732A54 - Big Data Analytics

BDA2 Lab

Sridhar Adhikarla (sriad858), Obaid Ur Rehman (obaur539)

Question 1:

- year, station with the max, maxValue ORDER BY maxValue DESC
- year, station with the min, minValue ORDER BY minValue DESC

Code:

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-

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

iFile = 'data/temperature-readings.csv'
oFile1 = 'outputs/out1_1'
oFile2 = 'outputs/out1_2'

fromYear = 1950
toYear = 2014

sc = SparkContext(appName = "Lab2_Q1_SparkSQLJob")
sqlContext = SQLContext(sc)

temperature_file = sc.textFile(iFile)

lines = temperature_file.map(lambda line: line.split(";"))
temp = lines.filter(lambda x:(int(x[1][0:4]) >= fromYear and int(x[1][0:4]) <=toYear))
temp = temp.map(lambda x: Row(station=x[0], year=int(x[1][0:4]), temp = float(x[3])))

schemaTempReadings =sqlContext.createDataFrame(temp)
schemaTempReadings.registerTempTable("tempReadings")

schemaTempReadingsMin =
  schemaTempReadings.groupBy('year','station').agg(F.min('temp').alias('mintemp'))\
  .orderBy(['mintemp'],descending=1)

schemaTempReadingsMax =
  schemaTempReadings.groupBy('year','station').agg(F.max('temp').alias('maxtemp'))\
  .orderBy(['maxtemp'], ascending = False)

schemaTempReadingsMin.rdd.repartition(1).saveAsTextFile(oFile1)
schemaTempReadingsMax.rdd.repartition(1).saveAsTextFile(oFile2)
```

```
print(schemaTempReadingsMin.take(15))
print(schemaTempReadingsMax.take(15))
```

MIN Temp:

```
out1_1 = read.csv("outputs/out1_1/part-00000", header = FALSE)
head(out1_1)
```

```
##              V1                 V2              V3
## 1 Row(year=1978  station=u'147100'  mintemp=-42.6)
## 2 Row(year=1978  station=u'159970'  mintemp=-42.6)
## 3 Row(year=1987  station=u'124020'  mintemp=-42.6)
## 4 Row(year=1967  station=u'181900'  mintemp=-42.6)
## 5 Row(year=1999  station=u'160970'  mintemp=-42.6)
## 6 Row(year=1956  station=u'155910'  mintemp=-42.6)
```

MAX Temp:

```
out1_2 = read.csv("outputs/out1_2/part-00000", header = FALSE)
head(out1_2)
```

```
##              V1                V2             V3
## 1 Row(year=1983  station=u'97390'  maxtemp=32.7)
## 2 Row(year=1994  station=u'52350'  maxtemp=32.7)
## 3 Row(year=2010  station=u'95130'  maxtemp=32.7)
## 4 Row(year=2006  station=u'75240'  maxtemp=32.7)
## 5 Row(year=1992  station=u'74440'  maxtemp=32.7)
## 6 Row(year=2000  station=u'76000'  maxtemp=32.7)
```

Question 2:

- year, month, value ORDER BY value DESC
- year, month, value ORDER BY value DESC

Code:

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-

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

iFile = 'data/temperature-readings.csv'
oFile1 = 'outputs/out2_a1'
oFile2 = 'outputs/out2_a2'

#2A
sc = SparkContext(appName = "Lab2_Q2_SparkSQLJob")
sqlContext = SQLContext(sc)

temperature_file = sc.textFile(iFile)
lines = temperature_file.map(lambda line: line.split(";"))
temp = lines.filter(lambda x:(int(x[1][0:4]) >= 1950 and int(x[1][0:4]) <=2014))
temp = temp.map(lambda x: Row(station=x[0], year=int(x[1][0:4]),
                              month = int(x[1][5:7]), temp = float(x[3])))

schemaTempReadings =sqlContext.createDataFrame(temp)
schemaTempReadings.registerTempTable("tempReadings")

#API method
schemaNumReadings = schemaTempReadings \
.filter(schemaTempReadings['temp']>10).groupBy('year','month').count()
schemaNumReadings.rdd.repartition(1).saveAsTextFile(oFile1)
print(schemaNumReadings.take(10))

#Sql method
SQL_NumReadings = sqlContext.sql(
    "SELECT year, month, count(temp) as count FROM tempReadings WHERE temp>10
    GROUP BY year, month ORDER BY count DESC")
SQL_NumReadings.rdd.repartition(1).saveAsTextFile(oFile2)
print(SQL_NumReadings.take(10))


#2B
oFile3 = 'outputs/out2_b1'
oFile4 = 'outputs/out2_b2'

temperature_file = sc.textFile(iFile)
lines = temperature_file.map(lambda line: line.split(";"))
temp = lines.filter(lambda x:(int(x[1][0:4]) >= 1950 and int(x[1][0:4]) <=2014))
```

```
temp = temp.map(lambda x: Row(station=x[0], year=int(x[1][0:4]),
                              month = int(x[1][5:7]), temp = float(x[3])))

schemaTempReadings =sqlContext.createDataFrame(temp)
schemaTempReadings.registerTempTable("tempReadings")

#API method
schemaNumReadings_st = schemaTempReadings.filter(schemaTempReadings['temp']>10) \
.groupBy('year','month').agg(F.countDistinct("station")) \
.orderBy(['count(station)'],ascending = 0)
schemaNumReadings_st.rdd.repartition(1).saveAsTextFile(oFile3)
print(schemaNumReadings_st.take(10))

#SQL Method
SQL_NumReadings_st = sqlContext.sql(
    "SELECT year, month, count(DISTINCT station) as count FROM tempReadings WHERE
    temp>10 GROUP BY year, month ORDER BY count DESC")
SQL_NumReadings_st.rdd.repartition(1).saveAsTextFile(oFile4)
print(SQL_NumReadings_st.take(10))
```

