cat input.txt  |pyhton3 mapper.py

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

hadoop jar '/home/hdoop/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.6.jar' -file mapper.py -mapper mapper.py -file reducer.py -reducer reducer.py -input /home/hdoop/210968054/ri.txt -output /home/hdoop/210968054/out

hdfs dfs -cat /home/hdoop/210968054/out/part-00000

hdfs dfs -get /home/hdoop/210968054/out/part-00000 /home/hdoop/210968054/

**Sure, here are some common Linux file operations and their corresponding commands:**

1. **Create a file**: touch filename
2. **Write to a file**: echo "content" > filename
3. **Read a file**: cat filename
4. **Append to a file**: echo "more content" >> filename
5. **Delete a file**: rm filename
6. **Rename a file**: mv oldname newname
7. **Copy a file**: cp source destination
8. **Change file permissions**: chmod permissions filename
9. **Change file ownership**: chown user:group filename

**Here are the Hadoop File System (HDFS) commands for the operations you mentioned. Please replace directory, file, localpath, hdfspath, newpath, replication\_factor, user, group, and permission with your actual values:**

1. **Create a directory in HDFS**: hadoop fs -mkdir /user/hadoop/directory
2. **Create an empty file**: hadoop fs -touchz /user/hadoop/file
3. **Copy files/folders from local file system to HDFS**: hadoop fs -copyFromLocal localpath hdfspath
4. **Print file contents**: hadoop fs -cat /user/hadoop/file
5. **Copy files/folders from HDFS to local file system**: hadoop fs -copyToLocal hdfspath localpath
6. **Move file from local to HDFS**: hadoop fs -moveFromLocal localpath hdfspath
7. **Copy files within HDFS**: hadoop fs -cp hdfspath newpath
8. **Move files within HDFS**: hadoop fs -mv hdfspath newpath
9. **Size of each file in directory**: hadoop fs -du -s -h /user/hadoop/directory/\*
10. **Total size of directory/file**: hadoop fs -du -s -h /user/hadoop/directory
11. **Last modified time of directory or path**: hadoop fs -ls /user/hadoop/directory
12. **Change the replication factor of a file/directory in HDFS**: hadoop fs -setrep -w replication\_factor /user/hadoop/file
13. **List the contents of a directory in HDFS**: hadoop fs -ls /user/hadoop/directory
14. **Remove a file from HDFS**: hadoop fs -rm /user/hadoop/file
15. **Change File Permissions**: hadoop fs -chmod permission /user/hadoop/file
16. **Change File Ownership**: hadoop fs -chown user:group /user/hadoop/file
17. **Checksum Calculation**: hadoop fs -checksum /user/hadoop/file
18. **File Concatenation**: hadoop fs -getmerge /user/hadoop/directory localfile
19. **File Compression/Decompression**: This depends on the type of compression. For example, for gzip files: hadoop fs -text /user/hadoop/file.gz
20. **File Block Location Information**: hadoop fsck /user/hadoop/file -files -blocks -locations

**Consider the text file (consider larger file size) of your choice and perform word count using MapReduce technique**

#!/usr/bin/python3

import sys

for line in sys.stdin:

    line = line.strip()

    words = line.split()

    for word in words:

        print(word,"\t",1)

#!/usr/bin/python3

import sys

word = None

current\_word = None

current\_count = 0

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(current\_word,"\t",current\_count)

        current\_count = count

        current\_word = word

if current\_word == word:

    print(current\_word,"\t",current\_count)

**Perform Matrix operations using MapReduce by considering 3 \* 3 matrix and perform following operations**

#!/usr/bin/python3

import sys

for line in sys.stdin:

    line = line.strip()

    mat,i,j,val = line.split(',')

    print('%s\t%s\t%s\t%s' % (mat,i,j,val))

#!/usr/bin/python3

import sys

A = [[0,0,0],[0,0,0],[0,0,0]]

B = [[0,0,0],[0,0,0],[0,0,0]]

for line in sys.stdin:

    line = line.strip()

    mat,i,j,val = line.split('\t')

    if mat=='a':

        A[int(i)][int(j)] = int(val)

    if mat=='b':

