1.15 通过某个字段将记录分组¶

问题¶

你有一个字典或者实例的序列，然后你想根据某个特定的字段比如 `date` 来分组迭代访问。

解决方案¶

`itertools.groupby()` 函数对于这样的数据分组操作非常实用。 为了演示，假设你已经有了下列的字典列表：

```
rows = [
    {'address': '5412 N CLARK', 'date': '07/01/2012'},
    {'address': '5148 N CLARK', 'date': '07/04/2012'},
    {'address': '5800 E 58TH', 'date': '07/02/2012'},
    {'address': '2122 N CLARK', 'date': '07/03/2012'},
    {'address': '5645 N RAVENSWOOD', 'date': '07/02/2012'},
    {'address': '1060 W ADDISON', 'date': '07/02/2012'},
    {'address': '4801 N BROADWAY', 'date': '07/01/2012'},
    {'address': '1039 W GRANVILLE', 'date': '07/04/2012'},
]
```

现在假设你想在按 date 分组后的数据块上进行迭代。为了这样做，你首先需要按照指定的字段(这里就是 `date`)排序，然后调用 `itertools.groupby()` 函数：

```
from operator import itemgetter
from itertools import groupby

# Sort by the desired field first
rows.sort(key=itemgetter('date'))
# Iterate in groups
for date, items in groupby(rows, key=itemgetter('date')):
    print(date)
    for i in items:
        print(' ', i)
```

运行结果：

```
07/01/2012
  {'date': '07/01/2012', 'address': '5412 N CLARK'}
  {'date': '07/01/2012', 'address': '4801 N BROADWAY'}
07/02/2012
  {'date': '07/02/2012', 'address': '5800 E 58TH'}
  {'date': '07/02/2012', 'address': '5645 N RAVENSWOOD'}
  {'date': '07/02/2012', 'address': '1060 W ADDISON'}
07/03/2012
  {'date': '07/03/2012', 'address': '2122 N CLARK'}
07/04/2012
  {'date': '07/04/2012', 'address': '5148 N CLARK'}
  {'date': '07/04/2012', 'address': '1039 W GRANVILLE'}
```

讨论¶

`groupby()` 函数扫描整个序列并且查找连续相同值（或者根据指定 key 函数返回值相同）的元素序列。 在每次迭代的时候，它会返回一个值和一个迭代器对象， 这个迭代器对象可以生成元素值全部等于上面那个值的组中所有对象。

一个非常重要的准备步骤是要根据指定的字段将数据排序。 因为 `groupby()` 仅仅检查连续的元素，如果事先并没有排序完成的话，分组函数将得不到想要的结果。

如果你仅仅只是想根据 `date` 字段将数据分组到一个大的数据结构中去，并且允许随机访问， 那么你最好使用 `defaultdict()` 来构建一个多值字典，关于多值字典已经在 1.6 小节有过详细的介绍。比如：

```
from collections import defaultdict
rows_by_date = defaultdict(list)
for row in rows:
    rows_by_date[row['date']].append(row)
```

这样的话你可以很轻松的就能对每个指定日期访问对应的记录：

```
>>> for r in rows_by_date['07/01/2012']:
... print(r)
...
{'date': '07/01/2012', 'address': '5412 N CLARK'}
{'date': '07/01/2012', 'address': '4801 N BROADWAY'}
>>>
```

在上面这个例子中，我们没有必要先将记录排序。因此，如果对内存占用不是很关心， 这种方式会比先排序然后再通过 `groupby()` 函数迭代的方式运行得快一些。