Code

Vivek Pandiyan

```
import org.apache.Spark.sql.functions
import org.joda.time.format.DateTimeFormat
Val inputPath = "C:/Users/vivek/Downloads/pressure.csv"
Val Pressure = Sql Context.read.
                 format ("com.databricks.Spark.csv")
                 .option ("header", "true")
                 .option ("delimiter", ",")
                 .option ("inferSchema", "true")
                 .load (inputpath)
Val temperature . Sql Context.read
                   . format ("com.databricks.Spark.BSV")
                   . option ("header", "true")
                   . option ("delimiter", ",")
                   . option ("inferschema", "true")
                   . load (inputpath)
Val inputdew: "C:/Users/vivek/downloads/dew.csv"
Val dew = Sql. Context.read.
                 .format ("com.databricks.Spark.csv").
                 .option ("header", "true")
                 .option ("delimiter", ",")
                 .option ("iferschema", "true")
                 .load ( inputdew
Val inputhumi = " C:/users/vivek/downloads/humidity.csv"
 Val humidity : Sql.Context.read
                 .format ("com.databricks.Spark.csv
                 .option ("header", "true")
                 .option ("delimiter", ",")
                 .option ("inferschema", "true").
```

Viren Pandiyan

```
                          . load(inputpathms)
Val inputWindd   = " c:/Users/ Viren/ dowloads/windDirection.csv
Val winddir      =   Sql. context .read.
                          . format ("com.databricks. Spark.com")
                          . format ("header", "true" )
                          . format ("delimiter", ",")
                          . Option ("inferSchema", "true")
                          . ~~load (inputpath)~~.
                          . load (inputwindd).

Val   input winds.   = "c:/Users/viren/downloads/windSpeed.csv"
Val   winds          = Sql . context . read
                          . format( "com.databricks. Spark.com" )
                          . Option ("header", "true" )
                          . option(" delimiter", ",")
                          . option(" inferSchema", "true" )
                          . load (input winds)
Val  inputVis =  "C:/users/viren/ downloads /Visibility.csv"
Val  Visibility =  Sql. context .read.
                          . format ("com.databricks. Spark.com" )
                          . Option ("header", "true" )
                          . option ( "delimiter ", ",")
                          . Option ("inferSchema", "true")
                          . load (inputVis)
```

Vivek Panchigar

```
(53) import matplotlib
(54) import matplotlib.pyplot as plt
(55) import StringIO
(56) import Pandas as pd.
(57) import Seaborn as sns
(58) df = Sql Context.sql ("Select timestamp, Pressure from Pressure
                              group by timestamp").
(59) %. pyspark .
(60) heatmap [0:100].
(61) From pyspark.Sql import SQL Context
(62) df = SqlContext.table( "Pressure").
(63) df.head() .
(64) import os
(65) import pandas as pd .
(66) list = []
(67) for row in glob.glob("c:/users/Vivek/downloads/pressure.csv")
(68)     data = pd.read_csv(row)
(69) data.count().
(70) % pyspark .
(71) def show(p):-
(72)  img = StringIO.StringIO()
(73)  p.savefig(img, format='svg')
(74)  img.seek(0)
```

Viven Panchyan

```
(75) Print "%html" + img.buf
(76) df = SqlContext.sql(" Select Pressure, timpstamp,
(77)        Month, count(*) as No of reading, Max(tempature),
(78)        Min(temperature) from Pressure P inner join
(79)        temperature T on P.timestamp = T.timestamp")
(80)   Value = " Max temp"
(81)   grouping = ["Month"]

(82)   Val Fireday = List("01/01/2016", "01/16/2016",
(83)                   "01/20/2016", "02/21/2016")
                       "03/07/2016, 03/07/206)
(84)

(85) def Forest_Fire : (String Array [Double]) = {
(86)   Val values = Array( Day.todouble,. humidity.todouble
(87)                       winds.todouble, temp.todouble
(88)                       daysfrom_maxtemp(year.toint,
(89)                       month.toint ,day.toint ))
(90) - import java.io._
(91)   import org.joda.time._
(92)   import org.joda.time.format._
(93)   import org.joda.time.frame.Datetime Fromat
(94)   import org.joda.time.Datetime
(95)   import org.joda.time.Days.
```

Vivek Pandey

```
def get-date ( year: Int, month: Int, day: Int ) =
    "%04d%02d%02d".format(year, month, day)

%sql

Select    timestamp, temperature, Pressure, humidity
          wind speed, wind direction, Visibility from
          Forest fire ~~groupby Item~~
          order by timestamp.

%sql

Select    max (temperature), Month( timestamp) from
          temperature group by Month(Timestamp)
          Order by month (timestamp).

%sql.

Select.   day (T.timestamp), day ( F.timpstamp ),
          Max (F.temperature), Max(F.pressure),
          From temperature T join Forest Fire F
       day (F.Timpestamp)=day(T.timestamp.)

% PySpark.
from pandas import Series, Dataframe.
import pandas as pd.
import numpy as np.
dataframe ( pressure ).
```