# Lab 2 - Spark SQL

Ahmad Alhasan, Ali Etminan

6/13/2020

```
import findspark
findspark.init()
findspark.find()
import pyspark
findspark.find()
from pyspark import SparkContext
from pyspark.sql import Row
from pyspark.sql.context import SQLContext
from pyspark.sql import functions as F
from pyspark.sql import SparkSession
from pyspark import SparkConf
```

```
APP_NAME = "Lab2 - Ex1"
conf = SparkConf()
conf = conf.setAppName(APP_NAME)
sc = SparkContext(conf=conf)
sqlContext = SQLContext(sc)
```

## Assignment 1

### Part 1 - Min Values

```
# Reading in the data
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")

lines = temperature_file.map(lambda line: line.split(";"))

year_temperature = lines.map(lambda x: (x[1][0:4], int(x[0]), float(x[3])))

# Defining Column names
tempNames = ["Year","Station","Temperature"]
# Converting RDD to dataframe
temperature_data = sqlContext.createDataFrame(year_temperature, tempNames)
# Registering dataframe as temporary table for SQL operations
temperature_data.registerTempTable("TempReadings")
# Filter with the specified time span
temperature_data = temperature_data.filter((temperature_data['Year']>=1950)
&(temperature_data['Year']<=2014))
```

```
# Get the minimum yearly temperature and sort w.r.t. max temperature
# in a descending order.
min_temps = temperature_data.groupBy('Year','Station').agg
(F.min('Temperature').alias('YearlyMin'))\
.orderBy('YearlyMin', ascending=False)

min_temps = min_temps.rdd
min_temps = min_temps.repartition(1)


# Materialize the outputs
min_temps.saveAsTextFile("BDA/output/Lab_2/A1")
```

**Snippet of the output:**

Row(Year=u'1975', Station=86200, YearlyMax=36.1)

Row(Year=u'1975', Station=95160, YearlyMax=35.8)

Row(Year=u'1975', Station=96550, YearlyMax=35.6)

Row(Year=u'1975', Station=106100, YearlyMax=35.5)

Row(Year=u'1992', Station=63600, YearlyMax=35.4)

Row(Year=u'1975', Station=75240, YearlyMax=35.4)

Row(Year=u'1992', Station=63050, YearlyMax=35.2)

Row(Year=u'1975', Station=95350, YearlyMax=35.0)

Row(Year=u'1975', Station=98210, YearlyMax=35.0)

Row(Year=u'1975', Station=85220, YearlyMax=35.0)

# Assignment 1

## Part 2 - Max Values

```
# Get the maximum yearly temperature and sort w.r.t. max temperature
# in a descending order.
max_temps = temperature_data.groupBy('Year','Station')
.agg(F.max('Temperature').alias('YearlyMax'))\
.orderBy('YearlyMax', ascending=False)

# Converting back to RDD and making it into one partition
max_temps = max_temps.rdd
max_temps = max_temps.repartition(1)

# Materialize the output
max_temps.saveAsTextFile("BDA/output/Lab_2/A1")
```

**Snippet of the output:**

Row(Year=u'1975', Station=86200, YearlyMax=36.1)

Row(Year=u'1975', Station=95160, YearlyMax=35.8)

Row(Year=u'1975', Station=96550, YearlyMax=35.6)

Row(Year=u'1975', Station=106100, YearlyMax=35.5)

Row(Year=u'1992', Station=63600, YearlyMax=35.4)

Row(Year=u'1975', Station=75240, YearlyMax=35.4)

Row(Year=u'1992', Station=63050, YearlyMax=35.2)

Row(Year=u'1975', Station=95350, YearlyMax=35.0)

Row(Year=u'1975', Station=98210, YearlyMax=35.0)

Row(Year=u'1975', Station=85220, YearlyMax=35.0)

# Assignment 2

## Part 1 - Total Readings

```
### Part 1

# Reading in the data
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")



lines = temperature_file.map(lambda line: line.split(";"))

# Constructing the dataframe with the specified columns
colNames = ["Year","Month", "Station", "Temperature"]
monthly_temperature = lines.map(lambda x: (int(x[1][0:4]), int(x[1][5:7]), int(x[0]), float(x[3])))
monthly_tempData = sqlContext.createDataFrame(monthly_temperature, colNames)
monthly_tempData.registerTempTable("MonthReadings")

# Filtering specified years and minimum temperature
new_tempData = monthly_tempData.filter((monthly_tempData['Year']>=1950) &
                                       (monthly_tempData['Year']<=2014) &
                                       (monthly_tempData['Temperature'] > 10))


# Aggregating total monthly readings above 10
summed_readings = new_tempData.groupBy('Year','Month')
.agg(F.count('Month').alias('TotalReadings')) \
.orderBy('TotalReadings', ascending=False)

# Converting back to RDD and making it into one partition
summed_readings = summed_readings.rdd
summed_readings = summed_readings.repartition(1)
```

```
# Materialize the output
summed_readings.saveAsTextFile("BDA/output/Lab_2/A1")
```

