.apache.org (downloads.apache.org)|135.181.214.104|:443...
 connected.
HTTP request sent, awaiting response... 200 OK
Length: 695457782 (663M) [application/x-gzip]
Saving to: ‘hadoop-3.3.4.tar.gz.1’

hadoop-3.3.4.tar.gz. 100%[===================>] 663.24M  1.66MB/s    in 6m 52s

2023-11-29 09:37:14 (1.61 MB/s) - ‘hadoop-3.3.4.tar.gz.1’ saved [695457782/695457
782]

hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

14. Extract the downloaded tar file

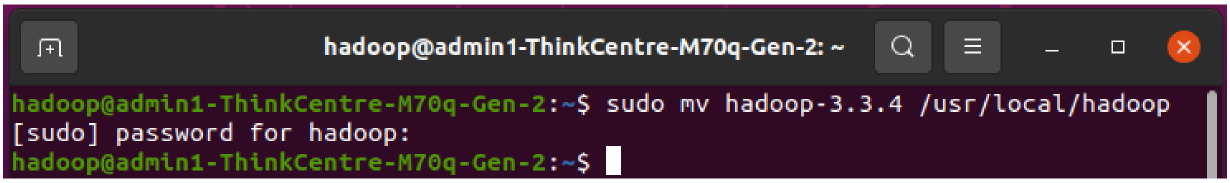
```
tar -xvzf hadoop-3.3.4.tar.gz
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ tar -xvzf hadoop-3.3.4.tar.gz
```

15. Now, rename hadoop-3.3.4 as hadoop and move to /usr/local

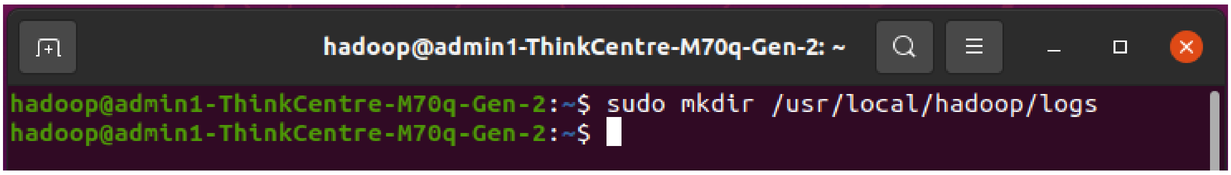
```
sudo mv hadoop-3.3.4 /usr/local/hadoop
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo mv hadoop-3.3.4 /usr/local/hadoop
[sudo] password for hadoop:
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

16. To maintain hadoop logs, create a different directory inside of usr/local/hadoop called logs.

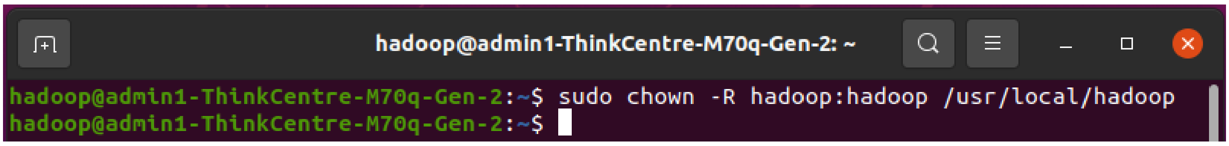
```
sudo mkdir /usr/local/hadoop/logs
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo mkdir /usr/local/hadoop/logs
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

17. Finally, use the following command to modify the directory's ownership.

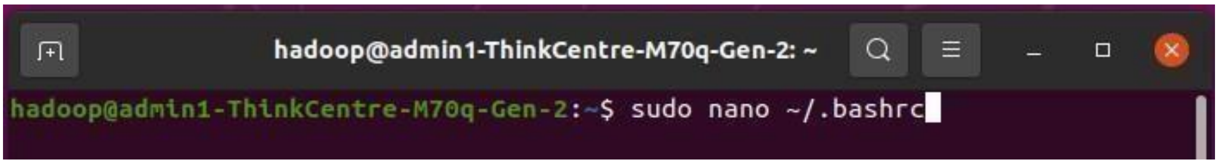
```
sudo chown -R hadoop:hadoop /usr/local/hadoop
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo chown -R hadoop:hadoop /usr/local/hadoop
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

18. Edit the bashrc for the Hadoop user via setting up the following Hadoop environment variables:

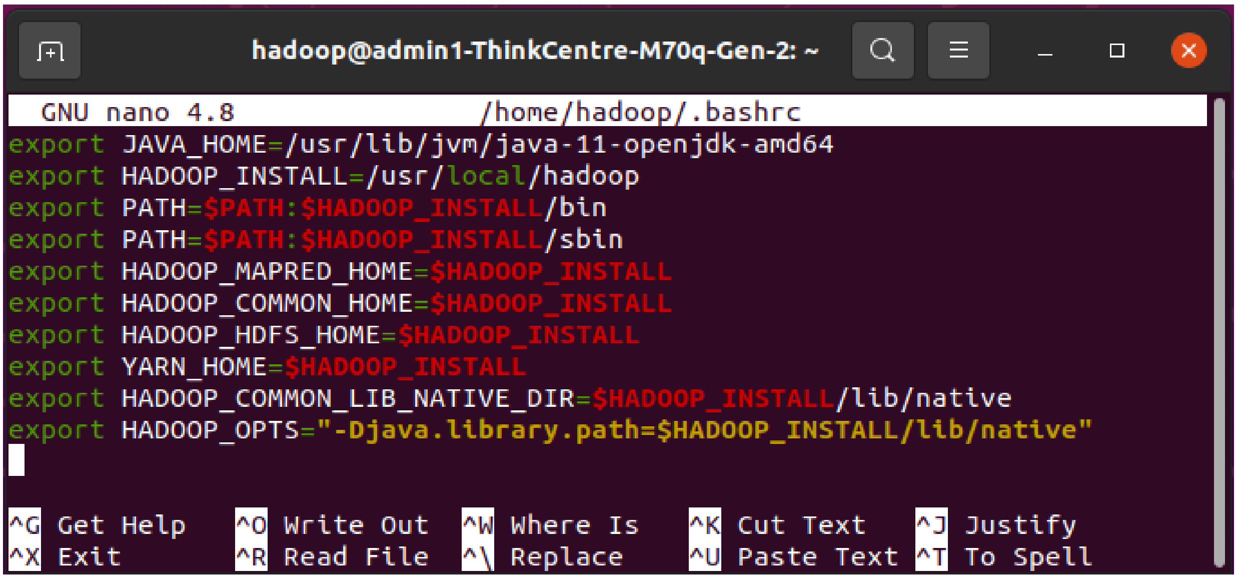
```
sudo nano ~/.bashrc
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo nano ~/.bashrc
```

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64 export
HADOOP_INSTALL=/usr/local/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin export
PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL export
YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native export
HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib/native"
```

*** Define the Hadoop environment variables by adding the following content to the end of the file.

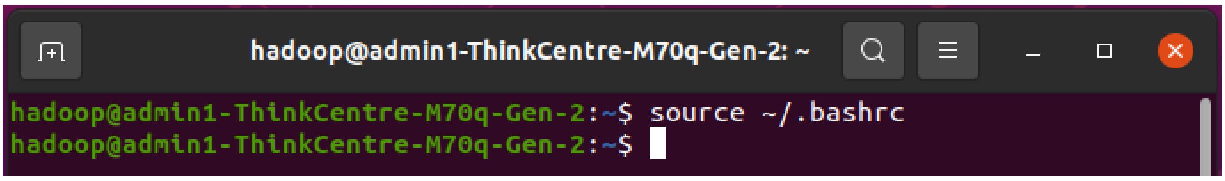


```
  GNU nano 4.8                    /home/hadoop/.bashrc
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_INSTALL=/usr/local/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib/native"


^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell
```

Press CTRL + S to save and CTRL + X to exit the nano editor after copying the lines above.

19. Use the following command to activate environmental variables after closing the nano editor.

```
source ~/.bashrc
```



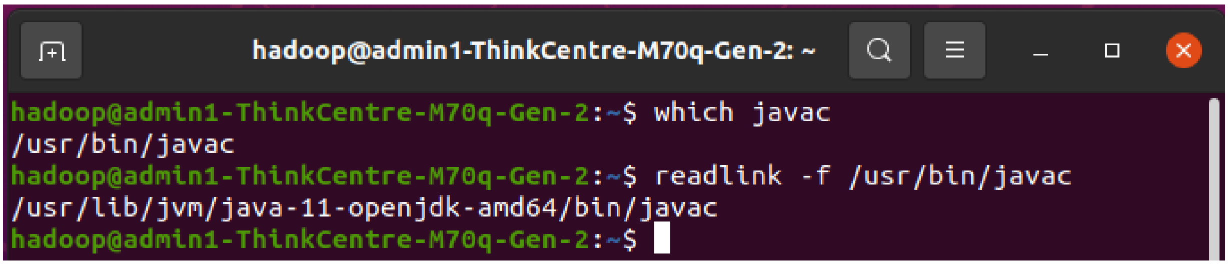
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ source ~/.bashrc
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

Configure Java Environmental variables

20. Find Java path and Open-JDK directory with help of following commands

```
which javac
```

```
readlink -f /usr/bin/javac
```

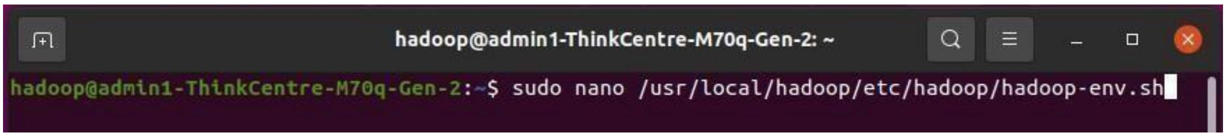


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ which javac
/usr/bin/javac
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ readlink -f /usr/bin/javac
/usr/lib/jvm/java-11-openjdk-amd64/bin/javac
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

21. Edit Hadoop-env.sh file

The path needs to match the location of the Java installation on your system

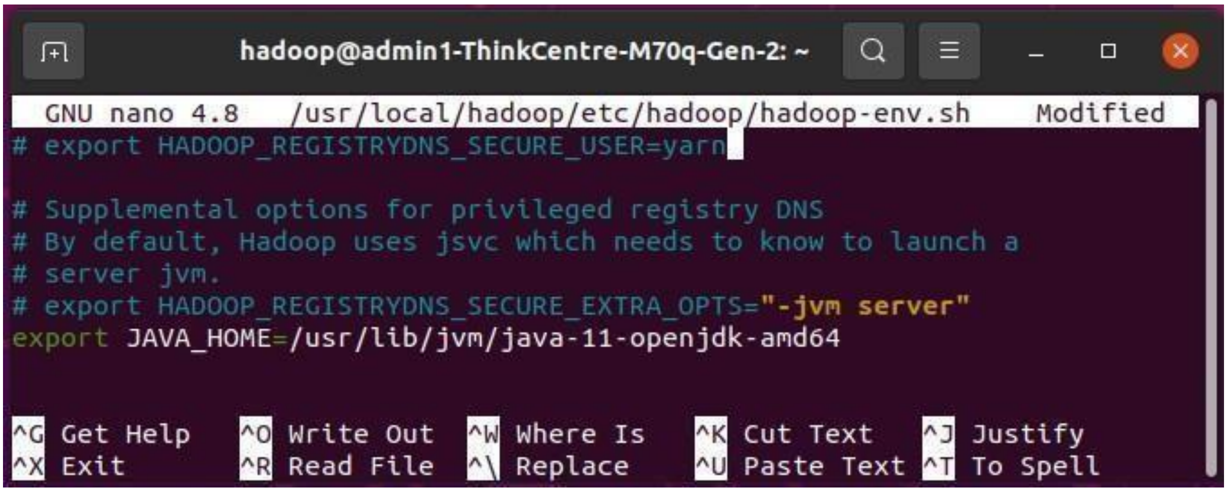
```
sudo nano /usr/local/hadoop/etc/hadoop/hadoop-env.sh
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo nano /usr/local/hadoop/etc/hadoop/hadoop-env.sh
```

Add the following content to the end of the file

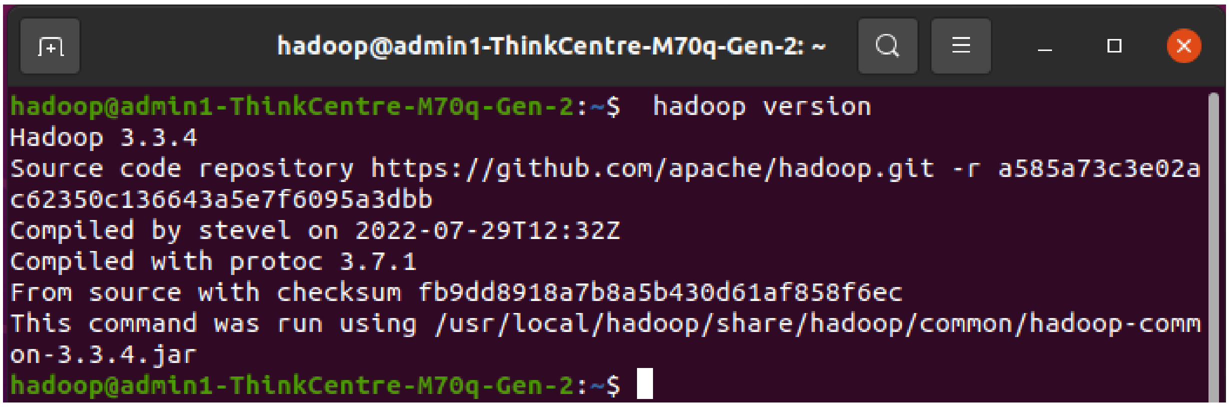
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64



```
  GNU nano 4.8    /usr/local/hadoop/etc/hadoop/hadoop-env.sh     Modified
# export HADOOP_REGISTRYDNS_SECURE_USER=yarn

# Supplemental options for privileged registry DNS
# By default, Hadoop uses jsvc which needs to know to launch a
# server jvm.
# export HADOOP_REGISTRYDNS_SECURE_EXTRA_OPTS="-jvm server"
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64


^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell
```

22. Now we can check the current Hadoop version, you can use below command:

```
hadoop version
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$  hadoop version
Hadoop 3.3.4
Source code repository https://github.com/apache/hadoop.git -r a585a73c3e02ac62350c136643a5e7f6095a3dbb
Compiled by stevel on 2022-07-29T12:32Z
Compiled with protoc 3.7.1
From source with checksum fb9dd8918a7b8a5b430d61af858f6ec
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-3.3.4.jar
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

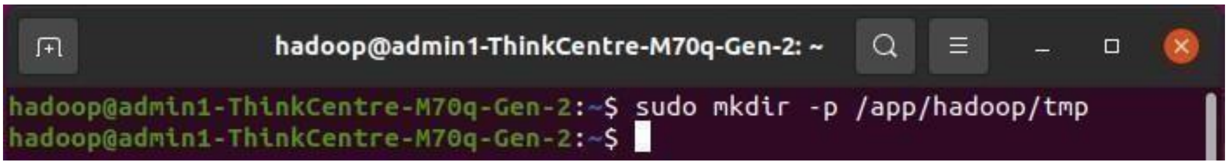
Configure related xml files to HDFS

A Hadoop environment is configured by editing a set of configuration files:

- core-site.xml
- hdfs-site.xml
- mapred-site-xml

25. create a tmp directory.

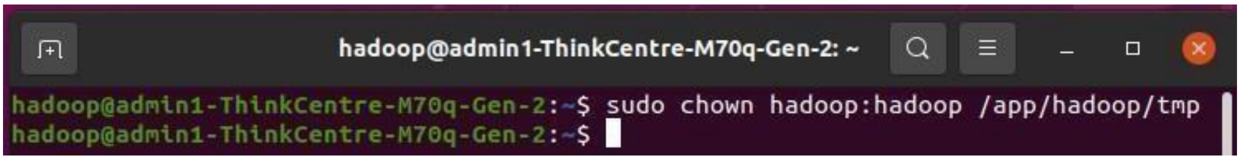
```
sudo mkdir -p /app/hadoop/tmp
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo mkdir -p /app/hadoop/tmp
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

26. Grant permissions to the directory

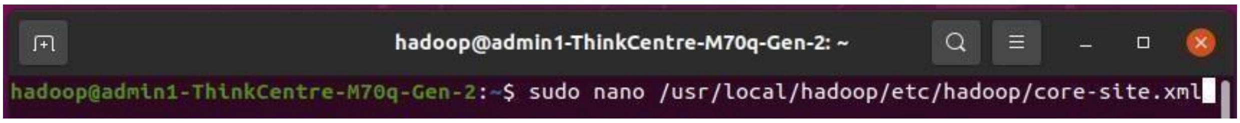
```
sudo chown hadoop:hadoop /app/hadoop/tmp
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo chown hadoop:hadoop /app/hadoop/tmp
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

27. Edit core-site.xml

