lab2_sparksql_bigdata

zahra jalilpour(zahja96), Zhixuan_Duan(zhidu838)

2021-05-16

Contents

Lab2 - Spark SQL - Exercises **2**

1) 2

2) 4

3) 7

4) 9

5) 10

Lab2 - Spark SQL - Exercises

1)

What are the lowest and highest temperatures measured each year for the period 1950-2014. Provide the lists sorted in the descending order with respect to the maximum temperature. In this exercise you will use the temperature-readings.csv file. The output should at least contain the following information (You can also include a Station column so that you may find multiple stations that record the highest (lowest) temperature.):

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

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

sc = SparkContext(appName = "exe 1")
sqlContext = SQLContext(sc)
Temp=sc.textFile("BDA/input/temperature-readings.csv")
# Split by a ';'
lines = Temp.map(lambda line: line.split(";"))
# Map key as year and value as temperature
tempReadings = lines.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]),
int(p[1].split("-")[1]), p[2], float(p[3]), p[4] ))

tempReadingsString = ["station", "date", "year", "month", "time", "value","quality"]
schemaTempReadings = sqlContext.createDataFrame(tempReadings,tempReadingsString)
schemaTempReadings.registerTempTable("tempReadingsTable")
filtered_years = schemaTempReadings.where('year >= 1950 and year <= 2014')
max_temp = filtered_years.groupBy('year').agg(F.max('value').alias('value'))
min_temp = filtered_years.groupBy('year').agg(F.min('value').alias('value'))
max_temp = max_temp.join(filtered_years, ['year', 'value']).select('year', 'station', 'value').orderBy(
min_temp = min_temp.join(filtered_years, ['year', 'value']).select('year', 'station', 'value').orderBy(
max_temp.withColumnRenamed('value', 'yearlyMax').rdd.saveAsTextFile("BDA/output/max_temperature")
min_temp.withColumnRenamed('value', 'yearlyMin').rdd.saveAsTextFile("BDA/output/min_temperature")
```

max temperature

Row(year=1975, station=u'86200', yearlyMax=36.1)

Row(year=1992, station=u'63600', yearlyMax=35.4)

Row(year=1994, station=u'117160', yearlyMax=34.7)

Row(year=2014, station=u'96560', yearlyMax=34.4)

Row(year=2010, station=u'75250', yearlyMax=34.4)

Row(year=1989, station=u'63050', yearlyMax=33.9)

Row(year=1982, station=u'94050', yearlyMax=33.8)

Row(year=1968, station=u'137100', yearlyMax=33.7)

Row(year=1966, station=u'151640', yearlyMax=33.5)

```
Row(year=2002, station=u'78290', yearlyMax=33.3)
Row(year=2002, station=u'78290', yearlyMax=33.3)
Row(year=1983, station=u'98210', yearlyMax=33.3)
.
.
.
Row(year=1950, station=u'75040', yearlyMax=29.4)
Row(year=1960, station=u'173810', yearlyMax=29.4)
Row(year=1998, station=u'63600', yearlyMax=29.2)
Row(year=1965, station=u'116500', yearlyMax=28.5)
Row(year=1951, station=u'75040', yearlyMax=28.5)
Row(year=1962, station=u'86200', yearlyMax=27.4)
Row(year=1962, station=u'76380', yearlyMax=27.4)
```

min temperature

