Lab2 Python连接并访问MongoDB和CRUD操作深入

1. 连接monngodb数据库

In [3]:
```
#导入库
import pymongo
```

连接mongodb
如果是在本地连接，则采用 client= pymongo.MongoClient("mongodb://localhost:27017/") 的方式连接。
不过，本次实验是在水杉云环境中进行，需要连接阿里云环境下的mongodb，请大家采用以下方式连接：
已经为大家创建好了MongoDB用户，**账号:ecnu学号；密码:ECNU学号；数据库: ecnu学号。**
将下面句子中的中文替换为自己线上mongodb的账号，密码和数据库名

`client = pymongo.MongoClient("mongodb://用户名:密码@172.16.208.86:27017/admin")`

填写到下面的cell中

In [4]:
```
#连接mongodb
client = pymongo.MongoClient("mongodb://ecnu10222140402:ECNU10222140402@172.16.2
print(client.list_database_names())
```

```
['ecnu10222140402']
```

每位用户只有操作本人数据库的权限。 需要切换到到自己的数据库，以“ecnuXXXX数据库”为例
db = client['ecnuXXXX']
或者 db = client.ecnuXXXX

In [5]:
```
#切换到自己的数据库
db = client.ecnu10222140402
```

In [6]:
```
# print 数据库名字，应是 “stu学号” 的格式
print(client.list_database_names())
```

```
['ecnu10222140402']
```

In [7]:
```
#获取集合，如果没有则自动创建，这里设定集合是users，在python中用users_col指代
users_col = db['users']
#或者
# users_col = db.users
```

In [8]:
```
#先将该集合中所有数据删除（删除操作讲解在最末尾），保证数据和Lab2中第2部分插入数据
#如有插入错误，可使用该语句重新插入
users_col.delete_many({})
```

Out[8]:
```
DeleteResult({'n': 1, 'ok': 1.0}, acknowledged=True)
```

2. 插入数据

- insert_one（参数）：插入一条数据，插入的参数就是Python的字典
- insert_many(参数)：批量插入数据，参数为多个字典的列表

插入一条用户数据

In [9]:
```
user1 = {
    "name" : "xiaobu",
    "country" : "China",
    "address" : {
        "aCode" : "001",
        "aName" : "北京"},
    "favorites" : {
        "books" : [
            "西游记",
            "红楼梦",
            "三国演义",
            "水浒传"],
        "cites" : [
            "韶关",
            "深圳",
            "佛山"]},
    "age" : 26,
    "height" : 1.70
}
result = users_col.insert_one(user1)
```

In [10]:
```
#为"name"字段创建唯一索引约束，保证同一name的文档只有一条能够插入。如果重复插入同
users_col.create_index([("name", 1)], unique=True)
```

Out[10]:
```
'name_1'
```

如果直接对插入的操作打印，返回的结果是pymongo.results.InsertOneResult格式

In [11]:
```
print(result)
```

```
InsertOneResult(ObjectId('66ed528c841c3120d498bc29'), acknowledged=True)
```

需要使用下面的语句才能成功输出数据表中的信息

In [12]:
```
#下面2会讲到的find语句，现在先看看是否插入成功
content = users_col.find()
for each in content:
    print(each)
```

```
{'_id': ObjectId('66ed528c841c3120d498bc29'), 'name': 'xiaobu', 'country': 'Chin
a', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游
记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age':
26, 'height': 1.7}
```

如果显示一条数据，即插入成功。

练习1：

将下面一位用户的信息插入users表中：

```
{
    "name" : "juyi",
    "country" : "China",
    "address" : {
        "aCode" : "002",
        "aName" : "广州"},
    "favorites" : {
        "movies" : [
            "肖生克的救赎",
            "阿甘正传",
            "头号玩家"],
        "cites" : [
            "衡阳",
            "南宁",
            "上海",
            "深圳"]},
    "age" : 25,
    "height" : 1.69
}
```

