

RDDs, Datasets and DataFrames

# Spark - Exercises

# Exercise #30

- Log filtering
  - Input: a simplified log of a web server (i.e., a textual file)
    - Each line of the file is associated with a URL request
  - Output: the lines containing the word “google”
    - Store the output in an HDFS folder

# Exercise #30 - Example

## ■ Input file

```
66.249.69.97 - - [24/Sep/2014:22:25:44 +0000] "GET http://www.google.com/bot.html"  
66.249.69.97 - - [24/Sep/2014:22:26:44 +0000] "GET http://www.google.com/how.html"  
66.249.69.97 - - [24/Sep/2014:22:28:44 +0000] "GET http://dbdmg.polito.it/course.html"  
71.19.157.179 - - [24/Sep/2014:22:30:12 +0000] "GET http://www.google.com/faq.html"  
66.249.69.97 - - [24/Sep/2014:31:28:44 +0000] "GET http://dbdmg.polito.it/thesis.html"
```

## ■ Output

```
66.249.69.97 - - [24/Sep/2014:22:25:44 +0000] "GET http://www.google.com/bot.html"  
66.249.69.97 - - [24/Sep/2014:22:26:44 +0000] "GET http://www.google.com/how.html"  
71.19.157.179 - - [24/Sep/2014:22:30:12 +0000] "GET http://www.google.com/faq.html"
```

# Exercise #31

- Log analysis
  - Input: log of a web server (i.e., a textual file)
    - Each line of the file is associated with a URL request
  - Output: the list of distinct IP addresses associated with the connections to a google page (i.e., connections to URLs containing the term "www.google.com")
    - Store the output in an HDFS folder

# Exercise #31 - Example

## ■ Input file

```
66.249.69.97 - - [24/Sep/2014:22:25:44 +0000] "GET http://www.google.com/bot.html"  
66.249.69.97 - - [24/Sep/2014:22:26:44 +0000] "GET http://www.google.com/how.html"  
66.249.69.97 - - [24/Sep/2014:22:28:44 +0000] "GET http://dbdmg.polito.it/course.html"  
71.19.157.179 - - [24/Sep/2014:22:30:12 +0000] "GET http://www.google.com/faq.html"  
66.249.69.95 - - [24/Sep/2014:31:28:44 +0000] "GET http://dbdmg.polito.it/thesis.html"  
66.249.69.97 - - [24/Sep/2014:56:26:44 +0000] "GET http://www.google.com/how.html"  
56.249.69.97 - - [24/Sep/2014:56:26:44 +0000] "GET http://www.google.com/how.html"
```

## ■ Output

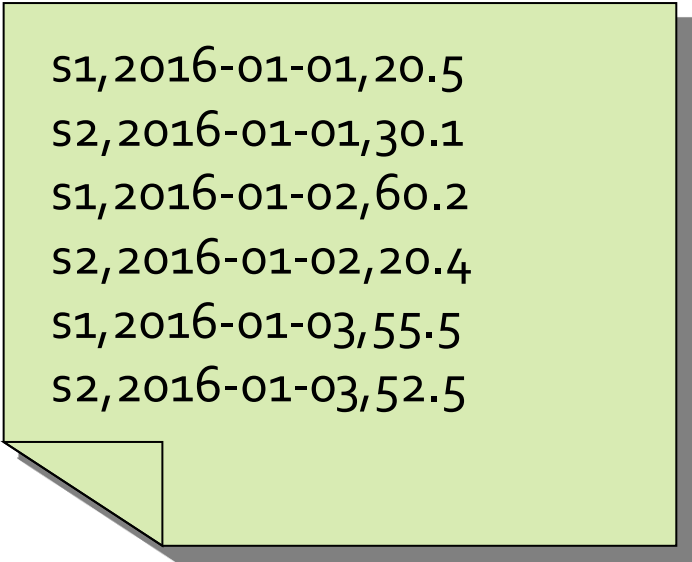
```
66.249.69.97  
71.19.157.179  
56.249.69.97
```

# Exercise #32

- Maximum value
  - Input: a collection of (structured) textual csv files containing the daily value of PM<sub>10</sub> for a set of sensors
    - Each line of the files has the following format  
sensorId,date,PM<sub>10</sub> value (µg/m<sup>3</sup>)\n
  - Output: report the maximum value of PM<sub>10</sub>
    - Print the result on the standard output

# Exercise #32 - Example

- Input file



```
s1,2016-01-01,20.5  
s2,2016-01-01,30.1  
s1,2016-01-02,60.2  
s2,2016-01-02,20.4  
s1,2016-01-03,55.5  
s2,2016-01-03,52.5
```

