# Big Data Analytics Lab 2

*Omkar Bhutra (omkbh878) and Vinay Bengaluru (vinbe289)*

*17 May 2019*

**Execution of the code**

```
[x_omkbh@heffa1 ~]$ ./runYarn.sh bda2alt.py
```

```
#Question 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 = "Max_MinTemperatures_sparksql")
sqlContext = SQLContext(sc)
rdddata = sc.textFile('/user/x_omkbh/data/temperature-readings.csv') \
            .map(lambda line: line.split(";")) \
            .filter(lambda obs:
                                (int(obs[1][:4]) >= 1950 and
                                 int(obs[1][:4]) <= 2014)) \
            .map(lambda obs: \
                Row(station = obs[0], date = obs[1], \
                    year = obs[1].split("-")[0], time = obs[2],
                    value = float(obs[3]), quality = obs[4]))
schemaTempReadings = sqlContext.createDataFrame(rdddata)
schemaTempReadings.registerTempTable("schemaTempReadings")

#Q1.1 year, station with the max, maxValue ORDER BY maxValue DESC

maxTemp = sqlContext.sql"""
        SELECT DISTINCT(table1.year) AS year,
                FIRST(table1.station) AS station,
                FIRST(value) AS value
        FROM schemaTempReadings AS table1
        INNER JOIN
        (
        SELECT year, MAX(value) AS max_value
        FROM schemaTempReadings
        GROUP BY year
        ) AS table2
        ON table1.year = table2.year
        WHERE table1.value = table2.max_value
        GROUP BY table1.year
        ORDER BY value DESC
        """)
maxTemp.rdd.repartition(1) \
             .sortBy(ascending=False, keyfunc=lambda (year,station, value): value)
```

```
maxTemp.take(10)
maxTemp.write.save('/user/x_omkbh/bda2.11')


#Q1.2 year, station with the min, minValue ORDER BY minValue DESC

minTemp = sqlContext.sql("""
        SELECT DISTINCT(table1.year) AS year,
                FIRST(table1.station) AS station,
                FIRST(value) AS value
        FROM schemaTempReadings AS table1
        INNER JOIN
        (
        SELECT year, MIN(value) AS min_value
        FROM schemaTempReadings
        GROUP BY year
        ) AS table2
        ON table1.year = table2.year
        WHERE table1.value = table2.min_value
        GROUP BY table1.year
        ORDER BY value DESC
        """)
minTemp.rdd.repartition(1) \
             .sortBy(ascending=False, keyfunc=lambda (year,station, value): value)
minTemp.take(10)
minTemp.write.save('/user/x_omkbh/bda2.12')
```

**Output: Max Temp**

```
maxTemp.take(10)
[Row(year=u'1975', station=u'86200', value=36.1),
Row(year=u'1992', station=u'63600', value=35.4),
Row(year=u'1994', station=u'117160', value=34.7),
Row(year=u'2014', station=u'96560', value=34.4),
Row(year=u'2010', station=u'75250', value=34.4),
Row(year=u'1989', station=u'63050', value=33.9),
Row(year=u'1982', station=u'94050', value=33.8),
Row(year=u'1968', station=u'137100', value=33.7),
Row(year=u'1966', station=u'151640', value=33.5),
Row(year=u'2002', station=u'78290', value=33.3)]
```

