LAB-2

Siddhesh Sreedar (sidsr770)
Hugo Morvan (hugmo418)

Question 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.): Year, temperature Please notice that filtering before the reduce step will save the time and resource for running your program

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

Code :

```
#!/usr/bin/env python3

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

sc = SparkContext(appName = "exercise 1")
sqlContext = SQLContext(sc)

# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (year,temperature)
data = lines.map(lambda x: Row(station = x[0], year=x[1][0:4],
temp=float(x[3])))

df = sqlContext.createDataFrame(data)

df.registerTempTable("tempReadings")

#filter
#year_temperature = year_temperature.filter(lambda x:
int(x[0])>=1950 and int(x[0])<=2014)
```

```
df=df.filter((df["year"] >= 1950) & (df["year"] <= 2014))

#====MAX====
#Getsmax temp per year per station
#dfmax = df.groupBy("year").agg(F.max("temp")).alias("yearly_max")

#sort by temperature
#dfmax = df.orderBy(["max(temp)"],ascending=[0])

#====MIN====
#Getsmin temp per year per station
dfmin = df.groupBy("year").agg(F.min("temp")).alias("yearly_min")

#sort by temperature
dfmin = df.orderBy(["min(temp)"],ascending=[0])

# Following code will save the result into
/user/ACCOUNT_NAME/BDA/output folder
#dfmax_RDD = dfmax.rdd
dfmin_RDD = dfmin.rdd

#dfmax_RDD.saveAsTextFile("BDA/output")
dfmin_RDD.saveAsTextFile("BDA/output")
```

Output:

MAX temperatures

```
================= FINAL OUTPUT
==========================================
Row(year='1975', max(temp)=36.1)
Row(year='1992', max(temp)=35.4)
Row(year='1994', max(temp)=34.7)
Row(year='2014', max(temp)=34.4)
Row(year='2010', max(temp)=34.4)
Row(year='1989', max(temp)=33.9)
Row(year='1982', max(temp)=33.8)
Row(year='1968', max(temp)=33.7)
Row(year='1966', max(temp)=33.5)
Row(year='2002', max(temp)=33.3)
Row(year='1983', max(temp)=33.3)
Row(year='1986', max(temp)=33.2)
Row(year='1970', max(temp)=33.2)
Row(year='1956', max(temp)=33.0)
Row(year='2000', max(temp)=33.0)
Row(year='1959', max(temp)=32.8)
Row(year='2006', max(temp)=32.7)
Row(year='1991', max(temp)=32.7)
Row(year='1988', max(temp)=32.6)
Row(year='2011', max(temp)=32.5)
Row(year='1999', max(temp)=32.4)
Row(year='1953', max(temp)=32.2)
Row(year='2008', max(temp)=32.2)
Row(year='1955', max(temp)=32.2)
Row(year='1973', max(temp)=32.2)
Row(year='2007', max(temp)=32.2)
Row(year='2003', max(temp)=32.2)
Row(year='2005', max(temp)=32.1)
Row(year='1969', max(temp)=32.0)
Row(year='1979', max(temp)=32.0)
Row(year='2001', max(temp)=31.9)
Row(year='1977', max(temp)=31.8)
Row(year='1997', max(temp)=31.8)
Row(year='2013', max(temp)=31.6)
Row(year='2009', max(temp)=31.5)
Row(year='2012', max(temp)=31.3)
Row(year='1972', max(temp)=31.2)
Row(year='1971', max(temp)=31.2)
Row(year='1964', max(temp)=31.2)
Row(year='1976', max(temp)=31.1)
Row(year='1961', max(temp)=31.0)
Row(year='1963', max(temp)=31.0)
Row(year='1958', max(temp)=30.8)
Row(year='1978', max(temp)=30.8)
…
```