- Output

60.2

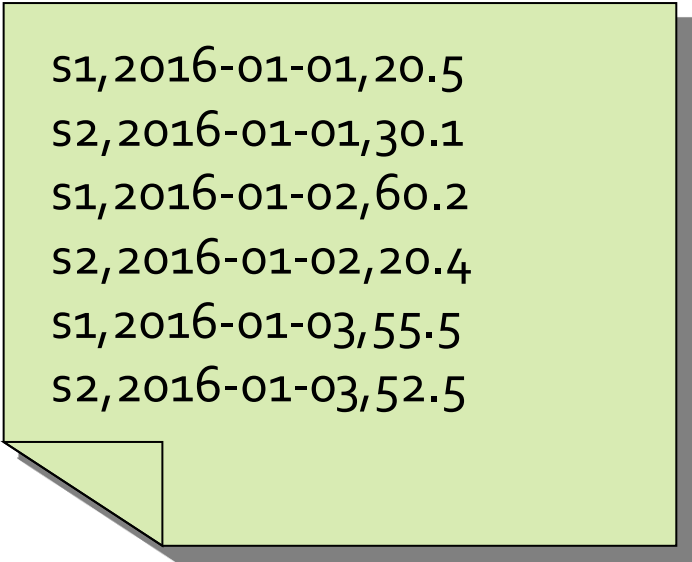
# Exercise #33

- Top-k maximum values
  - Input: a collection of (structured) textual csv files containing the daily value of PM<sub>10</sub> for a set of sensors
    - Each line of the files has the following format  
sensorId,date,PM10 value ( $\mu\text{g}/\text{m}^3$ )\n
  - Output: report the top-3 maximum values of PM<sub>10</sub>
    - Print the result on the standard output



# Exercise #33 - Example

- Input file



```
s1,2016-01-01,20.5  
s2,2016-01-01,30.1  
s1,2016-01-02,60.2  
s2,2016-01-02,20.4  
s1,2016-01-03,55.5  
s2,2016-01-03,52.5
```

- Output

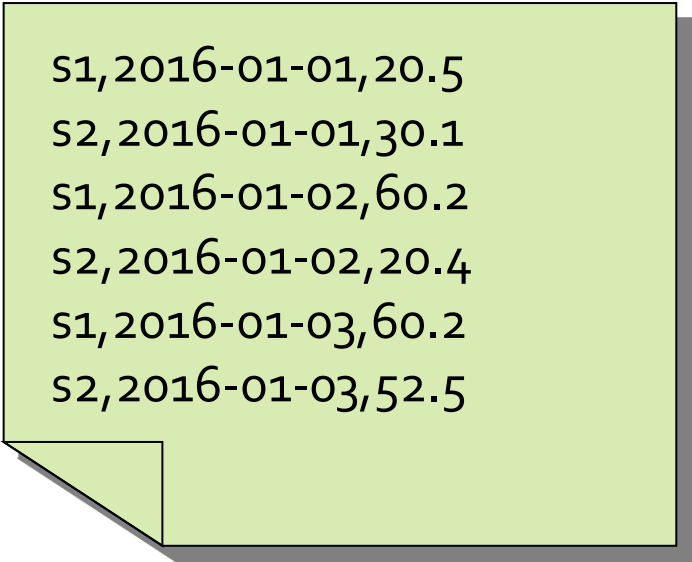
```
60.2  
55.5  
52.5
```

# Exercise #34

- Readings associated with the maximum value
  - Input: a collection of (structured) textual csv files containing the daily value of PM<sub>10</sub> for a set of sensors
    - Each line of the files has the following format  
sensorId,date,PM<sub>10</sub> value (µg/m<sup>3</sup>)\n
  - Output: the line(s) associated with the maximum value of PM<sub>10</sub>
    - Store the result in an HDFS folder

# Exercise #34 - Example

- Input file



```
s1,2016-01-01,20.5  
s2,2016-01-01,30.1  
s1,2016-01-02,60.2  
s2,2016-01-02,20.4  
s1,2016-01-03,60.2  
s2,2016-01-03,52.5
```

- Output

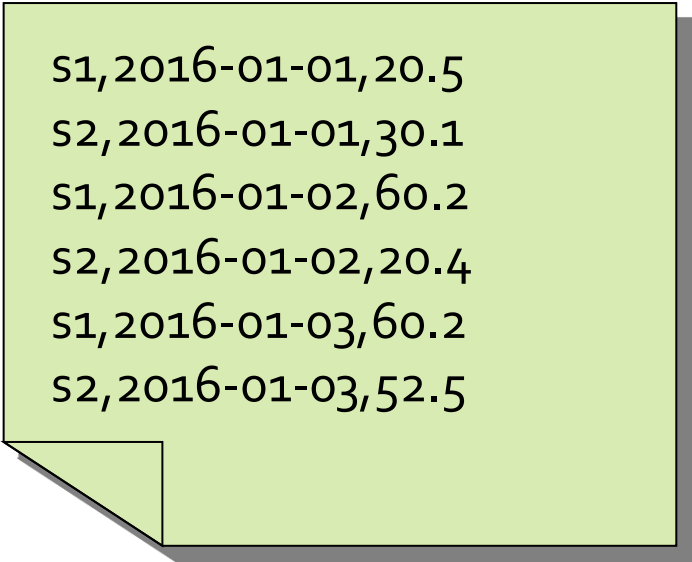
```
s1,2016-01-02,60.2  
s1,2016-01-03,60.2
```