解答1：

In [13]:
```
user2 = {
    "name" : "juyi",
    "country" : "China",
    "address" : {
        "aCode" : "002",
        "aName" : "广州"},
    "favorites" : {
        "movies" : [
            "肖生克的救赎",
            "阿甘正传",
            "头号玩家"],
        "cites" : [
            "衡阳",
            "南宁",
            "上海",
            "深圳"]},
    "age" : 25,
    "height" : 1.69
}
result = users_col.insert_one(user2)
```

In [14]:
```
# 可使用该语句判断是否插入成功
content = users_col.find()
for each in content:
    print(each)
```

```
{'_id': ObjectId('66ed528c841c3120d498bc29'), 'name': 'xiaobu', 'country': 'Chin
a', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游
记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age':
26, 'height': 1.7}
{'_id': ObjectId('66ed528f841c3120d498bc2a'), 'name': 'juyi', 'country': 'China',
'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的
救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age':
25, 'height': 1.69}
```

插入多条数据:

In [15]:
```
more_users = [
{
    "name" : "lilei",
    "country" : "China",
    "address" : {
        "aCode" : "003",
        "aName" : "上海"},
    "favorites" : {
        "books" : [
            "谁动了我的奶酪",
            "CSAPP"],
        "cites" : [
            "上海",
            "杭州"]},
    "age" : 18,
    "height" : 1.88
},
{
    "name" : "zhangsan",
    "country" : "China",
    "address" : {
        "aCode" : "003",
        "aName" : "上海"},
    "favorites" : {
        "movies" : [
            "头号玩家",
            "肖生克的救赎"
            ],
        "cites" : [
            "旧金山",
            "上海"
            ]},
    "age" : 23,
    "height" : 1.72
},
{
    "name" : "tony",
    "country" : "USA",
    "address" : {
        "aCode" : "004",
        "aName" : "洛杉矶"},
    "favorites" : {
        "books" : [
            "权利的游戏",
            "飘",
            "谁动了我的奶酪"],
        "cites" : [
            "芝加哥",
```

```
            "洛杉矶"]},
    "age" : 28,
    "height" : 1.79
}
]
users_col.insert_many(more_users)
content = users_col.find()
for each in content:
    print(each)
```

```
{'_id': ObjectId('66ed528c841c3120d498bc29'), 'name': 'xiaobu', 'country': 'Chin
a', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游
记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age':
26, 'height': 1.7}
{'_id': ObjectId('66ed528f841c3120d498bc2a'), 'name': 'juyi', 'country': 'China',
'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的
救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age':
25, 'height': 1.69}
{'_id': ObjectId('66ed5291841c3120d498bc2b'), 'name': 'lilei', 'country': 'Chin
a', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了
我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88}
{'_id': ObjectId('66ed5291841c3120d498bc2c'), 'name': 'zhangsan', 'country': 'Chi
na', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'movies': ['头号
玩家', '肖生克的救赎'], 'cites': ['旧金山', '上海']}, 'age': 23, 'height': 1.72}
{'_id': ObjectId('66ed5291841c3120d498bc2d'), 'name': 'tony', 'country': 'USA',
'address': {'aCode': '004', 'aName': '洛杉矶'}, 'favorites': {'books': ['权利的游
戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '洛杉矶']}, 'age': 28, 'heigh
t': 1.79}
```

练习2:

插入下面多位user的信息（要求使用insert_many):

```
user6 = {
    "name" : "xiaoxiaobu",
    "country" : "USA",
    "address" : {
        "aCode" : "001",
        "aName" : "纽约"},
    "favorites" : {
        "books" : [
            "权利的游戏",
            "教父"],
        "cites" : [
            "芝加哥",
            "芝加哥",
            "洛杉矶"]},
    "age" : 26,
    "height" : 1.78
}

user7 = {
    "name" : "lisa",
    "country" : "USA",
    "address" : {
        "aCode" : "001",
        "aName" : "纽约"},
    "favorites" : {
```

```
            "books" : [
                "权利的游戏",
                "飘",
                "谁动了我的奶酪"],
            "cites" : [
                "芝加哥",
                "芝加哥",
                "洛杉矶"]},
        "age" : 20,
        "height" : 1.71
    }

    user8 = {
        "name" : "xiaoxiao",
        "country" : "USA",
        "address" : {
            "aCode" : "005",
            "aName" : "费城"},
        "favorites" : {
            "books" : [
                "权利的游戏",
                "飘"],
            "cites" : [
                "芝加哥",
                "芝加哥",
                "洛杉矶"]},
        "age" : 21,
        "height" : 1.75
    }
```