**Output: Min Temp**

```
minTemp.take(10)
[Row(year=u'1990', station=u'147270', value=-35.0),
Row(year=u'1952', station=u'192830', value=-35.5),
Row(year=u'1974', station=u'166870', value=-35.6),
Row(year=u'1954', station=u'113410', value=-36.0),
Row(year=u'1992', station=u'179960', value=-36.1),
Row(year=u'1975', station=u'157860', value=-37.0),
Row(year=u'1972', station=u'167860', value=-37.5),
Row(year=u'1995', station=u'182910', value=-37.6),
Row(year=u'2000', station=u'169860', value=-37.6),
Row(year=u'1957', station=u'159970', value=-37.8)]
```

```
#Question 2:
#year, month, value ORDER BY value DESC
#year, month, value ORDER BY value DESC
rdddata2 = sc.textFile('/user/x_omkbh/data/temperature-readings.csv') \
            .map(lambda line: line.split(";")) \
            .map(lambda obs: \
                Row(station = obs[0], date = obs[1],  \
                    year = obs[1].split("-")[0], month = obs[1].split("-")[1], \
                    yymm = obs[1][:7], \
                    time = obs[2],value = float(obs[3]), quality = obs[4]))
schemaTempReadings2 = sqlContext.createDataFrame(rdddata2)
schemaTempReadings2.registerTempTable("schemaTempReadings2")


#Q2.1 Temperatures readings higher than 10 degrees

overTenTemp = sqlContext.sql(" \
                        SELECT FIRST(year), FIRST(month), COUNT(value) AS counts\
                        FROM schemaTempReadings2 \
                        WHERE value >= 10 AND year >= 1950 AND year <= 2014\
                        GROUP BY year, month \
                        ORDER BY counts DESC")

#Q2.2 Distinct Temperatures readings higher than 10 degrees

overTenTempDistinct = schemaTempReadings2.filter(schemaTempReadings2["value"] > 10) \
                                .groupBy("yymm") \
                                .agg(F.countDistinct("station").alias("count"))
overTenTempDistinct = overTenTempDistinct.rdd.repartition(1) \
                            .sortBy(ascending = False, keyfunc = lambda \
                                    (yymm, counts): counts)
overTenTempDistinct.saveAsTextFile('/user/x_omkbh/bda2.2')
```

### Output of Distinct Temperatures readings counts:

```
print overTenTempDistinct.take(10)
[Row(yymm=u'1972-10', count=378),
 Row(yymm=u'1973-05', count=377),
 Row(yymm=u'1973-06', count=377),
 Row(yymm=u'1973-09', count=376),
 Row(yymm=u'1972-08', count=376),
 Row(yymm=u'1972-05', count=375),
 Row(yymm=u'1972-06', count=375),
 Row(yymm=u'1972-09', count=375),
 Row(yymm=u'1971-08', count=375),
 Row(yymm=u'1972-07', count=374)]
```

```
#Question 3.
#year, month, station, avgMonthlyTemperature ORDER BY avgMonthlyTemperature DESC
rdddata3 = sc.textFile('/user/x_omkbh/data/temperature-readings.csv') \
            .map(lambda line: line.split(";")) \
                       .filter(lambda p:
                               (int(p[1][:4]) >= 1950 and
                                int(p[1][:4]) <= 2014)) \
                       .map(lambda p: Row(station=int(p[0]),
                                          day=p[1],
                                          month=p[1][:7],
                                          value=float(p[3])))
schemaTempReadings3 = sqlContext.createDataFrame(rdddata3)
schemaTempReadings3.registerTempTable("schemaTempReadings3")
avgMonthTemp = sqlContext.sql(
        """
        SELECT mytbl.month, mytbl.station, AVG(mytbl.max_value + mytbl.min_value) / 2 AS avg_value
        FROM
        (
        SELECT month, station, MIN(value) AS min_value, MAX(value) AS max_value
        FROM schemaTempReadings3
        GROUP BY day, month, station
        ) AS mytbl
        GROUP BY mytbl.month, mytbl.station
        ORDER BY AVG(mytbl.max_value + mytbl.min_value) / 2 DESC
        """
    )
avgMonthTemp.rdd.repartition(1).sortBy(ascending=False,
                                keyfunc=lambda (month, station, value): value)
avgMonthTemp.write.save('/user/x_omkbh/bda2.3')
```

