Week-01

Aim: a) Demonstrate all HDFs or Hadoop commands in Ubuntu.

b) Working with hadoop file system i) Reading ii) Writing iii) copying.

→ To check Hadoop version

» hadoop version ⟹ (3.3.6)

→ Start the hadoop services

» start-all.sh

→ To check if Hadoop services are running

>> jps

```
4656  Resource Manager
4785  Node Manager.
4387  Secondary Name Node.
5189  JPS
4014  Name Node
4175  Data Node.
```

→ ls command is used to list all the files

» hdfs dfs -ls /

→ mkdir command used to create a new directory

» hdfs dfs -mkdir /bda33

→ touchz to create an empty file

» hdfs dfs -touchz /file1.txt.

→ copying file from the Local system

>> hdfs dfs -copyfromLocal /home/hadoop/input.txt /bda73

→ view the contents from the file using cat command

>> hdfs dfs -cat \60\22\7flom bda73/file1.txt

olp: Hello

→ To copy within HDFS use cp command

>> hdfs dfs -mkdir/copied

>> hdfs dfs -cp /bda73/input.txt /copied.

→ To move within HDFS use mv command

>> hdfs dfs -mv /bda73/file1.txt /copied

→ creating a file using put command.

>> echo "Hello Welcome" | hadoop fs.-appendToFile -bda73/input.txt.

>> hdfs dfs -cat bda73/input.txt.

olp: Hello Welcome.

→ copying file from HDFS to Local.

>> hdfs dfs -get /bda73/file1.txt /home/hadoop/bda73

→ du command

>> hdfs dfs -du/

20 20 /bda73/file1.txt

29 29 /bda73/input.txt

→ df command (disk free)

» hdfs dfs -df /

→ tail

>> hdfs dfs -tail /bdata3/input.txt

→ head

>> hdfs dfs -head /bdata3/input.txt

olp: Hello.

Aim:- To execute word count Application using Map-Reduce on a single Node cluster.

Procedure:-

In Local file system, create two files wcmap.py and wcred.py using gedit in your folder (bda73).

```
>> cd /home/hadoop1/bda73.
>> gedit wcmap.py.
import sys
for line in sys.stdin:
    line = line.strip()
    words = line.split()
    for word in words:
        print("%s\t%s" %(word,1))
>> gedit wcred.py
from operator import itemgetter
for line in sys.stdin:
    line = line.strip()
    word, count = line.split('\t', 1)
    try:
        count = int(count)
    except ValueError:
        continue
    if current_word == word:
        current_count += count
```

```
        else:
            if current_word:
                print("%s\t%s" %(current_word, current_count))
            current_count = count
            current_word = word
    if current_word == word:
        print("%s\t%s" %(current_word, current_count))
```

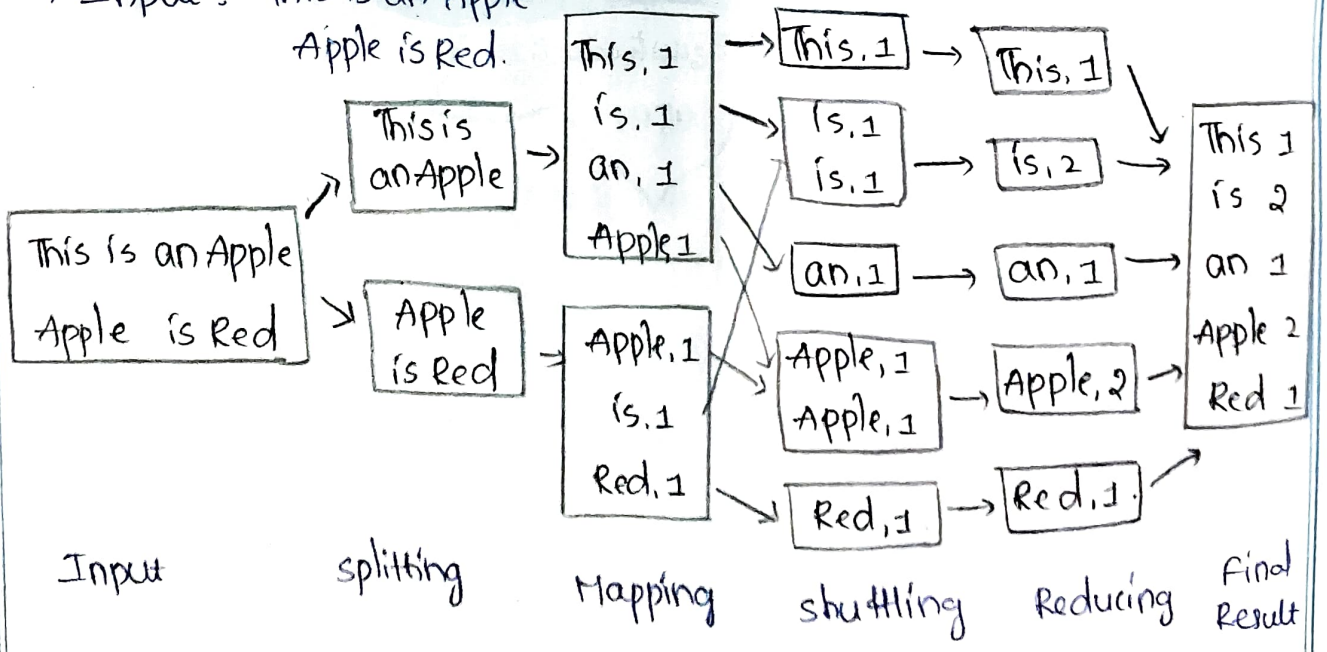
Example:-

→ Map-Reduce consists of two phases Map phase and Reduce phase.

→ Map phase: Splits a text file into words and assigns each word a count of 1.

→ Reduce phase: Aggregates counts for each word, producing the final word frequencies.

→ Input: This is an Apple
Apple is Red.



Sheet No.:

Date:

Execution:

→ To Execute this

>> cd /home/hadoop/bda+3

>> ls

-This gives the details of files wcmap.py wcred.py

>> python3 wcmap.py <input.txt

This command performs mapping by running wcmap.py file

>> cat input.txt | python3 wcmap.py | sort -k1.1 |
python3 wcred.py

This performs reducing by running the file wcred.py

→ To see the output in HDFS (Hadoop). Open Hadoop in bda+3 directory in local system and run below commands.

>> bash

>> start-all.sh

>> jps

>> hadoop fs -mkdir /wcount

>> hadoop fs -put input.txt /wcount

>> hadoop jar /home/hadoop/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.6.jar -file wcmap.py -mapper "python3 wcmap.py" -file wcred.py -reducer "python3 wcred.py" -input /wcount/input.txt -output /wcount/output.txt.

→ This performs word count and stores o/p in output.txt file.

olpi

T	1
H	1
I	1
S	1
:	:
R	1
e	1
d	1

olpi

This	1
is	2
an	1
Apple	2
Red	1