Spark programs

bash file

export JAVA\_HOME=$(readlink -f $(which javac) | awk 'BEGIN {FS="/bin"} {print $1}')

if ! command -v spark-shell --version &> /dev/null

then

export PATH=$(echo $PATH):$(pwd)/bin

fi

program 1

**1.Write a spark map-reduce to analyze the given weather report data and to generate a report with cities having maximum temperature for a particular year**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (int(x[15:19]),int(x[87:92])))

maxi=temp.reduceByKey(lambda a,b:a if a>b else b)

maxi.saveAsTextFile(sys.argv[2])

**2.Write a spark map-reduce program to analyze the given Insurance data and generate a statistics report with the construction building name and the count of building.**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[16],1))

data=temp.countByKey()

dd=sc.parallelize(data.items())

dd.saveAsTextFile(sys.argv[2])

**3.Write a spark map-reduce program to analyze the given employee record data and generate a statistics report with the total Sales for female and male employees**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split('\t')[3],float(x.split('\t')[8])))

total=temp.reduceByKey(lambda a,b : a+b)

total.saveAsTextFile(sys.argv[2])

**4. Write a spark map-reduce program to analyze the given sales records over a period and generate data about the country’s total sales, and the total number of the products**

import sys

if(len(sys.argv)!=3):

print("Provide Input File and Output Directory")

sys.exit(0)

from pyspark import SparkContext

sc =SparkContext()

f = sc.textFile(sys.argv[1])

temp=f.map(lambda x: (x.split(',')[7],1))

data=temp.countByKey()

dd=sc.parallelize(data.items())

dd.saveAsTextFile(sys.argv[2])

pig programs

export JAVA\_HOME=$(readlink -f $(which javac) | awk 'BEGIN {FS="/bin"} {print $1}')  
if ! command -v pig &> /dev/null  
then  
export PATH=$(echo $PATH):$(pwd)/bin  
fi

1.Write Pig program to filter and Group Student\_Detatils data

student\_detail = LOAD 'student.txt' USING

PigStorage(',') as (id:int, firstname:chararray, lastname:chararray, age:int,

phone:chararray,

city:chararray);

filter\_data = FILTER student\_detail BY city == 'Chennai';

group\_data = GROUP student\_detail by age;

STORE filter\_data INTO 'filter';

STORE group\_data INTO 'group';

2.Write Pig program to JOIN AND SORT coustomer and order details data

customers = LOAD 'customer.txt' USING PigStorage(',') as (id:int, name:chararray, age:int,address:chararray, salary:int);

orders = LOAD 'order.txt' USING PigStorage(',') as (oid:int, date:chararray, customer\_id:int,amount:int);

join\_result = JOIN customers BY id, orders BY customer\_id;

STORE join\_result INTO 'joinoutput';

customers = LOAD 'customer.txt' USING PigStorage(',') as (id:int, name:chararray, age:int,address:chararray, salary:int);

orders = LOAD 'order.txt' USING PigStorage(',') as (oid:int, date:chararray, customer\_id:int,amount:int);

join\_result = JOIN customers BY id, orders BY customer\_id;

sorting = ORDER join\_result BY age ASC;

STORE sorting INTO 'sortoutput';