```
Row(year=1990, station=u'147270', yearlyMin=-35.0)
Row(year=1990, station=u'166870', yearlyMin=-35.0)
Row(year=1952, station=u'192830', yearlyMin=-35.5)
Row(year=1974, station=u'166870', yearlyMin=-35.6)
Row(year=1974, station=u'179950', yearlyMin=-35.6)
Row(year=1954, station=u'113410', yearlyMin=-36.0)
Row(year=1992, station=u'179960', yearlyMin=-36.1)
Row(year=1975, station=u'157860', yearlyMin=-37.0)
.
.
.
Row(year=1967, station=u'166870', yearlyMin=-45.4)
Row(year=1987, station=u'123480', yearlyMin=-47.3)
Row(year=1978, station=u'155940', yearlyMin=-47.7)
Row(year=1999, station=u'192830', yearlyMin=-49.0)
Row(year=1999, station=u'192830', yearlyMin=-49.0)
Row(year=1966, station=u'179950', yearlyMin=-49.4)
```

2)

Count the number of readings for each month in the period of 1950-2014 which are higher than 10 degrees.Repeat the exercise,this time taking only distinct readings from each station. That is, if a station reported a reading above 10 degrees in some month, then it appears only once in the count for that month.

year, month, value ORDER BY value DESC

year, month, value ORDER BY value DESC

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

sc = SparkContext(appName = "exe 2")
sqlContext = SQLContext(sc)
Temp=sc.textFile("BDA/input/temperature-readings.csv")
# Split  by a ';'
lines = Temp.map(lambda line: line.split(";"))
tempReadings = lines.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]),
int(p[1].split("-")[1]), p[2], float(p[3]), p[4] ))

tempReadingsString = ["station", "date", "year", "month", "time", "value","quality"]
schemaTempReadings = sqlContext.createDataFrame(tempReadings,tempReadingsString)
schemaTempReadings.registerTempTable("tempReadingsTable")
filtered_years = schemaTempReadings.where('year >= 1950 and year <= 2014 and value > 10')
month_over_10 = filtered_years.groupBy(['year', 'month']).agg(F.count('value').alias('value'))
month_over_10_distinct = filtered_years.groupBy(['year', 'month', 'station']).agg(F.countDistinct('year
month_10_distinct = month_over_10_distinct.groupBy(['year', 'month']).agg(F.count('value').alias('value
sorted_over_10 = month_over_10.orderBy('value', ascending=0).rdd.saveAsTextFile("BDA/output/month_10")
sorted_distinct_over_10 = month_10_distinct.orderBy('value', ascending=0).rdd.saveAsTextFile("BDA/outpu
```

month higher than 10 degree

Row(year=2014, month=7, value=147681)

Row(year=2011, month=7, value=146656)

Row(year=2010, month=7, value=143419)

Row(year=2012, month=7, value=137477)

Row(year=2013, month=7, value=133657)

Row(year=2009, month=7, value=133008)

Row(year=2011, month=8, value=132734)

Row(year=2009, month=8, value=128349)

Row(year=2013, month=8, value=128235)

Row(year=2003, month=7, value=128133)

Row(year=2002, month=7, value=127956)

Row(year=2006, month=8, value=127622)

Row(year=2008, month=7, value=126973)

Row(year=2002, month=8, value=126073)

```
.
.
.
Row(year=1960, month=12, value=2)
Row(year=1950, month=11, value=2)
Row(year=1957, month=2, value=2)
Row(year=1990, month=12, value=2)
Row(year=1993, month=12, value=2)
Row(year=1991, month=1, value=2)
Row(year=1955, month=1, value=2)
Row(year=1955, month=3, value=2)
Row(year=1956, month=1, value=2)
Row(year=1979, month=1, value=1)
Row(year=1984, month=3, value=1)
Row(year=1952, month=3, value=1)
Row(year=1970, month=3, value=1)
Row(year=1950, month=12, value=1)
Row(year=1995, month=2, value=1)
Row(year=1962, month=2, value=1)
Row(year=1958, month=2, value=1)
Row(year=1975, month=12, value=1)
Row(year=1984, month=1, value=1)
Row(year=1962, month=3, value=1)
Row(year=1988, month=11, value=1)
Row(year=1963, month=3, value=1)
Row(year=1954, month=2, value=1)
Row(year=1960, month=2, value=1)
Row(year=2001, month=3, value=1)
Row(year=1952, month=11, value=1)
Row(year=1993, month=1, value=1)
Row(year=1960, month=1, value=1)
Row(year=1951, month=2, value=1)
Row(year=2005, month=12, value=1)
Row(year=1958, month=1, value=1)
```

