BDA2

Rabnawaz jansher & Saman Zahid

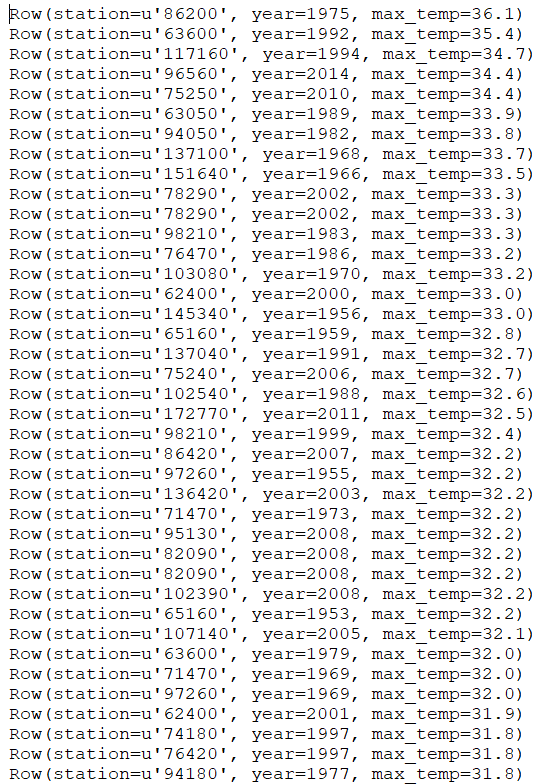
5/21/2018

# BDA2 - Spark Sql - Exercises

# Question 1

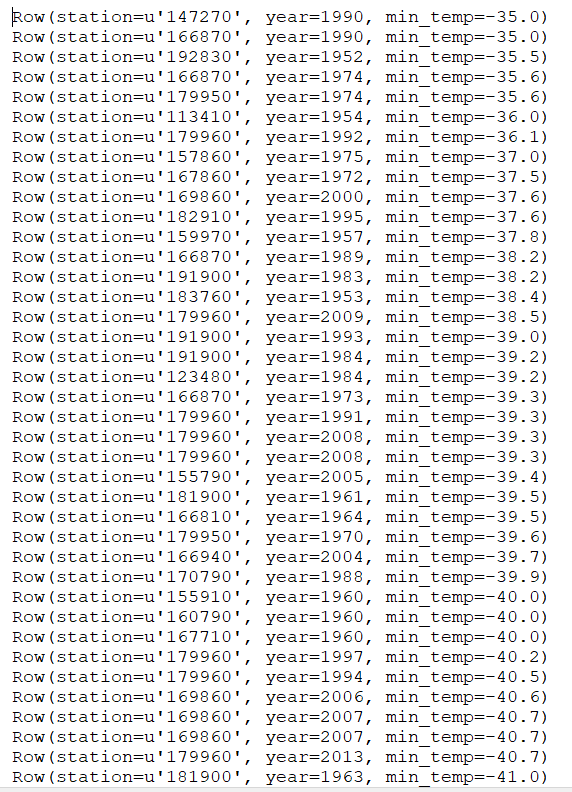
## Part A

#sql spark imports  
from pyspark import SparkContext  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
#create a spark objec  
sc = SparkContext(appName = "Lab2 Q1")  
#create a sql context  
sqlContext = SQLContext(sc)  
#read a temperatue data from file  
temperatureFile = sc.textFile("/user/x\_rabsh/data/temperature-readings.csv")  
lines = temperatureFile.map(lambda line: line.split(";"))  
#create temperature dataframe   
tempReadingsRows = lines.map(lambda x: (x[0],x[1],int(x[1][0:4]),float(x[3])))  
dataFrameString = ["station","date","year","temp"]  
df = sqlContext.createDataFrame(tempReadingsRows,dataFrameString)  
df.registerTempTable("tempReadingTable")  
#group by year and find max temperature  
df\_filter = df.groupBy('year').agg(F.max('temp').alias('max\_temp'))  
output = df.join(df\_filter,df\_filter.year == df.year ,'inner').select('station' , df\_filter.year,df\_filter.max\_temp , df.temp)  
#filter data by year   
output = output.where('temp = max\_temp').select('station','year','max\_temp').where( 'year >= 1950 and year <= 2014' )  
#order dataframe in descending order  
output = output.orderBy('max\_temp', ascending=False)  
#convert dataframe into Rdd  
output = output.rdd  
#repartion result and save into file  
output = output.coalesce(1)  
output.saveAsTextFile("lab2\_1a")