MIN Temperatures

```
================= FINAL OUTPUT
==========================================
Row(year='1990', min(temp)=-35.0)
Row(year='1952', min(temp)=-35.5)
Row(year='1974', min(temp)=-35.6)
Row(year='1954', min(temp)=-36.0)
Row(year='1992', min(temp)=-36.1)
Row(year='1975', min(temp)=-37.0)
Row(year='1972', min(temp)=-37.5)
Row(year='2000', min(temp)=-37.6)
Row(year='1995', min(temp)=-37.6)
Row(year='1957', min(temp)=-37.8)
Row(year='1989', min(temp)=-38.2)
Row(year='1983', min(temp)=-38.2)
Row(year='1953', min(temp)=-38.4)
Row(year='2009', min(temp)=-38.5)
Row(year='1993', min(temp)=-39.0)
Row(year='1984', min(temp)=-39.2)
Row(year='2008', min(temp)=-39.3)
Row(year='1973', min(temp)=-39.3)
Row(year='1991', min(temp)=-39.3)
Row(year='2005', min(temp)=-39.4)
Row(year='1964', min(temp)=-39.5)
Row(year='1961', min(temp)=-39.5)
Row(year='1970', min(temp)=-39.6)
Row(year='2004', min(temp)=-39.7)
Row(year='1988', min(temp)=-39.9)
Row(year='1960', min(temp)=-40.0)
Row(year='1997', min(temp)=-40.2)
Row(year='1994', min(temp)=-40.5)
Row(year='2006', min(temp)=-40.6)
Row(year='2013', min(temp)=-40.7)
Row(year='2007', min(temp)=-40.7)
Row(year='1963', min(temp)=-41.0)
Row(year='1955', min(temp)=-41.2)
Row(year='1969', min(temp)=-41.5)
Row(year='2003', min(temp)=-41.5)
Row(year='1996', min(temp)=-41.7)
Row(year='2010', min(temp)=-41.7)
Row(year='1962', min(temp)=-42.0)
Row(year='1968', min(temp)=-42.0)
Row(year='2011', min(temp)=-42.0)
Row(year='1951', min(temp)=-42.0)
Row(year='1950', min(temp)=-42.0)
Row(year='1982', min(temp)=-42.2)
Row(year='2002', min(temp)=-42.2)
Row(year='1976', min(temp)=-42.2)
Row(year='1977', min(temp)=-42.5)
Row(year='2014', min(temp)=-42.5)
Row(year='2012', min(temp)=-42.7)
Row(year='1998', min(temp)=-42.7)
Row(year='1958', min(temp)=-43.0)
Row(year='1985', min(temp)=-43.4)
…
```

Question 2

Count the number of readings for each month in the period of 1950-2014 which are higher than 10 degrees.
The output should contain the following information: year, month, value ORDER BY value DESC

Code:

```
#!/usr/bin/env python3
```

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

sc = SparkContext(appName = "exercise 1")
sqlContext = SQLContext(sc)

# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (yearmonth,temperature)
data = lines.map(lambda x: Row(station = x[0], yearmonth=x[1][0:7],
temp=float(x[3])))

df = sqlContext.createDataFrame(data)

df.registerTempTable("tempReadings")

#filter
df=df.filter((df["yearmonth"][0:4] >= 1950) & (df["yearmonth"][0:4]
<= 2014))

#filter temperature
df = df.filter(df["temp"] > 10)

df = df.groupBy("yearmonth").agg(F.count("temp")).alias("count")

df = df.orderBy(["count(temp)"], ascending=[0])

# Following code will save the result into
/user/ACCOUNT_NAME/BDA/output folder

counts = df.rdd

counts.saveAsTextFile("BDA/output")
```

Output:

```
================= FINAL OUTPUT
========================================
Row(yearmonth='2014-07', count(temp)=147681)
Row(yearmonth='2011-07', count(temp)=146656)
Row(yearmonth='2010-07', count(temp)=143419)
Row(yearmonth='2012-07', count(temp)=137477)
Row(yearmonth='2013-07', count(temp)=133657)
Row(yearmonth='2009-07', count(temp)=133008)
Row(yearmonth='2011-08', count(temp)=132734)
Row(yearmonth='2009-08', count(temp)=128349)
Row(yearmonth='2013-08', count(temp)=128235)
Row(yearmonth='2003-07', count(temp)=128133)
Row(yearmonth='2002-07', count(temp)=127956)
Row(yearmonth='2006-08', count(temp)=127622)
Row(yearmonth='2008-07', count(temp)=126973)
Row(yearmonth='2002-08', count(temp)=126073)
Row(yearmonth='2005-07', count(temp)=125294)
Row(yearmonth='2011-06', count(temp)=125193)
Row(yearmonth='2012-08', count(temp)=125037)
Row(yearmonth='2006-07', count(temp)=124794)
Row(yearmonth='2010-08', count(temp)=124417)
Row(yearmonth='2014-08', count(temp)=124045)
Row(yearmonth='1997-07', count(temp)=123496)
Row(yearmonth='2007-07', count(temp)=123218)
Row(yearmonth='2013-06', count(temp)=122181)
Row(yearmonth='1997-08', count(temp)=121154)
Row(yearmonth='2001-07', count(temp)=120529)
Row(yearmonth='1998-07', count(temp)=120230)
Row(yearmonth='2000-07', count(temp)=119769)
Row(yearmonth='2004-07', count(temp)=119536)
Row(yearmonth='1999-07', count(temp)=116385)
Row(yearmonth='2008-08', count(temp)=114272)
Row(yearmonth='2004-08', count(temp)=114168)
Row(yearmonth='2002-06', count(temp)=114034)
Row(yearmonth='2005-08', count(temp)=113950)
Row(yearmonth='2001-08', count(temp)=113937)
Row(yearmonth='2007-08', count(temp)=110428)
Row(yearmonth='2000-08', count(temp)=109201)
Row(yearmonth='2003-08', count(temp)=108501)
Row(yearmonth='1996-08', count(temp)=107758)
Row(yearmonth='1997-06', count(temp)=104696)
Row(yearmonth='1999-06', count(temp)=103227)
Row(yearmonth='2007-06', count(temp)=103046)
Row(yearmonth='2008-06', count(temp)=102900)
Row(yearmonth='2010-06', count(temp)=102716)
Row(yearmonth='2006-06', count(temp)=102588)
Row(yearmonth='2014-06', count(temp)=101711)
Row(yearmonth='1998-08', count(temp)=101387)
Row(yearmonth='1996-07', count(temp)=99916)
Row(yearmonth='2003-06', count(temp)=99693)
Row(yearmonth='2011-09', count(temp)=99335)
Row(yearmonth='1999-08', count(temp)=97437)
Row(yearmonth='2006-09', count(temp)=97181)
Row(yearmonth='2012-06', count(temp)=94513)
Row(yearmonth='2001-06', count(temp)=93375)
Row(yearmonth='2005-06', count(temp)=90724)
…
```

Part-2

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. In this exercise you will use the temperature-readings.csv file.

Code:

```
#!/usr/bin/env python3
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext(appName = "exercise 1")
sqlContext = SQLContext(sc)

# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (yearmonth,temperature)
data = lines.map(lambda x: Row(station = x[0], yearmonth=x[1][0:7],
temp=float(x[3])))

df = sqlContext.createDataFrame(data)

df.registerTempTable("tempReadings")

#filter
df=df.filter((df["yearmonth"][0:4] >= 1950) & (df["yearmonth"][0:4]
<= 2014))

#filter temperature
df = df.filter(df["temp"] > 10)

#df = df.groupBy("station",
"yearmonth").agg(F.count("count"),F.count("temp")).alias("count")

df = df.withColumn("distinct",F.lit(1))

df =
df.groupBy("yearmonth").agg(F.count("distinct")).alias("count_distin
ct")
```

```
df = df.orderBy(["count(distinct)"], ascending=[0])

# Following code will save the result into
/user/ACCOUNT_NAME/BDA/output folder

counts = df.rdd

counts.saveAsTextFile("BDA/output")
```

Output:

```
================= FINAL OUTPUT
=========================================
Row(yearmonth='1972-10', count(distinct)=378)
Row(yearmonth='1973-05', count(distinct)=377)
Row(yearmonth='1973-06', count(distinct)=377)
Row(yearmonth='1973-09', count(distinct)=376)
Row(yearmonth='1972-08', count(distinct)=376)
Row(yearmonth='1972-06', count(distinct)=375)
Row(yearmonth='1972-05', count(distinct)=375)
Row(yearmonth='1971-08', count(distinct)=375)
Row(yearmonth='1972-09', count(distinct)=375)
Row(yearmonth='1971-06', count(distinct)=374)
Row(yearmonth='1972-07', count(distinct)=374)
Row(yearmonth='1971-09', count(distinct)=374)
Row(yearmonth='1973-08', count(distinct)=373)
Row(yearmonth='1971-05', count(distinct)=373)
Row(yearmonth='1974-08', count(distinct)=372)
Row(yearmonth='1974-06', count(distinct)=372)
Row(yearmonth='1970-08', count(distinct)=370)
Row(yearmonth='1974-05', count(distinct)=370)
Row(yearmonth='1973-07', count(distinct)=370)
Row(yearmonth='1974-09', count(distinct)=370)
Row(yearmonth='1971-07', count(distinct)=370)
Row(yearmonth='1970-09', count(distinct)=369)
Row(yearmonth='1976-05', count(distinct)=369)
Row(yearmonth='1975-09', count(distinct)=369)
Row(yearmonth='1970-06', count(distinct)=369)
Row(yearmonth='1975-06', count(distinct)=368)
Row(yearmonth='1976-06', count(distinct)=368)
Row(yearmonth='1975-08', count(distinct)=367)
Row(yearmonth='1975-05', count(distinct)=367)
Row(yearmonth='1970-05', count(distinct)=366)
Row(yearmonth='1976-09', count(distinct)=365)
Row(yearmonth='1977-06', count(distinct)=364)
Row(yearmonth='1967-05', count(distinct)=363)
Row(yearmonth='1976-08', count(distinct)=363)
Row(yearmonth='1970-07', count(distinct)=362)
Row(yearmonth='1974-07', count(distinct)=362)
Row(yearmonth='1967-09', count(distinct)=361)
Row(yearmonth='1966-06', count(distinct)=360)
Row(yearmonth='1966-09', count(distinct)=360)
Row(yearmonth='1969-09', count(distinct)=359)
Row(yearmonth='1966-08', count(distinct)=359)
…
```

Question 3

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 timeframe. In this exercise you will use the temperature-readings.csv file.
The output should contain the following information:
year, month, station, avgMonthlyTemperature ORDER BY avgMonthlyTemperature DESC

Code:

```
#!/usr/bin/env python3
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext(appName = "exercise 1")
sqlContext = SQLContext(sc)

# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (yearmonth,temperature)
data = lines.map(lambda x: Row(station = x[0], yearmonth=x[1][0:7], 
temp=float(x[3])))

df = sqlContext.createDataFrame(data)

df.registerTempTable("tempReadings")

#filter
df=df.filter((df["yearmonth"][0:4] >= 1960) & (df["yearmonth"][0:4] 
<= 2014))
```

```
df = df.groupBy("station",
"yearmonth").agg(F.avg("temp")).alias("average")
df = df.orderBy(["avg(temp)"], ascending=[0])

df=df.rdd

# Following code will save the result into
/user/ACCOUNT_NAME/BDA/output folder
df.saveAsTextFile("BDA/output")
```

Output:

```
================= FINAL OUTPUT
========================================
Row(station='96000', yearmonth='2014-07', avg(temp)=26.3)
Row(station='65450', yearmonth='1994-07', avg(temp)=23.65483870967742)
Row(station='95160', yearmonth='1994-07', avg(temp)=23.505376344086027)
Row(station='75120', yearmonth='1994-07', avg(temp)=23.26881720430107)
Row(station='105260', yearmonth='1994-07', avg(temp)=23.143820224719107)
Row(station='85280', yearmonth='1994-07', avg(temp)=23.108602150537635)
Row(station='54550', yearmonth='1983-08', avg(temp)=23.0)
Row(station='54550', yearmonth='1975-08', avg(temp)=22.9625)
Row(station='96550', yearmonth='1994-07', avg(temp)=22.957894736842114)
Row(station='96000', yearmonth='1994-07', avg(temp)=22.931182795698923)
Row(station='106070', yearmonth='1994-07', avg(temp)=22.822580645161295)
Row(station='173960', yearmonth='1972-07', avg(temp)=22.776666666666667)
Row(station='54300', yearmonth='1994-07', avg(temp)=22.76021505376344)
Row(station='85210', yearmonth='1994-07', avg(temp)=22.755913978494615)
Row(station='65450', yearmonth='2006-07', avg(temp)=22.74086021505376)
Row(station='75120', yearmonth='2006-07', avg(temp)=22.73010752688173)
Row(station='103080', yearmonth='1994-07', avg(temp)=22.708602150537626)
Row(station='92100', yearmonth='1994-07', avg(temp)=22.698924731182792)
Row(station='94180', yearmonth='1994-07', avg(temp)=22.68172043010753)
Row(station='83230', yearmonth='1994-07', avg(temp)=22.577419354838707)
Row(station='97490', yearmonth='1994-07', avg(temp)=22.57419354838709)
Row(station='82110', yearmonth='1994-07', avg(temp)=22.546236559139782)
Row(station='76530', yearmonth='2006-07', avg(temp)=22.534408602150542)
Row(station='83270', yearmonth='1994-07', avg(temp)=22.49354838709678)
Row(station='86470', yearmonth='1994-07', avg(temp)=22.46559139784947)
Row(station='76530', yearmonth='1994-07', avg(temp)=22.46021505376344)
Row(station='74080', yearmonth='1994-07', avg(temp)=22.45806451612903)
Row(station='76000', yearmonth='1994-07', avg(temp)=22.451612903225808)
Row(station='54300', yearmonth='1997-08', avg(temp)=22.446236559139784)
Row(station='86330', yearmonth='1994-07', avg(temp)=22.40537634408603)
Row(station='86200', yearmonth='1994-07', avg(temp)=22.381720430107528)
Row(station='53430', yearmonth='1994-07', avg(temp)=22.324731182795706)
Row(station='62400', yearmonth='1994-07', avg(temp)=22.320430107526885)
Row(station='105370', yearmonth='1994-07', avg(temp)=22.313978494623658)
Row(station='54300', yearmonth='2006-07', avg(temp)=22.31290322580645)
Row(station='78140', yearmonth='1994-07', avg(temp)=22.279569892473123)
Row(station='83130', yearmonth='1994-07', avg(temp)=22.276344086021513)
Row(station='78140', yearmonth='2002-08', avg(temp)=22.270967741935497)
Row(station='98180', yearmonth='2010-07', avg(temp)=22.270967741935483)
Row(station='75100', yearmonth='1994-07', avg(temp)=22.26236559139785)
Row(station='64290', yearmonth='1994-07', avg(temp)=22.249462365591405)
Row(station='53440', yearmonth='1994-07', avg(temp)=22.234125269978385)
Row(station='98180', yearmonth='1994-07', avg(temp)=22.183870967741935)
Row(station='78140', yearmonth='2006-07', avg(temp)=22.1774193548387)
(...)
```

Question 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. In this exercise you will use the temperature-readings.csv and precipitation-readings.csv files. The output should contain the following information: Station number, maximum measured temperature, maximum daily precipitation

Code:

```
#!/usr/bin/env python3
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext(appName = "exercise 1")
sqlContext = SQLContext(sc)

# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

prec_file = sc.textFile("BDA/input/precipitation-readings.csv")
lines2 = prec_file.map(lambda line: line.split(";"))



# (station, yearmonth, temp)
data = lines.map(lambda x: Row(station = x[0], yearmonth=x[1][0:7],
temp=float(x[3])))
df = sqlContext.createDataFrame(data)
df.registerTempTable("tempReadings")
```

```
df =
df.groupBy("station").agg(F.max("temp")).alias("station_max_temp")
df = df.filter((df["max(temp)"] >= 25) & (df["max(temp)"] <= 30))