distinct month higher 10 degree

```
Row(year=1972, month=10, value=378)
Row(year=1973, month=5, value=377)
Row(year=1973, month=6, value=377)
Row(year=1972, month=8, value=376)
Row(year=1973, month=9, value=376)
Row(year=1972, month=6, value=375)
Row(year=1972, month=5, value=375)
Row(year=1972, month=9, value=375)
Row(year=1971, month=8, value=375)
Row(year=1972, month=7, value=374)
Row(year=1971, month=6, value=374)
Row(year=1971, month=9, value=374)
.
.
.
Row(year=1959, month=1, value=3)
Row(year=1954, month=1, value=3)
Row(year=2000, month=2, value=3)
Row(year=1980, month=3, value=3)
Row(year=2007, month=12, value=3)
Row(year=1996, month=3, value=3)
Row(year=1954, month=12, value=3)
Row(year=1955, month=2, value=3)
Row(year=1990, month=1, value=3)
Row(year=1956, month=2, value=2)
Row(year=1960, month=12, value=2)
Row(year=1955, month=1, value=2)
Row(year=1957, month=2, value=2)
Row(year=1957, month=1, value=2)
Row(year=1997, month=2, value=2)
Row(year=1988, month=1, value=2)
Row(year=2003, month=2, value=2)
Row(year=1950, month=11, value=2)
Row(year=1990, month=12, value=2)
Row(year=1956, month=1, value=2)
Row(year=1993, month=12, value=2)
Row(year=1979, month=1, value=1)
Row(year=1951, month=2, value=1)
Row(year=2005, month=12, value=1)
Row(year=1950, month=12, value=1)
Row(year=1995, month=2, value=1)
Row(year=1960, month=1, value=1)
Row(year=1963, month=3, value=1)
Row(year=1954, month=2, value=1)
Row(year=1984, month=3, value=1)
Row(year=1952, month=3, value=1)
Row(year=1993, month=1, value=1)
Row(year=1970, month=3, value=1)
Row(year=1993, month=11, value=1)
Row(year=1955, month=3, value=1)
Row(year=1975, month=12, value=1)
Row(year=1960, month=2, value=1)
Row(year=1987, month=12, value=1)
Row(year=2001, month=3, value=1)
Row(year=2014, month=12, value=1)
Row(year=1952, month=11, value=1)
Row(year=1962, month=3, value=1)
Row(year=1988, month=11, value=1)
Row(year=1958, month=1, value=1)
Row(year=1984, month=1, value=1)
Row(year=1962, month=2, value=1)
Row(year=1991, month=1, value=1)
Row(year=1958, month=2, value=1)
```

3)

Find the average monthly temperature for each available station in Sweden. Your result should include average temperature for each station for each month in the period of 1960- 2014. Bear in mind that not every station has the readings for each month in this time frame. In this exercise you will use the temperature-readings.csv file.

year, month, station, avgMonthlyTemperature ORDER BY avgMonthlyTemperature DESC

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

sc = SparkContext(appName = "exe 3")
sqlContext = SQLContext(sc)
Temp=sc.textFile("BDA/input/temperature-readings.csv")
lines = Temp.map(lambda line: line.split(";"))
tempReadings = lines.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]),
int(p[1].split("-")[1]), p[2], float(p[3]), p[4] ))

