In [ ]:

```
import findspark
findspark.init()
import pyspark
sc = pyspark.SparkContext(appName="SMA")
import datetime
```

In [2]:

```
def convertstringtodate(stringdate) :
    date_time_obj = datetime.datetime.strptime(stringdate, '%Y-%m-%d')
    date_obj = date_time_obj.date()
    return date_obj
```

In [4]:

```
rdd = sc.parallelize([(17.00, "2018-03-10T15:27:18+00:00"), # The first six days are se
quential
  (13.00, "2018-03-11T12:27:18+00:00"),  # included ...
  (25.00, "2018-03-12T11:27:18+00:00"),  # included ...
  (20.00, "2018-03-13T15:27:18+00:00"),  # included ...
  (56.00, "2018-03-14T12:27:18+00:00"),  # included...
  (99.00, "2018-03-15T11:27:18+00:00"),  # This one will be included with the next wind
ow
  (156.00, "2018-03-22T11:27:18+00:00"), # This one is inside the 7 day window of the p
revious
  (122.00, "2018-03-31T11:27:18+00:00"), # This one is a new window, outside the 7 day
 window of any previous...
  (7000.00, "2018-04-15T11:27:18+00:00"),# This starts a * brand new window * with the
 next entry included next
  (9999.00, "2018-04-16T11:27:18+00:00")])
rdd = rdd.map(lambda x : (x[0],x[1].split("T")[0]))
rdd = rdd.map(lambda x: (convertstringtodate(x[1]), x[0]))
rdd.collect()
```

Out[4]:

```
[(datetime.date(2018, 3, 10), 17.0),
 (datetime.date(2018, 3, 11), 13.0),
 (datetime.date(2018, 3, 12), 25.0),
 (datetime.date(2018, 3, 13), 20.0),
 (datetime.date(2018, 3, 14), 56.0),
 (datetime.date(2018, 3, 15), 99.0),
 (datetime.date(2018, 3, 22), 156.0),
 (datetime.date(2018, 3, 31), 122.0),
 (datetime.date(2018, 4, 15), 7000.0),
 (datetime.date(2018, 4, 16), 9999.0)]
```

In [8]:

```
#dates example
rdd2 = rdd.flatMap(lambda x : [(x[0] + datetime.timedelta(days=i),x[1]) for i in range(
8) ])
rdd2 = rdd2.groupByKey().map(lambda x : (x[0], list(x[1])))
rdd3 = rdd2.join(rdd)
rdd3 = rdd3.map(lambda x : (x[0], x[1][0]))
rdd3 = rdd3.map(lambda x : (x[0], sum(x[1])/len(x[1])))
rdd3 = rdd3.sortBy(lambda x : x[0])
rdd3.collect()
```

Out[8]:

```
[(datetime.date(2018, 3, 10), 17.0),
 (datetime.date(2018, 3, 11), 15.0),
 (datetime.date(2018, 3, 12), 18.333333333333332),
 (datetime.date(2018, 3, 13), 18.75),
 (datetime.date(2018, 3, 14), 26.2),
 (datetime.date(2018, 3, 15), 38.333333333333336),
 (datetime.date(2018, 3, 22), 127.5),
 (datetime.date(2018, 3, 31), 122.0),
 (datetime.date(2018, 4, 15), 7000.0),
 (datetime.date(2018, 4, 16), 8499.5)]
```

SIMPLE EXAMPLE WITHOUT DATES

In [33]:

```
#simple example without dates
rdd = sc.parallelize([(1,0.5),(2,0.8),(3,4),(4,2),(5,6),(6,3)])
n=3
rdd2 = rdd.flatMap(lambda x : [(i + x[0],x[1]) for i in range(0,4) ])
rdd2 = rdd2.groupByKey().map(lambda x : (x[0], list(x[1])))
#value_when_true if condition else value_when_false
rdd2 = rdd2.map(lambda x : (x[0], x[1] if len(x[1])<=3 else x[1][1:]))
rdd2 = rdd2.join(rdd)
rdd2 = rdd2.map(lambda x : (x[0],x[1][0]))
rdd2 = rdd2.map(lambda x : (x[0], sum(x[1])/len(x[1])))
rdd2.collect()
```

Out[33]:

```
[(1, 0.5),
 (2, 0.65),
 (3, 1.7666666666666666),
 (4, 2.2666666666666666),
 (5, 4.0),
 (6, 3.6666666666666665)]
```

In [91]:

```
rdd = sc.parallelize([(1,0.5),(2,0.8),(3,4),(4,2),(5,6),(6,3)])
```

In [95]:

```
rdd2 = rdd.flatMap(lambda x : [(i + x[0],x[1]) for i in range(0,3) ])
rdd2 = rdd2.groupByKey().map(lambda x : (x[0], list(x[1])))
rdd2 = rdd2.join(rdd)
rdd2.collect()
```

Out[95]:

```
[(1, ([0.5], 0.5)),
 (2, ([0.5, 0.8], 0.8)),
 (3, ([0.5, 0.8, 4], 4)),
 (4, ([0.8, 4, 2], 2)),
 (5, ([4, 2, 6], 6)),
 (6, ([2, 6, 3], 3))]
```