# (station, yearmonth, prec)
data2 = lines2.map(lambda x: Row(station = x[0], yearmonth=x[1],
prec=float(x[3])))
df2 = sqlContext.createDataFrame(data2)
df2.registerTempTable("precReadings")

df2 =
df2.groupBy("station","yearmonth").agg(F.sum("prec")).alias("station
_daily_prec")
df2 =
df2.groupBy("station").agg(F.max("sum(prec)")).alias("station_max_da
ily_prec")
df2 = df2.filter((df2["max(sum(prec))"] >= 100) &
(df2["max(sum(prec))"] <= 200))

df_join= df.join(df2,df.station == df2.station,"inner")


df_join=df_join.rdd

# Following code will save the result into
/user/ACCOUNT_NAME/BDA/output folder
df_join.saveAsTextFile("BDA/output")
```

Output:

================= FINAL OUTPUT
===
(empty)

Question 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 order to do this, you will first need to calculate the total monthly precipitation for each station before calculating the monthly average (by averaging over stations).
In this exercise you will use the precipitation-readings.csv and stations-Ostergotland.csv files. HINT (not for the SparkSQL lab): Avoid using joins here! stations-Ostergotland.csv is small and if distributed will cause a number of unnecessary shuffles when joined with precipitationRDD.

If you distribute precipitation-readings.csv then either repartition your
stations RDD to 1 partition or make use of the collect function to acquire a python list and broadcast function to broadcast the list to all nodes.
The output should contain the following information:
year, month, avgMonthlyPrecipitation ORDER BY year DESC, month DESC

Code:

```
#!/usr/bin/env python3
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext(appName = "exercise 1")
sqlContext = SQLContext(sc)

prec_file = sc.textFile("BDA/input/precipitation-readings.csv")
lines2 = prec_file.map(lambda line: line.split(";"))

data2 = lines2.map(lambda x: Row(station = x[0], year=
int(x[1][0:4]),month=int(x[1][6:7]), prec=float(x[3])))
df2 = sqlContext.createDataFrame(data2)
df2.registerTempTable("precReadings")

df2=df2.filter((df2["year"] >= 1993) & (df2["year"] <= 2016))

df2 =
df2.groupBy("station","year","month").agg(F.sum("prec")).alias("stat
ion_daily_prec")


stations_file = sc.textFile("BDA/input/stations-Ostergotland.csv")
stations = stations_file.map(lambda line: line.split(";"))

stationsdata = stations.map(lambda x: Row(station = x[0]))
stations = sqlContext.createDataFrame(stationsdata)
stations.registerTempTable("stations")

df_filter = df2.join(stations,df2.station ==
stations.station,"inner")

df_filter =
df_filter.groupBy("year","month").agg(F.avg("sum(prec)")).alias("ave
rage")

df_filter = df_filter.orderBy(["year","month"], ascending=[0,0])

df_filter = df_filter.rdd
```

```
# Following code will save the result into
/user/ACCOUNT_NAME/BDA/output folder
df_filter.saveAsTextFile("BDA/output")
```

Output:

```
================= FINAL OUTPUT
=========================================
Row(year=2016, month=7, avg(sum(prec))=0.0)
Row(year=2016, month=6, avg(sum(prec))=47.662499999999994)
Row(year=2016, month=5, avg(sum(prec))=29.250000000000007)
Row(year=2016, month=4, avg(sum(prec))=26.900000000000006)
Row(year=2016, month=3, avg(sum(prec))=19.962500000000006)
Row(year=2016, month=2, avg(sum(prec))=21.5625)
Row(year=2016, month=1, avg(sum(prec))=22.325)
Row(year=2015, month=9, avg(sum(prec))=101.29999999999998)
Row(year=2015, month=8, avg(sum(prec))=26.9875)
Row(year=2015, month=7, avg(sum(prec))=119.09999999999997)
Row(year=2015, month=6, avg(sum(prec))=78.66250000000002)
Row(year=2015, month=5, avg(sum(prec))=93.22499999999998)
Row(year=2015, month=4, avg(sum(prec))=15.337499999999999)
Row(year=2015, month=3, avg(sum(prec))=42.61250000000001)
Row(year=2015, month=2, avg(sum(prec))=53.750000000000014)
Row(year=2015, month=1, avg(sum(prec))=122.99999999999991)
Row(year=2015, month=0, avg(sum(prec))=2.2625)
Row(year=2014, month=9, avg(sum(prec))=48.45000000000001)
Row(year=2014, month=8, avg(sum(prec))=90.81249999999999)
Row(year=2014, month=7, avg(sum(prec))=22.987500000000004)
Row(year=2014, month=6, avg(sum(prec))=75.13750000000002)
Row(year=2014, month=5, avg(sum(prec))=58.000000000000014)
Row(year=2014, month=4, avg(sum(prec))=31.762500000000003)
Row(year=2014, month=3, avg(sum(prec))=36.56250000000001)
Row(year=2014, month=2, avg(sum(prec))=79.17500000000001)
Row(year=2014, month=1, avg(sum(prec))=114.99999999999997)
Row(year=2014, month=0, avg(sum(prec))=72.13749999999999)
Row(year=2013, month=9, avg(sum(prec))=26.18750000000001)
Row(year=2013, month=8, avg(sum(prec))=54.07500000000001)
Row(year=2013, month=7, avg(sum(prec))=54.56249999999999)
Row(year=2013, month=6, avg(sum(prec))=61.324999999999996)
Row(year=2013, month=5, avg(sum(prec))=47.92500000000001)
Row(year=2013, month=4, avg(sum(prec))=38.28750000000001)
Row(year=2013, month=3, avg(sum(prec))=7.387499999999998)
Row(year=2013, month=2, avg(sum(prec))=67.78750000000002)
Row(year=2013, month=1, avg(sum(prec))=67.9)
Row(year=2013, month=0, avg(sum(prec))=53.875)
Row(year=2012, month=9, avg(sum(prec))=72.75)
Row(year=2012, month=8, avg(sum(prec))=68.81666666666665
(...)
```

Q.3) **Fix**

Code:

```
#!/usr/bin/env python3
from pyspark import SparkContext
from pyspark.sql import SQLContext, Row
from pyspark.sql import functions as F

sc = SparkContext(appName = "exercise 1")
sqlContext = SQLContext(sc)

# This path is to the file on hdfs
temperature_file = sc.textFile("BDA/input/temperature-readings.csv")
lines = temperature_file.map(lambda line: line.split(";"))

# (key, value) = (station,yearmonthday,temperature)
#data = lines.map(lambda x: Row(station = x[0]+x[1][0:10], temp=float(x[3])))
data = lines.map(lambda x: Row(station = x[0], date = x[1], temp=float(x[3])))

df = sqlContext.createDataFrame(data)

df.registerTempTable("tempReadings")

#filter
df=df.filter((df["date"][0:4] >= 1960) & (df["date"][0:4] <= 2014))


df_max = df.groupBy("station","date").agg(F.max("temp")).alias("max_day")
df_min =  df.groupBy("station","date").agg(F.min("temp")).alias("min_day")


#df_join = df_max.join(df_min,(df_max["station"] == df_min["station"]) & (df_min["date"] ==
df_max["date"]))

df_join = df_max.join(df_min,on = ["station","date"])


#df_day_average = df_join.map(lambda x: x[0],(x[1]+x[2])/2)
```

```
df_daily_avg = df_join.withColumn("temp_day_avg", (F.col("max(temp)") +
F.col("min(temp)")) / 2)


#removed day from key
#df = df_day_average.map(lambda x: x[0][0:7]+x[-10:-3],x[1])
df_daily_avg = df_daily_avg.withColumn("year_month", F.date_trunc("MM","date"))



df =
df_daily_avg.groupBy("station","year_month").agg(F.avg("temp_day_avg")).alias("average")
df = df.orderBy(["avg(temp_day_avg)"], ascending=[0])

df=df.rdd

# Following code will save the result into /user/ACCOUNT_NAME/BDA/output folder
df.saveAsTextFile("BDA/output")
```

Output:

```
================= FINAL OUTPUT
==========================================
Row(station='96000', year_month=datetime.datetime(2014, 7, 1, 0, 0),
avg(temp_day_avg)=26.3)
Row(station='96550', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=23.071052631578947)
Row(station='54550', year_month=datetime.datetime(1983, 8, 1, 0, 0),
avg(temp_day_avg)=23.0)
Row(station='78140', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.970967741935485)
Row(station='85280', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.87258064516129)
Row(station='75120', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.858064516129033)
Row(station='65450', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.85645161290323)
Row(station='96000', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.80806451612903)
Row(station='95160', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.76451612903226)
Row(station='86200', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.711290322580645)
Row(station='78140', year_month=datetime.datetime(2002, 8, 1, 0, 0),
avg(temp_day_avg)=22.700000000000003)
```

