LAB REPORT 2

TDDE31

Prepared by: Aswathi Karunakaran Beena(aswka169)
Nandana Krishna (nankri143)

Question 1
a. Maximum temperature

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext()
sqlContext = SQLContext(sc)
# This path is to load text file and convert each line to Row
rdd = sc.textFile("BDA/input/temperature-readings.csv")
lines = rdd.map(lambda line: line.split(";"))

tempReadings = lines.map(lambda p: Row(station=p[0], date=p[1], year=p[1].split("-")[0], time=p[2],
value=float(p[3]), quality=p[4]))

#Inferring the schema and registering the DataFrame as a table
schemaTempReadings = sqlContext.createDataFrame(tempReadings)
schemaTempReadings.registerTempTable("tempReadings")

#filter
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] >= 1950)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] <= 2014)

#Get max
max_temperatures = schemaTempReadings.groupBy('year').agg(F.max('value').alias('value'))
max_temperatures = max_temperatures.join(schemaTempReadings, ['year', 'value']).select
('year','station','value').orderBy(F.desc("value"))

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
max_temperatures.rdd.saveAsTextFile("BDA/output")
```

Output

```
Row(year=u'2003', station=u'136420', value=32.2)
Row(year=u'1953', station=u'65160', value=32.2)
Row(year=u'1973', station=u'71470', value=32.2)
Row(year=u'2007', station=u'86420', value=32.2)
Row(year=u'2008', station=u'95130', value=32.2)
Row(year=u'2008', station=u'102390', value=32.2)
Row(year=u'2008', station=u'82090', value=32.2)
Row(year=u'2008', station=u'82090', value=32.2)
Row(year=u'1955', station=u'97260', value=32.2)
```

b. Minimum temperature readings

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext()
sqlContext = SQLContext(sc)
# This path is to load text file and convert each line to Row
rdd = sc.textFile("BDA/input/temperature-readings.csv")
lines = rdd.map(lambda line: line.split(";"))

tempReadings = lines.map(lambda p: Row(station=p[0], date=p[1], year=p[1].split("-")[0], time=p[2],
value=float(p[3]), quality=p[4]))

#Inferring the schema and registering the DataFrame as a table
schemaTempReadings = sqlContext.createDataFrame(tempReadings)
schemaTempReadings.registerTempTable("tempReadings")

#filter
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] >= 1950)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] <= 2014)


#Get min
min_temperatures = schemaTempReadings.groupBy('year').agg(F.min('value').alias('value'))
min_temperatures = min_temperatures.join(schemaTempReadings, ['year', 'value']).select
('year','station','value').orderBy(F.desc("value"))

#print(max_temperatures.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
min_temperatures.rdd.saveAsTextFile("BDA/output")
```

Output

```
Row(year=u'1950', station=u'155910', value=-42.0)
Row(year=u'1962', station=u'181900', value=-42.0)
Row(year=u'1962', station=u'181900', value=-42.0)
Row(year=u'1951', station=u'155910', value=-42.0)
Row(year=u'2011', station=u'179960', value=-42.0)
Row(year=u'1968', station=u'179950', value=-42.0)
```

Question 2

a. Non distinct

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext()
sqlContext = SQLContext(sc)
# This path is to load text file and convert each line to Row
rdd = sc.textFile("BDA/input/temperature-readings.csv")
lines = rdd.map(lambda line: line.split(";"))

tempReadings = lines.map(lambda p: Row(station=p[0], date=p[1], year=p[1].split("-")[0], month =
p[1].split("-")[1],time = p[2], value = float(p[3]), quality=p[4]))

#Inferring the schema and registering the DataFrame as a table
schemaTempReadings = sqlContext.createDataFrame(tempReadings)
schemaTempReadings.registerTempTable("tempReadings")

#filter
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] >= 1950)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] <= 2014)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['value'] >= 10)
#Get count
count_temperatures =
schemaTempReadings.groupBy('year','month').agg(F.count('value').alias('value')).orderBy(F.desc('value'))
#print(max_temperatures.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
count_temperatures.rdd.saveAsTextFile("BDA/output")
```

Output

```
Row(year=u'2001', month=u'07', value=121192)
Row(year=u'1998', month=u'07', value=120912)
Row(year=u'2000', month=u'07', value=120460)
Row(year=u'2004', month=u'07', value=120189)