```
sudo nano /usr/local/hadoop/etc/hadoop/core-site.xml
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo nano /usr/local/hadoop/etc/hadoop/core-site.xml
```

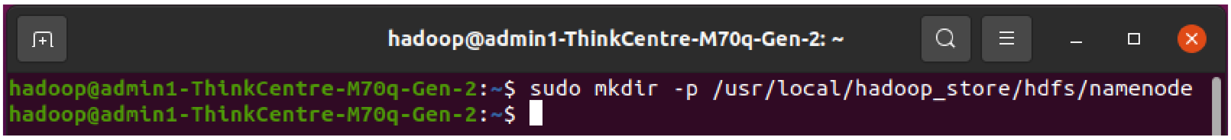
Copy below line in between tags <configuration></configuration>

```
<configuration>
<property>
<name>hadoop.tmp.dir</name>
```

```
<value>/app/hadoop/tmp</value>
<description>A base for other temporary directories.</description>
</property>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:54310</value>
<description>The name of the default file system.
A URI whose scheme and authority determine the FileSystem implementation.
The uri's scheme determines the config property (fs.SCHEME.impl) naming the FileSystem
implementation class.
used to The uri's authority is determine the host, port, etc. for a filesystem.
</description>
</property>
<property>
<name>fs.trash.interval</name>
<value>3</value>
</property>
<property>
<name>fs.trash.checkpoint.interval</name>
<value>1</value>
</property>
<property>
<name>hadoop.http.staticuser.user</name>
<value>hadoop</value>
</property>
</configuration>
```

28. create the Namenode directory.

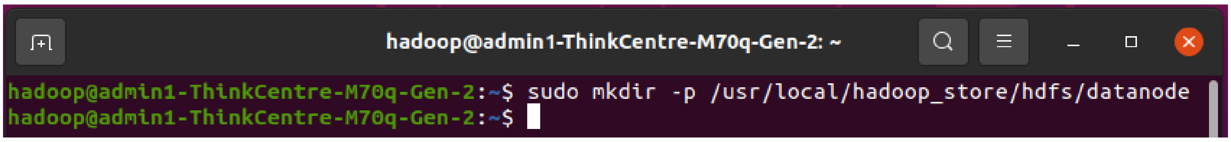
```
sudo mkdir -p /usr/local/hadoop_store/hdfs/namenode
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo mkdir -p /usr/local/hadoop_store/hdfs/namenode
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

29. create a Datanode directory.

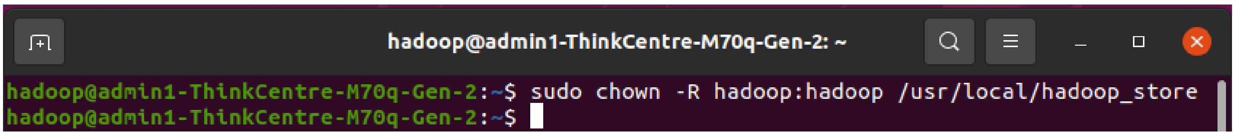
```
sudo mkdir -p /usr/local/hadoop_store/hdfs/datanode
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo mkdir -p /usr/local/hadoop_store/hdfs/datanode
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

30. after creating the directories and give it hadoop’s ownership .

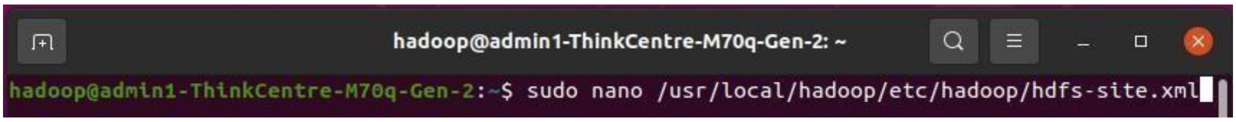
```
sudo chown -R hadoop:hadoop /usr/local/hadoop_store
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo chown -R hadoop:hadoop /usr/local/hadoop_store
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

31. Edit hdfs-site.xml

```
sudonano/usr/local/hadoop/etc/hadoop/hdfs-site.xml
```



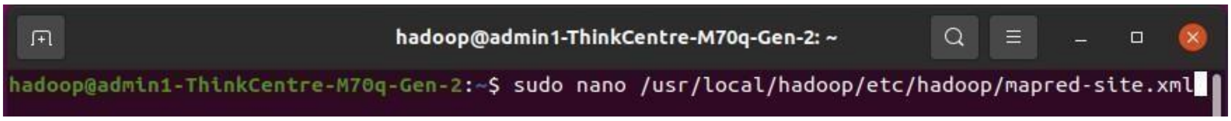
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo nano /usr/local/hadoop/etc/hadoop/hdfs-site.xml
```

Add below lines of setting in between tags <configuration> and </configuration>

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
<description>
Default block replication.
The actual number of replications can be specified when the file is created.
The default is used if replication is not specified in create time.
</description>
</property>
<property>
<name>dfs.block.size</name>
<value>1048576</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/usr/local/hadoop_store/hdfs/namenode</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:/usr/local/hadoop_store/hdfs/datanode</value>
</property>
<property>
<name>dfs.permissions.enabled</name>
<value>true</value>
</property>
</configuration>
```

32. Edit mapred-site.xml

```
sudo nano /usr/local/hadoop/etc/hadoop/mapred-site.xml
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo nano /usr/local/hadoop/etc/hadoop/mapred-site.xml
```

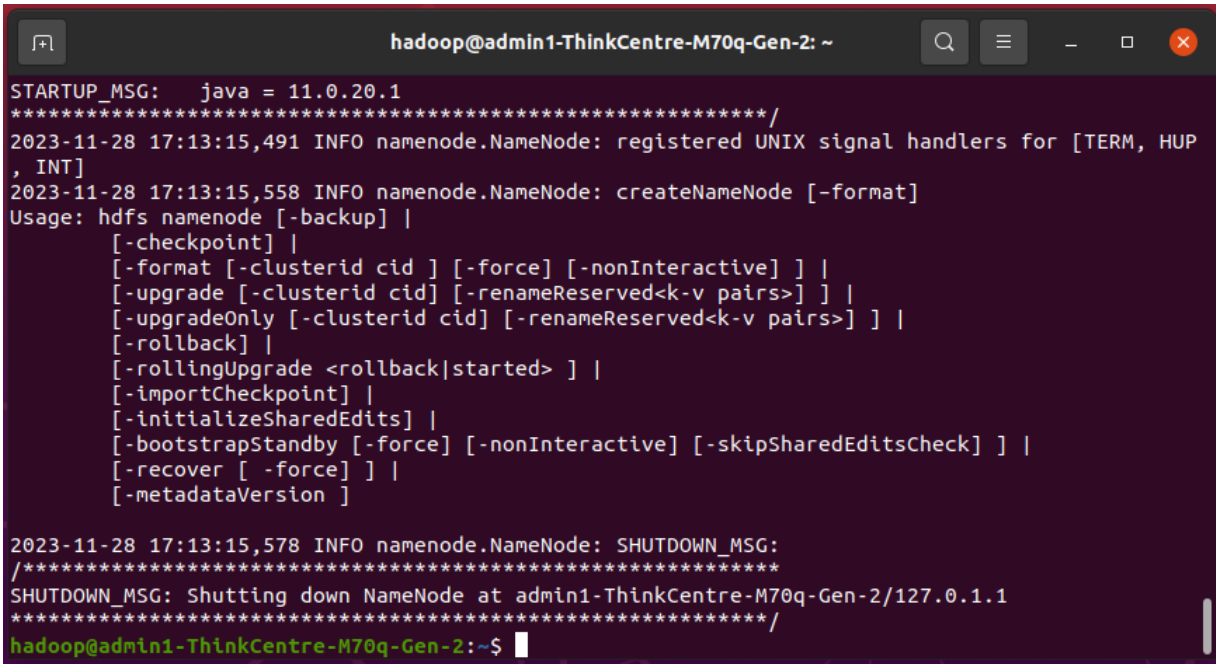
Add below lines of setting in between tags <configuration> and </configuration>

```
<configuration>
<property>
<name>mapred.job.tracker</name>
<value>localhost:54311</value>
<description>
The host and port that the MapReduce job tracker Runs at.
If "local", then jobs are run in-process as a single map and reduce task.
</description>
</property>
</configuration>
```

Starting the Hadoop Cluster

33. Format the namenode before using it for the first time.
    As HDFS user run the below command to format the Namenode.

```
hdfs namenode –format
```



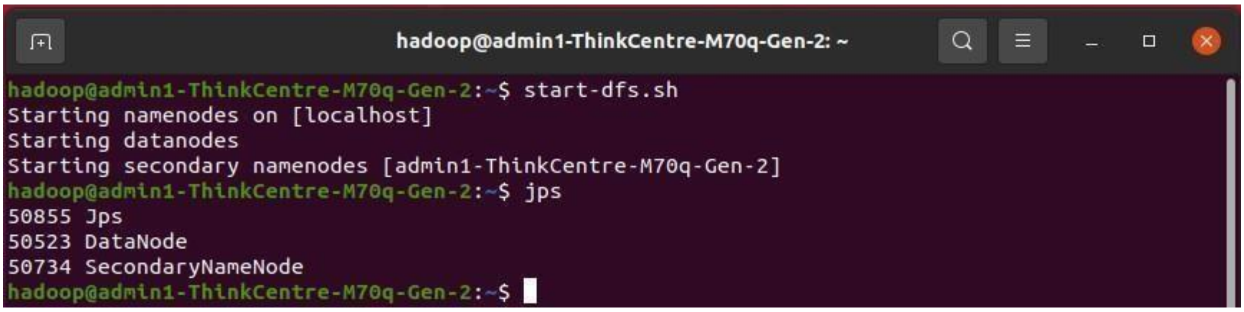
```
STARTUP_MSG:   java = 11.0.20.1
************************************************************/
2023-11-28 17:13:15,491 INFO namenode.NameNode: registered UNIX signal handlers for [TERM, HUP
, INT]
2023-11-28 17:13:15,558 INFO namenode.NameNode: createNameNode [-format]
Usage: hdfs namenode [-backup] |
        [-checkpoint] |
        [-format [-clusterid cid ] [-force] [-nonInteractive] ] |
        [-upgrade [-clusterid cid] [-renameReserved<k-v pairs>] ] |
        [-upgradeOnly [-clusterid cid] [-renameReserved<k-v pairs>] ] |
        [-rollback] |
        [-rollingUpgrade <rollback|started> ] |
        [-importCheckpoint] |
        [-initializeSharedEdits] |
        [-bootstrapStandby [-force] [-nonInteractive] [-skipSharedEditsCheck] ] |
        [-recover [ -force] ] |
        [-metadataVersion ]

2023-11-28 17:13:15,578 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at admin1-ThinkCentre-M70q-Gen-2/127.0.1.1
************************************************************/
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

Launch the Apache Hadoop Cluster

First way is start-dfs.sh & start-yarn.sh, to launch the name node data node , yarn resource and node manager

34. Once the Namenode has been formatted then start the HDFS using the start-dfs.sh script

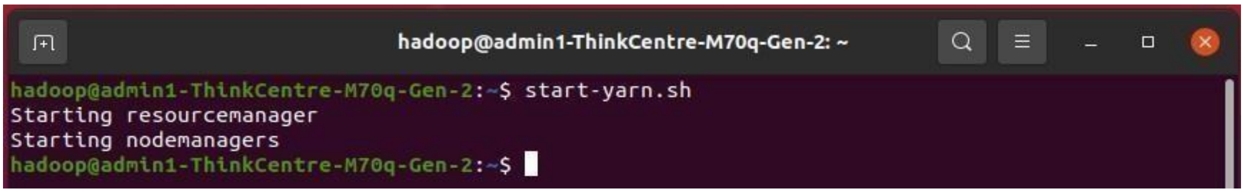
```
start-dfs.sh
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [admin1-ThinkCentre-M70q-Gen-2]
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ jps
50855 Jps
50523 DataNode
50734 SecondaryNameNode
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

35. To start the YARN services you need to execute the yarn start script i.e. start-yarn.sh

```
start-yarn.sh
```



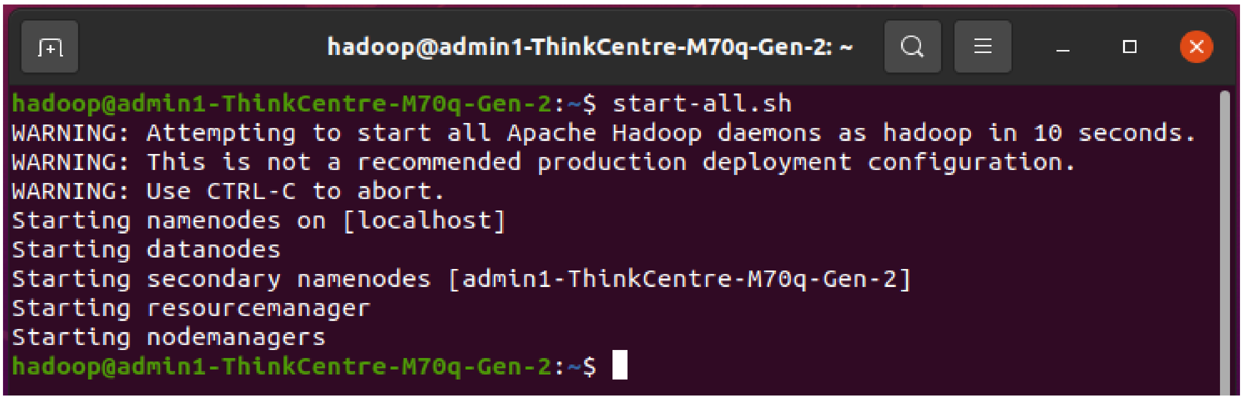
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

or

Second way is start-all.sh

To start all the daemons and bring up your hadoop cluster use the below command:

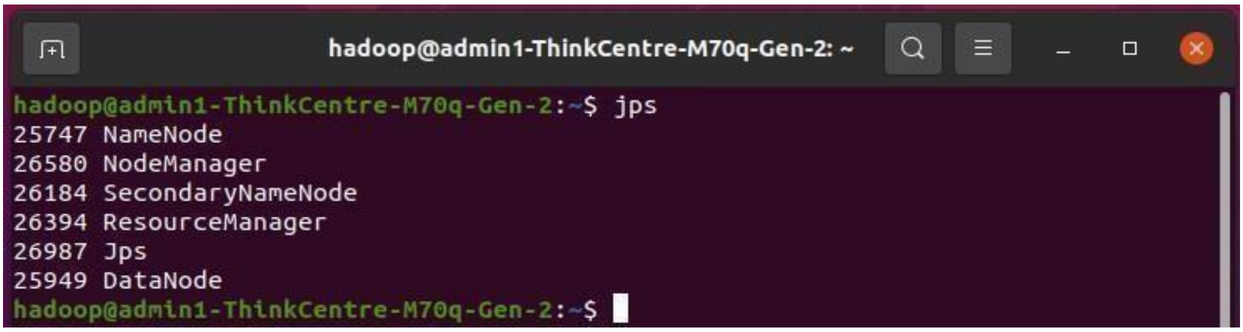
```
start-all.sh
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [admin1-ThinkCentre-M70q-Gen-2]
Starting resourcemanager
Starting nodemanagers
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

36. The **'jps'** command is used to check whether all the Hadoop processes are running or not.