## Part B

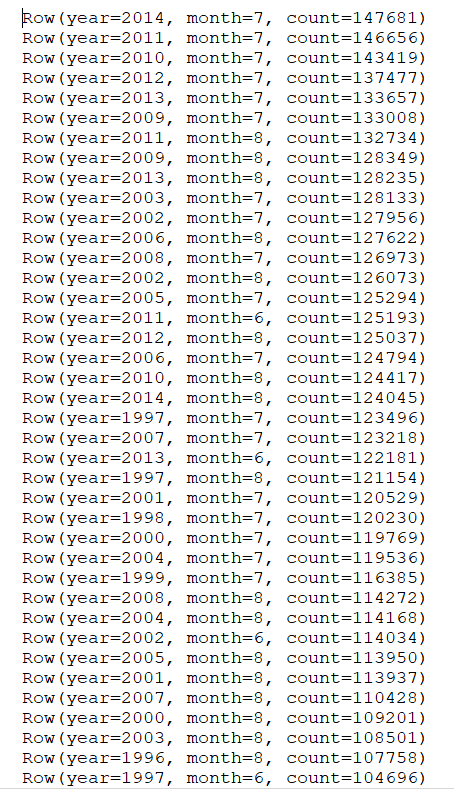
#import spark libraries  
from pyspark import SparkContext  
from operator import add  
#spark sql context  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
from pyspark.sql.functions import broadcast  
   
#spark context object  
sc = SparkContext(appName = "Lab2 Q5")  
#create a sql context  
sqlContext = SQLContext(sc)  
#temperature dataframe  
#read temperature data   
temperature\_file = sc.textFile("/user/x\_rabsh/data/temperature-readings.csv")  
temp\_lines = temperature\_file.map(lambda line: line.split(";"))  
#temperature dataframe  
tempRows = temp\_lines.map(lambda x: (x[0], x[1][0:10],int(x[1][0:4]),int(x[1][5:7]),int(x[1][8:10]), float(x[3]) ))  
tempDataString = ["station", "date", "year","month","day","temp"]  
#register temperature table  
dfTemp = sqlContext.createDataFrame(tempRows,tempDataString)  
dfTemp.registerTempTable("tempReadingTable")  
#filter stations by using broadcast Join  
dfTemp\_filter = dfTemp.where('year >= 1960 and year <= 2014')   
maxTemp = dfTemp\_filter.groupBy('date','station').agg(F.max('temp').alias('max\_temp'),F.min('temp').alias('min\_temp'))  
output = maxTemp.rdd  
#for now  
#output = output.coalesce(1)  
output.saveAsTextFile("ppp")



