Alice Velander, alive213
David Björelind, davbj395

TDDE31: Big Data Analytics, Lab 2

Assignment 1)

#----------CODE----------#

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

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

#Loading text file and convert each line to a Row
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.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], temp=float(p[3]), quality=p[4]))

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

schema_temp = schemaTempReadings.filter( (schemaTempReadings.year >= 1950) & (schemaTempReadings.year <= 2014))

#Get Max
schema_temp_max = schema_temp.groupBy('year').agg(F.max('temp').alias('temp'))
schema_temp_max = schema_temp_max.join(schema_temp, ['year','temp'], 'inner').\
                  select('year', 'station', 'temp').orderBy('temp', ascending=False).show()

#Get Min
schema_temp_min = schema_temp.groupBy('year').agg(F.min('temp').alias('temp'))
schema_temp_min = schema_temp_min.join(schema_temp, ['year','temp'], 'inner').\
                  select('year', 'station', 'temp').orderBy('temp', ascending=False).show()
```

#----------OUTPUT----------#

Max temperature

```
+----+-------+----+
|year|station|temp|
+----+-------+----+
|1975|  86200|36.1|
|1992|  63600|35.4|
|1994| 117160|34.7|
|2010|  75250|34.4|
|2014|  96560|34.4|
|1989|  63050|33.9|
|1982|  94050|33.8|
|1968| 137100|33.7|
|1966| 151640|33.5|
|2002|  78290|33.3|
|1983|  98210|33.3|
|2002|  78290|33.3|
|1986|  76470|33.2|
|1970| 103080|33.2|
|1956| 145340|33.0|
|2000|  62400|33.0|
|1959|  65160|32.8|
|2006|  75240|32.7|
|1991| 137040|32.7|
|1988| 102540|32.6|
+----+-------+----+
only showing top 20 rows
```

Alice Velander, alive213
David Björelind, davbj395

Showing Min temperature

```
+----+-------+-----+
|year|station| temp|
+----+-------+-----+
|1990| 147270|-35.0|
|1990| 166870|-35.0|
|1952| 192830|-35.5|
|1974| 179950|-35.6|
|1974| 166870|-35.6|
|1954| 113410|-36.0|
|1992| 179960|-36.1|
|1975| 157860|-37.0|
|1972| 167860|-37.5|
|1995| 182910|-37.6|
|2000| 169860|-37.6|
|1957| 159970|-37.8|
|1983| 191900|-38.2|
|1989| 166870|-38.2|
|1953| 183760|-38.4|
|2009| 179960|-38.5|
|1993| 191900|-39.0|
|1984| 191900|-39.2|
|1984| 123480|-39.2|
|2008| 179960|-39.3|
+----+-------+-----+
only showing top 20 rows
```

Assignment 2)

#----------CODE----------#

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

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

#Loading text file and convert each line to a Row
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

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

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

schema_temp = schemaTempReadings.filter( (schemaTempReadings.year >= 1950) & (schemaTempReadings.year <=
2014) & (schemaTempReadings.temp > 10) )

#Get Max
schema_temp_count = schema_temp.groupBy('year','month').count().orderBy('count',ascending=False).show()
```

#----------OUTPUT----------#

Alice Velander, alive213
David Björelind, davbj395

i)

Sorted by maxcount….

```
+----+-----+------+
|year|month| count|
+----+-----+------+
|2014|   07|147681|
|2011|   07|146656|
|2010|   07|143419|
|2012|   07|137477|
|2013|   07|133657|
|2009|   07|133008|
|2011|   08|132734|
|2009|   08|128349|
|2013|   08|128235|
|2003|   07|128133|
|2002|   07|127956|
|2006|   08|127622|
|2008|   07|126973|
|2002|   08|126073|
|2005|   07|125294|
|2011|   06|125193|
|2012|   08|125037|
|2006|   07|124794|
|2010|   08|124417|
|2014|   08|124045|
+----+-----+------+
```

ii)

Made some changes to the code to take the distinct constraint into consideration:

```
+----+-----+-----+
|year|month|count|
+----+-----+-----+
|1972|   10|  378|
|1973|   06|  377|
|1973|   05|  377|
|1973|   09|  376|
|1972|   08|  376|
|1972|   05|  375|
|1971|   08|  375|
|1972|   09|  375|
|1972|   06|  375|
|1972|   07|  374|
|1971|   09|  374|
|1971|   06|  374|
|1971|   05|  373|
|1973|   08|  373|
```

Alice Velander, alive213
David Björelind, davbj395

|1974| 06| 372|
|1974| 08| 372|
|1974| 09| 370|
|1974| 05| 370|
|1971| 07| 370|
|1970| 08| 370|
+----+-----+-----+

Assignment 3)

#----------CODE----------#

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

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

#Loading text file and convert each line to a Row
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

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

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

schema_temp = schemaTempReadings.filter( (schemaTempReadings.year >= 1960) & (schemaTempReadings.year <=
2014) )

#Get average
schema_temp_count = schema_temp.groupBy('station', 'year','month').avg('temp').orderBy('avg(temp)', ascen
ding=False).show()
```

#----------OUTPUT----------#

+-------+----+-----+------------------+
|station|year|month| avg(temp)|
+-------+----+-----+------------------+
96000	2014	07	26.3
65450	1994	07	23.65483870967742
95160	1994	07	23.505376344086027
75120	1994	07	23.26881720430107
105260	1994	07	23.143820224719107
85280	1994	07	23.108602150537635
54550	1983	08	23.0
54550	1975	08	22.9625
96550	1994	07	22.957894736842114
96000	1994	07	22.931182795698923
106070	1994	07	22.822580645161295
173960	1972	07	22.776666666666667
54300	1994	07	22.76021505376344
85210	1994	07	22.755913978494615
65450	2006	07	22.74086021505376

Alice Velander, alive213
David Björelind, davbj395

```
|  75120|2006|   07| 22.73010752688173|
| 103080|1994|   07|22.708602150537626|
|  92100|1994|   07|22.698924731182792|
|  94180|1994|   07| 22.68172043010753|
|  83230|1994|   07|22.577419354838707|
+-------+----+-----+------------------+
```

Assignment 4)

#----------CODE----------#

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F
from pyspark.sql import HiveContext
sc = SparkContext(appName = "spark")
sqlContext = SQLContext(sc)

#--RAIN READ--#
pre_file = sc.textFile("BDA/input/precipitation-readings.csv")
lines = pre_file.map(lambda line: line.split(";"))
rainReadings = lines.map(lambda p: Row(station=p[0], year=p[1].split("-")[0], month=p[1].split("-")
s)[1], time=p[2], date=p[1], rain=float(p[3]), quality=p[4]))
schemaRainReadings = sqlContext.createDataFrame(rainReadings)
schemaRainReadings.registerTempTable("rainReadings")

#--TEMP READ--#
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))
tempReadings = lines.map(lambda p: Row(station=p[0], year=p[1].split("-")[0], month=p[1].split("-")
s)[1], time=p[2], temp=float(p[3]), quality=p[4]))
schemaTempReadings = sqlContext.createDataFrame(tempReadings)
schemaTempReadings.registerTempTable("tempReadings")

#Get max temp for each station within the temperature interval
schema_temp_max = schemaTempReadings.groupBy('station').agg(F.max('temp').alias('temp'))
schema_temp = schema_temp_max.filter( (schema_temp_max.temp >= 25) & (schema_temp_max.temp <= 30))
schema_temp = schema_temp.join(schemaTempReadings, ['station','temp'], 'inner').\
                  select('year', 'station', 'temp')

# Get daily observations of rain and then filtering
schemaRainDaily = schemaRainReadings.groupBy('date').sum('rain')
schemaRainDaily.show()
schemaRainDaily = schemaRainDaily.join(schemaRainReadings, ['date'], 'inner')
schemaRainMax = schemaRainDaily.groupBy('station').agg(F.max('sum(rain)').alias('sumrain'))
schema_rain = schemaRainMax.filter ( (schemaRainMax.sumrain >= 100) & (schemaRainMax.sumrain <= 20
s0) )

# JOIN TEMP AND RAIN
schema_tot = schema_rain.join(schema_temp, ['station'], 'inner').\
             select('station', 'temp', 'sumrain').orderBy('station', acsending=False).show()
```

#----------OUTPUT----------#

```
+-------+----+-------+
|station|temp|sumrain|
+-------+----+-------+
+-------+----+-------+
```

No results! :o

Assignment 5)

Alice Velander, alive213
David Björelind, davbj395

#----------CODE----------#

```
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F
from pyspark.sql import HiveContext
sc = SparkContext(appName = "spark")
sqlContext = SQLContext(sc)

#--RAIN READ--#
pre_file = sc.textFile("BDA/input/precipitation-readings.csv")
lines = pre_file.map(lambda line: line.split(";"))
rainReadings = lines.map(lambda p: Row(station=p[0], year=p[1].split("-")[0], month=p[1].split("-")[1], time=p[2], date=p[1], rain=float(p[3]), quality=p[4]))
schemaRainReadings = sqlContext.createDataFrame(rainReadings)
schemaRainReadings.registerTempTable("rainReadings")

#--STATION READ--#
ost_file = sc.textFile("BDA/input/stations-Ostergotland.csv")
lines = ost_file.map(lambda line: line.split(";"))
ostReadings = lines.map(lambda p: Row(station=p[0]))
schemaOstReadings = sqlContext.createDataFrame(ostReadings)
schemaOstReadings.registerTempTable("ostReadings")

# Get MONTHLY observations of rain and then filtering
schemaRainReadings = schemaRainReadings.filter( (schemaRainReadings.year >= 1993) & (schemaRainReadings.year <= 2016) )
schemaRainMonth = schemaRainReadings.groupBy('year', 'month', 'station').sum('rain')
schemaRainMonth = schemaRainMonth.join(schemaRainReadings, ['year', 'month'], 'inner')

# Only Ost-stations
schemaOstRain = schemaRainMonth.join(schemaOstReadings, ['station'], 'inner')
schemaOstRain = schemaOstRain.groupBy('year', 'month').avg('sum(rain)')
schemaOstRain.select('year', 'month', 'avg(sum(rain))').orderBy('avg(sum(rain))', ascending=False).show()
```

#----------OUTPUT----------#

```
+----+-----+-----------------+
|year|month|    avg(sum(rain))|
+----+-----+-----------------+
|2006|   08|148.08333333336174|
|2008|   08|138.51666666669396|
|2000|   07|135.86666666662452|
|1995|   09|134.54999999997327|
|2012|   06|132.20000000000113|
|2015|   07|119.09999999991126|
|2006|   10|118.16666666664882|
|2003|   07| 113.4666666665552|
|2009|   07|113.16666666661368|
|2001|   09|110.63333333338788|
|2000|   10| 110.3000000000696|
|2007|   06| 108.9500000000062|
|2000|   11|108.11666666667975|
|2010|   08|108.04999999998998|
|2005|   07|104.35000000005213|
|2015|   09|101.29999999999579|
|2002|   06| 98.78333333329823|
|1995|   06| 97.20000000000515|
```

Alice Velander, alive213
David Björelind, davbj395

```
|2004|   07| 96.00000000002285|
|2007|   07| 95.96666666670036|
+----+-----+------------------+
```