解答2:

In [16]:

```
more_users = [
    {
    "name" : "xiaoxiaobu",
    "country" : "USA",
    "address" : {
        "aCode" : "001",
        "aName" : "纽约"},
    "favorites" : {
        "books" : [
            "权利的游戏",
            "教父"],
        "cites" : [
            "芝加哥",
            "芝加哥",
            "洛杉矶"]},
    "age" : 26,
    "height" : 1.78
},
{
    "name" : "lisa",
    "country" : "USA",
    "address" : {
        "aCode" : "001",
        "aName" : "纽约"},
    "favorites" : {
```

```
        "books" : [
            "权利的游戏",
            "飘",
            "谁动了我的奶酪"],
        "cites" : [
            "芝加哥",
            "芝加哥",
            "洛杉矶"]},
    "age" : 20,
    "height" : 1.71
},
{
    "name" : "xiaoxiao",
    "country" : "USA",
    "address" : {
        "aCode" : "005",
        "aName" : "费城"},
    "favorites" : {
        "books" : [
            "权利的游戏",
            "飘"],
        "cites" : [
            "芝加哥",
            "芝加哥",
            "洛杉矶"]},
    "age" : 21,
    "height" : 1.75
}
]
users_col.insert_many(more_users)
content = users_col.find()
for each in content:
    print(each)
```

{'_id': ObjectId('66ed528c841c3120d498bc29'), 'name': 'xiaobu', 'country': 'China', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age': 26, 'height': 1.7}
{'_id': ObjectId('66ed528f841c3120d498bc2a'), 'name': 'juyi', 'country': 'China', 'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age': 25, 'height': 1.69}
{'_id': ObjectId('66ed5291841c3120d498bc2b'), 'name': 'lilei', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88}
{'_id': ObjectId('66ed5291841c3120d498bc2c'), 'name': 'zhangsan', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'movies': ['头号玩家', '肖生克的救赎'], 'cites': ['旧金山', '上海']}, 'age': 23, 'height': 1.72}
{'_id': ObjectId('66ed5291841c3120d498bc2d'), 'name': 'tony', 'country': 'USA', 'address': {'aCode': '004', 'aName': '洛杉矶'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '洛杉矶']}, 'age': 28, 'height': 1.79}
{'_id': ObjectId('66ed5292841c3120d498bc2e'), 'name': 'xiaoxiaobu', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '教父'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 26, 'height': 1.78}
{'_id': ObjectId('66ed5292841c3120d498bc2f'), 'name': 'lisa', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 20, 'height': 1.71}
{'_id': ObjectId('66ed5292841c3120d498bc30'), 'name': 'xiaoxiao', 'country': 'USA', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 21, 'height': 1.75}

后续的操作需要保证8位用户的数据都插入成功。

！注意：第3部分中的练习都需要保证8条用户的记录都为原始记录，如果操作了更新和删除数据，可以使用 `users_col.delete_many({})` 删除所有数据，再重新插入所有用户数据。

3. 查找数据

MongoDB的查找功能对应的方法是

- find(查询条件，返回字段)
- find_one(查询条件，返回字段)：一次只返回一次信息

返回字段为一个字典 `{key:value}` ，key为字段名称，value为 `0` （不返回该字段）或 `1` （返回该字段）。
其中_id是一个特殊字段，只有指定 `_id:0` 才不会返回，否则默认返回。

3.1 基础查询

find()返回的是可迭代的PyMongo对象，这个对象可以被for循环展开。展开后得到每一条都是记录的字典。

查找年龄为25岁，并且只显示名字。

In [17]:
```
result = users_col.find({'age': 25}, {'_id': 0, 'name': 1})
print(type(result))
for each in result:
    print(each)
```

```
<class 'pymongo.cursor.Cursor'>
{'name': 'juyi'}
```

还可以以下面这种形式输出：
将for循环写在列表中，具体语法可参考
https://blog.csdn.net/xuxiao1991101/article/details/50815146

In [18]:
```
content = [x for x in users_col.find({'age': 25}, {'_id': 0, 'name': 1})]
print(content)
```

```
[{'name': 'juyi'}]
```

显示所有数据（在前面多次使用）

In [19]:
```
for each in users_col.find():
    print(each)
```

```
{'_id': ObjectId('66ed528c841c3120d498bc29'), 'name': 'xiaobu', 'country': 'Chin
a', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游
记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age':
26, 'height': 1.7}
{'_id': ObjectId('66ed528f841c3120d498bc2a'), 'name': 'juyi', 'country': 'China',
'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的
救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age':
25, 'height': 1.69}
{'_id': ObjectId('66ed5291841c3120d498bc2b'), 'name': 'lilei', 'country': 'Chin
a', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了
我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88}
{'_id': ObjectId('66ed5291841c3120d498bc2c'), 'name': 'zhangsan', 'country': 'Chi
na', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'movies': ['头号
玩家', '肖生克的救赎'], 'cites': ['旧金山', '上海']}, 'age': 23, 'height': 1.72}
{'_id': ObjectId('66ed5291841c3120d498bc2d'), 'name': 'tony', 'country': 'USA',
'address': {'aCode': '004', 'aName': '洛杉矶'}, 'favorites': {'books': ['权利的游
戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '洛杉矶']}, 'age': 28, 'heigh
t': 1.79}
{'_id': ObjectId('66ed5292841c3120d498bc2e'), 'name': 'xiaoxiaobu', 'country': 'U
SA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利
的游戏', '教父'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 26, 'height':
1.78}
{'_id': ObjectId('66ed5292841c3120d498bc2f'), 'name': 'lisa', 'country': 'USA',
'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游
戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 20,
'height': 1.71}
{'_id': ObjectId('66ed5292841c3120d498bc30'), 'name': 'xiaoxiao', 'country': 'US
A', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的
游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 21, 'height': 1.7
5}
```

练习3：

查询country为China的用户(要求：输出为列表)

解答3：

In [20]:
```
content = [x for x in users_col.find({'country':'China'}, {'_id': 0})]
print(content)
```

```
[{'name': 'xiaobu', 'country': 'China', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age': 26, 'height': 1.7}, {'name': 'juyi', 'country': 'China', 'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age': 25, 'height': 1.69}, {'name': 'lilei', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88}, {'name': 'zhangsan', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'movies': ['头号玩家', '肖生克的救赎'], 'cites': ['旧金山', '上海']}, 'age': 23, 'height': 1.72}]
```

3.2 复杂查询

查询年龄在20岁并且身高为1.71m的用户

In [21]:
```
result = users_col.find({'age':20,'height':1.71},{'_id':0})
for each in result:
    print(each)
```

```
{'name': 'lisa', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 20, 'height': 1.71}
```

3.2.1 逻辑查询

符号	含义	示例
$lt	小于	{'age': {'$lt': 20}}
$gt	大于	{'age': {'$gt': 20}}
$lte	小于等于	{'age': {'$lte': 20}}
$gte	大于等于	{'age': {'$gte': 20}}
$ne	不等于	{'age': {'$ne': 20}}
$in	在范围内	{'age': {'$in': [20, 23]}}
$nin	不在范围内	{'age': {'$nin': [20, 23]}}
$or	逻辑非	{'$or': [{'age':25},{'country':'USA'}]}
$and	逻辑与	{'$and': [{'age':25},{'country':'USA'}]}

查询用户年龄在21和25之间的记录（包括21和25岁）,并且不显示_id

In [22]:
```
result = users_col.find({'age': {'$gte':21, '$lte':25}},{'_id':0})
for each in result:
    print(each)
```

```
{'name': 'juyi', 'country': 'China', 'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age': 25, 'height': 1.69}
{'name': 'zhangsan', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'movies': ['头号玩家', '肖生克的救赎'], 'cites': ['旧金山', '上海']}, 'age': 23, 'height': 1.72}
{'name': 'xiaoxiao', 'country': 'USA', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 21, 'height': 1.75}
```