Row(year=u'2011', month=u'06', value=126084)
Row(year=u'2012', month=u'08', value=125947)
Row(year=u'2005', month=u'07', value=125651)
```

b. Distinct count

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext()
sqlContext = SQLContext(sc)
# This path is to load text file and convert each line to Row
rdd = sc.textFile("BDA/input/temperature-readings.csv")
lines = rdd.map(lambda line: line.split(";"))

tempReadings = lines.map(lambda p: Row(station=p[0], date=p[1], year=p[1].split("-")[0], month =
p[1].split("-")[1],time = p[2], value = float(p[3]), quality=p[4]))

#Inferring the schema and registering the DataFrame as a table
schemaTempReadings = sqlContext.createDataFrame(tempReadings)
schemaTempReadings.registerTempTable("tempReadings")

#filter
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] >= 1950)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] <= 2014)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['value'] >= 10)
#distinct temperature readings
countDistinct_temperatures =
schemaTempReadings.select('station','year','month').distinct().groupBy('year','month').count().orderBy(F.d
esc('count'))

#print(max_temperatures.collect())

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
countDistinct_temperatures.rdd.saveAsTextFile("BDA/output")
```

Output

```
Row(year=u'1966', month=u'06', count=360)
Row(year=u'1969', month=u'09', count=359)
Row(year=u'1967', month=u'06', count=359)
Row(year=u'1966', month=u'08', count=359)
Row(year=u'1974', month=u'06', count=372)
Row(year=u'1974', month=u'08', count=372)
Row(year=u'1970', month=u'08', count=370)
Row(year=u'1974', month=u'05', count=370)
Row(year=u'1971', month=u'07', count=370)
Row(year=u'1973', month=u'07', count=370)
```

Question 3

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext()
sqlContext = SQLContext(sc)
# This path is to load text file and convert each line to Row
rdd = sc.textFile("BDA/input/temperature-readings.csv")
lines = rdd.map(lambda line: line.split(";"))

tempReadings = lines.map(lambda p: Row(station=p[0], date=p[1], year=p[1].split("-")[0], month =
p[1].split("-")[1], day = p[1].split("-")[2], time = p[2], value = float(p[3]), quality=p[4]))

#Inferring the schema and registering the DataFrame as a table
schemaTempReadings = sqlContext.createDataFrame(tempReadings)
schemaTempReadings.registerTempTable("tempReadings")

#filter
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] >= 1960)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['year'] <= 2016)

#Get average
avg_temperatures = schemaTempReadings.groupBy('year'. 'month', 'day', 'station').agg((F.max('value') +
(F.min('value'))/2) .alias('value'))
avg_temperatures = avg_temperatures.groupBy
('year','month','station').agg((F.avg('value').alias('avg'))orderBy(F.desc('avg'))

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
avg_temperatures.rdd.saveAsTextFile("BDA/output")
```

Output

