SparkSQL R Sample - USA Daily Temperatures

In [ ]:
```
Sys.getenv("SPARK_HOME")
```

In [ ]:
```
# Set the correct value for SPARK_HOME if not set in your environment
if (nchar(Sys.getenv("SPARK_HOME")) < 1) {
  Sys.setenv(SPARK_HOME = "/Users/skalathur/MyApps/spark")
}
```

In [ ]:
```
Sys.setenv(SPARK_LOCAL_IP="localhost")
```

In [ ]:
```
# load the SparkR library (wait until it loads)
library(SparkR, lib.loc = c(file.path(Sys.getenv("SPARK_HOME"), "R", "lib")))
```

In [ ]:
```
# Start the Spark Session, wait until it starts
sparkR.session(master = "local[*]", sparkConfig = list(spark.driver.memory = "2g"
))
```

In [ ]:
```
inputFile <- "/temp/datasets/usa_daily_avg_temps.csv"
```

In [ ]:
```
# Read the csv file as a SparkDataFrame
usaDailyTemps <- read.df(inputFile, source = "csv",
                         header='true',
                         inferSchema='true')

usaDailyTemps
```

In [ ]:
```
printSchema(usaDailyTemps)
```

In [ ]:
```
count(usaDailyTemps)
```

In [ ]:
```
head(usaDailyTemps)
```

In [ ]:
```
persist(usaDailyTemps, "MEMORY_AND_DISK")
```

In [ ]:
```
# Register the DataFrame as a SQL table.
createOrReplaceTempView(usaDailyTemps, "usaDailyTempsTable")
```

Aggregate to find the maximum of avgtemp

In [ ]:
```
query <- "SELECT max(avgtemp) FROM usaDailyTempsTable"
query
```

In [ ]:
```
maxAvgTemp <- sql(query)
maxAvgTemp
```

In [ ]:
```
count(maxAvgTemp)
```

In [ ]:
```
# collect to local data frame
collect(maxAvgTemp)
```

In [ ]:
```
# Provide the appropriate column name (MaxValue)

query <- "SELECT max(avgtemp) AS MaxValue FROM usaDailyTempsTable"

maxAvgTemp <- sql(query)
maxAvgTemp
```

In [ ]:
```
localDf <- collect(maxAvgTemp)
localDf
```

In [ ]:
```
# Filter the SparkDataFrame to find the rows with the max value

query <- paste("SELECT * from usaDailyTempsTable WHERE avgtemp = ",
               localDf[1, 'MaxValue'])
query
```

In [ ]:
```
maxData <- sql(query)
maxData
```

In [ ]:
```
# collect to local data frame
collect(maxData)
```

Aggregate to find the maximum of avgtemp grouping by Year

In [ ]:
```
query <- "SELECT year, max(avgtemp) AS MaxValue FROM usaDailyTempsTable
             GROUP BY year"
query
```

In [ ]:
```
maxTempByYear <- sql(query)
maxTempByYear
```

In [ ]:
```
count(maxTempByYear)
```

In [ ]:
```
collect(maxTempByYear)
```

In [ ]:
```
query <- "SELECT year, max(avgtemp) AS MaxValue FROM usaDailyTempsTable
               GROUP BY year ORDER BY year"
query
```

In [ ]:
```
maxTempByYear <- sql(query)
maxTempByYear
```

In [ ]:
```
count(maxTempByYear)
collect(maxTempByYear)
```

Aggregate to find the maximum of avgtemp grouping by State

In [35]:
```
query <- "SELECT state, max(avgtemp) AS MaxValue FROM usaDailyTempsTable
               GROUP BY state ORDER BY state"
query
```

'SELECT state, max(avgtemp) AS MaxValue FROM usaDailyTempsTable GROUP BY state ORDER BY state'

In [36]:
```
maxTempByState <- sql(query)
maxTempByState
```

```
SparkDataFrame[state:string, MaxValue:double]
```

In [37]:
```
count(maxTempByState)
```

50

In [38]:
```
collect(maxTempByState)
```

state	MaxValue
Alabama	91.5
Alaska	79.5
Arizona	107.5
Arkansas	100.7
California	102.6
Colorado	94.7
Connecticut	89.8
Delaware	89.7
Florida	92.8
Georgia	97.7
Hawaii	87.2
Idaho	94.2
Illinois	92.3
Indiana	94.0
Iowa	93.0
Kansas	96.1
Kentucky	93.2
Louisiana	95.4
Maine	89.1
Maryland	92.8
Massachusetts	90.7
Michigan	89.4
Minnesota	92.0
Mississippi	92.8
Missouri	96.3
Montana	100.1
Nebraska	93.2
Nevada	105.5
New Hampshire	88.0
New Jersey	95.6
New Mexico	89.4
New York	93.7
North Carolina	91.0
North Dakota	91.7
Ohio	91.2
Oklahoma	100.4