### Output Average monthly temperatures:

```
print avgMonthTemp.take(10)
[Row(month=u'2014-07', station=96000, avg_value=26.3),
Row(month=u'1994-07', station=96550, avg_value=23.07105263157895),
Row(month=u'1983-08', station=54550, avg_value=23.0),
Row(month=u'1994-07', station=78140, avg_value=22.970967741935485),
Row(month=u'1994-07', station=85280, avg_value=22.872580645161293),
Row(month=u'1994-07', station=75120, avg_value=22.858064516129033),
Row(month=u'1994-07', station=65450, avg_value=22.856451612903232),
Row(month=u'1994-07', station=96000, avg_value=22.808064516129033),
Row(month=u'1994-07', station=95160, avg_value=22.764516129032256),
Row(month=u'1994-07', station=86200, avg_value=22.711290322580645)]
```

```
#Question 4.
#station, maxTemp, maxDailyPrecipitation ORDER BY station DESC
#Note: The correct result for this question should be empty.
```

```
# Temperatures
temperature_data = sc.textFile('/user/x_omkbh/data/temperature-readings.csv')
temperature_obs = temperature_data.map(lambda line: line.split(";")) \
                                    .map(lambda obs: Row(station=int(obs[0]),
                                                         temp=float(obs[3])))
schema_temp_readings = sqlContext.createDataFrame(temperature_obs)
schema_temp_readings.registerTempTable("temp_readings")
# precipitation
precipitation_data = sc.textFile('/user/x_omkbh/data/precipitation-readings.csv')
precipitation_obs = precipitation_data.map(lambda line: line.split(";")) \
                                        .map(lambda obs: Row(station=int(obs[0]),
                                                             day=obs[1],
                                                             precip=float(obs[3])))
schema_precip_readings = sqlContext.createDataFrame(precipitation_obs)
schema_precip_readings.registerTempTable("precip_readings")

combined = sqlContext.sql(
        """
        SELECT tr.station, MAX(temp) AS max_temp, MAX(precip) AS max_precip
        FROM schema_temp_readings AS tr
        INNER JOIN
        (
        SELECT station, SUM(precip) AS precip
        FROM schema_precip_readings
        GROUP BY day, station
        ) AS pr
        ON tr.station = pr.station
        WHERE (temp >= 25 AND temp <= 30)
        AND (precip >= 100 AND precip <= 200)
        GROUP BY tr.station
        ORDER BY tr.station DESC
        """
        )
tempPrec = combined.rdd.repartition(1) \
        .sortBy(ascending=False, keyfunc=lambda (station, temp, precip): station)

tempPrec.take(10)
tempPrec.saveAsTextFile('/user/x_omkbh/bda2.4')
```

## Output Max daily temperatures/precipitation:

```
#Station number, maximum measured temperature, maximum daily precipitation
(u'128510', (29.5, None))
(u'192830', (29.5, None))
(u'84660', (27.6, None))
(u'139110', (29.0, None))
(u'161670', (25.7, None))
(u'166940', (27.9, None))
(u'77180', (29.3, None))
(u'180740', (29.0, None))
(u'72340', (29.8, None))
```

```
(u'147560', (29.9, None))
(u'180750', (29.3, None))
(u'83460', (28.0, None))
(u'83620', (29.4, None))
(u'159680', (26.2, None))
(u'139340', (28.9, None))
```

```
#Question 5.
#year, month, avgMonthlyPrecipitation ORDER BY year DESC, month DESC
ostergotlandStations = sc.textFile('/user/x_omkbh/data/stations-Ostergotland.csv') \
                           .map(lambda line: line.split(";")) \
                           .map(lambda obs: int(obs[0])) \
                           .distinct().collect()
ostergotlandStations = sc.broadcast(ostergotlandStations)
ostergotlandStations = {station: True for station in ostergotlandStations.value}
precipitations = sc.textFile('/user/x_omkbh/data/precipitation-readings.csv') \
                                          .map(lambda line: line.split(";")) \
                                          .filter(lambda obs:
                                                    ostergotlandStations.get(int(obs[0]), False)) \
                                          .map(lambda obs: Row(day=obs[1],
                                                               month=obs[1][:7],
                                                               station=int(obs[0]),
                                                               precip=float(obs[3])))
precSchema = sqlContext.createDataFrame(precipitations)
precSchema.registerTempTable("PrecSchema")
avgMthPrec = sqlContext.sql(
        """
        SELECT mytbl2.month, AVG(mytbl2.precip) AS avg_precip
        FROM
        (
        SELECT mytbl1.month, mytbl1.station, SUM(mytbl1.precip) AS precip
        FROM
        (
        SELECT month, station, SUM(precip) AS precip
        FROM PrecSchema
        GROUP BY day, month, station
        ) AS mytbl1
        GROUP BY mytbl1.month, mytbl1.station
        ) AS mytbl2
        GROUP BY mytbl2.month
        ORDER BY mytbl2.month DESC
        """
    )
avgMthPrec.rdd.repartition(1).sortBy(ascending=False, keyfunc=lambda (month, precip): month)
avgMthPrec.saveAsTextFile('/user/x_omkbh/bda2.5')
```

