Cloud Computing and Big Data Laboratory

- A. Write a spark program using Python, to analyze the given Weather Report Data and to generate a report with cities having maximum and minimum temperature for a particular year.

Code: weather.py

```
import sys
from pyspark import SparkContext

# Check for correct number of arguments
if len(sys.argv) != 4:
    print("Usage: script.py <input_file> <min_output_dir> <max_output_dir>")
    sys.exit(1)

# Initialize Spark context
sc = SparkContext()

try:
    # Read the input file
    f = sc.textFile(sys.argv[1])

    # Map to (key, value) pairs
    temp = f.map(lambda x: (int(x[15:19]), int(x[87:92])))

    # Calculate minimum values
    mini = temp.reduceByKey(lambda a, b: a if a < b else b)
    mini.saveAsTextFile(sys.argv[2])

    # Calculate maximum values
    maxi = temp.reduceByKey(lambda a, b: a if a > b else b)
    maxi.saveAsTextFile(sys.argv[3])

except Exception as e:
    print(f"An error occurred: {e}")
finally:
    # Stop the Spark context
    sc.stop()
```

Execution:

spark-submit weather.py input.txt minimum maximum

Output:

- A. Write a spark program using Python, to analyze the given Earthquake Data and generate statistics with region and magnitude/ region and depth/ region and latitude/ region and longitude

Code: earthquake.py

```
import sys
from pyspark import SparkContext

if(len(sys.argv)!=6):
   print("Provide Input File and Output Directory")
   sys.exit(0)

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

# Region with Magnitude
temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[8])))
maxi=temp.reduceByKey(lambda a,b:a if a>b else b)
maxi.saveAsTextFile(sys.argv[2])

# Region with Depth
temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[9])))
maxi=temp.reduceByKey(lambda a,b:a if a>b else b)
maxi.saveAsTextFile(sys.argv[3])
```

```
# Region with latitude
temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[6])))
maxi=temp.reduceByKey(lambda a,b:a if a>b else b)
maxi.saveAsTextFile(sys.argv[4])

# Region with longitude
temp=f.map(lambda x: (x.split(',')[11],float(x.split(',')[7])))
maxi=temp.reduceByKey(lambda a,b:a if a>b else b)
maxi.saveAsTextFile(sys.argv[5])
```

Execution:

spark-submit earthquake.py earthquake-input.csv magnitude depth latitude longitude

Output:

```
ritlab-01@ritlab01-ThinkCentre-M70t-Gen-3:~/1MS22CS105/hadoop-3.2.2/spark-3.5.2-bin-hadoop3$ cat magnitude/*
('Aegean Sea', 5.7)
('Alaska Peninsula', 3.1)
('"Andreanof Islands', 2.7)
('Arizona', 3.1)
('Arkansas', 1.8)
('"Arunachal Pradesh', 4.2)
('"Babuyan Islands region', 4.5)
('"Baja California', 2.8)
('"Acme Islands', 8.9)
('"Andaman Islands', 5.0)
('"Antofagasta', 5.1)
('Central Alaska', 1.5)
ritlab-01@ritlab01-ThinkCentre-M70t-Gen-3:~/1MS22CS105/hadoop-3.2.2/spark-3.5.2-bin-hadoop3$ cat longitude/*
('Aegean Sea', 25.6298)
('Alaska Peninsula', -154.6988)
('"Andreanof Islands', -174.3559)
('Arizona', -111.8563)
('Arkansas', -91.9482)
('"Arunachal Pradesh', 94.3088)
('"Babuyan Islands region', 121.2571)
('"Baja California', -115.2127)
('"Acme Islands', -175.8648)
('"Andaman Islands', 92.3832)
('"Antofagasta', -69.522)
('Central Alaska', -147.3775)
ritlab-01@ritlab01-ThinkCentre-M70t-Gen-3:~/1MS22CS105/hadoop-3.2.2/spark-3.5.2-bin-hadoop3$
```

- A. Write a spark program using Python, to analyze the given Insurance Data and generate

a statistics report with the construction building name and the count of building/ county name and its frequency

Code: insurance.py

```
import sys
from pyspark import SparkContext

if(len(sys.argv)!=4):
   print("Provide Input File and Output Directory")
   sys.exit(1)

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

# Construction building or Count of building
temp=f.map(lambda x: (x.split(',')[16],1))
data=temp.countByKey()
dd=sc.parallelize(data.items())
dd.saveAsTextFile(sys.argv[2])

# County name and its frequency
temp=f.map(lambda x: (x.split(',')[2],1))
data=temp.countByKey()
dd=sc.parallelize(data.items())
dd.saveAsTextFile(sys.argv[3])
```

Execution:

spark-submit insurance.py input-insurance.csv construction county

Output:

```
ritlab-01@ritlab01-ThinkCentre-M70t-Gen-3:~/1MS22CS105/hadoop-3.2.2/spark-3.5.2-bin-hadoop3$ cat construction/*
('Wood', 17)
('Reinforced Masonry', 2)
('Reinforced Concrete', 3)
('Masonry', 2)
ritlab-01@ritlab01-ThinkCentre-M70t-Gen-3:~/1MS22CS105/hadoop-3.2.2/spark-3.5.2-bin-hadoop3$ cat county/*
('ALACHUA COUNTY', 24)
ritlab-01@ritlab01-ThinkCentre-M70t-Gen-3:~/1MS22CS105/hadoop-3.2.2/spark-3.5.2-bin-hadoop3$
```

- A. Write a spark program using Python, to analyze the given Sales Records over a period of time and generate data about the country's total sales, and the total number of the products. / Country's total sales and the frequency of the payment mode.

Code: sales.py

```
import sys
from pyspark import SparkContext

if(len(sys.argv)!=4):
   print("Provide Input File and Output Directory")
   sys.exit(0)

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

# Total products
temp=f.map(lambda x: (x.split(',')[7],1))
data=temp.countByKey()
dd=sc.parallelize(data.items())
dd.saveAsTextFile(sys.argv[2])

# Frequency
temp=f.map(lambda x: (x.split(',')[3],1))
data=temp.countByKey()
dd=sc.parallelize(data.items())
dd.saveAsTextFile(sys.argv[3])
```

Execution:

spark-submit sales.py input-sales.csv products frequency

Output: