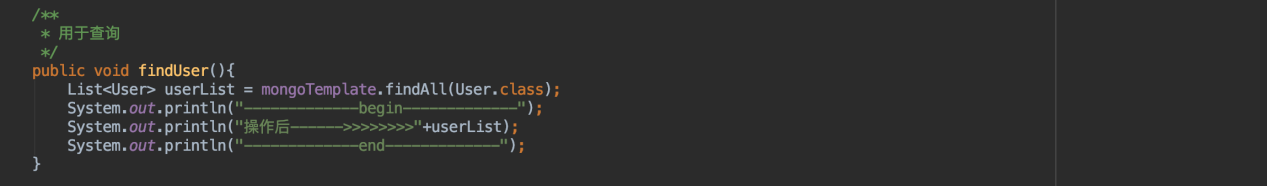
一：【mongdb】基本操作

基础数据



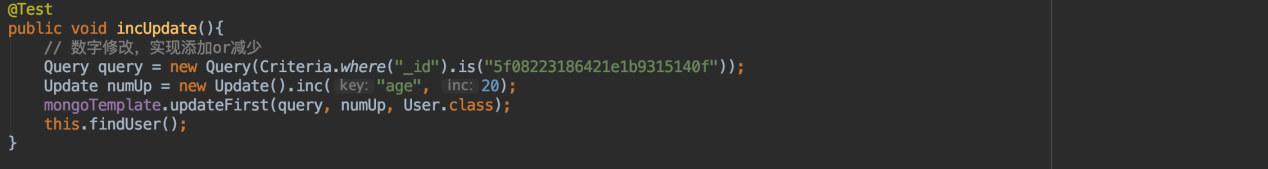
定一个用于查询的方法



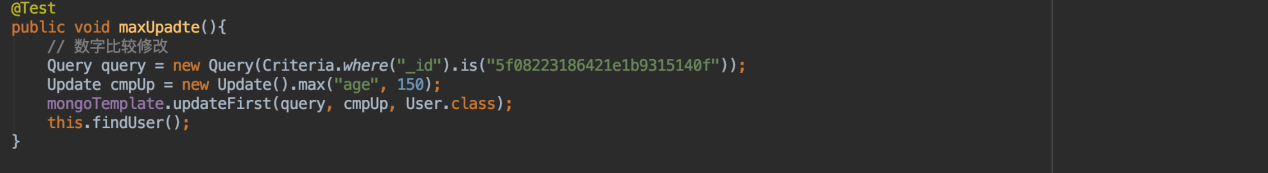
1. 基本类型修改：mongodb支持我们常见的各种基本类型，而MongoTemplate也封装了不少对应的修改方法，最基础的修改，主要是借助Update来实现。
2. 基本使用修改



b.数字增加/减少



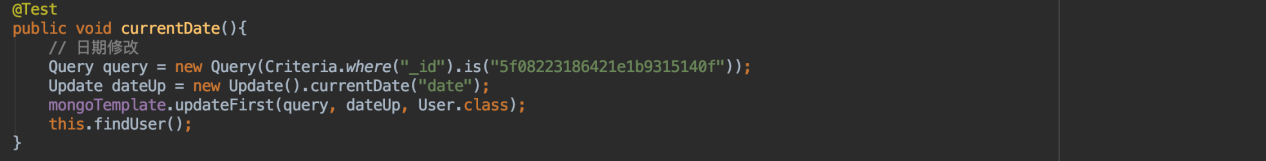
c.数字比较修改



d.乘法



e.日期修改



1. 数组操作：在MongoDB的document中，有两个有意思的类型，一个是数组，一个是document（即可以嵌套），这里则主要介绍下如何操作数组中的成员
2. 添加到数组中：

在数组中新增一个数据，提供了两种方式

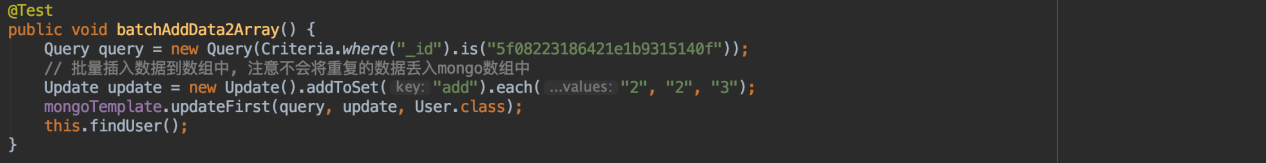
一个是addToSet(java.lang.String, java.lang.Object)