查询favorites.movies中，只有"头号玩家","肖生克的救赎"的用户的名字，并且喜欢的电影以该顺序排序。

In [23]:
```
result = users_col.find({'favorites.movies': ['头号玩家','肖生克的救赎']},{'_id'
for each in result:
    print(each)
```

```
{'name': 'zhangsan'}
```

3.2.2 匹配查询

符号	含义	示例	示例含义
$regex	匹配正则表达式	{'name': {'$regex': '^M.*'}}	name以M开头
$exists	属性是否存在	{'name': {'$exists': True}}	name属性存在
$type	类型判断	{'age': {'$type': 'int'}}	age的类型为int
$mod	数字模操作	{'age': {'$mod': [5, 0]}}	年龄模5余0
$text	文本查询	{'$text': {'$search': 'Mike'}}	text类型的属性中包含Mike字符串
$where	高级条件查询	{'$where': 'obj.fans_count == obj.follows_count'}	自身粉丝数等于关注数

查询用户名字以'xiao'开头的数据

In [24]:
```
results = users_col.find({'name': {'$regex': '^xiao.*'}},{'_id':0})
for each in results:
    print(each)
```

```
{'name': 'xiaobu', 'country': 'China', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age': 26, 'height': 1.7}
{'name': 'xiaoxiao', 'country': 'USA', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 21, 'height': 1.75}
{'name': 'xiaoxiaobu', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '教父'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 26, 'height': 1.78}
```

3.3 排序

- sort()：在其中传入排序的字段及升降序标志
  `collection.find().sort('列名', 升降序标志 )`

`pymongo.ASCENDING` 指定升序(或者1）；
`pymongo.DESCENDING` 指定降序（或者-1）；

In [25]:
```
results = users_col.find().sort('name', pymongo.ASCENDING)
print([result['name'] for result in results])
```

```
['juyi', 'lilei', 'lisa', 'tony', 'xiaobu', 'xiaoxiao', 'xiaoxiaobu', 'zhangsan']
```

In [26]:
```
results = users_col.find().sort('name', -1)
print([result['name'] for result in results])
```

```
['zhangsan', 'xiaoxiaobu', 'xiaoxiao', 'xiaobu', 'tony', 'lisa', 'lilei', 'juyi']
```

3.4 偏移

- skip(x)：偏移x个位置，忽略前x个元素，得到第x+1个及以后的元素

In [27]:
```
results = users_col.find().sort('name', pymongo.ASCENDING).skip(2)
print([result['name'] for result in results])
```

```
['lisa', 'tony', 'xiaobu', 'xiaoxiao', 'xiaoxiaobu', 'zhangsan']
```

- limit()：指定要取的结果个数

In [28]:
```
results = users_col.find().sort('name', pymongo.ASCENDING).skip(2).limit(2)
print([result['name'] for result in results])
```

```
['lisa', 'tony']
```

在数据库数量非常庞大的时候，如千万、亿级别，最好不要使用大的偏移量来查询数据，因为这样很可能导致内存溢出。

3.5 练习

练习4:

查询所有age > 25的记录，并且不显示_id

解答4:

In [29]:
```
result = users_col.find({'age': {'$gt':25}},{'_id':0})
for each in result:
    print(each)
```

```
{'name': 'xiaobu', 'country': 'China', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age': 26, 'height': 1.7}
{'name': 'tony', 'country': 'USA', 'address': {'aCode': '004', 'aName': '洛杉矶'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '洛杉矶']}, 'age': 28, 'height': 1.79}
{'name': 'xiaoxiaobu', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '教父'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 26, 'height': 1.78}
```