```
jps
```

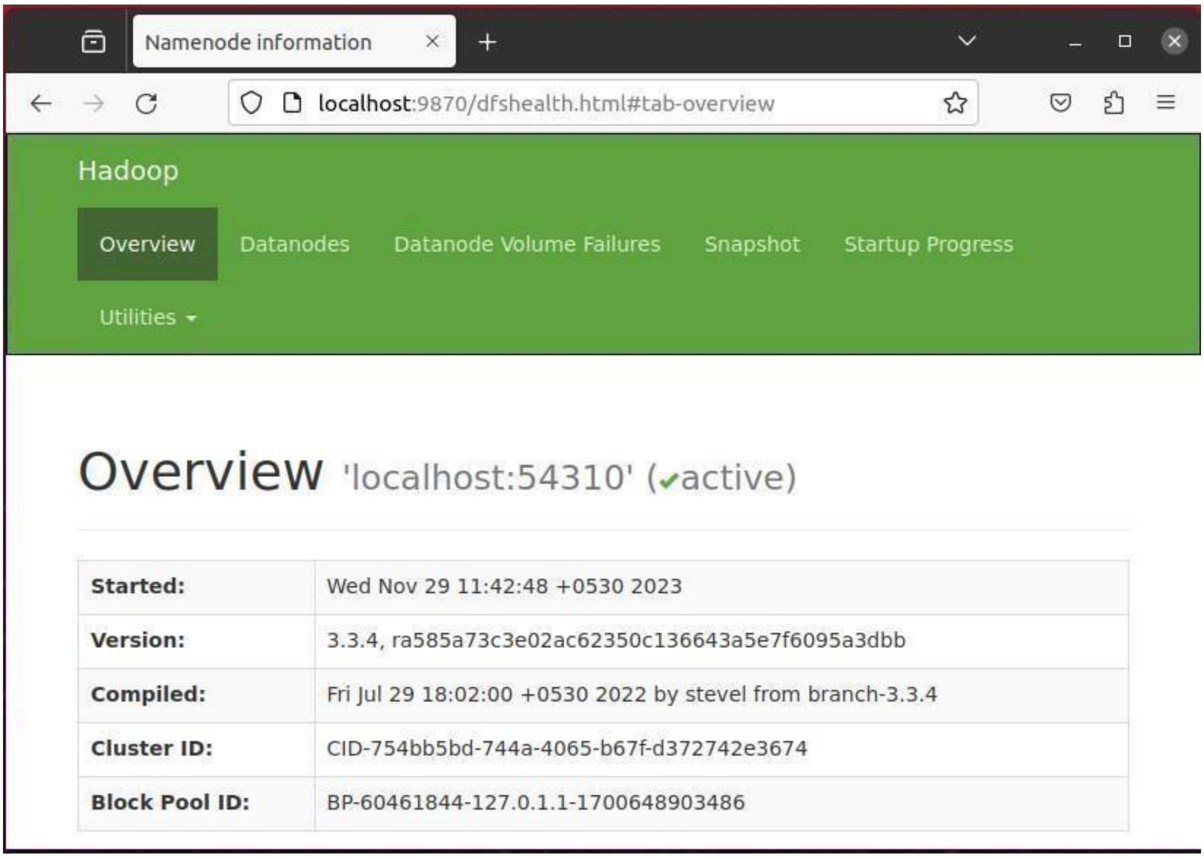


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ jps
25747 NameNode
26580 NodeManager
26184 SecondaryNameNode
26394 ResourceManager
26987 Jps
25949 DataNode
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

If Hadoop has started successfully then an output of jps should show NameNode, NodeManager, SecondaryNameNode, ResourceManager, DataNode.

Access Hadoop UI from Browser

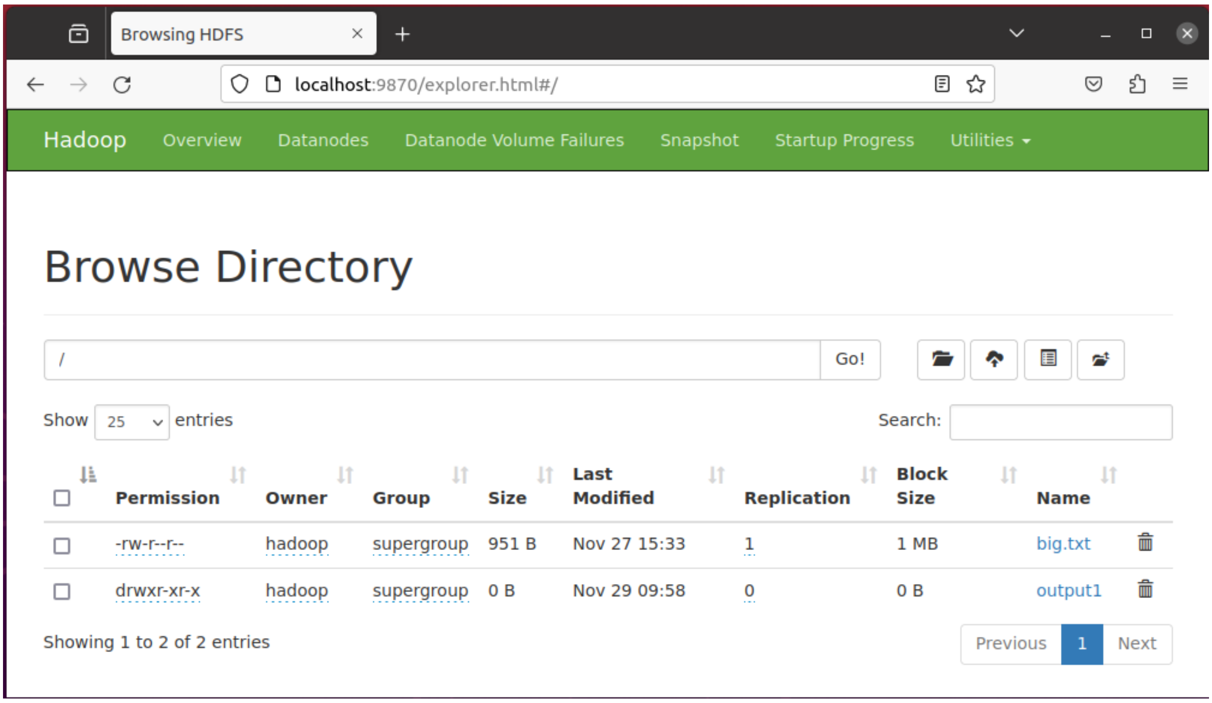
Use your preferred browser and navigate to your localhost URL or IP.
The default port number **9870** gives you access to the Hadoop NameNode UI:
Example : localhost:9870/



Overview 'localhost:54310' (✓active)

Started:	Wed Nov 29 11:42:48 +0530 2023
Version:	3.3.4, ra585a73c3e02ac62350c136643a5e7f6095a3dbb
Compiled:	Fri Jul 29 18:02:00 +0530 2022 by stevel from branch-3.3.4
Cluster ID:	CID-754bb5bd-744a-4065-b67f-d372742e3674
Block Pool ID:	BP-60461844-127.0.1.1-1700648903486

HDFS Directory Browsing via Namenode Web User Interface.



Browse Directory

/

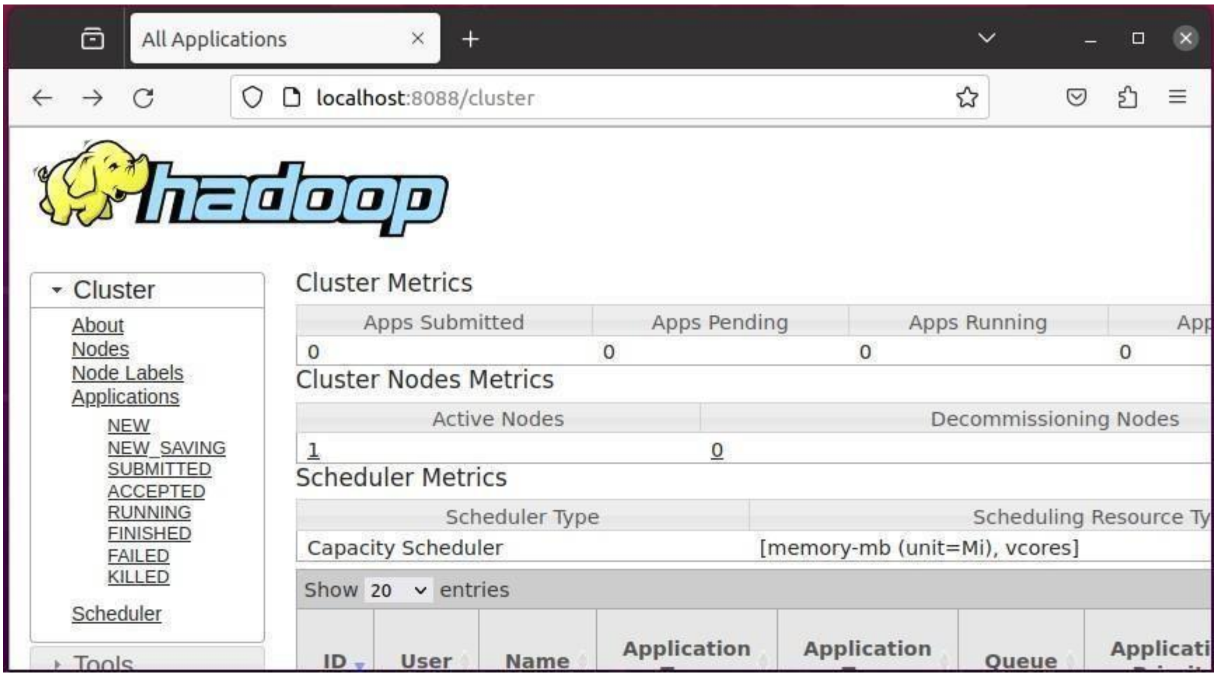
Show 25 entries Search:

☐	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
☐	-rw-r--r--	hadoop	supergroup	951 B	Nov 27 15:33	1	1 MB	big.txt
☐	drwxr-xr-x	hadoop	supergroup	0 B	Nov 29 09:58	0	0 B	output1

Showing 1 to 2 of 2 entries

The YARN Resource Manager (RM) web interface will display all running jobs on current Hadoop Cluster.

http://localhost:8088



Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	App
0	0	0	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes
1	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Ty
Capacity Scheduler	[memory-mb (unit=Mi), vcores]

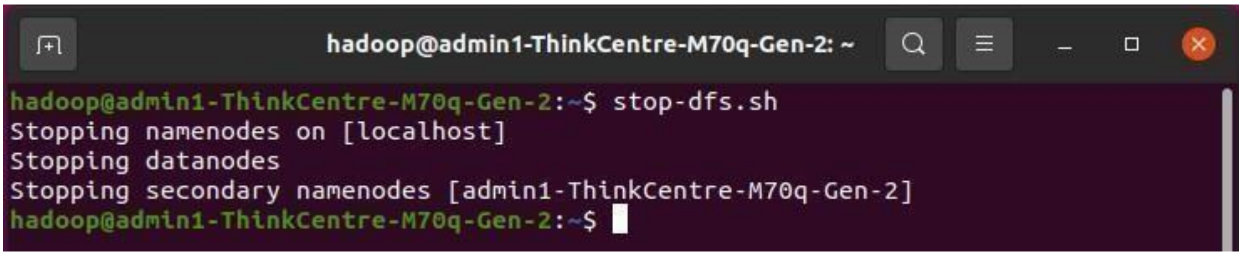
Show 20 entries

ID	User	Name	Application	Application	Queue	Applicati

Stopping the Apache Hadoop clusters

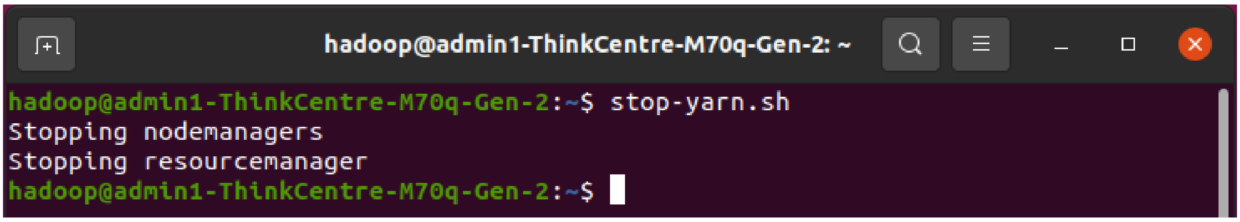
First way is stop-dfs.sh & stop-yarn.sh, to stop the namenode datanode , yarn resource and node manager .

38.stop-yarn.sh



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ stop-dfs.sh
Stopping namenodes on [localhost]
Stopping datanodes
Stopping secondary namenodes [admin1-ThinkCentre-M70q-Gen-2]
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

38.stop-yarn.sh

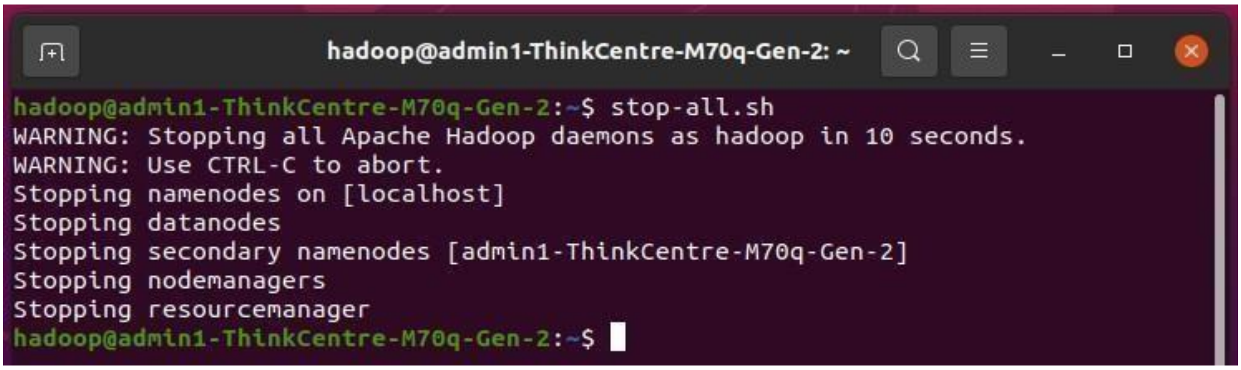


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ stop-yarn.sh
Stopping nodemanagers
Stopping resourcemanager
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

or

Second way is to use start-all.sh

Stop-all.sh



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ stop-all.sh
WARNING: Stopping all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: Use CTRL-C to abort.
Stopping namenodes on [localhost]
Stopping datanodes
Stopping secondary namenodes [admin1-ThinkCentre-M70q-Gen-2]
Stopping nodemanagers
Stopping resourcemanager
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

HDFS Commands

With the help of the HDFS command, we can perform Hadoop HDFS file operations like changing the file permissions, viewing the file contents, creating files or directories, copying file/directory from the local file system to HDFS or vice-versa, etc.

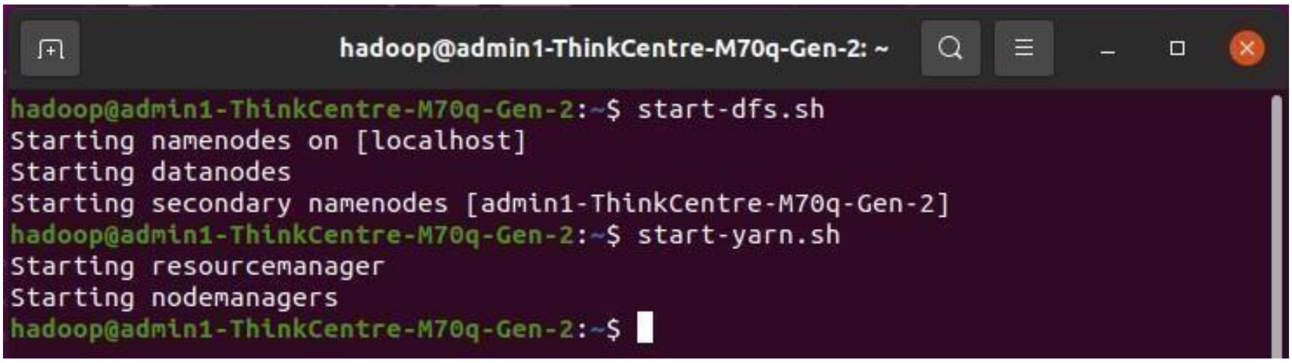
Before starting with the HDFS command, we have to start the Hadoop services.

To start the Hadoop services, do the following:

```
start-dfs.sh
```

&

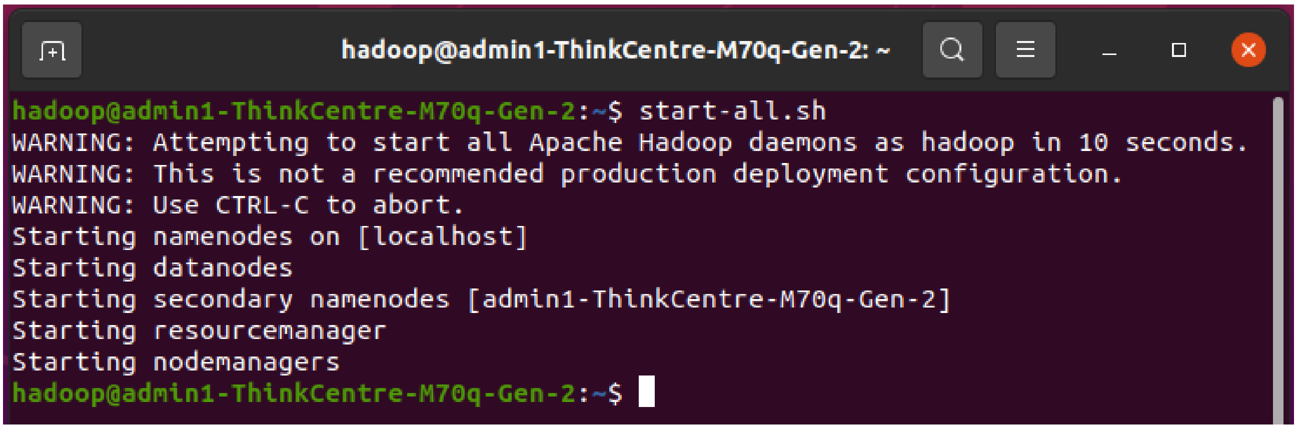
```
start-yarn.sh
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [admin1-ThinkCentre-M70q-Gen-2]
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

or

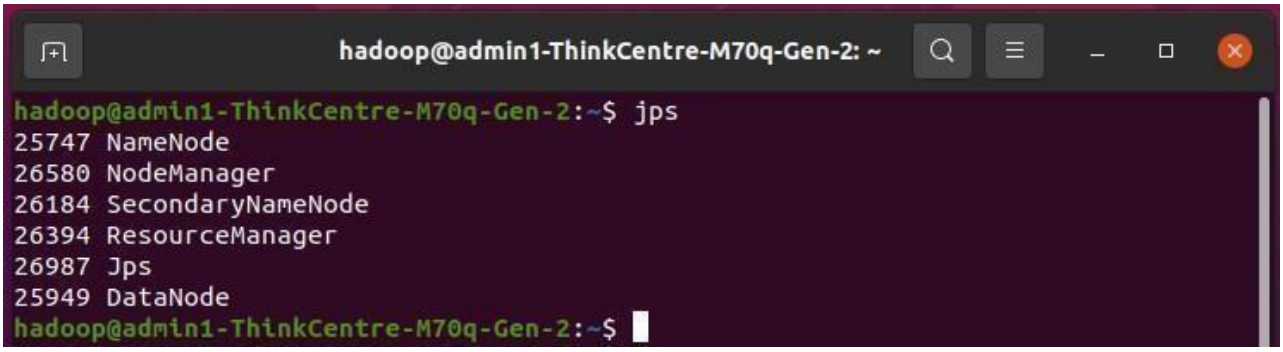
```
start-all.sh
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [admin1-ThinkCentre-M70q-Gen-2]
Starting resourcemanager
Starting nodemanagers
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

To check the Hadoop services are up and running use the following command:

```
jps
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ jps
25747 NameNode
26580 NodeManager
26184 SecondaryNameNode
26394 ResourceManager
26987 Jps
25949 DataNode
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

To create a directory: mkdir

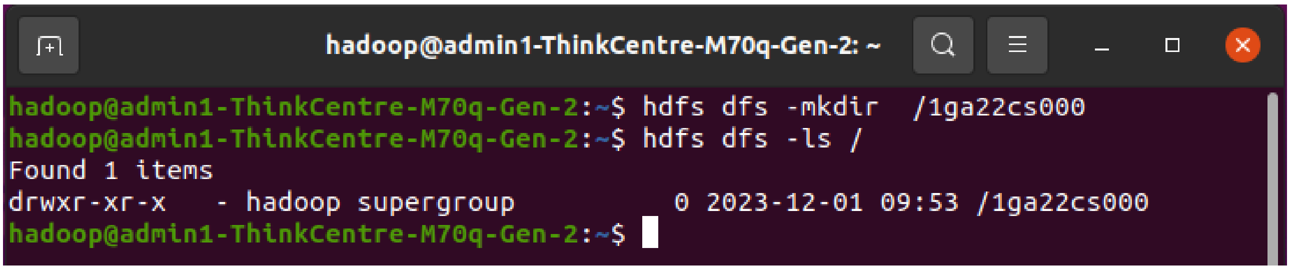
1. In Hadoop dfs there is no home directory by default. So let’s first create it.

Syntax:

hdfs dfs -mkdir <folder name>

Example:1

```
hdfs dfs -mkdir  /1ga22cs000
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -mkdir  /1ga22cs000
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls /
Found 1 items
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 09:53 /1ga22cs000
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

To create an empty file: touchz

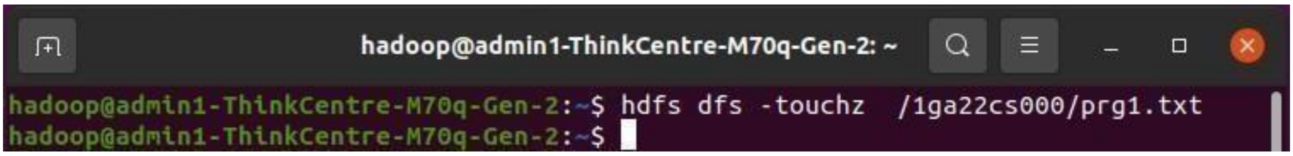
2. It creates an empty file.

Syntax:

hdfs dfs -touchz <file_path>

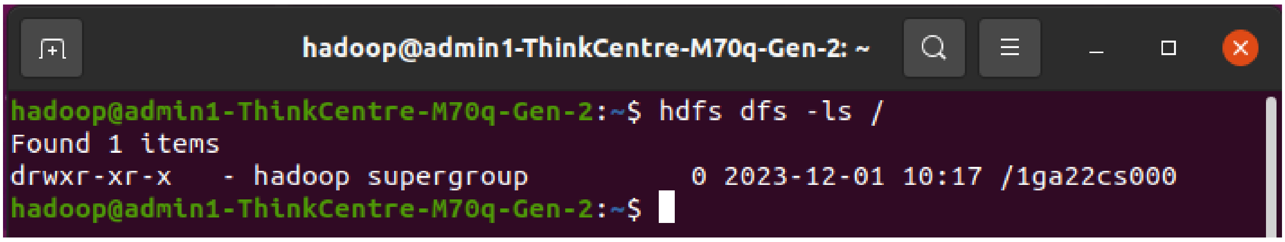
Example:1

```
hdfs dfs -touchz  /1ga22cs000/prg1.txt
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -touchz  /1ga22cs000/prg1.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

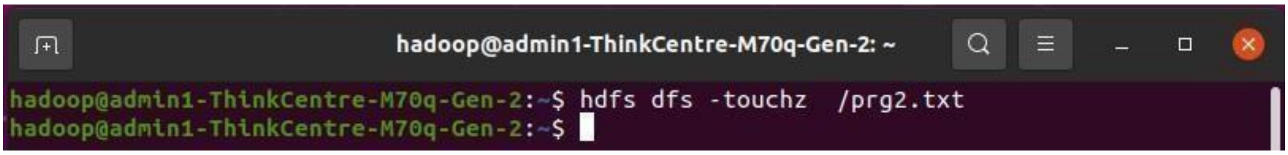
```
hdfs dfs -ls /
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls /
Found 1 items
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

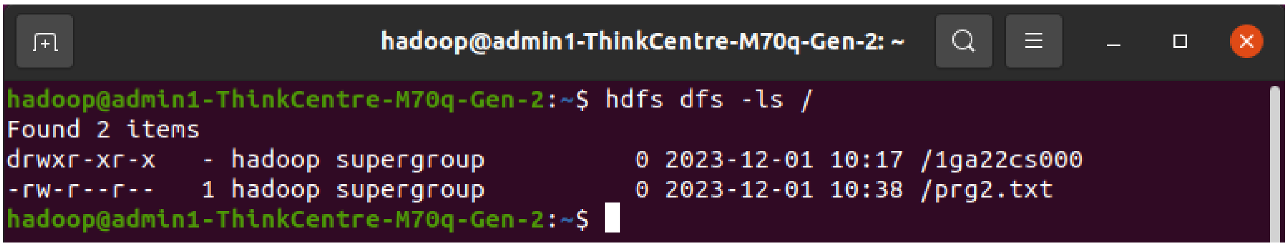
Example:2

```
hdfs dfs -touchz  /prg2.txt
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -touchz  /prg2.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

```
hdfs dfs -ls /
```

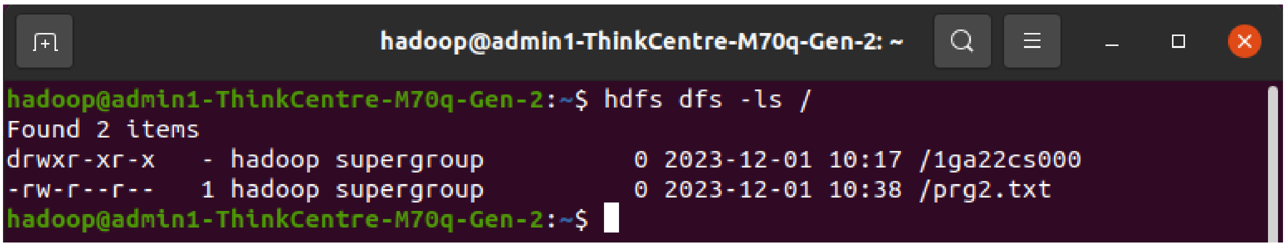


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls /
Found 2 items
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:38 /prg2.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

File Listing Commands: ls

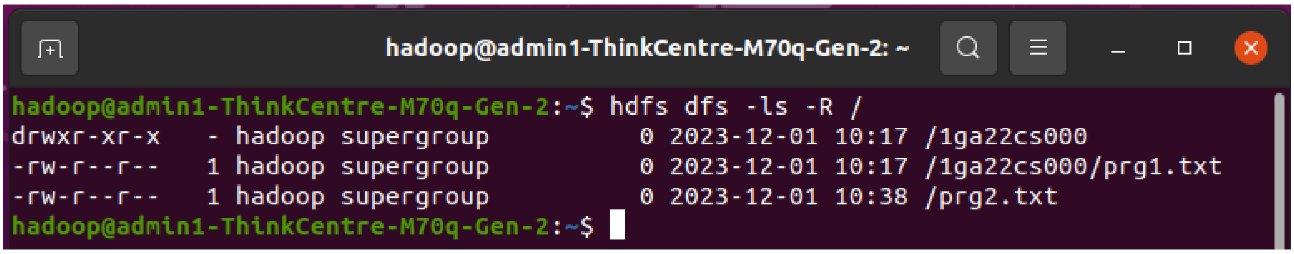
3. the ls command to enlist the files and directories present in HDFS.

```
hdfs dfs -ls /
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls /
Found 2 items
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:38 /prg2.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

```
hdfs dfs -ls -R /
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls -R /
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000/prg1.txt
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:38 /prg2.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

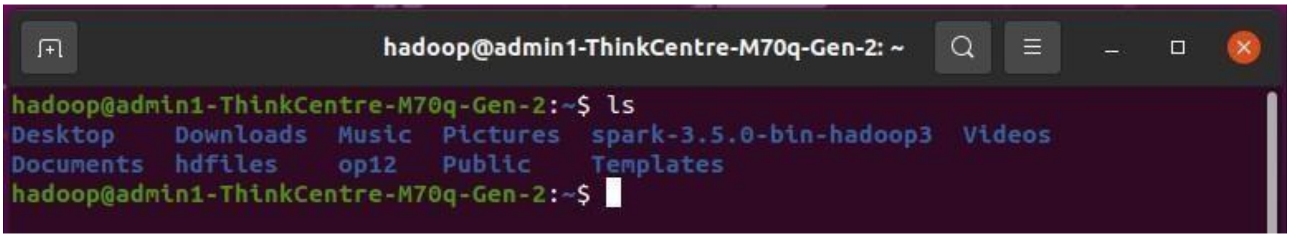
copyFromLocal (or) put:

4. To copy files/folders from local file system to hdfs store. This is the most important command. Local filesystem means the files present on the OS.

Syntax:
hdfs dfs -copyFromLocal <local file path> <dest(present on hdfs)>

Example:

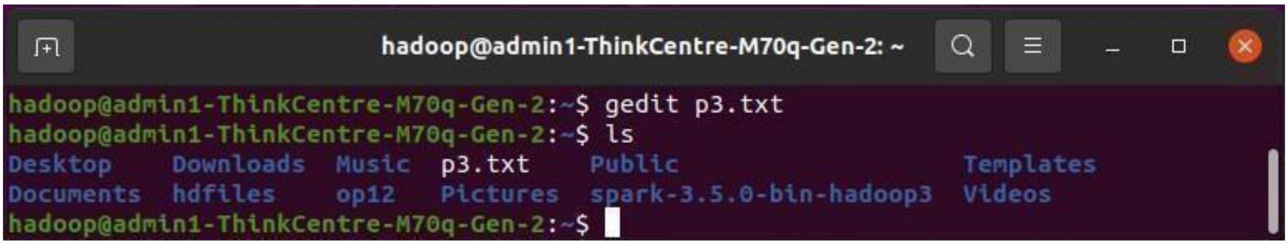
```
ls
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ ls
Desktop    Downloads  Music  Pictures  spark-3.5.0-bin-hadoop3  Videos
Documents  hdfiles    op12   Public    Templates
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

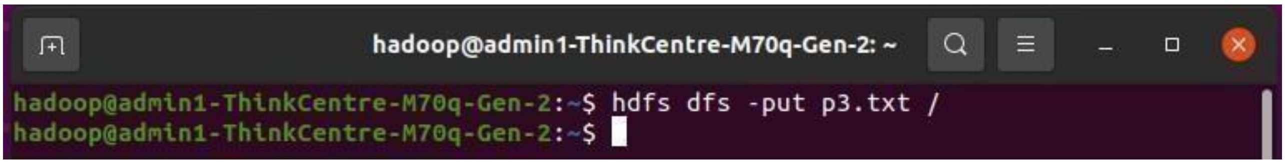
gedit p3.txt

ls



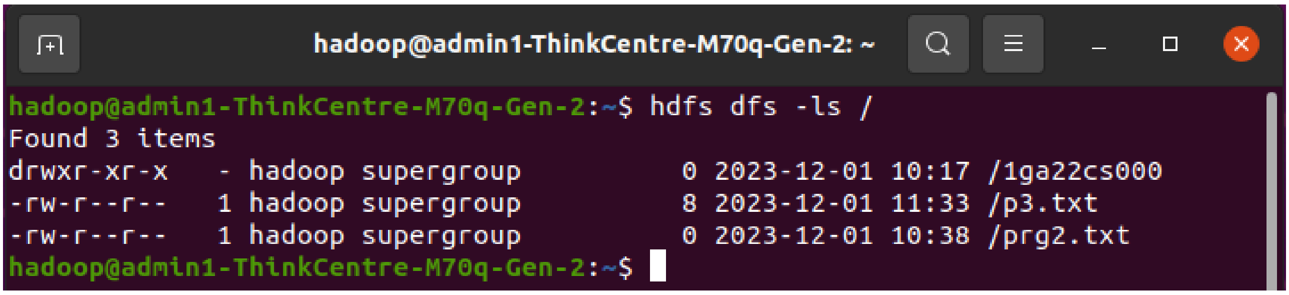
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ gedit p3.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ ls
Desktop    Downloads  Music  p3.txt    Public                   Templates
Documents  hdfiles    op12   Pictures  spark-3.5.0-bin-hadoop3  Videos
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

hdfs dfs -put p3.txt /



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -put p3.txt /
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

hdfs dfs -ls /



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls /
Found 3 items
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          8 2023-12-01 11:33 /p3.txt
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:38 /prg2.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

To print file contents: cat

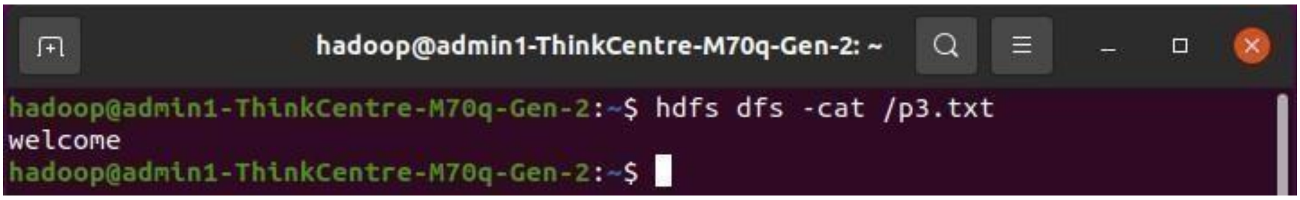
5. the cat command to display the content of the file present

Syntax:

hdfs dfs -cat <path>

Example:

hdfs dfs -cat /p3.txt



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -cat /p3.txt
welcome
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

copyToLocal (or) get

6. To copy files/folders from hdfs store to local file system.

Syntax:

hdfs dfs -copyToLocal <<srcfile(on hdfs)> <local file dest>

Example:

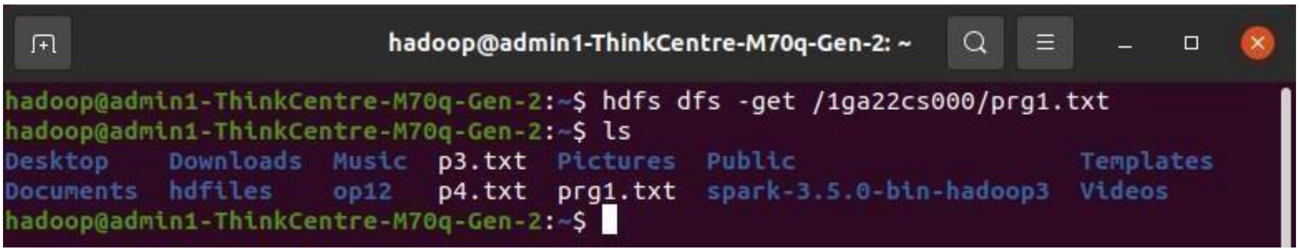
hdfs dfs -get /1ga22cs000/prg1.txt

(OR)

hdfs dfs - copyToLocal /1ga22cs000/prg1.txt

prg1.txt from 1ga22cs000 folder will be copied to home directory

ls

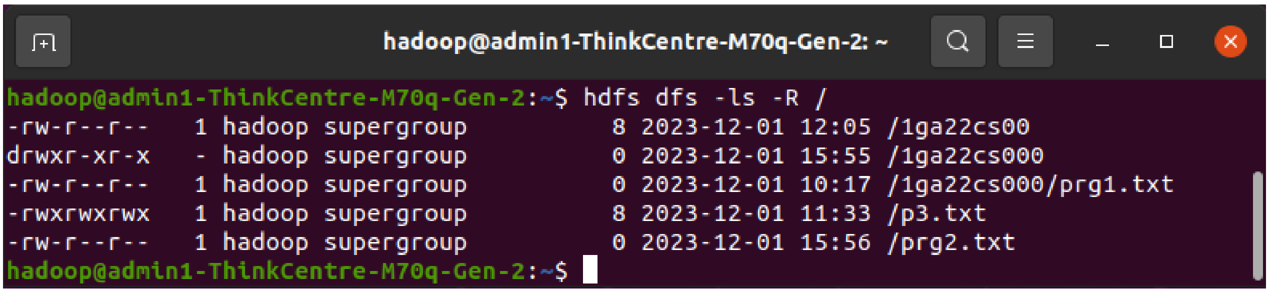


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -get /1ga22cs000/prg1.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ ls
Desktop    Downloads  Music  p3.txt  Pictures  Public                   Templates
Documents  hdfiles    op12   p4.txt  prg1.txt  spark-3.5.0-bin-hadoop3  Videos
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

mv

7. mv command to move files and directories from one directory to another or to rename a file or directory.

hdfs dfs -ls -R /

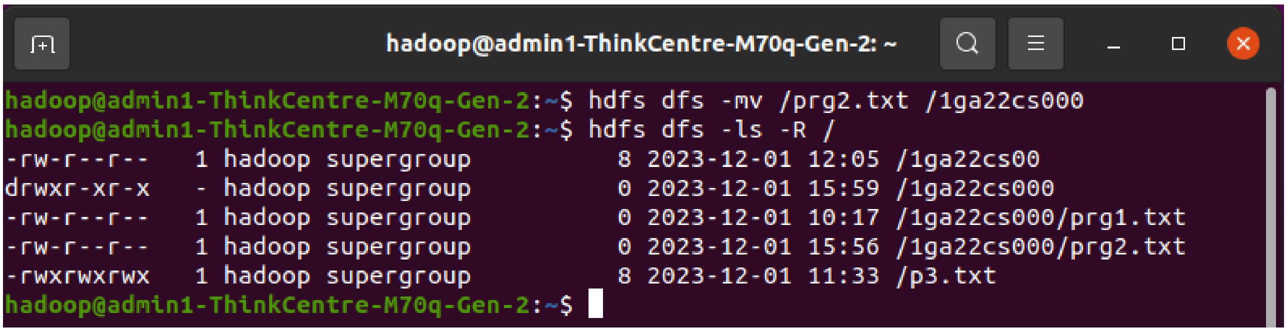


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls -R /
-rw-r--r--   1 hadoop supergroup          8 2023-12-01 12:05 /1ga22cs00
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 15:55 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000/prg1.txt
-rwxrwxrwx   1 hadoop supergroup          8 2023-12-01 11:33 /p3.txt
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 15:56 /prg2.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

Example: 1

hdfs dfs -mv /prg2.txt /1ga22cs000

hdfs dfs -ls -R /



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -mv /prg2.txt /1ga22cs000
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls -R /
-rw-r--r--   1 hadoop supergroup          8 2023-12-01 12:05 /1ga22cs00
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 15:59 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000/prg1.txt
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 15:56 /1ga22cs000/prg2.txt
-rwxrwxrwx   1 hadoop supergroup          8 2023-12-01 11:33 /p3.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

cp:

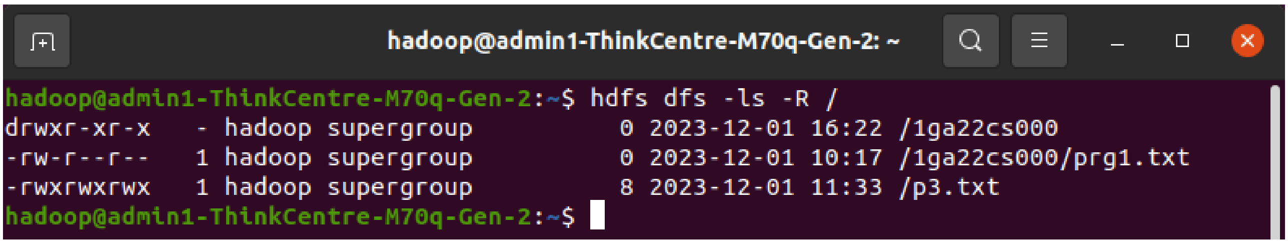
8. This command is used to copy files within hdfs.

Syntax:

hdfs dfs -cp <src(on hdfs)> <dest(on hdfs)>

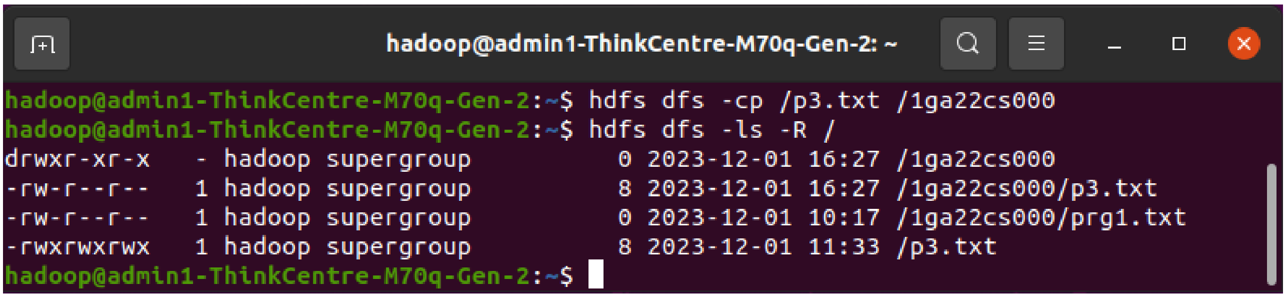
Example:

hdfs dfs -ls -R /



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls -R /
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 16:22 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000/prg1.txt
-rwxrwxrwx   1 hadoop supergroup          8 2023-12-01 11:33 /p3.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

hdfs dfs -cp /p5.txt /1ga22cs000



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -cp /p3.txt /1ga22cs000
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls -R /
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 16:27 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          8 2023-12-01 16:27 /1ga22cs000/p3.txt
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000/prg1.txt
-rwxrwxrwx   1 hadoop supergroup          8 2023-12-01 11:33 /p3.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

du:

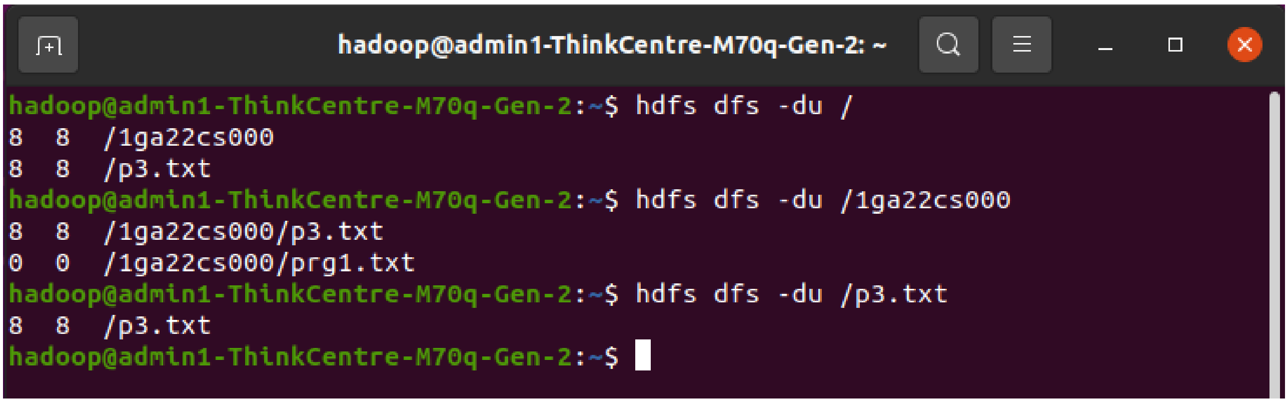
It will give the size of each file in directory.

Syntax:

hdfs dfs -du <dirName>

Example:

hdfs dfs -du /
hdfs dfs -du /1ga22cs000
hdfs dfs -du /p3.txt



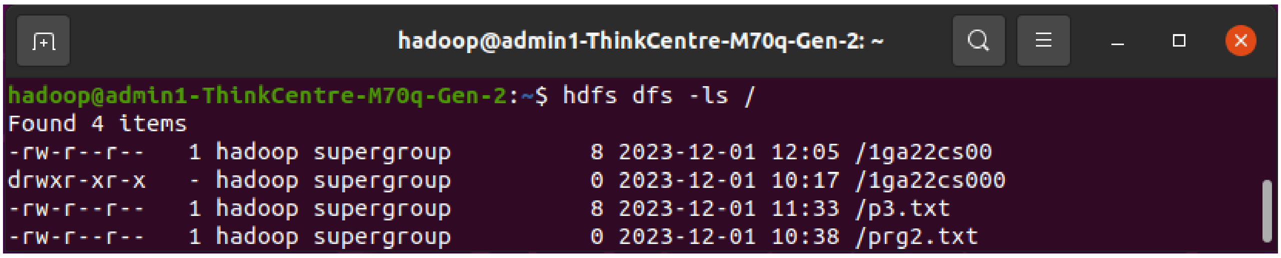
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -du /
8  8  /1ga22cs000
8  8  /p3.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -du /1ga22cs000
8  8  /1ga22cs000/p3.txt
0  0  /1ga22cs000/prg1.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -du /p3.txt
8  8  /p3.txt
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ 
```

chmod

the **chmod** command is used to change the access mode of a file.

Example:

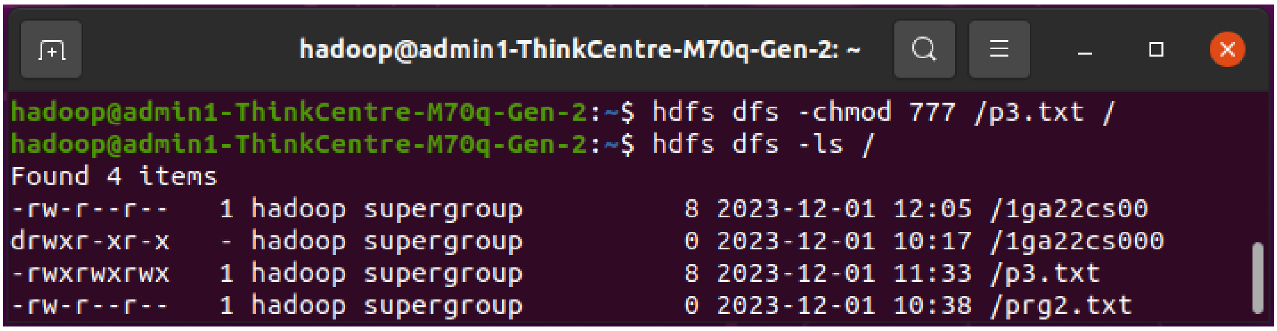
hdfs dfs -ls /



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls /
Found 4 items
-rw-r--r--   1 hadoop supergroup          8 2023-12-01 12:05 /1ga22cs00
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000
-rw-r--r--   1 hadoop supergroup          8 2023-12-01 11:33 /p3.txt
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:38 /prg2.txt
```

hdfs dfs -chmod 777 /p3.txt /

hdfs dfs -ls /



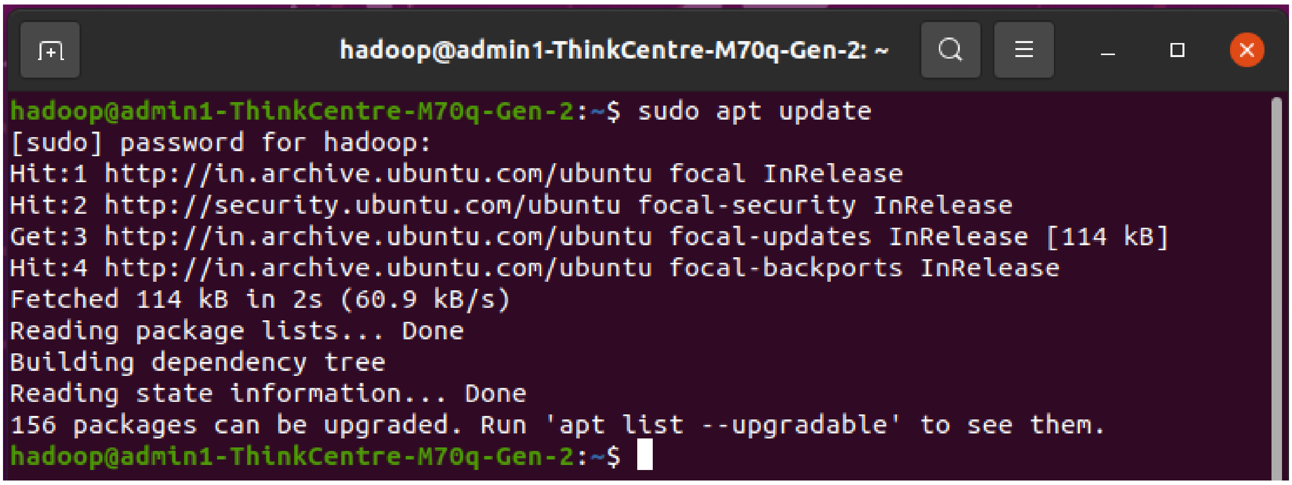
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -chmod 777 /p3.txt /
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ hdfs dfs -ls /
Found 4 items
-rw-r--r--   1 hadoop supergroup          8 2023-12-01 12:05 /1ga22cs00
drwxr-xr-x   - hadoop supergroup          0 2023-12-01 10:17 /1ga22cs000
-rwxrwxrwx   1 hadoop supergroup          8 2023-12-01 11:33 /p3.txt
-rw-r--r--   1 hadoop supergroup          0 2023-12-01 10:38 /prg2.txt
```

Demo 2 – Apache Spark Installation

Steps to Install Apache Spark and Configure a single node cluster on Ubuntu 18 0r 20

1. Use the following command to update your system before initiating a new installation.

```
sudo apt update
```



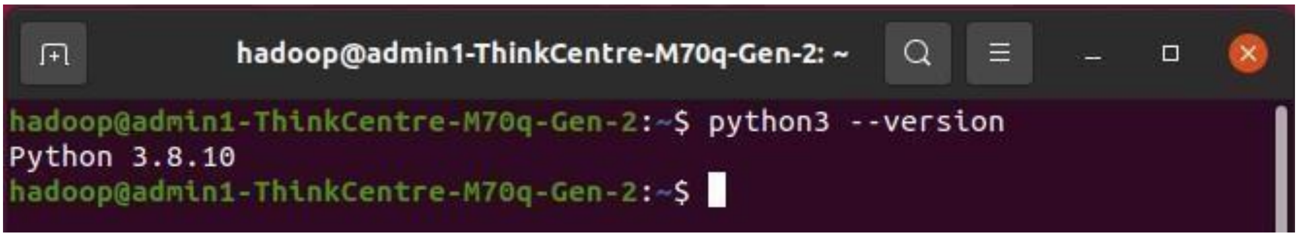
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo apt update
[sudo] password for hadoop:
Hit:1 http://in.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Hit:4 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease
Fetched 114 kB in 2s (60.9 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
156 packages can be upgraded. Run 'apt list --upgradable' to see them.
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

Install Python3 and Python3-pip

In Ubuntu 20.04 LTS, the python included in the base system is Python 3.8.