Aggregate to find the number of entries grouping by State

In [40]:
```
query <- "SELECT state, count(*) AS count FROM usaDailyTempsTable
            GROUP BY state ORDER BY count DESC"
query
```

'SELECT state, count(*) AS count FROM usaDailyTempsTable
GROUP BY state ORDER BY count DESC'

In [41]:
```
stateCounts <- sql(query)
stateCounts
```

```
SparkDataFrame[state:string, count:bigint]
```

In [42]:
```
collect(stateCounts)
```

state	count
Texas	106736
Ohio	53368
Florida	51495
Pennsylvania	43871
Michigan	38120
California	38120
New York	38120
Oregon	30496
Illinois	30496
Georgia	30496
North Carolina	30496
Alabama	30496
Tennessee	30496
Indiana	30496
Colorado	30496
Louisiana	28670
Arizona	23202
Nebraska	22872
Washington	22872
Alaska	22872
Missouri	22872
Montana	22872
Virginia	22872
Kansas	22872
Kentucky	22872
Wisconsin	22872
Minnesota	15248
Arkansas	15248
North Dakota	15248
Connecticut	15248
Nevada	15248
Oklahoma	15248
West Virginia	15248
Wyoming	15248
New Jersey	15248
Maryland	15248

Aggregate to find the number of entries grouping by State and City

In [43]:
```
query <- "SELECT state, city, count(*) AS count FROM usaDailyTempsTable
            GROUP BY state, city ORDER BY state, city"
query
```

'SELECT state, city, count(*) AS count FROM usaDailyTempsTable
GROUP BY state, city ORDER BY state, city'

In [44]:
```
stateCityCounts <- sql(query)
stateCityCounts
```

```
SparkDataFrame[state:string, city:string, count:bigint]
```

In [45]:
```
collect(stateCityCounts)
```

state	city	count
Alabama	Birmingham	7624
Alabama	Huntsville	7624
Alabama	Mobile	7624
Alabama	Montgomery	7624
Alaska	Anchorage	7624
Alaska	Fairbanks	7624
Alaska	Juneau	7624
Arizona	Flagstaff	3574
Arizona	Phoenix	7624
Arizona	Tucson	7624
Arizona	Yuma	4380
Arkansas	Fort Smith	7624
Arkansas	Little Rock	7624
California	Fresno	7624
California	Los Angeles	7624
California	Sacramento	7624
California	San Diego	7624
California	San Francisco	7624
Colorado	Colorado Springs	7624
Colorado	Denver	7624
Colorado	Grand Junction	7624
Colorado	Pueblo	7624
Connecticut	Bridgeport	7624
Connecticut	Hartford Springfield	7624
Delaware	Wilmington	5751
Florida	Daytona Beach	5751
Florida	Jacksonville	7624
Florida	Miami Beach	7624
Florida	Orlando	7624
Florida	Tallahassee	7624
⋮	⋮	⋮
Tennessee	Nashville	7624
Texas	Abilene	7624
Texas	Amarillo	7624
Texas	Austin	7624
Texas	Brownsville	7624

Number of cities for each state in the dataset

In [46]:
```
# Register the DataFrame as a SQL table.

createOrReplaceTempView(stateCityCounts, "stateCityCountsTable")
```

In [47]:
```
query <- "SELECT state, count(*) AS count FROM stateCityCountsTable
              GROUP BY state ORDER BY state"
query
```

```
'SELECT state, count(*) AS count FROM stateCityCountsTable
        GROUP BY state ORDER BY state'
```

In [48]:
```
collect(sql(query))
```

state	count
Alabama	4
Alaska	3
Arizona	4
Arkansas	2
California	5
Colorado	4
Connecticut	2
Delaware	1
Florida	7
Georgia	4
Hawaii	1
Idaho	2
Illinois	4
Indiana	4
Iowa	2
Kansas	3
Kentucky	3
Louisiana	4
Maine	2
Maryland	2
Massachusetts	1
Michigan	5
Minnesota	2
Mississippi	2
Missouri	3
Montana	3
Nebraska	3
Nevada	2
New Hampshire	1
New Jersey	2
New Mexico	1
New York	5
North Carolina	4
North Dakota	2
Ohio	7
Oklahoma	2