练习5：

查询用户的country是USA并且年龄大于20岁的数据。

解答5：

In [30]:
```
results = users_col.find({'age':{'$gt':20},'country':'USA'},{'_id':0})
for each in results:
    print(each)
```

```
{'name': 'tony', 'country': 'USA', 'address': {'aCode': '004', 'aName': '洛杉矶'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '洛杉矶']}, 'age': 28, 'height': 1.79}
{'name': 'xiaoxiaobu', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '教父'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 26, 'height': 1.78}
{'name': 'xiaoxiao', 'country': 'USA', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 21, 'height': 1.75}
```

练习6：

查找喜欢书籍的，且身高在1.68-1.72m之间的中国用户

解答6：

In [31]:
```
results = users_col.find({'favorites.books': {'$exists': True},'height': {'$gte'
for each in results:
    print(each)
```

```
{'name': 'xiaobu', 'country': 'China', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age': 26, 'height': 1.7}
```

练习7：

查找喜欢书籍“谁动了我的奶酪” 或者 年龄不小于25岁且家不在纽约的美国用户

解答7：

In [39]:
```
result= users_col.find({
'$or':[{'favorites.books':'谁动了我的奶酪'},
{'age':{'$gte':25},'address.aName':{'$ne':'纽约'},'country':'USA'}
]},{'_id':0})
for each in result:
    print(each)
```

```
{'name': 'lilei', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88}
{'name': 'tony', 'country': 'USA', 'address': {'aCode': '004', 'aName': '洛杉矶'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '洛杉矶']}, 'age': 29, 'height': 1.79}
{'name': 'lisa', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 20, 'height': 1.71}
```

练习8：

查找既喜欢cites也喜欢books，且身高在1.80以上的用户

解答8：

In [33]:
```
results = users_col.find({
'$and':[
{'favorites.books':{'$exists': True}},
{'favorites.cites':{'$exists': True}},
{'height':{'$gt':1.80}}
]},{'_id':0})
for each in results:
    print(each)
```

```
{'name': 'lilei', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88}
```

练习9：

查找名字以xiao开头，且年龄在20-25岁之间的用户

解答9：

In [34]:
```
results = users_col.find({
'$and':[
{'name':{'$regex': '^xiao.*'}},
{'age':{'$gte':20, '$lte':25}}
]},{'_id':0})
for each in results:
    print(each)
```

```
{'name': 'xiaoxiao', 'country': 'USA', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 21, 'height': 1.75}
```

练习10：

查找用户中身高按降序排序中，排名4-6的用户

解答10：

In [35]:
```
results = users_col.find().sort('height', -1).skip(3).limit(3)
print([result['name'] for result in results])
```

```
['xiaoxiao', 'zhangsan', 'lisa']
```

！注意：第4部分中的练习都需要保证8条用户的记录都为原始记录，如果操作了删除数据，可以使用 `users_col.delete_many({})` 删除所有数据，再重新插入所有用户数据。

4. 更新记录

- collection.update_one(参数1, 参数2)：只更新一条信息
- collection.update_many(参数1, 参数2)：更新所有符合要求的信息

参数1和参数2都是字典，都不能省略。参数1用来寻找需要更新的记录，参数2用来更新记录的内容。

将名字为xiaoxiao的人的身高改为1.81(使用$set)

In [36]:
```
condition = {'name': 'xiaoxiao'}
users_col.update_one(condition, {'$set':{'height': 1.81}})
result = users_col.find(condition)
for each in result:
    print(each) #输出结果，看名字的不同
```

```
{'_id': ObjectId('66ed5292841c3120d498bc30'), 'name': 'xiaoxiao', 'country': 'US
A', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的
游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 21, 'height': 1.8
1}
```

将年龄大于20岁的用户的年龄增加1岁（使用$inc)