一个是push(java.lang.String, java.lang.Object)

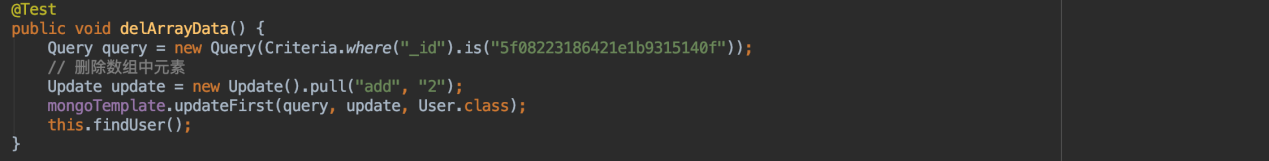
两个的区别在于前者不能插入重复数据，后者可以



b.批量添加：一次添加多个，借助addToSet的each来实现



c.删除：借助pull来精确删除某个值



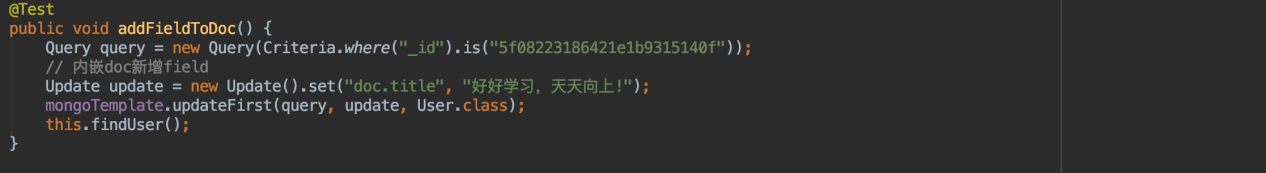
d. 修改：修改，首先的问题是要定位，确定删除数组中某个下标的元素，这里借助了一个有意思的站位。



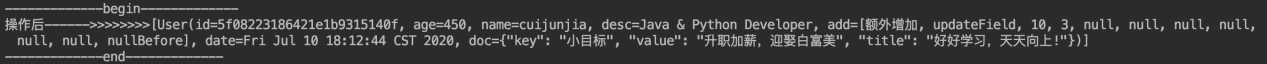


1. document操作：内嵌文档，可以所是MongoDB的一个特色了，我们则来看下如何进行操作。
2. 添加：借助前面的站位思想，就很好实现了，定位元素的方式采用