# Quesrion 2

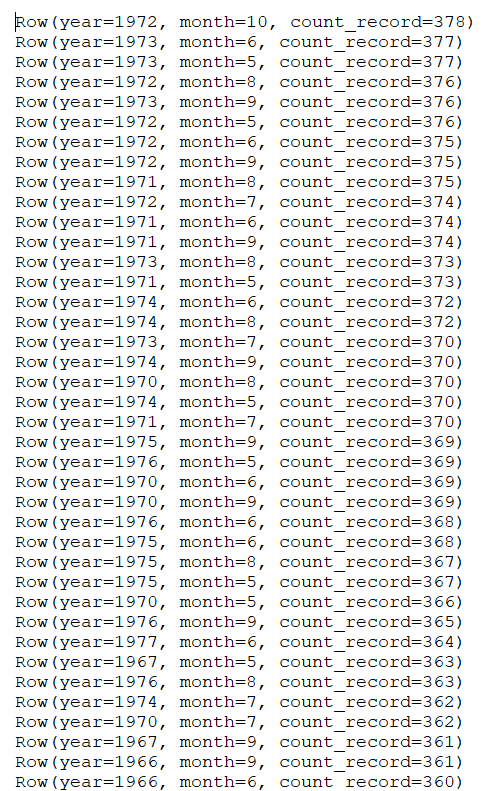
## Part A

#import spark libraries  
from pyspark import SparkContext  
from operator import add  
#spark sql context  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
#spark context object  
sc = SparkContext(appName = "Lab1 Q2-count-records")  
#create a sql context  
sqlContext = SQLContext(sc)  
#read temperature data from file  
temperature\_file = sc.textFile("/user/x\_rabsh/data/temperature-readings.csv")  
lines = temperature\_file.map(lambda line: line.split(";"))  
#create a temperature datafreame  
tempReadingsRows = lines.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]), int(p[1].split("-")[1]), float(p[3]),1))  
dataFrameString = ["station", "date", "year", "month", "temp","counter"]   
df = sqlContext.createDataFrame(tempReadingsRows,dataFrameString)  
df.registerTempTable("tempReadingTable")  
#select and filter data on year  
df\_select = df.select('year','month','counter','year','temp').where( 'year >= 1950 and year <= 2014 and temp > 10' )  
#count records  
output = df\_select.groupBy('year','month').agg(F.count('counter').alias('count')).orderBy('count', ascending=False)  
#convert dataframe into rdd  
output = output.rdd  
#repartion data and save  
output = output.coalesce(1)  
output.saveAsTextFile("lab2\_2a")



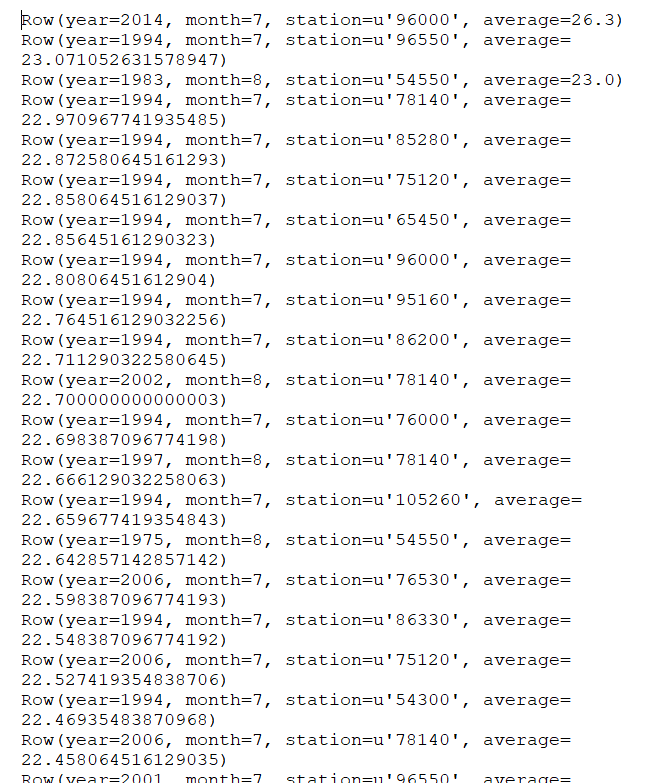
## Part B

#import spark libraries  
from pyspark import SparkContext  
from operator import add  
#spark sql context  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
#spark context object  
sc = SparkContext(appName = "Lab1 Q2b-count-distinct-records")  
#create a sql context  
sqlContext = SQLContext(sc)  
#read temperature from file  
temperature\_file = sc.textFile("/user/x\_rabsh/data/temperature-readings.csv")  
lines = temperature\_file.map(lambda line: line.split(";"))  
#create a data frame  
tempReadingsRows = lines.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]), int(p[1].split("-")[1]), float(p[3])))  
dataFrameString = ["station", "date", "year", "month", "temp"]  
df = sqlContext.createDataFrame(tempReadingsRows,dataFrameString)  
df.registerTempTable("tempReadingTable")  
#filter data year wise  
output = df.where(df.year >= 1950)\  
 .where(df.year <= 2014)\  
 .where(df.temp >= 10)  
#groupby data count distinct count   
output = output.groupBy(output.year, output.month).agg(F.countDistinct(output.station).alias('count\_record'))\  
 .orderBy('count\_record', ascending = False)  
#convert dataframe into rdd  
output = output.rdd  
#repartion data and save  
output = output.coalesce(1)  
output.saveAsTextFile("lab2\_2b")