        B[int(i)][int(j)] = int(val)

multresult = [[0,0,0],[0,0,0],[0,0,0]]

addresult = [[0,0,0],[0,0,0],[0,0,0]]

subresult = [[0,0,0],[0,0,0],[0,0,0]]

atrans = [[0,0,0],[0,0,0],[0,0,0]]

btrans = [[0,0,0],[0,0,0],[0,0,0]]

for i in range(3):

    for j in range(3):

        for k in range(3):

            multresult[i][k]+=A[i][j]\*B[j][k]

for i in range(3):

    for j in range(3):

        sum=A[i][j]+B[i][j]

        addresult[i][j]=sum

for i in range(3):

    for j in range(3):

        subresult[i][j]=A[i][j]-B[i][j]

        transpose = [[0,0,0],[0,0,0],[0,0,0]]

for i in range(3):

    for j in range(3):

        atrans[j][i] = A[i][j]

        btrans[j][i] = B[i][j]

print("The addition result\n")

print(addresult)

print("The subtraction result\n")

print(subresult)

print("The multiplication result\n")

print(multresult)

print("The A transpose result\n")

print(atrans)

print("The B transpose result\n")

print(btrans)

**Create a text file containing the 20 student details such as registration number, name and marks (ex: 1001, john,45 ) .Write a MapReduce program to sort data by student name.**

#!/usr/bin/python3

import sys

for line in sys.stdin:

    line = line.strip()

    regno,name,marks = line.split(',')

    print('%s\t%s\t%s'%(regno,name,marks))

#!/usr/bin/python3

import sys

reg = []

name1 = []

mrk = []

cnt=0

for line in sys.stdin:

    emp = []

    line = line.strip()

    regno,name,marks = line.split('\t')

    reg.append(regno)

    name1.append(name)

    mrk.append(marks)

    cnt+=1;

x=0

y=0

for x in range (cnt):

    for y in range(cnt-x-1):

        if name1[y]>name1[y+1]:

            temp = name1[y]

            name1[y] = name1[y+1]

            name1[y+1] = temp

            temp = reg[y]

            reg[y] = reg[y+1]

            reg[y+1] = temp

            temp = mrk[y]

            mrk[y] = mrk[y+1]

            mrk[y+1] = temp

for i in range(cnt):

    print(reg[i],'\t',name1[i],'\t',mrk[i])

**Write a MapReduce program to find unit wise salary for the bellow given data.**

#!/usr/bin/python3

import sys

for line in sys.stdin:

    line = line.strip()

    eno,ename,unit,desig,sal = line.split()

    print(unit,'\t',sal)

#!/usr/bin/python3

import sys

u = {}

cnt=0

for line in sys.stdin:

    line = line.strip()

    unit,sal = line.split()

    if cnt==0:

        cnt+=1

    else:

        if unit not in u:

            u[unit] = list()

            u[unit].append(int(sal))

        else:

            u[unit].append(int(sal))

for x in u:

    print(x,sum(u[x]))

#!/usr/bin/python3

import sys

cnt = 0

for line in sys.stdin:

    line = line.strip()

    empno,empname,unit,designation,salary = line.split(',')

    if cnt==0:

        cnt+=1

    else:

        print('%s\t%s\t%s\t%s\t%s\t'%(empno,empname,unit,designation,salary))

#!/usr/bin/python3

import sys

unitwise = {}

for line in sys.stdin:

    line = line.strip()

    empno,empname,unit,desig,salary = line.split('\t')

    if unit in unitwise:

        sal = unitwise[unit]

        sal+=int(salary)

        unitwise[unit] = sal

    else:

        unitwise[unit] = int(salary)

for unit in unitwise:

    print('%s\t%s'%(unit,unitwise[unit]))

**Consider the following sample text file to compute the the average, minimum and maximum recorded temperature by year wise using concept of Map Reduce.**

#!/usr/bin/python3

import sys

for line in sys.stdin:

    line = line.strip()

    year,temp = line.split()

    print(year,'\t',temp)

#!/usr/bin/python3

import sys

dic={}

for line in sys.stdin:

    line = line.strip()

    year,temp = line.split('\t')

    if year not in  dic:

        dic[year] = list()

        dic[year].append(int(temp))

    else:

        dic[year].append(int(temp))

ans={}

print("year\tmin\tmax\tavg")

for x in dic:

    print(x,'\t',min(dic[x]),'\t',max(dic[x]),'\t',sum(dic[x])/len(dic[x]))