2. Find the current version of the python

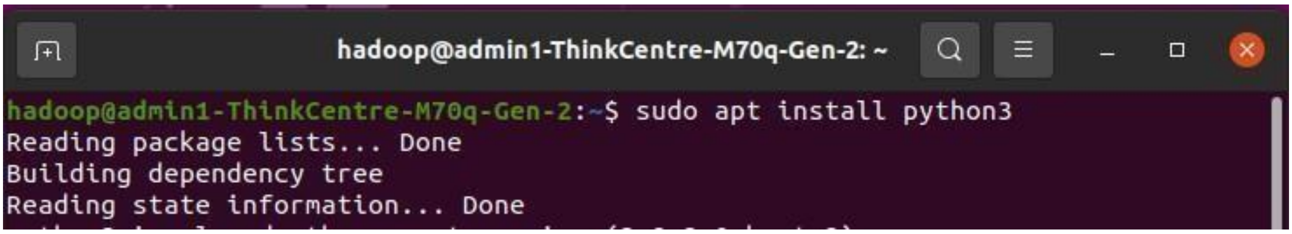
```
python  --version
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ python3 --version
Python 3.8.10
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

3. ***if not found, Install Python3.

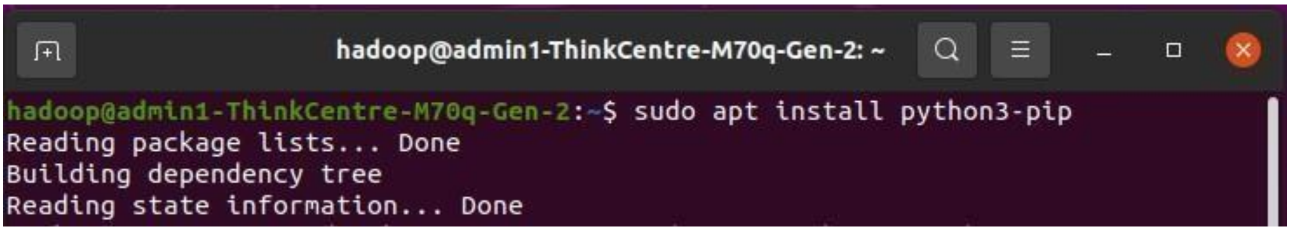
```
sudo apt install python3
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo apt install python3
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

4. Install python3-pip

```
sudo apt install python3-pip
```

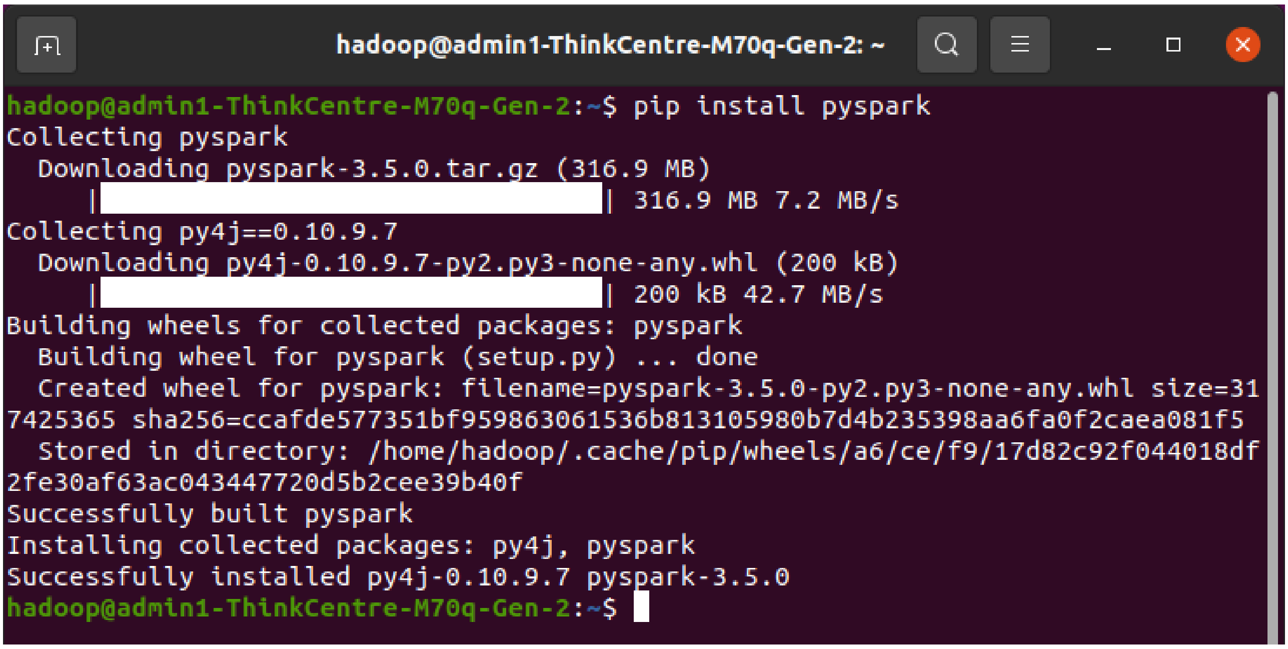


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo apt install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Install PySpark using pip

5. Install PySpark

```
pip install pyspark
```

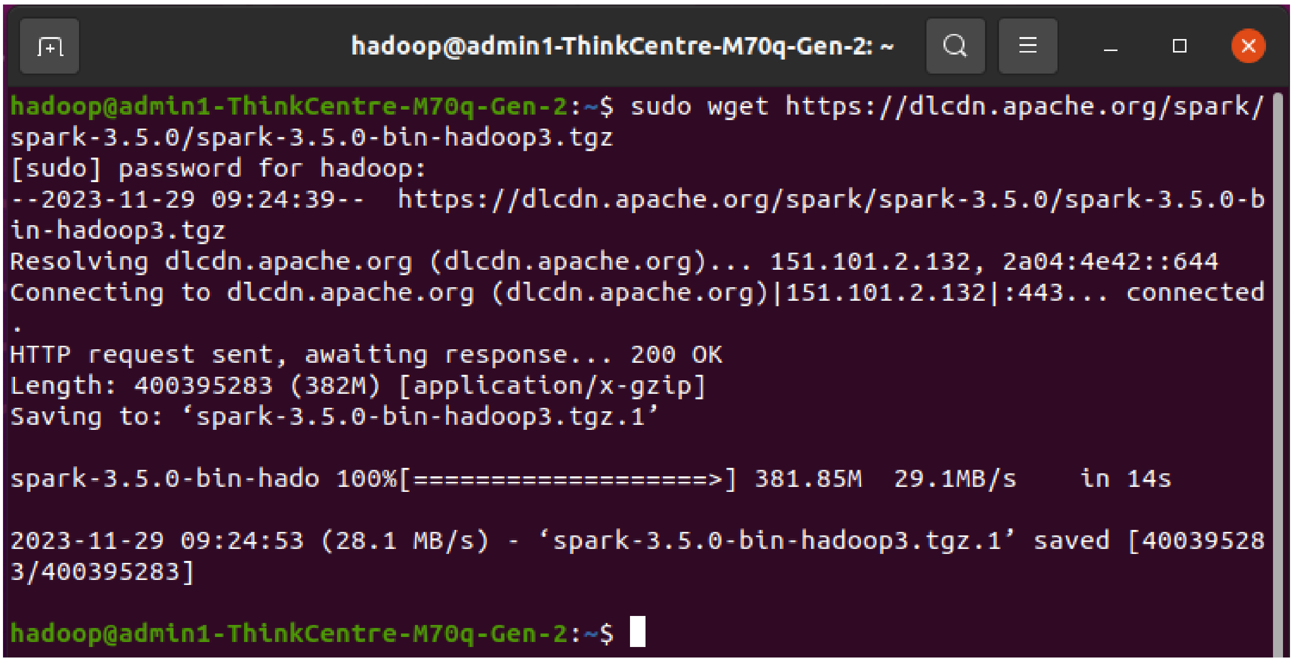


```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ pip install pyspark
Collecting pyspark
  Downloading pyspark-3.5.0.tar.gz (316.9 MB)
     |████████████████████████████████| 316.9 MB 7.2 MB/s
Collecting py4j==0.10.9.7
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
     |████████████████████████████████| 200 kB 42.7 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.5.0-py2.py3-none-any.whl size=31
7425365 sha256=ccafde577351bf959863061536b813105980b7d4b235398aa6fa0f2caea081f5
  Stored in directory: /home/hadoop/.cache/pip/wheels/a6/ce/f9/17d82c92f044018df
2fe30af63ac043447720d5b2cee39b40f
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.7 pyspark-3.5.0
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

Download and Set Up Spark

6. Use the following commands to download

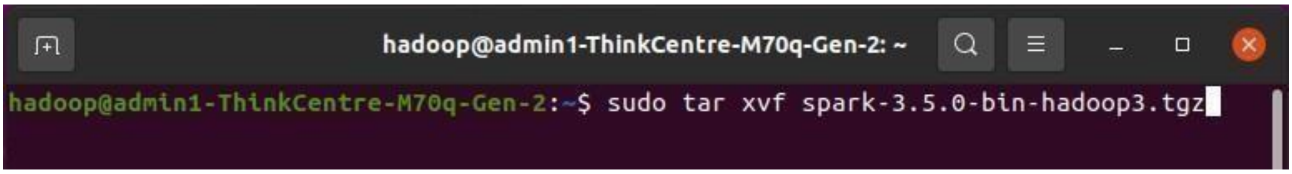
```
sudo wget https://dlcdn.apache.org/spark/spark-3.5.0/spark-3.5.0-bin-hadoop3.tgz
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo wget https://dlcdn.apache.org/spark/
spark-3.5.0/spark-3.5.0-bin-hadoop3.tgz
[sudo] password for hadoop:
--2023-11-29 09:24:39--  https://dlcdn.apache.org/spark/spark-3.5.0/spark-3.5.0-b
in-hadoop3.tgz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected
.
HTTP request sent, awaiting response... 200 OK
Length: 400395283 (382M) [application/x-gzip]
Saving to: ‘spark-3.5.0-bin-hadoop3.tgz.1’

spark-3.5.0-bin-hado 100%[===================>] 381.85M  29.1MB/s    in 14s

2023-11-29 09:24:53 (28.1 MB/s) - ‘spark-3.5.0-bin-hadoop3.tgz.1’ saved [40039528
3/400395283]

hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

7. Now, extract the saved archive using tar:

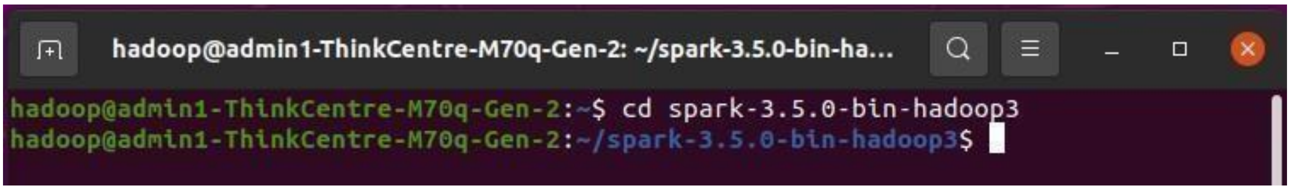
```
sudo tar xvf spark-3.5.0-bin-hadoop3.tgz
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ sudo tar xvf spark-3.5.0-bin-hadoop3.tgz
```

8. Move to spark-3.5.0-bin-hadoop3 directory

```
cd spark-3.5.0-bin-hadoop3
```

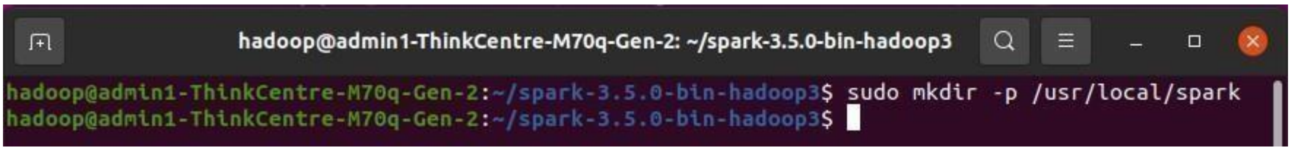


hadoop@admin1-ThinkCentre-M70q-Gen-2: ~/spark-3.5.0-bin-ha...

```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ cd spark-3.5.0-bin-hadoop3
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$
```

9. Create a spark directory

```
sudo mkdir -p /usr/local/spark
```

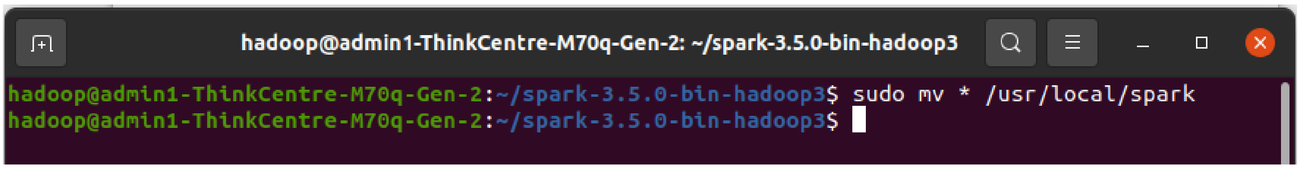


hadoop@admin1-ThinkCentre-M70q-Gen-2: ~/spark-3.5.0-bin-hadoop3

```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$ sudo mkdir -p /usr/local/spark
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$
```

10. Move the extracted folder to the / spark directory

```
sudo mv * /usr/local/spark
```


hadoop@admin1-ThinkCentre-M70q-Gen-2: ~/spark-3.5.0-bin-hadoop3

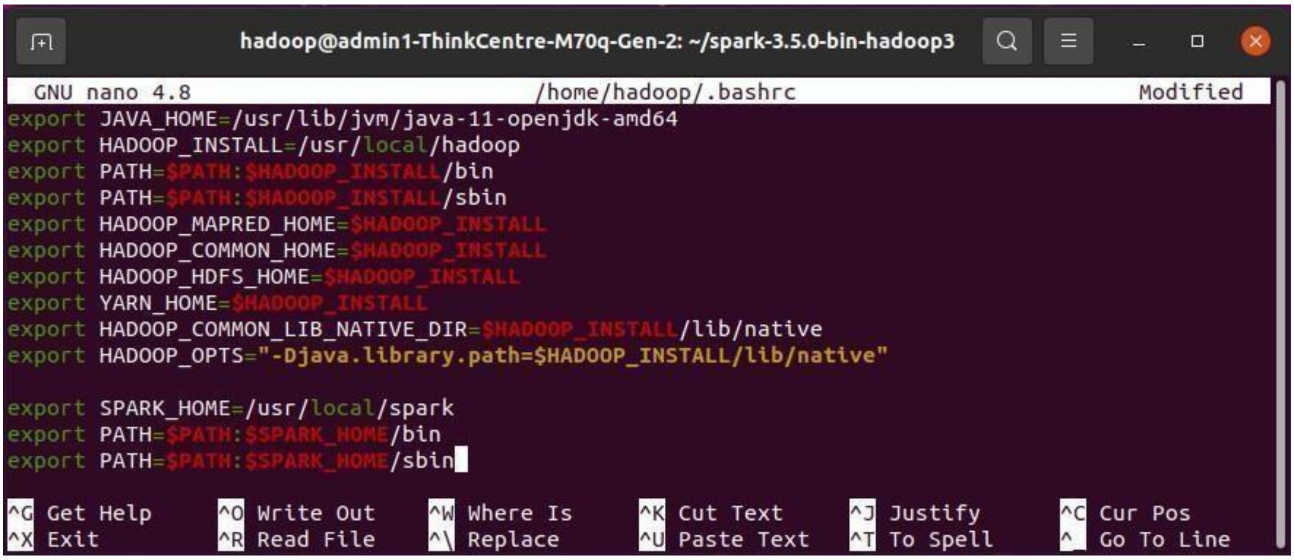
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$ sudo mv * /usr/local/spark
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$ 
```

11. Set Up Environment Variables

```
sudo nano ~/.bashrc
```

12. Add the following lines to your ~/.bashrc file to set up the required environment variables:

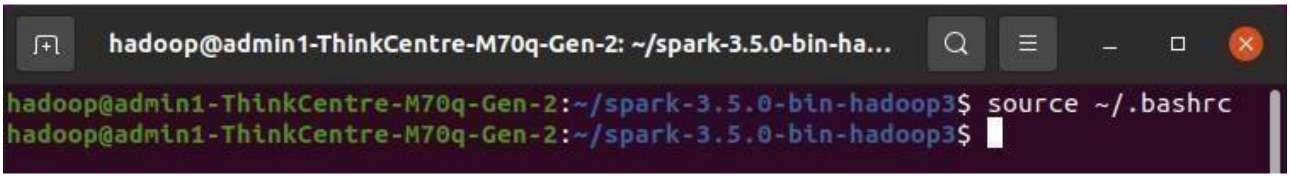
```
export SPARK_HOME=/usr/local/spark
export PATH=$PATH:$SPARK_HOME/bin
export PATH=$PATH:$SPARK_HOME/sbin
```


hadoop@admin1-ThinkCentre-M70q-Gen-2: ~/spark-3.5.0-bin-hadoop3