# Exercise #35

- Dates associated with the maximum value
  - Input: a collection of (structured) textual csv files containing the daily value of PM<sub>10</sub> for a set of sensors
    - Each line of the files has the following format  
sensorId,date,PM<sub>10</sub> value (μg/m<sup>3</sup>)\n
  - Output: the date(s) associated with the maximum value of PM<sub>10</sub>
    - Store the result in an HDFS folder

# Exercise #35 - Example

- Input file



```
s1,2016-01-01,20.5  
s2,2016-01-01,30.1  
s1,2016-01-02,60.2  
s2,2016-01-02,20.4  
s1,2016-01-03,60.2  
s2,2016-01-03,52.5
```

- Output

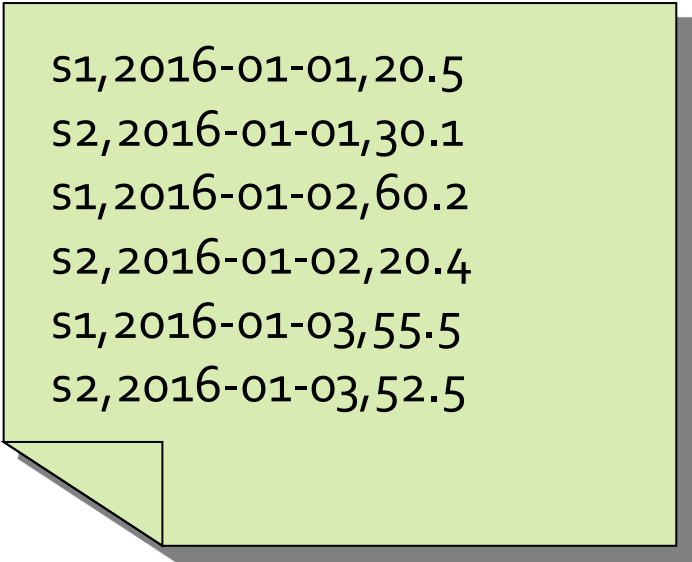
```
2016-01-02  
2016-01-03
```

# Exercise #36

- Average value
  - Input: a collection of (structured) textual csv files containing the daily value of PM<sub>10</sub> for a set of sensors
    - Each line of the files has the following format  
sensorId,date,PM<sub>10</sub> value (μg/m<sup>3</sup>)\n
  - Output: compute the average PM<sub>10</sub> value
    - Print the result on the standard output

# Exercise #36 - Example

- Input file



```
s1,2016-01-01,20.5  
s2,2016-01-01,30.1  
s1,2016-01-02,60.2  
s2,2016-01-02,20.4  
s1,2016-01-03,55.5  
s2,2016-01-03,52.5
```

- Output

39.86

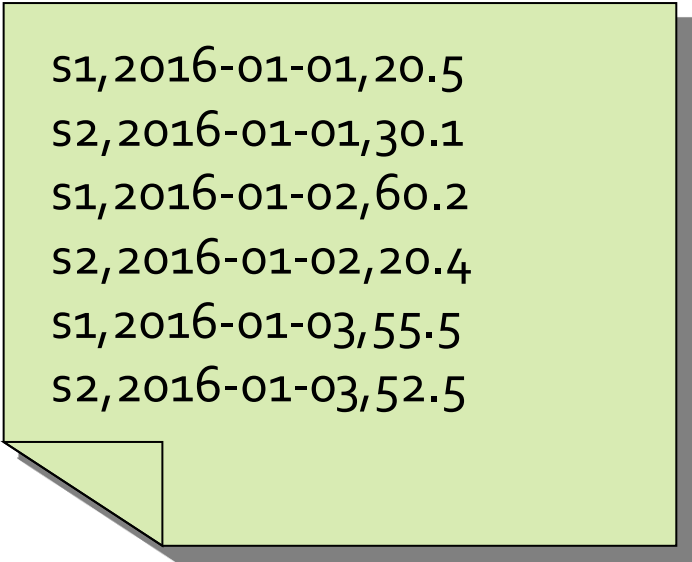
# Exercise #37

- Maximum values
  - Input: a textual csv file containing the daily value of PM<sub>10</sub> for a set of sensors
    - Each line of the files has the following format  
sensorId,date,PM<sub>10</sub> value (μg/m<sup>3</sup>)\n
  - Output: the maximum value of PM<sub>10</sub> for each sensor
    - Store the result in an HDFS file



# Exercise #37 - Example

- Input file



```
s1,2016-01-01,20.5  
s2,2016-01-01,30.1  
s1,2016-01-02,60.2  
s2,2016-01-02,20.4  
s1,2016-01-03,55.5  
s2,2016-01-03,52.5
```

- Output

```
(s1,60.2)  
(s2,52.5)
```