Temperatures readings counts:

```
out2_a1 = read.csv("outputs/out2_a1/part-00000", header = FALSE)
head(out2_a1, 10)
```

```
##                V1        V2             V3
## 1  Row(year=1950   month=9    count=3612)
## 2  Row(year=2004  month=11    count=1066)
## 3  Row(year=1977  month=10   count=13659)
## 4  Row(year=1961   month=2      count=89)
## 5  Row(year=1988   month=3      count=14)
## 6  Row(year=1999   month=6  count=103227)
## 7  Row(year=1972   month=5   count=21192)
## 8  Row(year=1961  month=12       count=8)
## 9  Row(year=1956   month=6   count=21075)
## 10 Row(year=2010   month=8  count=124417)
```

```
out2_a2 = read.csv("outputs/out2_a2/part-00000", header = FALSE)
head(out2_a2, 10)
```

```
##                V1       V2             V3
## 1  Row(year=2009  month=8  count=128349)
## 2  Row(year=2013  month=8  count=128235)
## 3  Row(year=2003  month=7  count=128133)
## 4  Row(year=1996  month=8  count=107758)
## 5  Row(year=1997  month=6  count=104696)
## 6  Row(year=1999  month=6  count=103227)
## 7  Row(year=2005  month=9   count=75494)
## 8  Row(year=2010  month=9   count=74816)
## 9  Row(year=1997  month=9   count=74472)
## 10 Row(year=2009  month=5   count=60867)
```

Distinct Station Temperatures readings counts:

```
out2_b1 = read.csv("outputs/out2_b1/part-00000", header = FALSE)
head(out2_b1, 10)
```

```
##                V1       V2                    V3
## 1  Row(year=1971  month=9  count(station)=374)
## 2  Row(year=1971  month=6  count(station)=374)
## 3  Row(year=1972  month=7  count(station)=374)
## 4  Row(year=1974  month=7  count(station)=362)
## 5  Row(year=1970  month=7  count(station)=362)
## 6  Row(year=1967  month=9  count(station)=361)
## 7  Row(year=1977  month=5  count(station)=351)
## 8  Row(year=1979  month=6  count(station)=351)
## 9  Row(year=1967  month=7  count(station)=351)
## 10 Row(year=1979  month=9  count(station)=351)
```

```
out2_b2 = read.csv("outputs/out2_b2/part-00000", header = FALSE)
head(out2_b2, 10)
```

```
##                V1        V2          V3
## 1  Row(year=1972  month=10  count=378)
## 2  Row(year=1973   month=6  count=377)
## 3  Row(year=1973   month=5  count=377)
## 4  Row(year=1972   month=8  count=376)
## 5  Row(year=1973   month=9  count=376)
## 6  Row(year=1972   month=7  count=374)
## 7  Row(year=1971   month=6  count=374)
## 8  Row(year=1971   month=9  count=374)
## 9  Row(year=1974   month=6  count=372)
## 10 Row(year=1974   month=8  count=372)
```

Question 3:

- year, month, station, avgMonthlyTemperature ORDER BY avgMonthlyTemperature DESC

Code:

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-

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

iFile = 'data/temperature-readings.csv'
oFile = 'outputs/out3'

sc = SparkContext(appName="Lab2_Q3_SparkSQLJob")

sqlContext = SQLContext(sc)

temperature_file = sc.textFile(iFile)
lines = temperature_file.map(lambda line: line.split(";"))
temp = lines.filter(lambda x:(int(x[1][0:4]) >= 1960 and int(x[1][0:4]) <=2014))
temp = temp.map(lambda x: Row(station=x[0], year=int(x[1][0:4]),
                              month = int(x[1][5:7]), temp = float(x[3])))

schemaTempReadings =sqlContext.createDataFrame(temp)
schemaTempReadings.registerTempTable("tempReadings")

avgTemp = schemaTempReadings.groupBy('year','month','station') \
.agg(F.avg('temp').alias('avgtemp')).orderBy(['avgtemp'],ascending = False)
avgTemp.rdd.repartition(1).saveAsTextFile(oFile)
print(avgTemp.take(15))
```

Average monthly temperatures:

```
out3 = read.csv("outputs/out3/part-00000", header = FALSE)
head(out3, 10)
```

```
##                V1       V2                 V3                              V4
## 1  Row(year=2005  month=7  station=u'66500'  avgtemp=19.719354838709673)
## 2  Row(year=1972  month=7  station=u'78390'  avgtemp=19.718279569892474)
## 3  Row(year=2006  month=7  station=u'78280'   avgtemp=19.71805929919136)
## 4  Row(year=1983  month=7  station=u'98210'  avgtemp=19.717297297297296)
## 5  Row(year=1997  month=8  station=u'62250'  avgtemp=19.716199376947067)
## 6  Row(year=1973  month=7  station=u'76160'  avgtemp=19.716129032258067)
## 7  Row(year=1997  month=7  station=u'81010'  avgtemp=19.716129032258067)
## 8  Row(year=1994  month=7  station=u'65020'   avgtemp=19.71435643564357)
## 9  Row(year=2005  month=7  station=u'83440'  avgtemp=19.713978494623653)
## 10 Row(year=2006  month=7  station=u'97280'   avgtemp=19.71347708894878)
```

Question 4:

- station, maxTemp, maxDailyPrecipitation ORDER BY station DESC

Code:

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-

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

iFile = 'data/temperature-readings.csv'
iFile2 = 'data/precipitation-readings.csv'
oFile = 'outputs/out4'

sc = SparkContext(appName = "Lab2_Q4_SparkSQLJob")
sqlContext = SQLContext(sc)

temperature_file = sc.textFile(iFile)
lines = temperature_file.map(lambda line: line.split(";"))
temp = lines.map(lambda x: Row(station=x[0], temp = float(x[3])))
schemaTempReadings =sqlContext.createDataFrame(temp)
schemaTempReadings.registerTempTable("tempReadings")

preci_file = sc.textFile(iFile2)
plines = preci_file.map(lambda line: line.split(";"))
prec = plines.map(lambda x: Row(station=x[0], date = x[1], prec = float(x[3])))
schemaPrecReadings =sqlContext.createDataFrame(prec)
schemaPrecReadings.registerTempTable("precReadings")

#finding max temperature and filtering
maxTemp = schemaTempReadings.groupBy('station').agg(F.max('temp').alias('maxtemp'))
maxTemp = maxTemp.filter(maxTemp.maxtemp>25).filter(maxTemp.maxtemp<30)

#finding max daily prec and filtering
dailyPrec = schemaPrecReadings.groupBy('station','date') \
.agg(F.sum('prec').alias('dailyPrec'))

maxDailyPrec = dailyPrec.groupBy('station') \
.agg(F.max('dailyPrec').alias('maxdailyPrec'))

maxDailyPrec = maxDailyPrec.filter(maxDailyPrec.maxdailyPrec>=100) \
.filter(maxDailyPrec.maxdailyPrec<=200)

#joining
StationMax = maxTemp.join(maxDailyPrec, "station") \
.orderBy(['station'],ascending = False)

StationMax.rdd.repartition(1).saveAsTextFile(oFile)
print(StationMax.take(15))
```

Max daily temperatures/precipitation:

We get an empty file as output for this question

```
out4 = read.csv("outputs/out4/part-00000", header = FALSE)
```

Question 5:

- station, maxTemp, maxDailyPrecipitation ORDER BY station DESC

Code:

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-

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

iFile = 'data/stations-Ostergotland.csv'
iFile2 = 'data/precipitation-readings.csv'
oFile = 'outputs/out5'

sc = SparkContext(appName="Lab2_Q5_SparkSQLJob")
sqlContext = SQLContext(sc)

stations = sc.textFile(iFile)
stations = stations.map(lambda line:line.split(";"))
stations = stations.map(lambda x:Row(station=x[0], name=x[1]))
stations = sqlContext.createDataFrame(stations)
stations.registerTempTable("O_Stations")

preci_file = sc.textFile(iFile2)
plines = preci_file.map(lambda line: line.split(";"))
prec = plines.filter(lambda x:(int(x[1][0:4]) >= 1993 and int(x[1][0:4]) <=2016))
prec = plines.map(lambda x: Row(station=x[0], year = x[1][0:4],
                                month = x[1][5:7], prec = float(x[3])))
schemaPrecReadings =sqlContext.createDataFrame(prec)
schemaPrecReadings.registerTempTable("precReadings")

avgPrec = stations.join(schemaPrecReadings, "station")
avgPrec = avgPrec.groupBy("year", "month", "station") \
.agg(F.sum("prec").alias("monthlyPrec"))

avgPrec = avgPrec.groupBy("year","month") \
.agg(F.avg("monthlyPrec").alias("avgMonthlyPrec")) \
.orderBy(["year", "month"], ascending=[0,0])

avgPrec.rdd.repartition(1).saveAsTextFile(oFile)
print(avgPrec.take(10))
```

Ostergotland average monthly precipitation:

```
out5 = read.csv("outputs/out5/part-00000", header = FALSE)
head(out5, 10)
```

```
##                  V1           V2                                  V3
## 1  Row(year=u'2016'  month=u'07'                 avgMonthlyPrec=0.0)
## 2  Row(year=u'2016'  month=u'06'             avgMonthlyPrec=47.6625)
## 3  Row(year=u'2016'  month=u'05'  avgMonthlyPrec=29.250000000000007)
## 4  Row(year=u'2016'  month=u'04'   avgMonthlyPrec=26.90000000000001)
## 5  Row(year=u'2016'  month=u'03'  avgMonthlyPrec=19.962500000000006)
## 6  Row(year=u'2016'  month=u'02'             avgMonthlyPrec=21.5625)
## 7  Row(year=u'2016'  month=u'01'  avgMonthlyPrec=22.325000000000003)
## 8  Row(year=u'2015'  month=u'12'  avgMonthlyPrec=28.925000000000004)
## 9  Row(year=u'2015'  month=u'11'             avgMonthlyPrec=63.8875)
## 10 Row(year=u'2015'  month=u'10'              avgMonthlyPrec=2.2625)
```

Question 6:

- Year, month, difference ORDER BY year DESC, month DESC

Code:

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-

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

iFile = 'data/stations-Ostergotland.csv'
iFile2 = 'data/temperature-readings.csv'
oFile = 'outputs/out6'

sc = SparkContext(appName="Lab2_Q6_SparkSQLJob")
sqlContext = SQLContext(sc)

stations = sc.textFile(iFile)
stations = stations.map(lambda line:line.split(";"))
stations = stations.map(lambda x:Row(station=x[0], name=x[1]))
stations = sqlContext.createDataFrame(stations)
stations.registerTempTable("O_Stations")

temperature_file = sc.textFile(iFile2)
lines = temperature_file.map(lambda line: line.split(";"))
temp = lines.filter(lambda x:(int(x[1][0:4]) >= 1950 and int(x[1][0:4]) <=2014))
temp = temp.map(lambda x: Row(station=x[0], year=int(x[1][0:4]),
                              month = int(x[1][5:7]),temp = float(x[3])))
schemaTempReadings =sqlContext.createDataFrame(temp)
schemaTempReadings.registerTempTable("tempReadings")

avgMonthTemp = schemaTempReadings.groupBy('year','month','station') \
.agg(F.avg('temp').alias('stationavg'))

#Average monthly temperature in Ostergotland stations
avgMonthTemp = stations.join(avgMonthTemp, "station")
avgMonthTemp = avgMonthTemp.groupBy('year','month') \
.agg(F.avg('stationavg').alias('avgMonthTemp'))

#filtering to find longterm average
longMonthTemp = avgMonthTemp.filter(avgMonthTemp.year <= 1980)
longMonthTemp = longMonthTemp.groupBy("month") \
.agg(F.avg("avgMonthTemp").alias("longAvg"))

#Joining the long term average and monthly average dataframes
MonthlyAvgDiff = avgMonthTemp.join(longMonthTemp, "month")
MonthlyAvgDiff = MonthlyAvgDiff.withColumn("difference",
                              MonthlyAvgDiff.avgMonthTemp-MonthlyAvgDiff.longAvg)
```

```
MonthlyAvgDiff = MonthlyAvgDiff.select("year","month","difference") \
.orderBy(["year","month"],ascending=[0,0])

MonthlyAvgDiff = MonthlyAvgDiff.map(lambda line: '%s,%s,%s'%(int(line[0]),
                                                             int(line[1]),
                                                             float(line[2])))
print(MonthlyAvgDiff.take(10))

MonthlyAvgDiff.repartition(1).saveAsTextFile(oFile)
print(MonthlyAvgDiff.take(10))
```