tempReadingsString = ["station", "date", "year", "month", "time", "value", "quality"]
schemaTempReadings = sqlContext.createDataFrame(tempReadings, tempReadingsString)
schemaTempReadings.registerTempTable("tempReadingsTable")
filtered_years = schemaTempReadings.where('year >= 1960 and year <= 2014')
temp_max = filtered_years.groupBy(['year', 'month', 'station', 'date']).agg(F.max('value').alias('max')
temp_min = filtered_years.groupBy(['year', 'month', 'station', 'date']).agg(F.min('value').alias('min')
temp_max_min = temp_max.join(temp_min, ['year', 'month', 'station', 'date'])
temp_max_min = temp_max_min.select('year', 'month', 'station', 'date', ((temp_max_min.max+temp_max_min.
ave_monthly_temp = temp_max_min.groupBy(['year', 'month', 'station']).agg(F.avg('maxmin').alias('avgMon
ave_monthly_temp.orderBy('avgMonthlyTemperature', ascending=0).rdd.saveAsTextFile("BDA/output/ave_month
```

Row(year=2014, month=7, station=u'96000', avgMonthlyTemperature=26.3)

Row(year=1994, month=7, station=u'96550', avgMonthlyTemperature=23.071052631578947)

Row(year=1983, month=8, station=u'54550', avgMonthlyTemperature=23.0)

Row(year=1994, month=7, station=u'78140', avgMonthlyTemperature=22.970967741935482)

Row(year=1994, month=7, station=u'85280', avgMonthlyTemperature=22.872580645161293)

Row(year=1994, month=7, station=u'75120', avgMonthlyTemperature=22.858064516129037)

Row(year=1994, month=7, station=u'65450', avgMonthlyTemperature=22.856451612903225)

Row(year=1994, month=7, station=u'96000', avgMonthlyTemperature=22.808064516129033)

Row(year=1994, month=7, station=u'95160', avgMonthlyTemperature=22.76451612903226)

Row(year=1994, month=7, station=u'86200', avgMonthlyTemperature=22.711290322580645)

Row(year=2002, month=8, station=u'78140', avgMonthlyTemperature=22.7)

Row(year=1994, month=7, station=u'76000', avgMonthlyTemperature=22.698387096774194)

Row(year=1997, month=8, station=u'78140', avgMonthlyTemperature=22.666129032258066)

Row(year=1994, month=7, station=u'105260', avgMonthlyTemperature=22.659677419354843)

Row(year=1975, month=8, station=u'54550', avgMonthlyTemperature=22.642857142857142)

Row(year=2006, month=7, station=u'76530', avgMonthlyTemperature=22.598387096774196)

Row(year=1994, month=7, station=u'86330', avgMonthlyTemperature=22.54838709677419)

Row(year=2006, month=7, station=u'75120', avgMonthlyTemperature=22.527419354838706)

Row(year=1994, month=7, station=u'54300', avgMonthlyTemperature=22.469354838709684)

Row(year=2006, month=7, station=u'78140', avgMonthlyTemperature=22.45806451612904)

Row(year=2001, month=7, station=u'96550', avgMonthlyTemperature=22.408333333333335)