```
Row(station='76000', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.69838709677419)
Row(station='78140', year_month=datetime.datetime(1997, 8, 1, 0, 0),
avg(temp_day_avg)=22.666129032258066)
Row(station='105260', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.659677419354836)
Row(station='54550', year_month=datetime.datetime(1975, 8, 1, 0, 0),
avg(temp_day_avg)=22.642857142857142)
Row(station='76530', year_month=datetime.datetime(2006, 7, 1, 0, 0),
avg(temp_day_avg)=22.598387096774196)
Row(station='86330', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.548387096774192)
Row(station='75120', year_month=datetime.datetime(2006, 7, 1, 0, 0),
avg(temp_day_avg)=22.527419354838713)
Row(station='54300', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.469354838709684)
Row(station='78140', year_month=datetime.datetime(2006, 7, 1, 0, 0),
avg(temp_day_avg)=22.45806451612903)
Row(station='96550', year_month=datetime.datetime(2001, 7, 1, 0, 0),
avg(temp_day_avg)=22.40833333333333)
Row(station='98180', year_month=datetime.datetime(2010, 7, 1, 0, 0),
avg(temp_day_avg)=22.37903225806452)
Row(station='65450', year_month=datetime.datetime(2006, 7, 1, 0, 0),
avg(temp_day_avg)=22.377419354838707)
Row(station='85210', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.375806451612906)
Row(station='98180', year_month=datetime.datetime(2014, 7, 1, 0, 0),
avg(temp_day_avg)=22.367741935483874)
Row(station='98180', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.36774193548387)
Row(station='98180', year_month=datetime.datetime(2002, 8, 1, 0, 0),
avg(temp_day_avg)=22.366129032258065)
Row(station='92100', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.317741935483873)
Row(station='86470', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.308064516129033)
Row(station='83230', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.27258064516129)
Row(station='64290', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.259677419354837)
Row(station='97490', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.258064516129032)
Row(station='94180', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.253225806451614)
Row(station='173960', year_month=datetime.datetime(1972, 7, 1, 0, 0),
avg(temp_day_avg)=22.244999999999997)
Row(station='74080', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.241935483870964)
```

```
Row(station='54300', year_month=datetime.datetime(2006, 7, 1, 0, 0),
avg(temp_day_avg)=22.237096774193546)
Row(station='98210', year_month=datetime.datetime(2002, 8, 1, 0, 0),
avg(temp_day_avg)=22.235483870967744)
Row(station='106070', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.23225806451613)
Row(station='75100', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.229032258064514)
Row(station='53440', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.1975)
Row(station='83270', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.177419354838708)
Row(station='103080', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.164516129032258)
Row(station='82110', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.161290322580644)
Row(station='97120', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.135483870967743)
Row(station='98210', year_month=datetime.datetime(2010, 7, 1, 0, 0),
avg(temp_day_avg)=22.11129032258065)
Row(station='53430', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.09677419354838)
Row(station='86330', year_month=datetime.datetime(1997, 8, 1, 0, 0),
avg(temp_day_avg)=22.07903225806452)
Row(station='66500', year_month=datetime.datetime(2006, 7, 1, 0, 0),
avg(temp_day_avg)=22.054838709677416)
Row(station='76530', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=22.033870967741933)
Row(station='98210', year_month=datetime.datetime(1997, 8, 1, 0, 0),
avg(temp_day_avg)=21.983870967741932)
Row(station='98210', year_month=datetime.datetime(2014, 7, 1, 0, 0),
avg(temp_day_avg)=21.962903225806446)
Row(station='62400', year_month=datetime.datetime(1994, 7, 1, 0, 0),
avg(temp_day_avg)=21.951612903225804)
Row(station='62400', year_month=datetime.datetime(1997, 8, 1, 0, 0),
avg(temp_day_avg)=21.93870967741935)
```