```
  GNU nano 4.8                    /home/hadoop/.bashrc                    Modified
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export HADOOP_INSTALL=/usr/local/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib/native"

export SPARK_HOME=/usr/local/spark
export PATH=$PATH:$SPARK_HOME/bin
export PATH=$PATH:$SPARK_HOME/sbin

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line
```

13. Source the updated ~/.bashrc file to apply the changes:

```
source ~/.bashrc
```


hadoop@admin1-ThinkCentre-M70q-Gen-2: ~/spark-3.5.0-bin-ha...

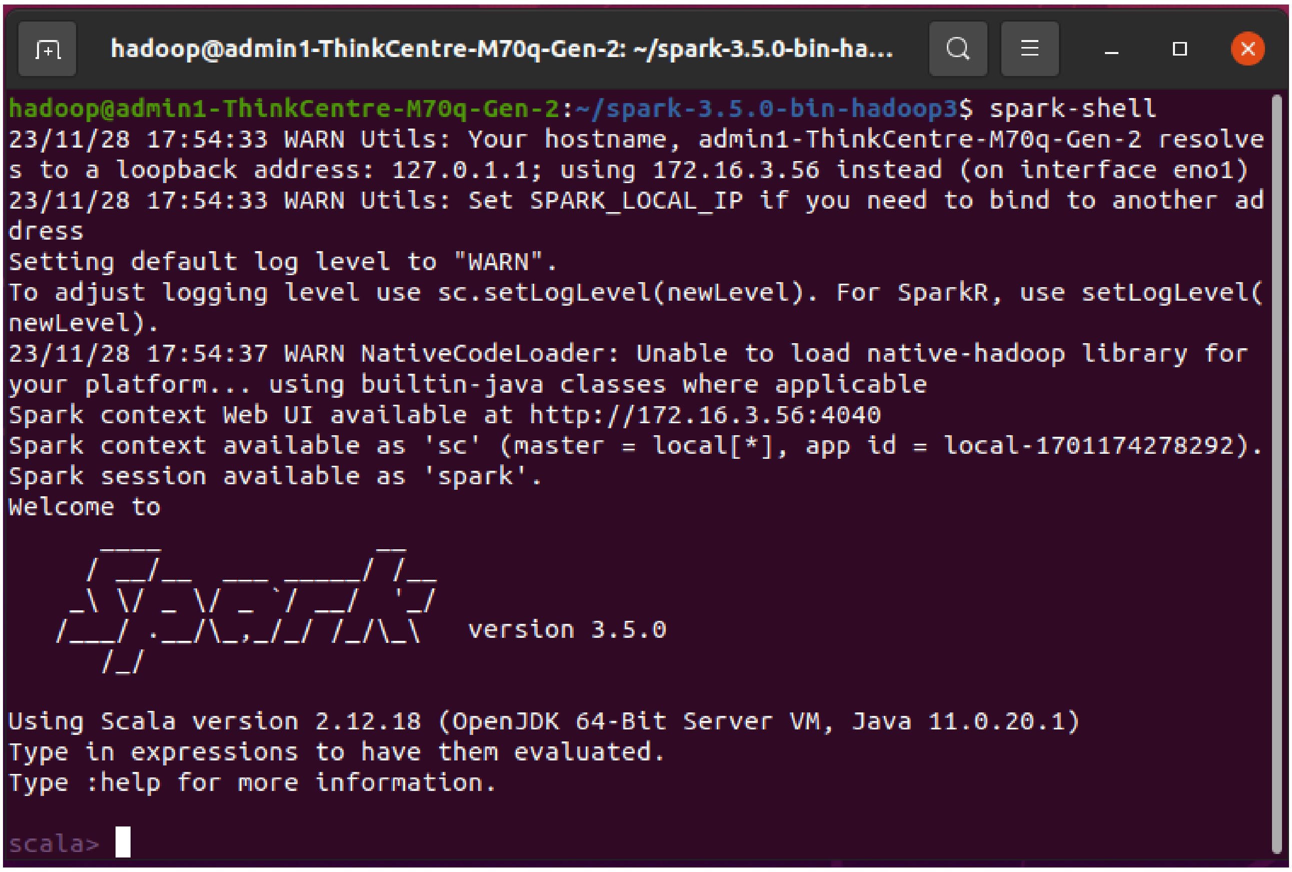
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$ source ~/.bashrc
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$ 
```

Spark Shell

14. This command loads the Spark and displays what version of Spark you are using.

```
spark-shell
```

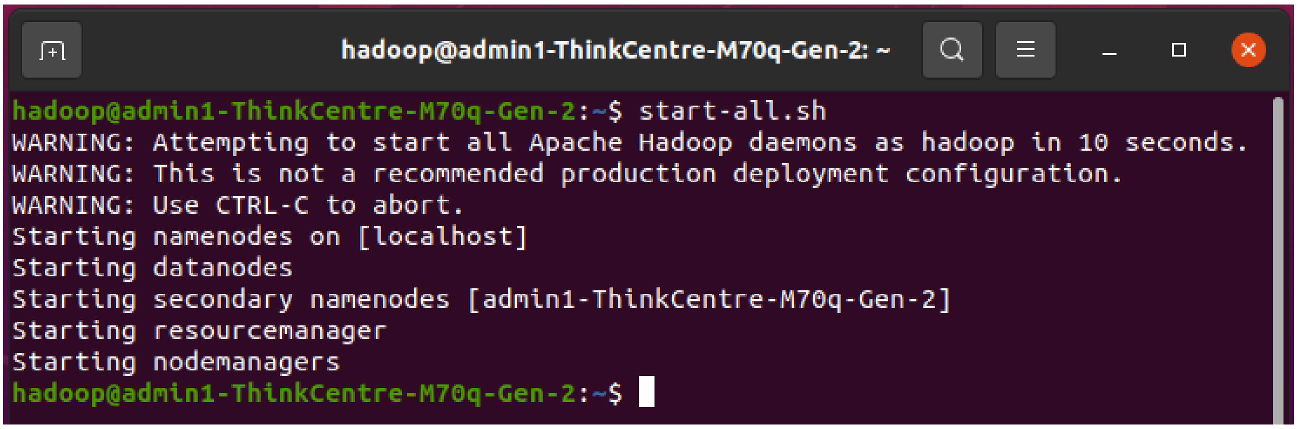
You will see an image like.


hadoop@admin1-ThinkCentre-M70q-Gen-2: ~/spark-3.5.0-bin-ha...

```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~/spark-3.5.0-bin-hadoop3$ spark-shell
23/11/28 17:54:33 WARN Utils: Your hostname, admin1-ThinkCentre-M70q-Gen-2 resolve
s to a loopback address: 127.0.1.1; using 172.16.3.56 instead (on interface eno1)
23/11/28 17:54:33 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another ad
dress
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(
newLevel).
23/11/28 17:54:37 WARN NativeCodeLoader: Unable to load native-hadoop library for
your platform... using builtin-java classes where applicable
Spark context Web UI available at http://172.16.3.56:4040
Spark context available as 'sc' (master = local[*], app id = local-1701174278292).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.0
      /_/

Using Scala version 2.12.18 (OpenJDK 64-Bit Server VM, Java 11.0.20.1)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

15. To start all the daemons and bring up your hadoop cluster use the below command:

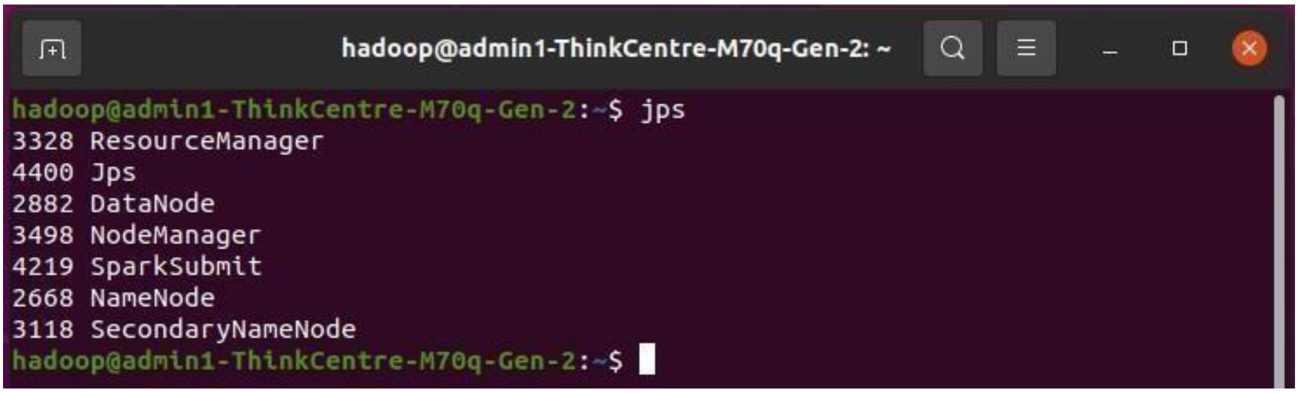
```
start-all.sh
```



```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [admin1-ThinkCentre-M70q-Gen-2]
Starting resourcemanager
Starting nodemanagers
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

16. The **‘jps’** command is used to check whether all the Hadoop processes are running or not.

```
jps
```



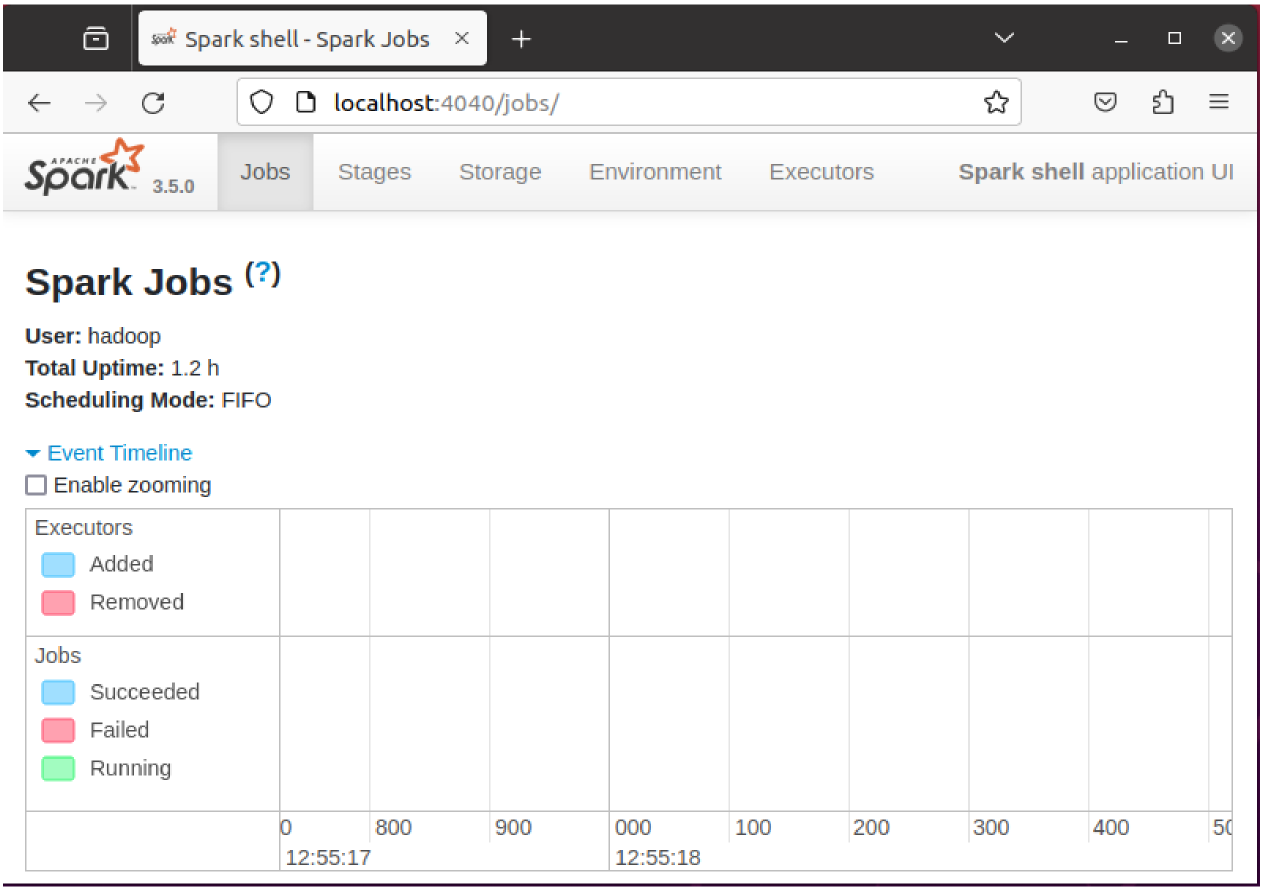
```
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$ jps
3328 ResourceManager
4400 Jps
2882 DataNode
3498 NodeManager
4219 SparkSubmit
2668 NameNode
3118 SecondaryNameNode
hadoop@admin1-ThinkCentre-M70q-Gen-2:~$
```

If Hadoop and spark have started successfully then an output of jps should show SparkSubmit,NameNode, NodeManager, SecondaryNameNode, ResourceManager, DataNode.

Spark Web UI

17. Apache Spark provides a suite of Web UIs
(Jobs, Stages, Tasks, Storage, Environment, Executors, and SQL)

```
http://localhost:4040
```



Spark Jobs (?)

User: hadoop
Total Uptime: 1.2 h
Scheduling Mode: FIFO

BIG DATA VIVA QUESTIONS

1. What do you know about the term “Big Data”?

Big Data is high-volume, high-velocity and high-variety information asset that requiresnew forms of processing for enhanced decision making, insightdiscovery and process optimization

2. What is structure data ?

Structured data conform and associate with data schemas and data models. Structured data are found in tables (rows and columns). Data in RDBMS. Structured data enables the following:

- Data insert, delete, update and append
- Indexing to enable faster data retrieval
- Scalability which enables increasing or decreasing capacities and data processing operations such as, storing, processing and analytics
- Transactions processing which follows ACID rules (Atomicity, Consistency, Isolation and Durability)
- Encryption and decryption for data security.

3. What is unstructured data ?

Semi-structured data does not conform and associate with formal data model structures, such as the relational database and table models. Examples of semi-structured data are XML and JSON documents.

4. What is semi structured data ?

Unstructured data does not possess data features such as a table or a database. Unstructured data are found in file types such as .TXT, .CSV, key- value pairs, e-mails. Unstructured data do not reveal relationships, hierarchy relationships or object-oriented features. The relationships, schema and features need to be separately established. Growth in data today can be characterized as mostly unstructured data

5. What are the 4 V’s of Big Data?

Volume: A considerable amount of data stored in data warehouses reflects the volume. The data may reach random heights; these large volumes of data need to be examined and processed. Which may exist up to or more than terabytes and petabytes.

Velocity: Velocity basically introduces the pace at which data is being produced in real-time. To give a simple example for recognition, imagine the rate atwhich Facebook, Instagram, or Twitter posts are generated per second, an hour or more.

Variety: Big Data comprises structured, unstructured, and semi-structured data collected from varied sources. This different variety of data requires very different and specific analysing and processing techniques with unique and appropriate algorithms.

Veracity: Data veracity basically relates to how reliable the data is, or ina fundamental way, we can define it as the quality of the data analysed.

6. How is Hadoop and Big Data related?

Hadoop is an open-source framework for saving, processing, and interpreting complex, disorganized data sets for obtaining insights and knowledge. So,that is how Hadoop and Big Data are related to each other.

7. Explain the importance of Hadoop technology in Big data analytics.

Since big data includes a large volume of data, i.e., structured, semi- structured, and unstructured data, analysing and processing this data is quite a big task. There was a need for a tool or technology to help processthe data at a rapid speed. Therefore, Hadoop is used because of its capabilities like storage, processing capability. Moreover, Hadoop is an open- source software. If you want to consider the cost, it's beneficial for business solutions.

The main reason for its popularity in recent years is that this framework permits distributed processing of enormous data sets using crosswise clusters of computerspracticing simple programming models.

8. Define HDFS and YARN, and talk about their respective components.

The HDFS is Hadoop's default storage unit and is responsible for storing different types of data in a distributed environment.

HDFS has the following two components:

Name Node – This is the master node that has the metadata information for all the data blocks in the HDFS.

Data Node – These are the nodes that act as slave nodes and are responsible for storing the data.

YARN, short for Yet Another Resource Negotiator, is responsible for managing resources and providing an execution environment for the said processes.