```
Row(year=2010, month=7, station=u'98180', avgMonthlyTemperature=22.379032258064516)
Row(year=2006, month=7, station=u'65450', avgMonthlyTemperature=22.377419354838707)
Row(year=1994, month=7, station=u'85210', avgMonthlyTemperature=22.375806451612902)
.
.
.
Row(year=1985, month=2, station=u'179950', avgMonthlyTemperature=-23.70535714285715)
Row(year=1966, month=2, station=u'179950', avgMonthlyTemperature=-23.78392857142857)
Row(year=1985, month=2, station=u'182930', avgMonthlyTemperature=-23.791071428571428)
Row(year=1993, month=12, station=u'166900', avgMonthlyTemperature=-23.8)
Row(year=1966, month=2, station=u'191900', avgMonthlyTemperature=-23.887499999999996)
Row(year=1985, month=2, station=u'172790', avgMonthlyTemperature=-23.992857142857144)
Row(year=1985, month=2, station=u'166810', avgMonthlyTemperature=-24.012499999999996)
Row(year=1985, month=2, station=u'159970', avgMonthlyTemperature=-24.03035714285715)
Row(year=1985, month=2, station=u'183980', avgMonthlyTemperature=-24.05)
Row(year=1985, month=2, station=u'191900', avgMonthlyTemperature=-24.06964285714286)
Row(year=1966, month=2, station=u'192830', avgMonthlyTemperature=-24.294642857142858)
Row(year=1966, month=2, station=u'189780', avgMonthlyTemperature=-24.37678571428571)
Row(year=1966, month=2, station=u'181900', avgMonthlyTemperature=-24.4125)
Row(year=1966, month=2, station=u'166870', avgMonthlyTemperature=-24.46607142857143)
Row(year=1985, month=2, station=u'166870', avgMonthlyTemperature=-24.719642857142855)
Row(year=1985, month=2, station=u'183760', avgMonthlyTemperature=-24.73571428571429)
Row(year=1966, month=2, station=u'167860', avgMonthlyTemperature=-24.758928571428577)
Row(year=1966, month=2, station=u'159970', avgMonthlyTemperature=-24.935714285714287)
Row(year=1985, month=2, station=u'169880', avgMonthlyTemperature=-25.79285714285714)
Row(year=1985, month=2, station=u'192830', avgMonthlyTemperature=-26.346428571428568)
Row(year=1985, month=2, station=u'181900', avgMonthlyTemperature=-26.6375)
```

4)

Provide a list of stations with their associated maximum measured temperatures and maximum measured daily precipitation. Show only those stations where the maximum temperature is between 25 and 30 degrees and maximum daily precipitation is between 100 mm and 200mm.

station, maxTemp, maxDailyPrecipitation ORDER BY station DESC

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

sc = SparkContext(appName = "exercise 4")
```

```
sqlContext = SQLContext(sc)
Temp=sc.textFile("BDA/input/temperature-readings.csv")
Precipitation=sc.textFile("BDA/input/precipitation-readings.csv")
lines_temperature = Temp.map(lambda line: line.split(";"))
lines_precipitation = Precipitation.map(lambda line: line.split(";"))
tempReadings = lines_temperature.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]),int(p[1].split("-")
precReadings = lines_precipitation.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]),int(p[1].split("-

tempReadingsString = ["station", "date", "year", "month", "time", "temp", "quality"]
precReadingsString = ["station", "date", "year", "month", "time", "prec", "quality"]

schemaTempReadings = sqlContext.createDataFrame(tempReadings, tempReadingsString)
schemaPrecReadings = sqlContext.createDataFrame(precReadings, precReadingsString)

schemaTempReadings.registerTempTable("tempReadingsTable")
schemaPrecReadings.registerTempTable("precReadingsTable")
temp_max = schemaTempReadings.groupBy('station').agg(F.max('temp').alias('maxtemp'))
prec_max = schemaPrecReadings.groupBy('station').agg(F.max('prec').alias('maxprec'))
joined_max = temp_max.join(prec_max, 'station').where('maxtemp >= 25 and maxtemp <= 30 and maxprec >= 1
joined_max.orderBy('station', ascending=0).rdd.saveAsTextFile("BDA/output/max_temp_prec")
```

output is empty

5)

Calculate the average monthly precipitation for the Östergotland region (list of stations is provided in the separate file) for the period 1993-2016. In orderto dothis, you willfirstneed to calculate the total monthly precipitation for each station before calculating the monthly average (by averaging over stations).

year, month, avgMonthlyPrecipitation ORDER BY year DESC, month DESC

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

sc = SparkContext(appName = "exercise 5")
sqlContext = SQLContext(sc)