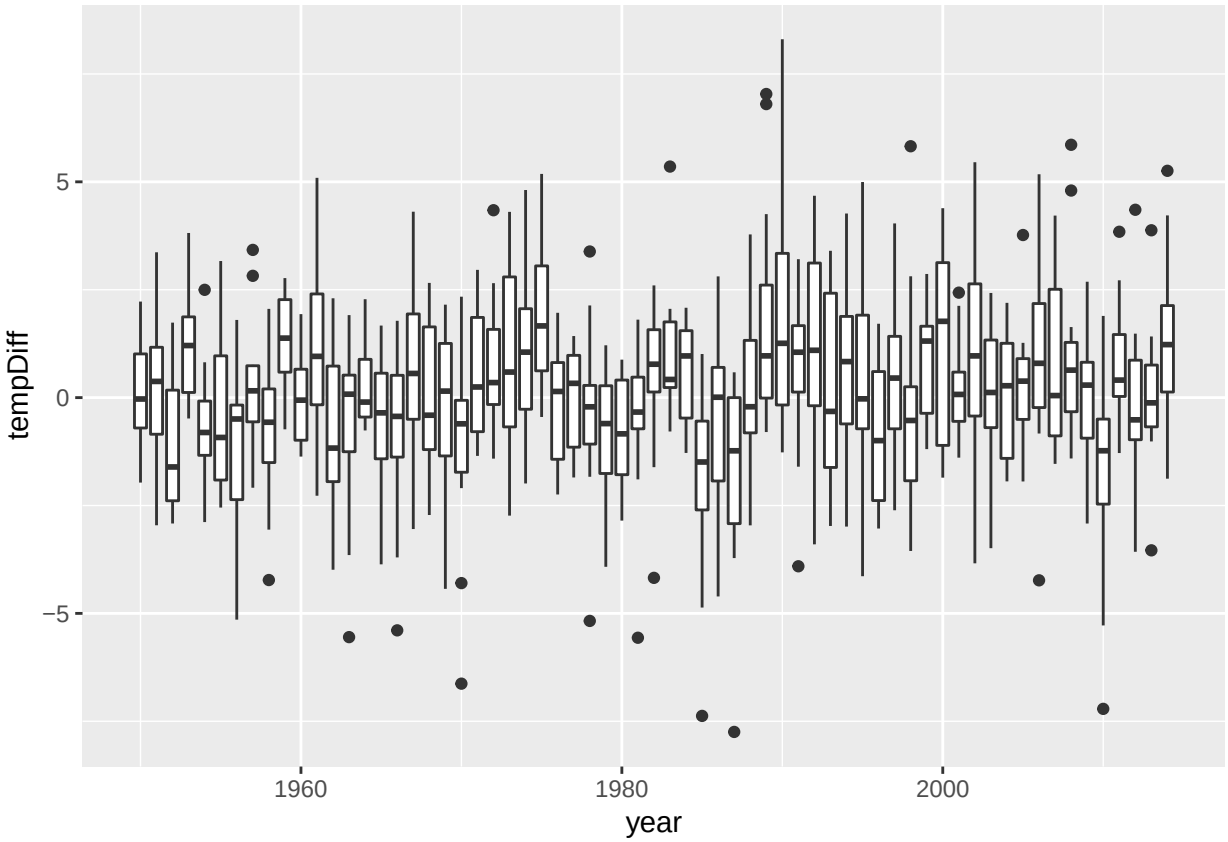
Ostergotland average monthly precipitation temperature difference:

```
out6 = read.csv("outputs/out6/part-00000", header = FALSE)
colnames(out6) = c("year", "month", "tempDiff")
head(out6, 10)
```

```
##    year month    tempDiff
## 1  2014    12  0.80900501
## 2  2014    11  2.09577458
## 3  2014    10  1.53698435
## 4  2014     9  0.04528869
## 5  2014     4  2.12110369
## 6  2014     3  4.22214051
## 7  2014     2  5.25560806
## 8  2014     1  0.91936869
## 9  2013     8 -0.38568784
## 10 2013     7  0.14340586
```

plot:

```
library(ggplot2)
ggplot(out6) +
  geom_boxplot(aes(x = year, y = tempDiff, group = year))
```


5
0
−5
tempDiff
1960
1980
2000
year