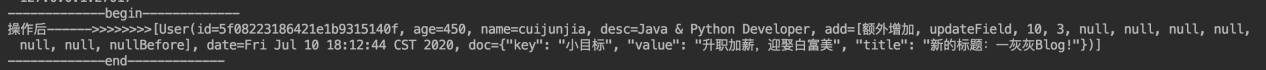
执行结果



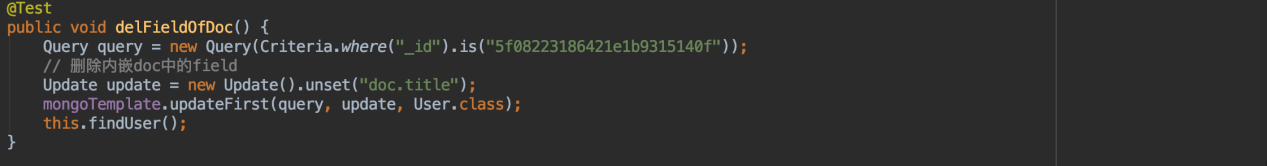
b.修改



执行结果



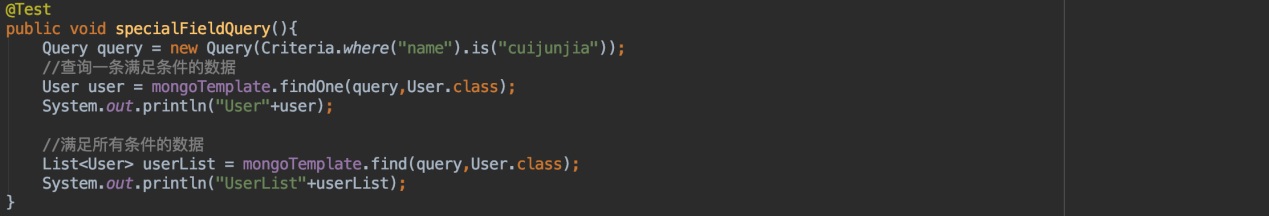
c.删除



执行结果

截屏2020-07-10 下午6.48.39

1. 根据字段进行查询：最常见的查询场景，比如我们根据查询user=cuijunjia的数据，这里主要会使用Query + Criteria 来完成。



Criteria.where(xxx).is(xxx)来指定具体的查询条件

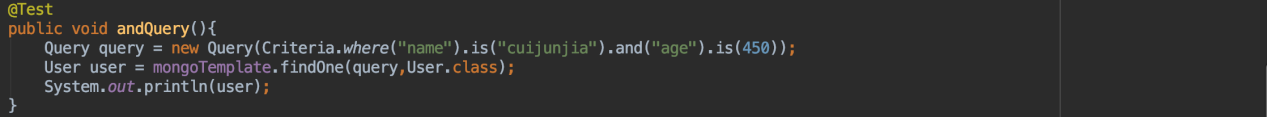
封装Query对象 new Query(criteria)

借助mongoTemplate执行查询 mongoTemplate.findOne(query, collectionName)

其中findOne表示只获取一条满足条件的数据；find则会将所有满足条件的返回。

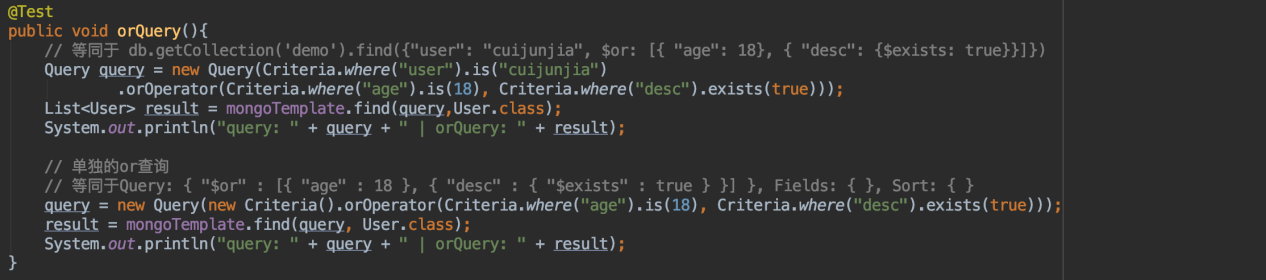
5. and多条件查询：

前面是只有一个条件满足，现在如果是要求同时满足多个条件，则利用org.springframework.data.mongodb.core.query.Criteria#and来斜街多个查询条件。

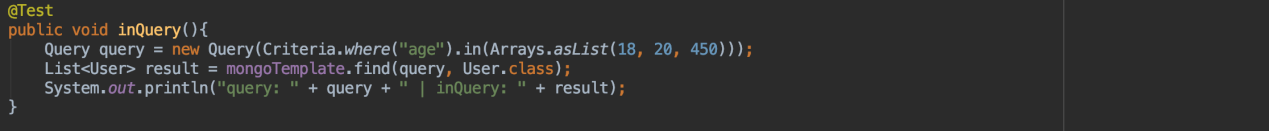


6.or或查询

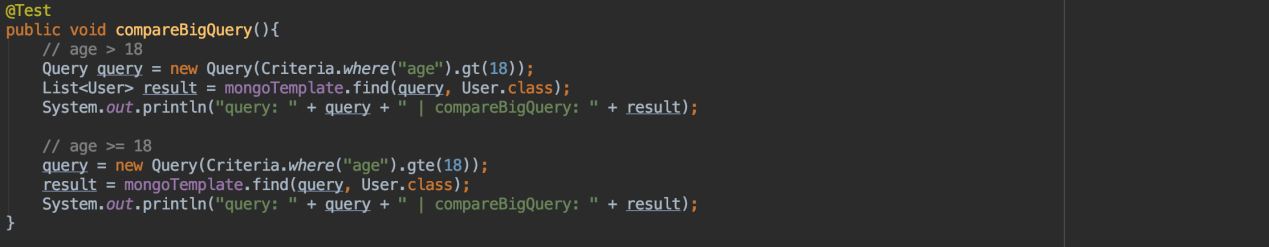
and对应的就是or，多个条件中只要一个满足即可，这个与and的使用有些区别, 借助org.springframework.data.mongodb.core.query.Criteria#orOperator来实现，传参为多个Criteria对象，其中每一个表示一种查询条件。