# Quesrion 3

#sql spark imports  
from pyspark import SparkContext  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
sc = SparkContext(appName = "Lab2 Q3")  
# create a sql context  
sqlContext = SQLContext(sc)  
# reading data  
temperatureFile = sc.textFile("/user/x\_samza/data/temperature-readings.csv")  
lines = temperatureFile.map(lambda line: line.split(";"))  
# creating dataframe of temperatures data   
tempReadingsRows = lines.map(lambda x: (x[0], int(x[1][8:10]) ,int(x[1][0:4]) ,int(x[1][5:7]) , float(x[3]) ))  
dataFrameString = ["station","date","year","month","temp"]  
df = sqlContext.createDataFrame(tempReadingsRows,dataFrameString)  
df.registerTempTable("tempReadingTable")  
# filtering data within the period 1950 and 2014  
df = df.where('year >= 1950 and year <= 2014')  
# finding max temperature  
df\_groupby\_df = df.groupBy('year','month','date','station').agg(F.max('temp').alias('max\_temperature'))  
# finding min temperature  
df\_groupby = df.groupBy('year','month','date','station').agg(F.min('temp').alias('min\_temperature'))  
# joining dataframes containing min and max temperature and selecting required attributes  
df\_join = df\_groupby\_df.join(df\_groupby,(df\_groupby\_df.year == df\_groupby.year) & (df\_groupby\_df.month == df\_groupby.month) &   
 (df\_groupby\_df.date == df\_groupby.date) & (df\_groupby\_df.station == df\_groupby.station))\  
 .select(df\_groupby.year,df\_groupby.month,df\_groupby.date,df\_groupby.station,  
 df\_groupby.min\_temperature,df\_groupby\_df.max\_temperature)   
# determining daily avg i-e (daily\_max + daily\_min)/2  
daily\_average = df\_join.withColumn('sum\_min\_max', (df\_join.min\_temperature + df\_join.max\_temperature)/2)  
# determining monthly avg by avg daily avg over stations  
monthly\_avg = daily\_average.groupBy('year','month','station').agg(F.avg(daily\_average.sum\_min\_max).alias('average'))  
# sorting in descending order  
monthly\_avg = monthly\_avg.orderBy('average',ascending = False)  
output = monthly\_avg.rdd  
output = output.coalesce(1)  
# saving output  
output.saveAsTextFile("lab\_q2")



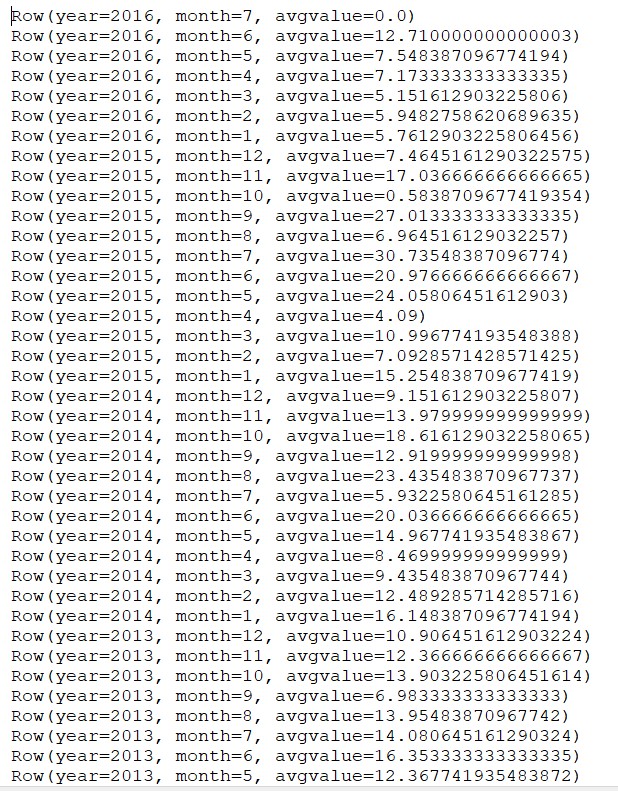
# Quesrion 4

#import spark libraries  
from pyspark import SparkContext  
from operator import add  
#spark sql context  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
   
