6、shell命令操作2聚合查询

1、mongo-》聚合查询

group工具：出现比较早，现在使用不多，官方不推荐。

aggregate工具：比较强大，基于管道（每一个操作都是基于上一个操作--类似于【工厂流水线】），提供各种操作（分组，过滤，筛选，排序，分页，拆分数组。。。），每一个操作可以按顺序多次执行。

mapreduce工具：分布式计算的模型，类似于haddop的mapreduce，提供map和reduce函数，
如果处理的数据量特别大，可以使用mapreduce。

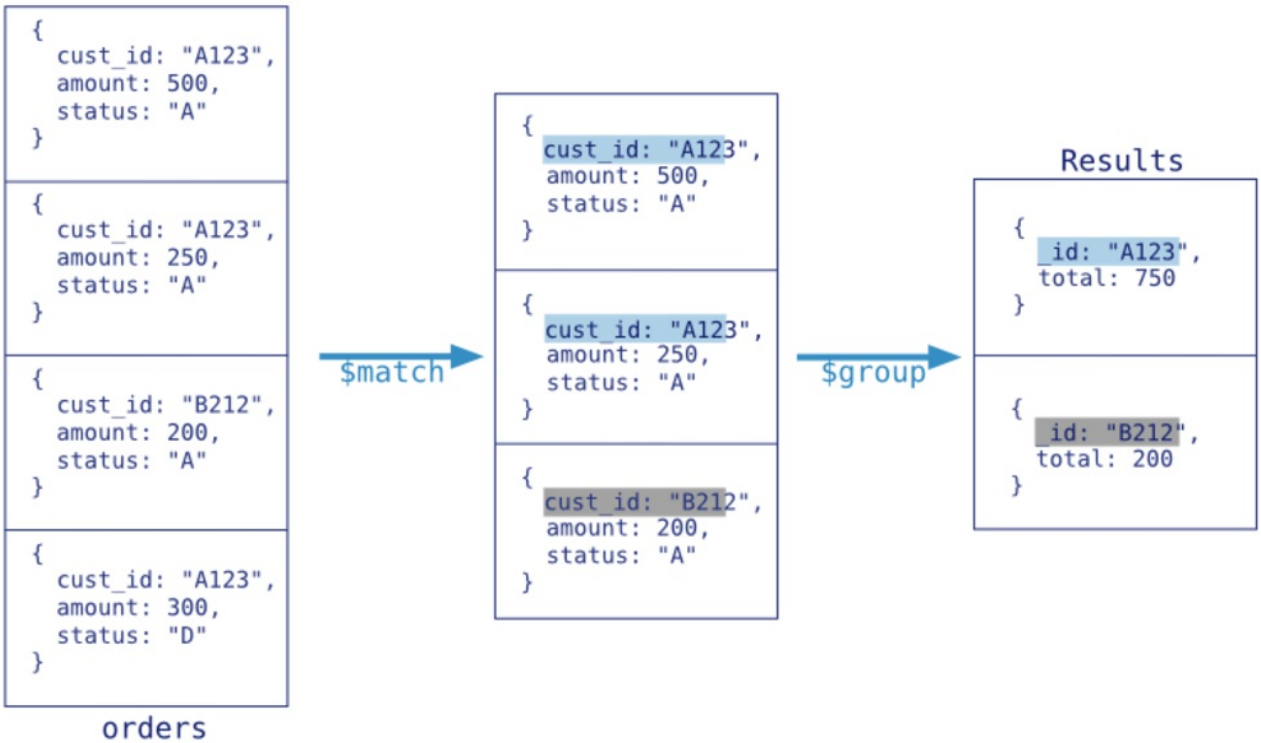
2、aggregate工具

主要用于把集合的数据进行复杂计算数据，类似sql中的group，having，sort，limit，sum()、avg()，count()等。。
包含了，前面的find()，count()，limit()，sort()。。。功能，非常强大！！！

语法：db.集合名称.aggregate([
{操作1:{表达式}},
{操作2:{表达式}},
{操作3:{表达式}},
...
])