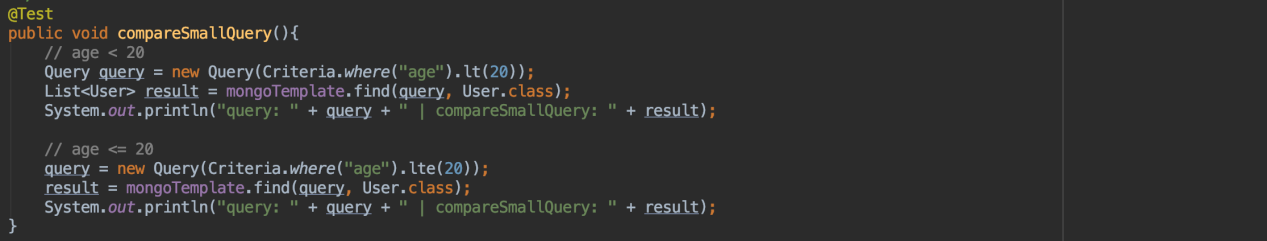


7.in查询：标准的in查询case

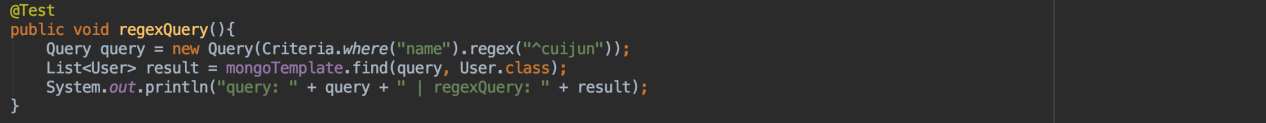


8.数值比较：数值的比较大小，主要使用的是 get, gt, lt, let





1. 正则查询



10.查询总数：统计常用，这个主要利用的是mongoTemplate.count方法

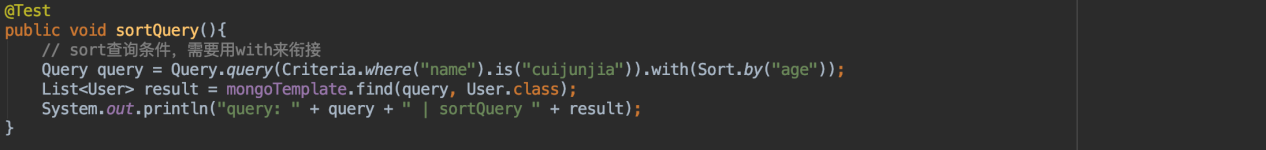


11.分组查询：

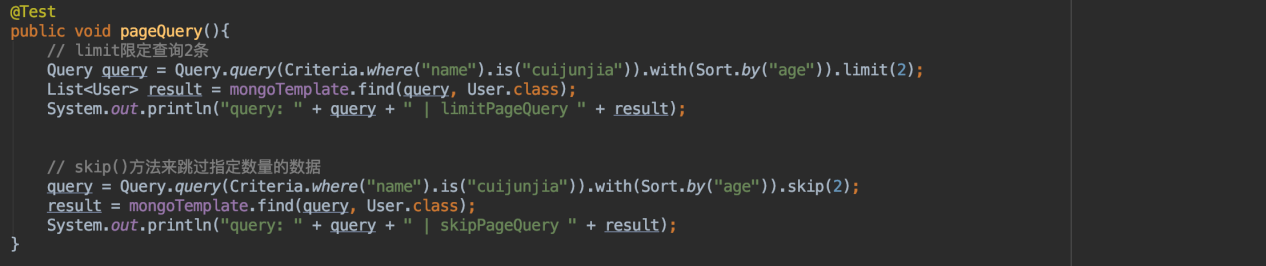
这个对应的是mysql中的group查询，但是在mongodb中，更多的是通过聚合查询，可以完成很多类似的操作，借助聚合。



1. 排序



1. 分页



二：【mongdb】聚合管道

1.阶段操作符（Stage Operators）

阶段操作符是使用于db.collection.aggregate方法里面，数组参数中的第一层。

db.collection.aggregate( [ { 阶段操作符：表述 }, { 阶段操作符：表述 }, ... ] )