### Output Ostergotland average monthly precipitation:

```
print avgMthPrec.take(10)
[Row(month=u'2016-07', avg_precip=0.0),
 Row(month=u'2016-06', avg_precip=47.6625),
 Row(month=u'2016-05', avg_precip=29.250000000000004),
 Row(month=u'2016-04', avg_precip=26.9),
 Row(month=u'2016-03', avg_precip=19.9625),
 Row(month=u'2016-02', avg_precip=21.5625),
 Row(month=u'2016-01', avg_precip=22.325),
 Row(month=u'2015-12', avg_precip=28.925),
 Row(month=u'2015-11', avg_precip=63.887499999999996),
 Row(month=u'2015-10', avg_precip=2.2625)]
```

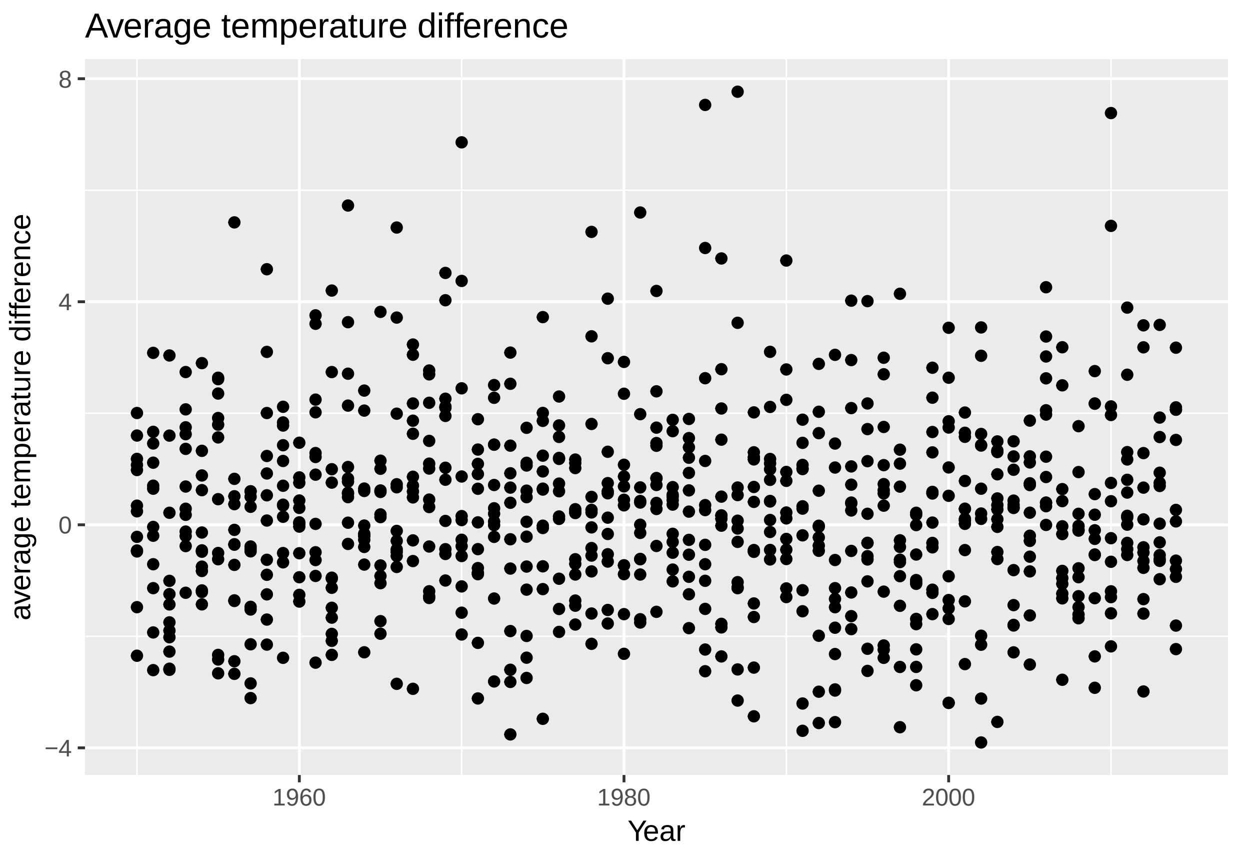
```
#Question 6.
#year, month, difference ORDER BY year DESC, month DESC
# Ostergotland Stations
ostergotlandStations = sc.textFile('/user/x_omkbh/data/stations-Ostergotland.csv') \
                           .map(lambda line: line.split(";")) \
                           .map(lambda obs: int(obs[0])) \
                           .distinct().collect()
ostergotlandStations = sc.broadcast(ostergotlandStations)
ostergotlandStations = {station: True for station in ostergotlandStations.value}
temperatures = sc.textFile('/user/x_omkbh/data/temperature-readings.csv') \
            .map(lambda line: line.split(";")) \
            .filter(lambda obs: ostergotlandStations.get(int(obs[0]), False)) \
            .map(lambda obs: \
                Row(station = obs[0], \
                    date = obs[1],  \
                    year = obs[1].split("-")[0], \
                    month = obs[1].split("-")[1], \
                    day = obs[1].split("-")[2], \
                    yymm = obs[1][:7], \
                    yymmdd = obs[1], \
                    time = obs[2], \
                    temp = float(obs[3]), \
                    quality = obs[4]))
tempSchema = sqlContext.createDataFrame(temperatures)
tempSchema.registerTempTable("TempSchema")
avgMthTemp = sqlContext.sql("""
        SELECT one.yymm,
            AVG(one.minTemp + one.maxTemp) / 2 AS avgTemp
        FROM
        (
        SELECT yymm,
                year,
