Atitit data algo group by 数据算法 分组

## 分组

首先，我们利用（lambda表达式出现之前的）命令式风格Java 程序对流中的雇员按城市进行分组：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | Map<String, List<Employee>> result = **new** HashMap<>();  **for** (Employee e : employees) {    String city = e.getCity();    List<Employee> empsInCity = result.get(city);  **if** (empsInCity == **null**) {      empsInCity = **new** ArrayList<>();      result.put(city, empsInCity);    }    emps |
| Map<String, List<Employee>> employeesByCity =    employees.stream().collect(groupingBy(Employee::getCity)); | |

结果如下面的 map 所示：

|  |  |
| --- | --- |
| 1 | {New York=[Charles], Hong Kong=[Dorothy], London=[Alice, Bob]} |

还可以计算每个城市中雇员的数量，只需传递一个计数收集器给 groupingBy 收集器。第二个收集器的作用是在流分类的同一个组中对每个元素进行递归操作。

|  |  |
| --- | --- |
| 1  2 | Map<String, Long> numEmployeesByCity =    employees.stream().collect(groupingBy(Employee::getCity, counting())); |

结果如下面的 map 所示：

|  |  |
| --- | --- |
| 1 | {New York=1, Hong Kong=1, London=2} |

顺便提一下，该功能与下面的 SQL 语句是等同的：

|  |  |
| --- | --- |
| 1 | select city, count(\*) from Employee group by city |

另一个例子是计算每个城市的平均年龄，这可以联合使用 averagingInt 和 groupingBy 收集器：

|  |  |
| --- | --- |
| 1  2  3 | Map<String, Double> avgSalesByCity =    employees.stream().collect(groupingBy(Employee::getCity,                                 averagingInt(Employee::getNumSales))); |

结果如下 map 所示：

|  |  |
| --- | --- |
| 1 | {New York=160.0, Hong Kong=190.0, London=175.0} |

## 分区

分区是一种特殊的分组，结果 map 至少包含两个不同的分组——一个true，一个false。例如，如果想找出最优秀的员工，你可以将所有雇员分为两组，一组销售量大于 N，另一组小于 N，使用 partitioningBy 收集器：

|  |  |
| --- | --- |
| 1  2 | Map<Boolean, List<Employee>> partitioned =    employees.stream().collect(partitioningBy(e -> e.getNumSales() > 150)); |

输出如下结果：

|  |  |
| --- | --- |
| 1 | {**false**=[Bob], **true**=[Alice, Charles, Dorothy]} |

你也可以将 groupingBy 收集器传递给 partitioningBy 收集器来将联合使用分区和分组。例如，你可以统计每个分区中的每个城市的雇员人数：

|  |  |
| --- | --- |
| 1  2  3 | Map<Boolean, Map<String, Long>> result =    employees.stream().collect(partitioningBy(e -> e.getNumSales() > 150,                                 groupingBy(Employee::getCity, counting()))); |

这样会生成一个二级 Map:

|  |  |
| --- | --- |
| 1 | {**false**={London=1}, **true**={New York=1, Hong Kong=1, London=1}} |

原文链