1. 表达式操作符（Expression Operators）

表达式操作符主要用于在管道中构建表达式时使用，使用类似于函数那样需要参数，主要用于$project操作符中，用于构建表达式，使用方法一般如下：

方法1:

{ <operator>: [ <argument1>, <argument2> ... ] }

方法2:

{ <operator>: <argument> }

1. 累加器（Accumulators）

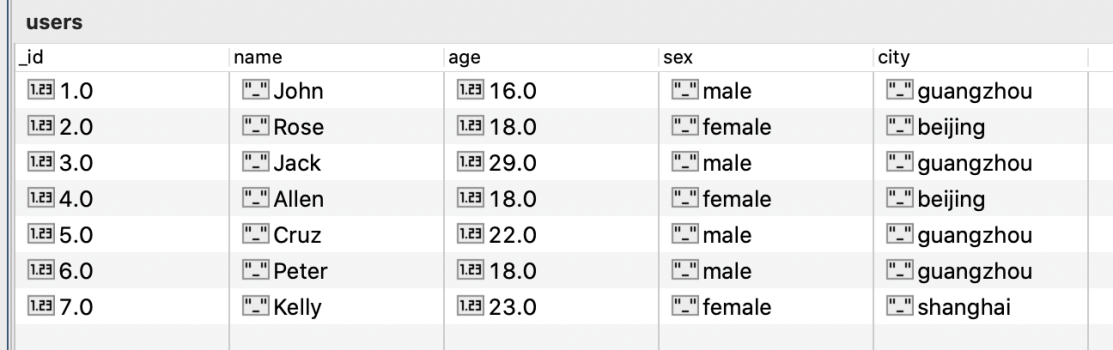
累加器本来只能使用与groud下，但是版本3.2或以上，部分累加器还能使用于project。当在group中使用时，累加器是针对每个分组使用的；当在project中使用时，累加器则是针对每个字面量起作用。