                yymmdd,
                MIN(temp) AS minTemp,
                MAX(temp) AS maxTemp
        FROM TempSchema
        GROUP BY yymmdd,
                    yymm,
                    year,
                    station
        ) AS one
        WHERE one.year >= 1950 AND one.year <= 2014
        GROUP BY one.yymm
        """)
```

```
longTermAvgTemp = avgMthTemp.filter(avgMthTemp.substring(avgMthTemp["yymm"], 1, 4) <= 1980) \
                    .groupBy(avgMthTemp.substring(avgMthTemp["yymm"], 6, 7).alias("month")) \
                    .agg(avgMthTemp.avg(avgMthTemp["avgTemp"]).alias("longTermAvgTemp"))
diffTemp = avgMthTemp.join(longTermAvgTemp,
                                (avgMthTemp.substring(avgMthTemp["yymm"], 6, 7) ==
                                  longTermAvgTemp["month"]), "inner")
diffTemp = diffTemp.select(diffTemp["yymm"],
                          (diffTemp.abs(diffTemp["avgTemp"]) -
                             diffTemp.abs(diffTemp["longTermAvgTemp"])).alias("diffTemp"))
diffTemp = diffTemp.rdd.repartition(1).sortBy(ascending = False,
                             keyfunc = lambda (yymm, diff): yymm)
diffTemp.write.save('/user/x_omkbh/bda2.6')
```

### Output Ostergotland average monthly precipitation temperature difference:

```
#take(13)
1950,01,2.00483133412
1950,02,-2.34798988599
1950,03,1.1819828212
1950,04,1.60069315899
1950,05,0.982351940463
1950,06,-0.216232256095
1950,07,-1.47714267742
1950,08,0.241517150903
1950,09,0.343179398558
1950,10,-0.460520515247
1950,11,-0.47779366064
1950,12,1.07259158462
1951,01,-0.19629769814
1951,02,-2.60656131457
1951,03,3.08359572443
1951,04,-0.0381401743418
1951,05,-1.93038999502
```

```
part.000006 <- read.csv("C:/Users/Omkar/Downloads/Big Data Analytics/lab2/part-000006", header=FALSE)
part.000006 %>% ggplot(aes(x=V1, y=V3))+geom_point()+
  ggtitle("Average temperature difference")+
  labs(x="Year",y="average temperature difference")
```


Average temperature difference
8
4
0
−4
average temperature difference
1960
1980
2000
Year