#spark context object  
sc = SparkContext(appName = "Lab1 Q4")  
#create a sql context  
sqlContext = SQLContext(sc)  
#read preciptate data   
precipitation\_file = sc.textFile("/user/x\_rabsh/data/precipitation-readings.csv")  
precipt\_lines = precipitation\_file.map(lambda line: line.split(";"))  
#preciptiate dataframe  
preciptRows = precipt\_lines.map(lambda x: (x[0], x[1][0:10], float(x[3]) ))  
preciptDataString = ["station", "date", "value"]  
#register temperature table  
dfPrecipt = sqlContext.createDataFrame(preciptRows,preciptDataString)  
dfPrecipt.registerTempTable("tempReadingTable")  
#sum preciptate  
precipt\_filter = dfPrecipt.groupBy('station','date') \  
 .agg(F.sum('value').alias('pvalue'))  
output\_2 = precipt\_filter.groupBy(precipt\_filter.station)\  
 .agg(F.max(precipt\_filter.pvalue).alias('max\_precipt'))   
max\_precipt\_r = output\_2.where('max\_precipt >= 100 and max\_precipt <= 200')  
#temperature file reading  
temperature\_file = sc.textFile("/user/x\_rabsh/data/temperature-readings.csv")  
temp\_lines = temperature\_file.map(lambda line: line.split(";"))  
#temperature dataframe   
tempReadingsRows = temp\_lines.map(lambda p: (p[0], p[1], int(p[1].split("-")[0]), int(p[1].split("-")[1]), float(p[3])))  
dataFrameString = ["station", "date", "year", "month", "temp"]  
#register temperature table  
dfTemperature = sqlContext.createDataFrame(tempReadingsRows,dataFrameString)  
dfTemperature.registerTempTable("tempReadingTable")  
#max temperature  
df\_filter = dfTemperature.groupBy(dfTemperature.station).agg(F.max('temp').alias('max\_temp'))  
output = dfTemperature.join(df\_filter,df\_filter.station == dfTemperature.station ,'inner')\  
 .select(dfTemperature.station , df\_filter.max\_temp , dfTemperature.temp)  
output = output.where('temp = max\_temp')  
output = output.where('max\_temp >= 20 and max\_temp <= 30')  
combine\_result = max\_precipt\_r.join(output, output.station == max\_precipt\_r.station)\  
 .select(output.station,output.max\_temp,max\_precipt\_r.max\_precipt)  
output = combine\_result.orderBy('station', ascending=False)  
output = output.rdd  
#for now  
output = output.coalesce(1)  
output.saveAsTextFile("lab2\_q4")

Because there is no matching between 2 Rdd, so result will be null.

# Quesrion 5

#import spark libraries  
from pyspark import SparkContext  
from operator import add  
#spark sql context  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
from pyspark.sql.functions import broadcast  
   
#spark context object  
sc = SparkContext(appName = "Lab2 Q5")  
#create a sql context  
sqlContext = SQLContext(sc)  
#reading data  
ostergotland\_file = sc.textFile("/user/x\_rabsh/data/stations-Ostergotland.csv")  
#partition data  
stations = ostergotland\_file.map(lambda line: line.split(";"))  
#station data frame   
stationRow = stations.map(lambda x: (x[0],x[1]) )  
stationDataFrameString = ["station","name"]  
stations = sqlContext.createDataFrame(stationRow,stationDataFrameString)  
stations.registerTempTable("tempReadingTable")  
#boradcast stations  
#stations = stations.distinct().collect()  
#Os\_stations = sc.broadcast(stations)  
#preciptiate dataframe  
#read preciptate data   
precipitation\_file = sc.textFile("/user/x\_rabsh/data/precipitation-readings.csv")  
precipt\_lines = precipitation\_file.map(lambda line: line.split(";"))  
#preciptiate dataframe  
preciptRows = precipt\_lines.map(lambda x: (x[0], x[1][0:10],int(x[1][0:4]),int(x[1][5:7]),int(x[1][8:10]), float(x[3]) ))  
preciptDataString = ["station", "date", "year","month","day","value"]  
#register temperature table  
dfPrecipt = sqlContext.createDataFrame(preciptRows,preciptDataString)  
dfPrecipt.registerTempTable("tempReadingTable")  
#filter stations by using broadcast Join  
filter\_stations = stations.join(dfPrecipt , dfPrecipt.station == stations.station)\  
 .select(stations.station,dfPrecipt.value,dfPrecipt.date,dfPrecipt.year,dfPrecipt.month,dfPrecipt.day)  
#groupby year,month and day and add precpitate values  
filter\_stations = filter\_stations.groupBy('year','month','day') \  
 .agg(F.sum('value').alias('pvalue'))  
filter\_stations = filter\_stations.groupBy(filter\_stations.year,filter\_stations.month).agg(F.avg('pvalue').alias('avgvalue'))  
#sort data by year and month in Descending order  
filter\_stations = filter\_stations.orderBy(['year', 'month'], ascending=False)  
#convert datafream into Rdd  
output = filter\_stations.rdd  
#parartion data and save  
output = output.coalesce(1)  
output.saveAsTextFile("lab2\_q5")