Create a subset SparkDataFrame for Boston

In [54]:
```
query <- "SELECT * FROM usaDailyTempsTable WHERE city == 'Boston'"
query
```

'SELECT * FROM usaDailyTempsTable WHERE city == \'Boston''

In [55]:
```
bostonDailyTemps <- sql(query)

bostonDailyTemps
```

```
SparkDataFrame[state:string, city:string, month:int, day:int, year:int, avgtemp:
double]
```

In [56]:
```
count(bostonDailyTemps)
```

7624

In [57]:
```
# Register the DataFrame as a SQL table.

createOrReplaceTempView(bostonDailyTemps, "bostonDailyTempsTable")
```

Boston Average Temperatures By Year

In [58]:
```
query <- "SELECT year, avg(avgtemp) AS Average FROM bostonDailyTempsTable
          GROUP BY year ORDER BY year"
query
```

'SELECT year, avg(avgtemp) AS Average FROM bostonDailyTempsTable
GROUP BY year ORDER BY year'

In [59]:
```
bostonAvgTempsByYear <- sql(query)
bostonAvgTempsByYear
```

```
SparkDataFrame[year:int, Average:double]
```

In [60]:
```
collect(bostonAvgTempsByYear)
```

year	Average
1995	51.32027
1996	47.71749
1997	50.83863
1998	51.51562
1999	52.33945
2000	50.36148
2001	52.42822
2002	50.41205
2003	49.73014
2004	50.52514
2005	50.97726
2006	53.02055
2007	51.12219
2008	50.95355
2009	50.32247
2010	53.47205
2011	53.22384
2012	53.86749
2013	51.69753
2014	50.95452
2015	52.46959

Boston Average Temperatures By Year

In [61]:
```
query <- "SELECT month, avg(avgtemp) AS Average FROM bostonDailyTempsTable
            GROUP BY month ORDER BY month"
query
```

'SELECT month, avg(avgtemp) AS Average FROM bostonDailyTempsTable
GROUP BY month ORDER BY month'

In [62]:
```
bostonAvgTempsByMonth <- sql(query)
bostonAvgTempsByMonth
```

```
SparkDataFrame[month:int, Average:double]
```

In [63]:
```
collect(bostonAvgTempsByMonth)
```

month	Average
1	29.76667
2	31.47032
3	37.57604
4	47.08413
5	57.57803
6	66.10714
7	73.55038
8	71.68909
9	65.05762
10	54.73456
11	44.89366
12	34.99742

Boston Average Temperatures By Year and Month

In [64]:
```
query <- "SELECT year, month, avg(avgtemp) AS Average FROM bostonDailyTempsTable
              GROUP BY year, month ORDER BY year, month"
query
```

'SELECT year, month, avg(avgtemp) AS Average FROM bostonDailyTempsTable
GROUP BY year, month ORDER BY year, month'

In [65]:
```
bostonAvgTempsByYearAndMonth <- sql(query)
bostonAvgTempsByYearAndMonth
```

```
SparkDataFrame[year:int, month:int, Average:double]
```

In [66]:
```
collect(bostonAvgTempsByYearAndMonth)
```

year	month	Average
1995	1	34.51935
1995	2	28.57500
1995	3	38.03871
1995	4	45.42000
1995	5	56.69677
1995	6	68.47667
1995	7	75.57419
1995	8	72.52581
1995	9	62.93667
1995	10	58.07742
1995	11	42.20333
1995	12	31.04194
1996	1	30.04516
1996	2	30.71034
1996	3	26.99355
1996	4	27.20000
1996	5	51.57419
1996	6	66.72667
1996	7	71.46452
1996	8	70.37419
1996	9	63.72667
1996	10	53.51613
1996	11	40.15000
1996	12	39.25484
1997	1	29.02258
1997	2	36.32500
1997	3	36.59032
1997	4	45.82333
1997	5	55.48710
1997	6	67.87333
⋮	⋮	⋮
2013	6	68.80000
2013	7	76.05161
2013	8	71.67419
2013	9	64.59667
2013	10	56.28065

Boston years in data

In [67]:
```
query <- "SELECT distinct(year) FROM bostonDailyTempsTable ORDER BY year"
query
```

'SELECT distinct(year) FROM bostonDailyTempsTable ORDER BY year'

In [68]:
```
yearsDF <- collect(sql(query))
yearsDF
```

year
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015

In [69]:
```
# Stop the SparkSession now
sparkR.session.stop()
```