**Snippet of the output:**

Row(Year=2014, Month=7, TotalReadings=147681)

Row(Year=2011, Month=7, TotalReadings=146656)

Row(Year=2010, Month=7, TotalReadings=143419)

Row(Year=2012, Month=7, TotalReadings=137477)

Row(Year=2013, Month=7, TotalReadings=133657)

Row(Year=2009, Month=7, TotalReadings=133008)

Row(Year=2011, Month=8, TotalReadings=132734)

Row(Year=2009, Month=8, TotalReadings=128349)

Row(Year=2013, Month=8, TotalReadings=128235)

Row(Year=2003, Month=7, TotalReadings=128133)

# Assignment 2

## Part 2 - Distinct Readings

Here we use the schema created in part 1.

```
# Aggregating total distinct readings from stations
distinct_tempData = new_tempData.groupBy('Year','Month')
.agg(F.countDistinct('Station').alias('UniqueReadings')) \
.orderBy('UniqueReadings', ascending=False)

# Converting back to RDD and making it into one partition
distinct_tempData = distinct_tempData.rdd
distinct_tempData = distinct_tempData.repartition(1)

# Materialize the output
distinct_tempData.saveAsTextFile("BDA/output/Lab_2/A1")
```

**Snippet of the output:**

Row(Year=1972, Month=10, UniqueReadings=378)

Row(Year=1973, Month=6, UniqueReadings=377)

Row(Year=1973, Month=5, UniqueReadings=377)

Row(Year=1972, Month=8, UniqueReadings=376)

Row(Year=1973, Month=9, UniqueReadings=376)

Row(Year=1972, Month=5, UniqueReadings=375)

Row(Year=1972, Month=9, UniqueReadings=375)

Row(Year=1972, Month=6, UniqueReadings=375)

Row(Year=1971, Month=8, UniqueReadings=375)

Row(Year=1971, Month=6, UniqueReadings=374)

# Assignment 3

## Average Monthly Temperatures

```
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")

lines = temperature_file.map(lambda line: line.split(";"))

# Constructing dataframe with specified columns
colNames = ["Year","Month", "Day", "Station", "Temperature"]
station_temp = lines.map(lambda x: (int(x[1][0:4]), int(x[1][5:7]), int(x[1][8:10]), x[0], float(x[3])))
monthly_tempData = sqlContext.createDataFrame(station_temp, colNames)
monthly_tempData.registerTempTable("StationReadings")

# Filtering the dataframe with specified years
station_tempData = monthly_tempData.filter((monthly_tempData['Year']>=1960) & (monthly_tempData['Year']<

# Finding and appending average monthly temperature
averageTemp = station_tempData.groupBy('Year','Month','Station',) \
.agg(F.avg('Temperature').alias('avgMonthlyTemperature')).orderBy('avgMonthlyTemperature', ascending=Fal

averageTemp = averageTemp.rdd
averageTemp = averageTemp.repartition(1)

averageTemp.saveAsTextFile("BDA/output/Lab_2/A1")
```

**Snippet of the output:**

Row(Year=2014, Month=7, Station=u'96000', avgMonthlyTemperature=26.3)

Row(Year=1994, Month=7, Station=u'65450', avgMonthlyTemperature=23.65483870967742)

Row(Year=1994, Month=7, Station=u'95160', avgMonthlyTemperature=23.505376344086027)

Row(Year=1994, Month=7, Station=u'75120', avgMonthlyTemperature=23.26881720430107)

Row(Year=1994, Month=7, Station=u'105260', avgMonthlyTemperature=23.143820224719107)

Row(Year=1994, Month=7, Station=u'85280', avgMonthlyTemperature=23.108602150537635)

Row(Year=1983, Month=8, Station=u'54550', avgMonthlyTemperature=23.0)

Row(Year=1975, Month=8, Station=u'54550', avgMonthlyTemperature=22.9625)

Row(Year=1994, Month=7, Station=u'96550', avgMonthlyTemperature=22.957894736842114)

Row(Year=1994, Month=7, Station=u'96000', avgMonthlyTemperature=22.931182795698923)