# Quesrion 6

#sql spark imports  
from pyspark import SparkContext  
from pyspark.sql import SQLContext, Row  
from pyspark.sql import functions as F  
#spark object   
sc = SparkContext(appName = "Lab2 Q6")  
#create a sql context  
sqlContext = SQLContext(sc)  
#reading station data  
ostergotland\_file = sc.textFile("/user/x\_rabsh/data/stations-Ostergotland.csv")  
#partition data  
stations = ostergotland\_file.map(lambda line: line.split(";"))  
#station data frame   
stationRow = stations.map(lambda x: (x[0],x[1]) )  
stationDataFrameString = ["station","name"]  
stations = sqlContext.createDataFrame(stationRow,stationDataFrameString)  
stations.registerTempTable("stationReadingTable")  
# reading temperature data from file  
temperatureFile = sc.textFile("/user/x\_rabsh/data/temperature-readings.csv")  
lines = temperatureFile.map(lambda line: line.split(";"))  
#create a temperature dataframe  
tempReadingsRows = lines.map(lambda x: (x[0], int(x[1][8:10]) ,int(x[1][0:4]) ,int(x[1][5:7]) , float(x[3]) ))  
dataFrameString = ["station","date","year","month","temp"]  
df = sqlContext.createDataFrame(tempReadingsRows,dataFrameString)  
df.registerTempTable("tempReadingTable")  
#filter stations from temeperature dataframe  
station\_filter = df.join(stations, df.station == stations.station)\  
 .select(stations.station,df.date,df.month,df.year,df.temp)  
#monthly average temperature  
max\_temp = station\_filter.groupBy('year','month','date','station').agg(F.max('temp').alias('max\_temperature'))  
min\_temp = station\_filter.groupBy('year','month','date','station').agg(F.min('temp').alias('min\_temperature'))  
df\_join = max\_temp.join(min\_temp,(min\_temp.year == max\_temp.year) & (min\_temp.month == max\_temp.month) & (min\_temp.date == max\_temp.date) & (min\_temp.station == max\_temp.station))\  
 .select(min\_temp.year,min\_temp.month,min\_temp.date,min\_temp.station,min\_temp.min\_temperature,max\_temp.max\_temperature)   
daily\_average = df\_join.withColumn('sum\_min\_max', (df\_join.min\_temperature + df\_join.max\_temperature)/2)  
monthly\_avg = daily\_average.groupBy('year','month','station').agg(F.avg(daily\_average.sum\_min\_max).alias('average'))  
#average by year  
year\_average\_r = monthly\_avg.groupBy('year','month').agg(F.avg(monthly\_avg.average).alias('year\_average'))  
#filter year  
year\_average\_r = year\_average\_r.where('year >= 1950 and year <= 1980')  
#long term average   
long\_term\_average\_r = year\_average\_r.groupBy(year\_average\_r.month).agg(F.avg(year\_average\_r.year\_average).alias('long\_term\_average'))  
long\_term\_average\_r = year\_average\_r.join(long\_term\_average\_r, long\_term\_average\_r.month == year\_average\_r.month,'left\_outer')\  
 .select(year\_average\_r.year,year\_average\_r.year\_average,year\_average\_r.month,long\_term\_average\_r.long\_term\_average)  
#find a difference in temperature  
difference\_temp = long\_term\_average\_r.withColumn('difference\_temp', (long\_term\_average\_r.year\_average - long\_term\_average\_r.long\_term\_average))  
difference\_temp = difference\_temp.select('year','month','difference\_temp').orderBy(['year', 'month'], ascending=False)   
output = difference\_temp.rdd  
output = output.coalesce(1)  
output.saveAsTextFile("lab2\_q6")