管道：管道可以对某些输入执行操作，并将输出用作下一个命令的输入。 类似于工厂的流水线。
生产手机的过程（来一个手机壳，来一个主板，来一个电池，来一个摄像头，来一个屏幕）
ps：（注意每一个操作的顺序，顺序不同，结果可能不同）

管道操作：对【集合数据的处理】
$group：将集合中的文档分组，可用于统计结果--》类似于sql的group
$match：过滤数据，只输出符合条件的文档--》类似于sql的where和having
$project：修改输入文档的结构，如重命名、增加、删除字段、创建计算结果--》类似于sql的select
$sort：将输入文档排序后输出
$limit：限制聚合管道返回的文档数
$skip：跳过指定数量的文档，并返回余下的文档
$unwind：将数组类型的字段进行拆分
$out ：将查询的数据写入集合中。要使用$out，它必须是流水线的最后一个阶段。
$lookup：（mongoDb3.2才支持）用于连接其他集合进行查询，类似于sql的left join。



聚合-表达式（类似于sql聚合函数）：对【group分组后的-【组数据的处理】】

$sum：计算总和，$sum:1同count表示计数
$avg：计算平均值
$min：获取最小值
$max：获取最大值
$push：在结果文档中插入值到一个数组中
$first：根据资源文档的排序获取第一个文档数据

$last：根据资源文档的排序获取最后一个文档数据

----------**帖子集合数据**----------

```
{
  _id: 102,
  title: 'mongoDb',
  description: 'mongoDb是文档数据库',
  by_user: '曾小贤',
  tags: ['mongoDb', 'database', 'NoSQL']
},
{
  _id: 103,
  title: 'MySQL',
  description: 'MySQL是关系型数据库',
  by_user: '曾小贤',
  tags: ['MySQL', 'database', 'SQL']
},
{
  _id: 104,
  title: 'spring mvc',
  description: 'spring mvc 是对servlet进行封装的web层框架',
  by_user: '吕子乔',
  tags: ['spring','mvc','spring mvc','java']
},
{
  _id: 105,
  title: 'mybatis',
  description: 'mybatis是对jdbc封装用来操作关系型数据库的框架',
  by_user: '一菲姐',
  tags: ['java','mybatis','jdbc','sql']
},
{
  _id: 106,
  title: 'js',
  description: 'js是一个强大的脚本语言，开发前端，后台，数据库！',
  by_user: '一菲姐',
  tags: ['js','database','javascript']
},
{
  _id: 107,
  title: 'json',
  description: 'json是一种轻量级数据格式，经常用于网络交互，js的对象也是json格式',
  by_user: '一菲姐',
  tags: ['js','json','javascript','数据格式']
}
```

a）查询出所有的用户所发帖数量和发帖内容的列表

ps--》按by_user分组，并sum：1求出每一组（用户）的帖子数,push每一个元素到数组

```
db.article.aggregate([
    {
        $group : {
            _id : "$by_user",      //按用户分组
            art_count : {$sum : 1},    //求出该用户帖子数量
            art_title_arr:{$push:'$title'} //合并该用户所有的帖子数据到数组
        }
    }]);
```

b）查询出所有的用户所发帖数量和发帖内容的列表---》更改显示的列，找到发帖数>1的人的信息，并按帖子数升序排列

```
db.article.aggregate([
    {
        $group : {
            _id : "$by_user",      //按用户分组
            art_count : {$sum : 1},    //求出该用户帖子数量
            art_title_arr:{$push:'$title'} //合并该用户所有的帖子数据到数组
        }
    },{
        $project:{
            _id:0,
            user:'$_id',
            art_count :1,
            art_title_arr:1
```

```
        }
    },{
        $match:{
            art_count:{$gt:1}
        }
    },{
        $sort:{
            art_count:1
        }
    }
    ]);
```

ps->project查询嵌套文档或数组，同样支持，下标或属性名访问子元素

"属性名.下标"

"属性名."

c）查询出102和103帖子里所有的标记列表，并去重

```
db.article.aggregate([
    {
        $match:{
            _id:{
                $in:[102,103]
            }
        }
    },{
        $project:{
            _id:0,
            tags:1
        }
    },
    {
        $unwind:'$tags'
    },{
        $group:{
        _id:'$tags'
        }
    }]);
```