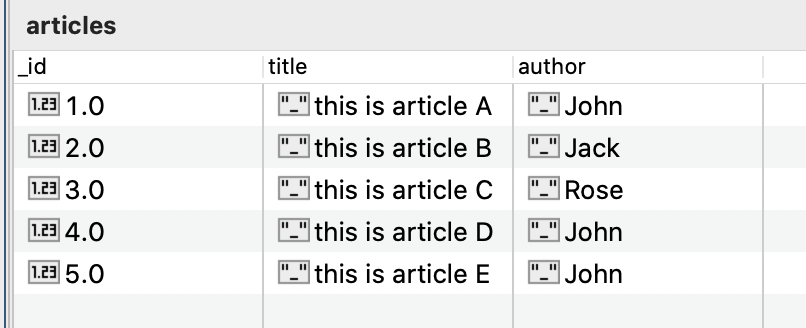
常用阶段操作符



1. 阶段操作符详解

假设有一个保存用户的集合Users，一个文章的集合Articles。





1. $match 匹配操作符：用于对文档集合进行筛选

语法：{ $match: { <query> } }

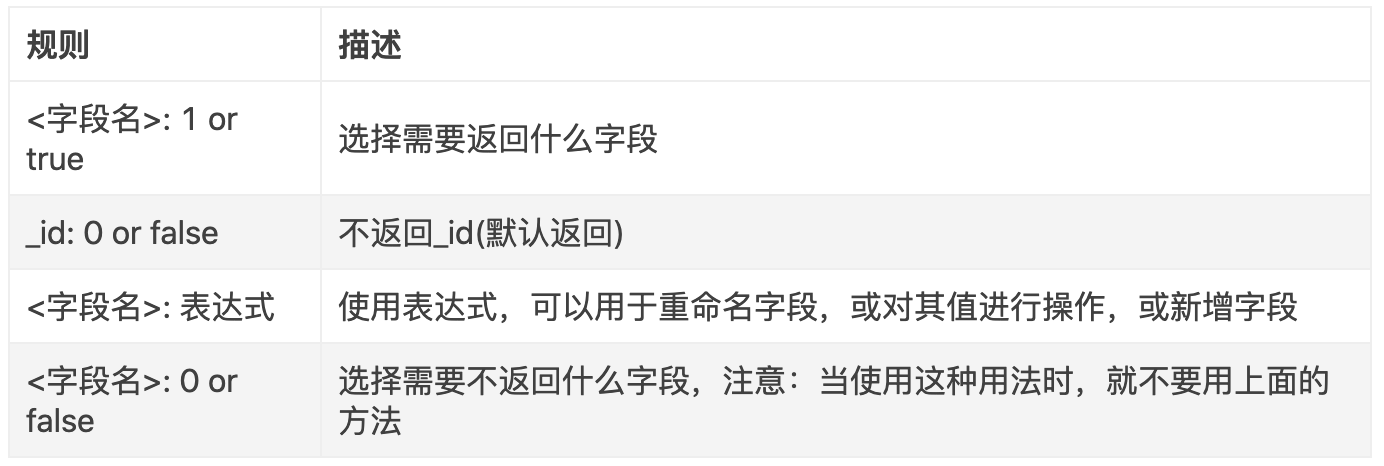
示例：查询用户年龄是18岁的用户

截屏2020-07-14 下午4.04.33

b.$project 投射操作符：用于重构每一个文档的字段，可以提取字段，重命名字段，甚至可以对原有字段进行操作后新增字段.

语法：{ $project: { <specification(s)> } }

specification的规则如下：



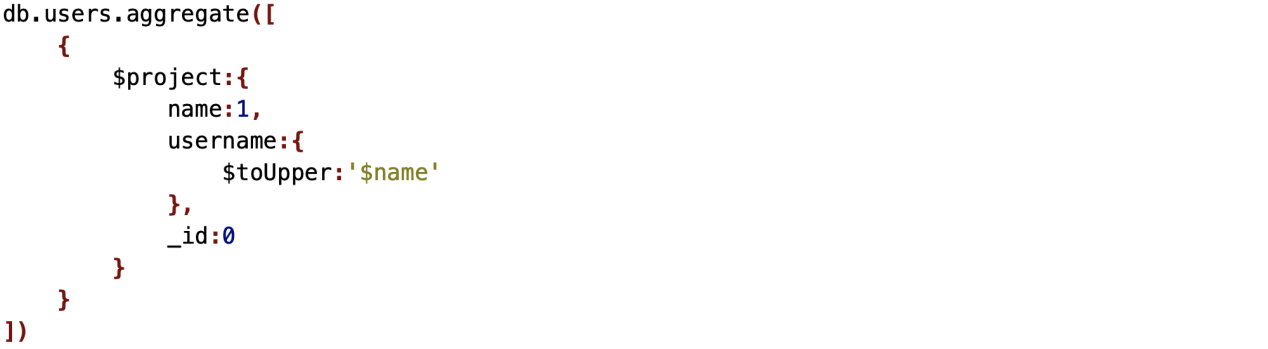
示例1:用户集体投射用户姓名

截屏2020-07-14 下午4.08.59

示例2:将返回的\_id重命名为userId

截屏2020-07-14 下午4.14.06

示例3:返回新字段username,并使用表达式让它的值为name的大写



c.$sort 排序操作符：用于根据一个或多个字段对文档进行排序

语法：{ $sort: { <field1>: <sort order>, <field2>: <sort order> ... } }

示例：users集合按照年龄age从低到高排序



d.$limit 限制操作符：用于限制返回文档的数量

语法：{ $limit: <positive integer> }

示例：返回5篇article



e.$skip 跳过操作符：用于跳过指定数量的文档

语法：{ $skip: <positive integer> }

示例：跳过1个文档



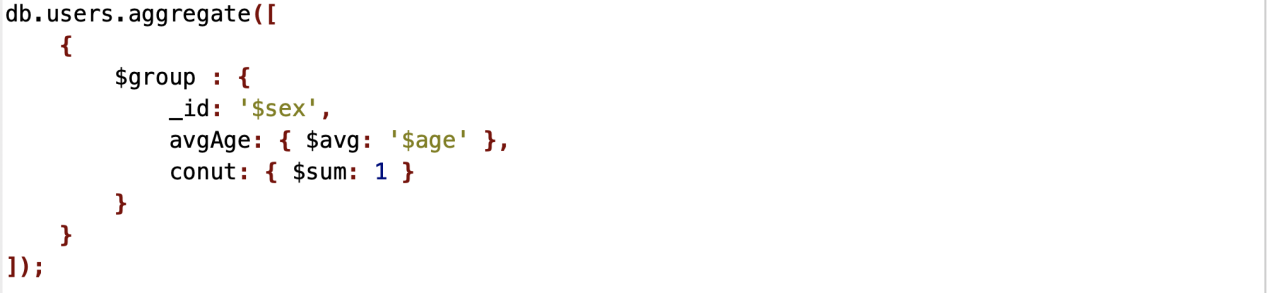
f.$group 分组操作符：用于对文档集合进行分组

语法：{ $group: { \_id: <expression>, <field1>: { <accumulator1> : <expression1> }, ... } } \_id是必须的，用作分组的依据条件