The two main components of YARN are –

Resource Manager – Responsible for allocating resources to respective Node Managers based on the needs.

Node Manager – Executes tasks on every Data Node

9. What is the purpose of the JPS command in Hadoop?

The JPS command is used for testing the working of all the Hadoop daemons.It specifically tests daemons like Name Node, Data Node, Resource Manager, Node Manager and more.

10. Explain the different features of Hadoop

Open-Source – Hadoop is an open-sourced platform. It allows the code to be rewritten or modified according to user and analytics requirements.

Scalability – Hadoop supports the addition of hardware resources to the new nodes.

Data Recovery – Hadoop follows replication which allows the recovery of data in the case of any failure.

Data Locality – This means that Hadoop moves the computation to the data and not the other way round. This way, the whole process speeds up.

11. Define Big Data and explain the Vs of Big Data.

Big Data can be defined as a collection of complex unstructured or semi-structured data sets which have the potential to deliver actionable insights.

The four Vs of Big Data are –

Volume – Talks about the amount of data

Variety – Talks about the various formats of data

Velocity – Talks about the ever increasing speed at which the data is growing

Veracity – Talks about the degree of accuracy of data available

12. How is Hadoop related to Big Data?

Hadoop is an open-source framework for storing, processing, and analyzing complex unstructured data sets for deriving insights and intelligence.

13. Define HDFS and YARN, and talk about their respective components.

The HDFS is Hadoop's default storage unit and is responsible for storing different types of data in a distributed environment.

HDFS has the following two components:

Name Node – This is the master node that has the metadata information for all the data blocks in the HDFS.

Data Node – These are the nodes that act as slave nodes and are responsible for storing the data.

YARN, short for Yet Another Resource Negotiator, is responsible for managing resources and providing an execution environment for the said processes.

The two main components of YARN are –

Resource Manager – Responsible for allocating resources to respective Node Managers based on the needs.

Node Manager – Executes tasks on every Data Node.

14. What do you mean by commodity hardware?

Commodity Hardware refers to the minimal hardware resources needed to run the ApacheHadoop framework. Any hardware that supports Hadoop's minimum requirements is known as 'Commodity Hardware.'

15. Define and describe the term FSCK.

FSCK stands for File System Check. It is a command used to run a Hadoop summary report that describes the state of HDFS. It only checks for errors and does not correct them. This command can be executed on either the whole system or a subset of files.

16. What is the purpose of the JPS command in Hadoop?

The JPS command is used for testing the working of all the Hadoop daemons. It specifically tests daemons like Name Node, Data Node, Resource Manager, Node Manager and more.

17. Name the different commands for starting up and shutting down Hadoop Daemons.

To start all the daemons:

```
./sbin/start-all.sh
```

To shut down all the daemons:

```
./sbin/stop-all.sh
```

18. Why do we need Hadoop for Big Data Analytics?

In most cases, Hadoop helps in exploring and analyzing large and unstructured data sets. Hadoop offers storage, processing and data collection capabilities that help in analytics.

19. Explain the different features of Hadoop.

Open-Source – Hadoop is an open-sourced platform. It allows the code to be rewritten or modified according to user and analytics requirements.

Scalability – Hadoop supports the addition of hardware resources to the new nodes.

Data Recovery – Hadoop follows replication which allows the recovery of data in the case of any failure. Data Locality – This means that Hadoop moves the computation to the data and not the other way round. This way, the whole process speeds up.

20. Define the Port Numbers for Name Node, Task Tracker and Job Tracker.

Name Node – Port 50070

Task Tracker – Port 50060

Job Tracker – Port 50030

21. What do you mean by indexing in HDFS?

HDFS indexes data blocks based on their sizes. The end of a data block points to the address of where the next chunk of data blocks gets stored. The Data Nodes store the blocks of data while Name Node stores these data blocks.

22. What are Edge Nodes in Hadoop?

Edge nodes refer to the gateway nodes which act as an interface between Hadoop cluster and the external network. These nodes run client applications and cluster management tools and are used as staging areas as well. Enterprise-class storage capabilities are required for Edge Nodes, and a single edge node usually suffices for multiple Hadoop clusters.

23. What are some of the data management tools used with Edge Nodes in Hadoop?

Oozy, Ambary, Pig and Flume are the most common data management tools that work with Edge Nodes in Hadoop.

24. Explain the core methods of a Reducer.

There are three core methods of a reducer. They are-

setup () – This is used to configure different parameters like heap size, distributed cache and input data.

reduce () – A parameter that is called once per key with the concerned reduce task

cleanup () – Clears all temporary files and called only at the end of a reducer task.

25. Talk about the different tombstone markers used for deletion purposes in Base.

There are three main tombstone markers used for deletion in Base. They are-

Family Delete Marker – For marking all the columns of a column family.

Version Delete Marker – For marking a single version of a single column.

Column Delete Marker – For marking all the versions of a single column.

Big Data Engineers: Myths vs. Realities

26. How can Big Data add value to businesses?

Big Data Analytics helps businesses to transform raw data into meaningful and actionableinsights that can shape their business strategies. The most important contribution of Big Data to business is data-driven business decisions. Big Data makes it possible for organizations to base their decisions on tangible information and insights.

27. How do you deploy a Big Data solution?

You can deploy a Big Data solution in three steps:

Data Ingestion – This is the first step in the deployment of a Big Data solution. You begin by collecting data from multiple sources, be it social media platforms, log files, business documents, anything relevant to your business. Data can either be extracted through real-time streaming or in batch jobs.

Data Storage – Once the data is extracted, you must store the data in a database. It can be HDFS or Base. While HDFS storage is perfect for sequential access, Base is ideal for random read/write access.

Data Processing – The last step in the deployment of the solution is data processing. Usually, data processing is done via frameworks like Hadoop, Spark, Map Reduce, Flank, and Pig, to namea few.

28. How is NFS different from HDFS?

The Network File System (NFS) is one of the oldest distributed file storage systems, while Hadoop Distributed File System (HDFS) came to the spotlight only recently after the upsurge of Big Data. The table below highlights some of the most notable differences between NFS and HDFS:

NFS HDFS

It can both store and process small volumes of data. It is explicitly designed to store and process Big Data.

The data is stored in dedicated hardware. Data is divided into data blocks that are distributed on the local drives of the hardware.

In the case of system failure, you cannot access the data. Data can be accessed even in the case of a system failure.

Since NFS runs on a single machine, there's no chance for data redundancy. HDFS runs on a cluster of machines, and hence, the replication protocol may lead to redundant data.

29. List the different file permissions in HDFS for files or directory levels.

The Hadoop distributed file system (HDFS) has specific permissions for files and directories. There are three user levels in HDFS – Owner, Group, and Others. For each of the user levels, there are three available permissions:

read (r)

write (w)

execute(x).

These three permissions work uniquely for files and directories.

For files –

The r permission is for reading a file

The w permission is for writing a file.

Although there's an execute(x) permission, you cannot execute HDFS files.

For directories –

The r permission lists the contents of a specific directory.

The w permission creates or deletes a directory.

The X permission is for accessing a child directory.

30. Elaborate on the processes that overwrite the replication factors in HDFS.

In HDFS, there are two ways to overwrite the replication factors – on file basis and on directory basis.

On File Basis

In this method, the replication factor changes according to the file using Hadoop FS shell. The following command is used for this:

$Hadoop fs – strep –w2/my/test file

Here, test file refers to the filename whose replication factor will be set to 2.

On Directory Basis

This method changes the replication factor according to the directory, as such, the replication factor for all the files under a particular directory, changes. The following command is used for this:

$Hadoop fs –strep –w5/my/testier

Here, testier refers to the name of the directory for which the replication factor and all the files contained within will be set to 5.

31. Name the three modes in which you can run Hadoop.

The three modes are:

Standalone mode – This is Hadoop's default mode that uses the local file system for both input and output operations. The main purpose of the standalone mode is debugging. It does not supportHDFS and also lacks custom configuration required for mapred-site.xml, core-site.xml, and hdfs-site.xml files.

Pseudo-distributed mode – Also known as the single-node cluster, the pseudo-distributed mode includes both Name Node and Data Node within the same machine. In this mode, all the Hadoop daemons will run on a single node, and hence, the Master and Slave nodes are the same.

Fully distributed mode – This mode is known as the multi-node cluster wherein multiple nodes function simultaneously to execute Hadoop jobs. Here, all the Hadoop daemons run on different nodes. So, the Master and Slave nodes run separately.

32. Explain "Overfitting."

Overfitting refers to a modelling error that occurs when a function is tightly fit (influenced) by a limited set of data points. Overfitting results in an overly complex model that makes it further difficult to explain the peculiarities or idiosyncrasies in the data at hand. As it adversely affects the generalization ability of the model, it becomes challenging to determine the predictivequotient of over fitted models. These models fail to perform when applied to external data (data that is not part of the sample data) or new datasets.

Overfitting is one of the most common problems in Machine Learning. A model is considered to be over fitted when it performs better on the training set but fails miserably on the test set. However, there are many methods to prevent the problem of overfitting, such as cross-validation, pruning, early stopping, regularization, and assembling.

33. What is Feature Selection?

Feature selection refers to the process of extracting only the required features from a specific dataset. When data is extracted from disparate sources, not all data is useful at all times – different business needs call for different data insights. This is where feature selection comes into identify and select only those features that are relevant for a particular business requirement orstage of data processing.

The main goal of feature selection is to simplify ML models to make their analysis and interpretation easier. Feature selection enhances the generalization abilities of a model and eliminates the problems of dimensionality, thereby, preventing the possibilities of overfitting. Thus, feature selection provides a better understanding of the data under study, improves the prediction performance of the model, and reduces the computation time significantly.

Feature selection can be done via three techniques:

Filters method

In this method, the features selected are not dependent on the designated classifiers. A variable ranking technique is used to select variables for ordering purposes. During the classification process, the variable ranking technique takes into consideration the importance and usefulness of a feature. The Chi-Square Test, Variance Threshold, and Information Gain are some examples of the filters method.

Wrappers method

In this method, the algorithm used for feature subset selection exists as a 'wrapper' around the induction algorithm. The induction algorithm functions like a 'Black Box' that produces aclassifier that will be further used in the classification of features. The major drawback or limitation of the wrappers method is that to obtain the feature subset, you need to perform heavy computation work. Genetic Algorithms, Sequential Feature Selection, and Recursive Feature Elimination are examples of the wrappers method.

Embedded method The embedded method combines the best of both worlds – it includes the best features of the filters and wrappers methods. In this method, the variable selection is done during the training process, thereby allowing you to identify the features that are the most accurate for a given model. L1 Regularization Technique and Ridge Regression are two popular examples of the embedded method.

34. Define "Outliers."

An outlier refers to a data point or an observation that lies at an abnormal distance from other values in a random sample. In other words, outliers are the values that are far removed from the group; they do not belong to any specific cluster or group in the dataset. The presence of outliers usually affects the behavior of the model – they can mislead the training process of ML algorithms. Some of the adverse impacts of outliers include longer training time, inaccurate models, and poor outcomes. However, outliers may sometimes contain valuable information. This is why they must be investigated thoroughly and treated accordingly.

35. Name some outlier detection techniques.

Here are six outlier detection methods:

Extreme Value Analysis – This method determines the statistical tails of the data distribution. Statistical methods like 'z-scores' on univariate data are a perfect example of extreme value analysis.

Probabilistic and Statistical Models – This method determines the 'unlikely instances' from a 'probabilistic model' of data. A good example is the optimization of Gaussian mixture models using 'expectation-maximization'.

Linear Models – This method models the data into lower dimensions. Proximity-based Models – In this approach, the data instances that are isolated from the data group are determined by Cluster, Density, or by the Nearest Neighbor Analysis.

Information-Theoretic Models – This approach seeks to detect outliers as the bad data instances that increase the complexity of the dataset.

High-Dimensional Outlier Detection – This method identifies the subspaces for the outliers according to the distance measures in higher dimensions.

36. Explain Rack Awareness in Hadoop.

Rach awareness is an algorithm that identifies and selects Data Nodes closer to the Name Node based on their rack information. It is applied to the Name Node to determine how data blocks and their replicas will be placed. During the installation process, the default assumption is that allnodes belong to the same rack.

Rack awareness helps to:

Improve data reliability and accessibility.

Improve cluster performance.
Improve network bandwidth.
Keep the bulk flow in-rack as and when possible.
Prevent data loss in case of a complete rack failure.

37. Can you recover a Name Node when it is down? If so, how?

Yes, it is possible to recover a Name Node when it is down. Here's how you can do it:
Use the Silage (the file system metadata replica) to launch a new Name Node.
Configure Data Nodes along with the clients so that they can acknowledge and refer to newly started Name Node.
When the newly created Name Node completes loading the last checkpoint of the Silage (that has now received enough block reports from the Data Nodes) loading process, it will be ready to start serving the client.
However, the recovery process of a Name Node is feasible only for smaller clusters. For large Hadoop clusters, the recovery process usually consumes a substantial amount of time, thereby making it quite a challenging task.

38. Name the configuration parameters of a Map Reduce framework.

The configuration parameters in the Map Reduce framework include:
The input format of data.
The output format of data.
The input location of jobs in the distributed file system.
The output location of jobs in the distributed file system.
The class containing the map function
The class containing the reduce function The JAR file containing the mapper, reducer, and driver classes.

39. What is a Distributed Cache? What are its benefits?

Distributed cache in Hadoop is a service offered by the Map Reduce framework used for caching files. If a file is cached for a specific job, Hadoop makes it available on individual Data Nodes both in memory and in system where the map and reduce tasks are simultaneously executing. Thisallows you to quickly access and read cached files to populate any collection (like arrays, hash maps, etc.) in a code.
Distributed cache offers the following benefits:
It distributes simple, read-only text/data files and other complex types like jars, archives, etc.
It tracks the modification timestamps of cache files which highlight the files that should not be modified until a job is executed successfully.

40. What is a Sequence File in Hadoop?

In Hadoop, a Sequence File is a flat-file that contains binary key-value pairs. It is most commonly used in Map Reduce I/O formats. The map outputs are stored internally as a SequenceFile which provides the reader, writer, and sorter classes.
There are three Sequence File formats:
Uncompressed key-value records
Record compressed key-value records (only 'values' are compressed).
Block compressed key-value records (here, both keys and values are collected in 'blocks' separately and then compressed).

41. Explain the role of a Job Tracker.

The primary function of the Job Tracker is resource management, which essentially means managing the Task Trackers. Apart from this, Job Tracker also tracks resource availability and handles task life cycle management (track the progress of tasks and their fault tolerance).

Some crucial features of the Job Tracker are:

It is a process that runs on a separate node (not on a Data Node).

It communicates with the Name Node to identify data location.

It tracks the execution of Map Reduce workloads.

It allocates Task Tracker nodes based on the available slots.

It monitors each Task Tracker and submits the overall job report to the client.

It finds the best Task Tracker nodes to execute specific tasks on particular nodes.

42. Name the common input formats in Hadoop.

Hadoop has three common input formats:

Text Input Format – This is the default input format in Hadoop.

Sequence File Input Format – This input format is used to read files in a sequence.

Key-Value Input Format – This input format is used for plain text files (files broken into lines).

43. What is the need for Data Locality in Hadoop?

In HDFS, datasets are stored as blocks in Data Nodes in the Hadoop cluster. When a Map Reduce job is executing, the individual Mapper processes the data blocks (Input Splits). If the data does is not present in the same node where the Mapper executes the job, the data must be copied from the Data Node where it resides over the network to the Mapper Data Node.

When a Map Reduce job has over a hundred Mappers and each Mapper Data Node tries to copy the data from another Data Node in the cluster simultaneously, it will lead to network congestion, thereby having a negative impact on the system's overall performance. This is whereData Locality enters the scenario. Instead of moving a large chunk of data to the computation, Data Locality moves the data computation close to where the actual data resides on the Data Node. This helps improve the overall performance of the system, without causing unnecessary delay.

44. What are the steps to achieve security in Hadoop?

In Hadoop, Kerberos – a network authentication protocol – is used to achieve security. Kerberos is designed to offer robust authentication for client/server applications via secret-keycryptography. When you use Kerberos to access a service, you have to undergo three steps, each of which involves a message exchange with a server. The steps are as follows:

Authentication – This is the first step wherein the client is authenticated via the authentication server, after which a time-stamped TGT (Ticket Granting Ticket) is given to the client.

Authorization – In the second step, the client uses the TGT for requesting a service ticket from the TGS (Ticket Granting Server).

Service Request – In the final step, the client uses the service ticket to authenticate themselves to the server.

45. What does a block in the Hadoop Distributed File System (HDFS) mean?

When a file is placed in HDFS, the entire file system is broken down into a collection of blocks, and HDFS is completely unaware of the contents of the file. Hadoop requires blocks to be 128MB in size. Individual files may have a different value for this.