```
Row(year=u'1995', month=u'08', station=u'64130', avg=30.895161290322577)
Row(year=u'2014', month=u'07', station=u'104580', avg=30.89193548387097)
Row(year=u'1973', month=u'07', station=u'71420', avg=30.88870967741936)
Row(year=u'1983', month=u'07', station=u'98210', avg=30.888709677419353)
Row(year=u'2010', month=u'07', station=u'53300', avg=30.887096774193548)
Row(year=u'1995', month=u'08', station=u'63340', avg=30.885483870967736)
Row(year=u'1969', month=u'08', station=u'97240', avg=30.883870967741935)
Row(year=u'1983', month=u'08', station=u'53560', avg=30.88225806451613)
Row(year=u'1996', month=u'08', station=u'86200', avg=30.88225806451613)
Row(year=u'1982', month=u'07', station=u'98210', avg=30.88064516129032)
Row(year=u'2014', month=u'07', station=u'85050', avg=30.88064516129032)
Row(year=u'2001', month=u'07', station=u'77210', avg=30.88064516129032)
Row(year=u'1973', month=u'07', station=u'106270', avg=30.879032258064512)
Row(year=u'2002', month=u'08', station=u'96040', avg=30.879032258064512)
Row(year=u'1992', month=u'06', station=u'76000', avg=30.878333333333337)
```

Question 4

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext()
sqlContext = SQLContext(sc)
# This path is to load text file and convert each line to Row
temprdd = sc.textFile("BDA/input/temperature-readings.csv")
lines = temprdd.map(lambda line: line.split(";"))

tempReadings = lines.map(lambda p: Row(station=p[0],  value = float(p[3])))

#Inferring the schema and registering the DataFrame as a table
schemaTempReadings = sqlContext.createDataFrame(tempReadings)
schemaTempReadings.registerTempTable("tempReadings")

precipitationrdd = sc.textFile("BDA/input/precipitation-readings.csv")
lines = precipitationrdd.map(lambda line: line.split(";"))

precipitation = lines.map(lambda p: Row(station=p[0], date=p[1], year=p[1].split("-")[0], month =
p[1].split("-")[1], day = p[1].split("-")[2], value = float(p[3])))

#Inferring the schema and registering the DataFrame as a table
schemaPrecip = sqlContext.createDataFrame(precipitation)
schemaPrecip.registerTempTable("precipitation")


#filter
schemaTempReadings =
schemaTempReadings.groupBy('station').agg(F.max('value').alias('max_temperature'))
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['max_temperature'] >= 25)
schemaTempReadings = schemaTempReadings.filter(schemaTempReadings['max_temperature'] <= 30)

schemaPrecip = schemaPrecip.groupBy('station','year','month','day').agg(F.sum('value').alias('value'))
schemaPrecip = schemaPrecip.filter(schemaPrecip['value'] >= 100)
schemaPrecip = schemaPrecip.filter(schemaPrecip['value'] <= 200)
```

```
joinedAnswer = schemaTempReadings.join(schemaPrecip,['station']).select('station',
'max_temperature','value').orderBy(F.desc('station'))
joinedAnswer.rdd.saveAsTextFile("BDA/output")
```

Output
No output

Question 5

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext()
sqlContext = SQLContext(sc)
# This path is to load text file and convert each line to Row
stationrdd = sc.textFile("BDA/input/stations-Ostergotland.csv")
lines = stationrdd.map(lambda line: line.split(";"))
stations = lines.map(lambda p: Row(station=p[0]))

#Inferring the schema and registering the DataFrame as a table
schemaStations = sqlContext.createDataFrame(stations)
schemaStations.registerTempTable("stations")
len = schemaStations.count()

precipitationrdd = sc.textFile("BDA/input/precipitation-readings.csv")
lines = precipitationrdd.map(lambda line: line.split(";"))

precipitation = lines.map(lambda p: Row(station=p[0], year=p[1].split("-")[0], month = p[1].split("-")[1],
day = p[1].split("-")[2], value = float(p[3])))

#Inferring the schema and registering the DataFrame as a table
schemaPrecip = sqlContext.createDataFrame(precipitation)
schemaPrecip.registerTempTable("precipitation")

#filter
precipOster = schemaPrecip.filter(schemaPrecip['year'] >= 1993)
precipOster = precipOster.filter(precipOster['value'] <= 2016)

precipOster = schemaPrecip.join(schemaStations, ['station']).select('year','month','value')
precipOster = precipOster.groupBy('year','month').agg((F.sum('value')/
len).alias('value')).orderBy(F.desc('value'))
```

```
precipOster.rdd.repartition(1).saveAsTextFile("BDA/output")
```

Output

```
Row(year=u'2012', month=u'06', value=23.329411764705878)
Row(year=u'2015', month=u'05', value=21.935294117647057)
Row(year=u'2007', month=u'06', value=19.22647058823529)
Row(year=u'2000', month=u'11', value=19.07941176470587)
Row(year=u'2004', month=u'07', value=16.94117647058823)
Row(year=u'2007', month=u'07', value=16.935294117647057)
Row(year=u'1997', month=u'06', value=15.35)
Row(year=u'1998', month=u'08', value=15.25882352941176)
Row(year=u'2002', month=u'07', value=14.208823529411765)
Row(year=u'2015', month=u'01', value=13.908823529411771)
Row(year=u'2004', month=u'10', value=13.797058823529412)
Row(year=u'2014', month=u'05', value=13.647058823529413)
Row(year=u'2005', month=u'08', value=13.582352941176474)
Row(year=u'2004', month=u'08', value=13.32058823529412)
Row(year=u'2012', month=u'09', value=12.838235294117647)
```