示例1：将用户(users)按性别（sex）分组

截屏2020-07-15 上午10.36.35

示例2：将用户(users)按性别（sex）分组，分组后使用计算各自性别的平均年龄，统计不同的性别的人数，并以count返回



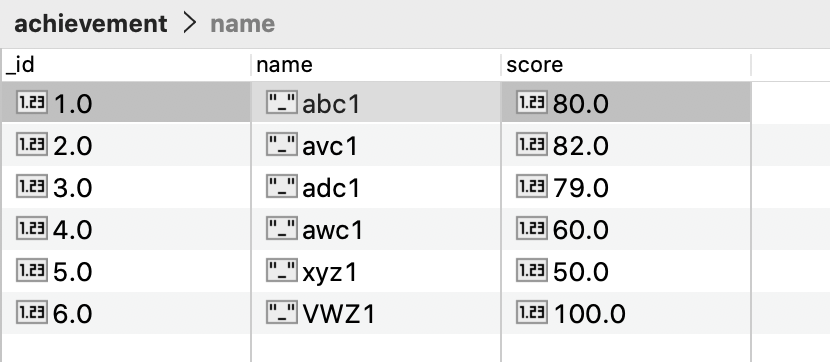
1. 表达式操作符（Expression Operators）：表达式操作符主要用于在管道中构建表达式时使用，使用类似于函数那样需要参数，主要用于$project操作符中，用于构建表达式
2. 布尔值操作符（Boolean Operators）