# Assignment 4

## Max Daily Precipitation

```
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
precipitation_file = sc.textFile("BDA/input/precipitation-readings.csv")

templines = temperature_file.map(lambda line: line.split(";"))
preclines = precipitation_file.map(lambda line: line.split(";"))

tempNames = ["Station", "Temperature"]
precNames = ["Station", "Date", "Precipitation"]

# Constructing dataframe for the Temperature data
station_temp = templines.map(lambda x: (x[0], float(x[3])))
tempData = sqlContext.createDataFrame(station_temp, tempNames)
tempData.registerTempTable("TempReadings")

# Constructing dataframe for the Precipitation data
station_prec = preclines.map(lambda x: (x[0], x[1], float(x[3])))
precData = sqlContext.createDataFrame(station_prec, precNames)
precData.registerTempTable("PrecReadings")

# Getting the maximum temperature for each station
maxTemp = tempData.groupBy('Station')
.agg(F.max('Temperature').alias('maxTemp')).orderBy('maxTemp', ascending=False)
# Getting the maximum precipitation for each station
sumPrec = precData.groupBy('Station','Date')
.agg(F.sum('Precipitation').alias('DailyTotal'))

maxPrec = sumPrec.groupBy('Station')
.agg(F.max('DailyTotal').alias('maxDailyPrecipitation'))\
.orderBy('maxDailyPrecipitation', ascending=False)

# Filtering out temperature and precipitation readings with the specified criteria
maxTemp = maxTemp.filter((maxTemp['maxTemp']>=25) & (maxTemp['maxTemp']<=30))
maxPrec = maxPrec.filter((maxPrec['maxDailyPrecipitation']>=100) & (maxPrec['maxDailyPrecipitation']<=2

# Joining the two dataframes to get a unified output
stationsData = maxTemp.join(maxPrec, on = ['Station'], how = 'inner')

stationsData = stationsData.rdd
stationsData = stationsData.repartition(1)

stationsData.saveAsTextFile("BDA/output/Lab_2")
```

**Snippet of the output:**

There are no stations within the specified criterea.

# Assignment 5

## Average Monthly Precipitation

```
ostergotland = sc.textFile("BDA/input/stations-Ostergotland.csv")
precipitation_file = sc.textFile("BDA/input/precipitation-readings.csv")

ostlines = ostergotland.map(lambda line: line.split(";"))
preclines = precipitation_file.map(lambda line: line.split(";"))

ostNames = ["Station"]
precNames = ["Station", "Year", "Month", "Day", "Precipitation"]

# Creating dataframe for the Ostergotland Stations
ost_stations = ostlines.map(lambda x: (x[0],))
ostData = sqlContext.createDataFrame(ost_stations, ostNames)
ostData.registerTempTable("OstStations")

# Creating dataframe for the precipitation readings
station_prec = preclines.map(lambda x: (x[0], int(x[1][0:4]), int(x[1][5:7]), int(x[1][8:10]), float(x[3
precData = sqlContext.createDataFrame(station_prec, precNames)
precData.registerTempTable("PrecReadings")

# Filtering precipitation readings with specified years
precData = precData.filter((precData['Year']>=1993) & (precData['Year']<=2016))

# Joining the two dataframes
ostPrec = ostData.join(precData, on = ['Station'], how = 'inner')

# Calculating and appending monthly average precipitation for stations in Ostergotland
monthlyAverages = ostPrec.groupBy('Year', 'Month').agg(F.avg('Precipitation').alias('avgMonthlyPrecipita
.orderBy('avgMonthlyPrecipitation', ascending=False)


monthlyAverages = monthlyAverages.rdd
monthlyAverages = monthlyAverages.repartition(1)

monthlyAverages.saveAsTextFile("BDA/output/Lab_2")
```

**Snippet of the output:**

Row(Year=2006, Month=8, avgMonthlyPrecipitation=0.20211555959963598)

Row(Year=1995, Month=9, avgMonthlyPrecipitation=0.18910751932536896)

Row(Year=2008, Month=8, avgMonthlyPrecipitation=0.18693207377417886)

Row(Year=2012, Month=6, avgMonthlyPrecipitation=0.185978898007034)

Row(Year=2000, Month=7, avgMonthlyPrecipitation=0.18489453390791555)

Row(Year=2015, Month=7, avgMonthlyPrecipitation=0.16872675757039127)

Row(Year=2006, Month=10, avgMonthlyPrecipitation=0.16040723981900448)

Row(Year=2003, Month=7, avgMonthlyPrecipitation=0.15532740132329453)

Row(Year=2001, Month=9, avgMonthlyPrecipitation=0.15447986967651842)

Row(Year=2007, Month=6, avgMonthlyPrecipitation=0.15327080890973038)