Precipitation=sc.textFile("BDA/input/precipitation-readings.csv")
Station=sc.textFile("BDA/input/stations-Ostergotland.csv")

lines_precipitation = Precipitation.map(lambda line: line.split(";"))
lines_stations = Station.map(lambda line: line.split(";"))
precReadings = lines_precipitation.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]),
int(p[1].split("-")[1]), p[2], float(p[3]), p[4] ))
stationReadings = lines_stations.map(lambda x: (x[0], x[1]))

stationReadingsString = ["station", "name"]
precReadingsString = ["station", "date", "year", "month", "time", "prec", "quality"]

schemaStationReadings = sqlContext.createDataFrame(stationReadings, stationReadingsString)
schemaPrecReadings = sqlContext.createDataFrame(precReadings, precReadingsString)
```

```
schemaStationReadings.registerTempTable("stationReadingsTable")
schemaPrecReadings.registerTempTable("precReadingsTable")

prec_province = schemaPrecReadings.join(schemaStationReadings, 'station').where("year>1993 and year<2016
monthly_prec = prec_province.groupBy(['station', 'year', 'month']).agg(F.sum('prec').alias('prec'))
prec_month_avg = monthly_prec.groupBy(['year', 'month']).agg(F.avg('prec').alias('avgMonthlyPrec'))
prec_month_avg.orderBy(['year', 'month'], ascending=[0,0]).rdd.saveAsTextFile("BDA/output/average_,onthl
```

```
Row(year=2016, month=7, avgMonthlyPrec=0.0)
Row(year=2016, month=6, avgMonthlyPrec=47.6625)
Row(year=2016, month=5, avgMonthlyPrec=29.250000000000004)
Row(year=2016, month=4, avgMonthlyPrec=26.90000000000001)
Row(year=2016, month=3, avgMonthlyPrec=19.962500000000002)
Row(year=2016, month=2, avgMonthlyPrec=21.562500000000004)
Row(year=2016, month=1, avgMonthlyPrec=22.325)
Row(year=2015, month=12, avgMonthlyPrec=28.925)
Row(year=2015, month=11, avgMonthlyPrec=63.887500000000024)
Row(year=2015, month=10, avgMonthlyPrec=2.2625)
Row(year=2015, month=9, avgMonthlyPrec=101.29999999999997)
Row(year=2015, month=8, avgMonthlyPrec=26.9875)
Row(year=2015, month=7, avgMonthlyPrec=119.09999999999997)
Row(year=2015, month=6, avgMonthlyPrec=78.66250000000002)
Row(year=2015, month=5, avgMonthlyPrec=93.225)
Row(year=2015, month=4, avgMonthlyPrec=15.3375)
Row(year=2015, month=3, avgMonthlyPrec=42.61250000000001)
Row(year=2015, month=2, avgMonthlyPrec=24.825000000000003)
Row(year=2015, month=1, avgMonthlyPrec=59.112500000000026)
Row(year=2014, month=12, avgMonthlyPrec=35.46250000000001)
Row(year=2014, month=11, avgMonthlyPrec=52.425000000000054)
.
.
.
Row(year=1994, month=5, avgMonthlyPrec=25.100000000000005)
Row(year=1994, month=4, avgMonthlyPrec=23.100000000000005)
Row(year=1994, month=3, avgMonthlyPrec=37.60000000000004)
Row(year=1994, month=2, avgMonthlyPrec=22.500000000000004)
Row(year=1994, month=1, avgMonthlyPrec=22.099999999999994)
Row(year=1993, month=12, avgMonthlyPrec=37.10000000000003)
Row(year=1993, month=11, avgMonthlyPrec=42.80000000000003)
Row(year=1993, month=10, avgMonthlyPrec=43.2)
Row(year=1993, month=9, avgMonthlyPrec=40.6)
Row(year=1993, month=8, avgMonthlyPrec=80.70000000000005)
Row(year=1993, month=7, avgMonthlyPrec=95.39999999999999)
Row(year=1993, month=6, avgMonthlyPrec=56.5)
Row(year=1993, month=5, avgMonthlyPrec=21.100000000000005)
Row(year=1993, month=4, avgMonthlyPrec=0.0)
```