b.比较操作符（Comparison Operators）



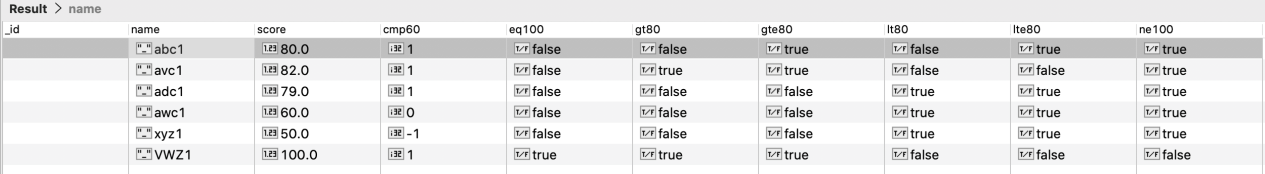
基本表结构



综合示例操作如下



返回结果



数学操作符（Arithmetic Operators）



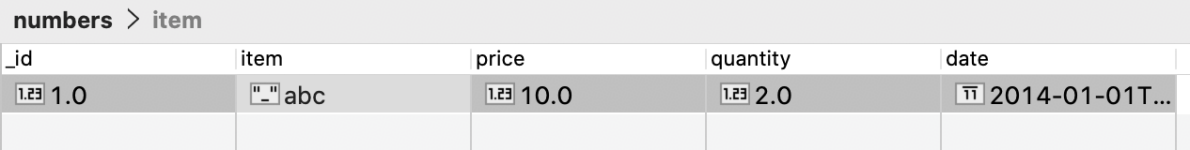
字符串操作符（String Operators）



日期操作符（Date Operators）



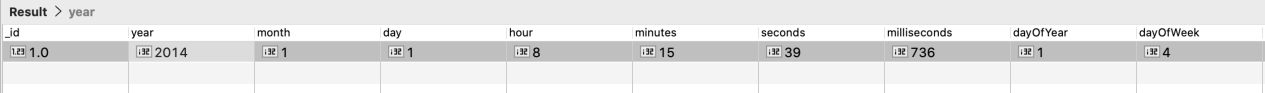
基本表结构



综合示例



结果如下

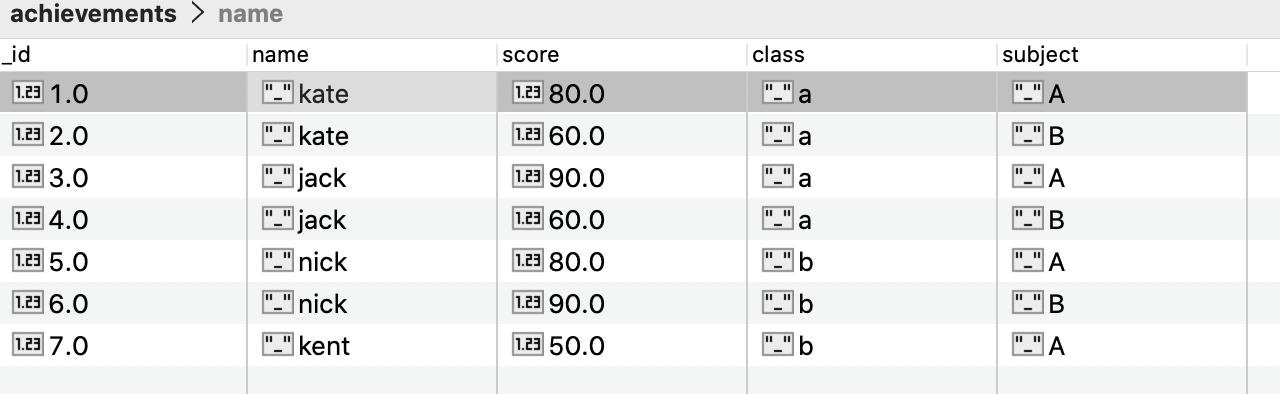


1. 累加器（Accumulators）：累加器本来只能使用与groud下，但是版本3.2或以上，部分累加器还能使用于project。当在group中使用时，累加器是针对每个分组使用的；当在project中使用时，累加器则是针对每个字面量起作用。

常用累加器

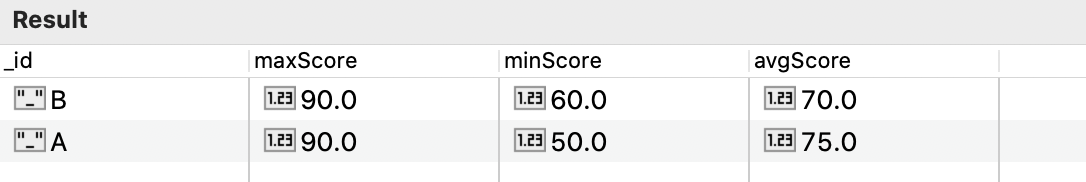


基本表数据

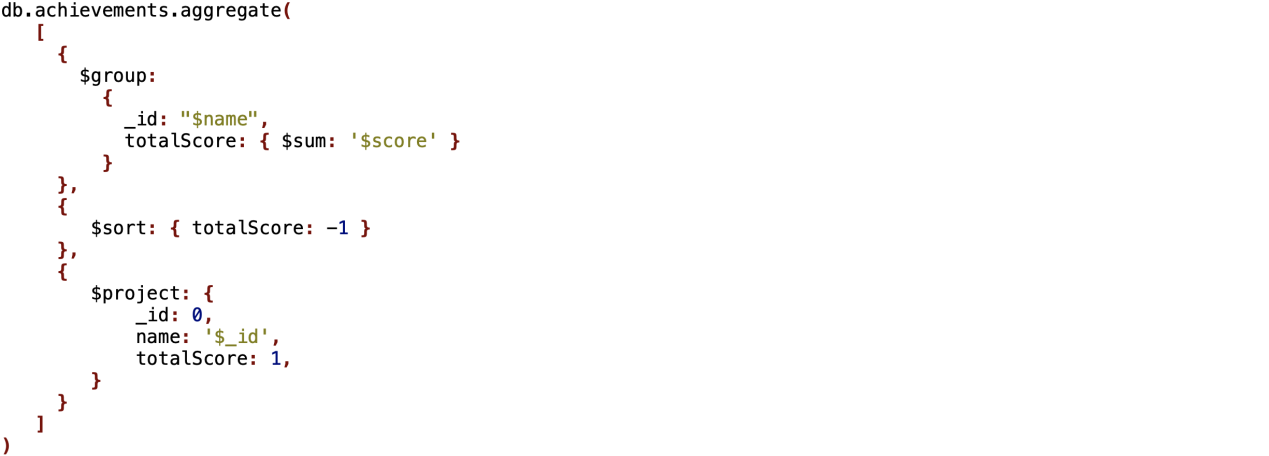


示例1:获取各科目的最高分，获取各科目的最低分，获取各科目的平均分

 结果集



示例2:获取每人的总分，从高到低排序



结果集