In [37]:
```
condition = {'age': {'$gt': 20}}
result = users_col.update_many(condition, {'$inc': {'age': 1}})
result = users_col.find(condition)
for each in result:
    print(each) #输出结果，看名字的不同
```

```
{'_id': ObjectId('66ed528c841c3120d498bc29'), 'name': 'xiaobu', 'country': 'Chin
a', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游
记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age':
27, 'height': 1.7}
{'_id': ObjectId('66ed528f841c3120d498bc2a'), 'name': 'juyi', 'country': 'China',
'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的
救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age':
26, 'height': 1.69}
{'_id': ObjectId('66ed5291841c3120d498bc2c'), 'name': 'zhangsan', 'country': 'Chi
na', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'movies': ['头号
玩家', '肖生克的救赎'], 'cites': ['旧金山', '上海']}, 'age': 24, 'height': 1.72}
{'_id': ObjectId('66ed5291841c3120d498bc2d'), 'name': 'tony', 'country': 'USA',
'address': {'aCode': '004', 'aName': '洛杉矶'}, 'favorites': {'books': ['权利的游
戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '洛杉矶']}, 'age': 29, 'heigh
t': 1.79}
{'_id': ObjectId('66ed5292841c3120d498bc2e'), 'name': 'xiaoxiaobu', 'country': 'U
SA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利
的游戏', '教父'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 27, 'height':
1.78}
{'_id': ObjectId('66ed5292841c3120d498bc30'), 'name': 'xiaoxiao', 'country': 'US
A', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的
游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 22, 'height': 1.8
1}
```

练习11：

将喜欢书籍且在中国的用户，所在城市改为上海

解答11：

In [38]:
```
condition = {'favorites.books':{'$exists': True},'country':'China'}
result = users_col.update_many(condition, {'$set':{'address.aName':'上海'}})
result = users_col.find(condition)
for each in result:
    print(each)
```

```
{'_id': ObjectId('66ed528c841c3120d498bc29'), 'name': 'xiaobu', 'country': 'Chin
a', 'address': {'aCode': '001', 'aName': '上海'}, 'favorites': {'books': ['西游
记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age':
27, 'height': 1.7}
{'_id': ObjectId('66ed5291841c3120d498bc2b'), 'name': 'lilei', 'country': 'Chin
a', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了
我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88}
```

对查询结果去重

- collection.distinct('列名')：返回一个去重后的列表

In [46]:
```
name = users_col.distinct('name')
print(name)
```

```
['juyi', 'lilei', 'lisa', 'tony', 'xiaobu', 'xiaoxiao', 'xiaoxiaobu', 'zhangsan']
```

练习12：

用户的年龄有多少种(最后输出数字,提示：可使用python的len())

解答12：

In [47]:
```
age = users_col.distinct('age')
print(len(age))
```

```
7
```

！注意：第5部分中的练习都需要保证8条用户的记录操作了更新数据。

5. 删除记录

- collection.delete_one(参数)：删除一条记录
- collection.delete_many(参数)：删除符合要求的所有记录

参数都是字典，不建议省略。

删除第1个country在USA的用户

In [48]:
```
users_col.delete_one({'country': 'USA'})
result = users_col.find({},{'_id':0})
for each in result:
    print(each) #输出结果，看名字的不同
```

```
{'name': 'xiaobu', 'country': 'China', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age': 27, 'height': 1.7, 'city': '上海'}
{'name': 'juyi', 'country': 'China', 'address': {'aCode': '002', 'aName': '广州'}, 'favorites': {'movies': ['肖生克的救赎', '阿甘正传', '头号玩家'], 'cites': ['衡阳', '南宁', '上海', '深圳']}, 'age': 26, 'height': 1.69}
{'name': 'lilei', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'books': ['谁动了我的奶酪', 'CSAPP'], 'cites': ['上海', '杭州']}, 'age': 18, 'height': 1.88, 'city': '上海'}
{'name': 'zhangsan', 'country': 'China', 'address': {'aCode': '003', 'aName': '上海'}, 'favorites': {'movies': ['头号玩家', '肖生克的救赎'], 'cites': ['旧金山', '上海']}, 'age': 24, 'height': 1.72}
{'name': 'xiaoxiaobu', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '教父'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 27, 'height': 1.78}
{'name': 'lisa', 'country': 'USA', 'address': {'aCode': '001', 'aName': '纽约'}, 'favorites': {'books': ['权利的游戏', '飘', '谁动了我的奶酪'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 20, 'height': 1.71}
{'name': 'xiaoxiao', 'country': 'USA', 'address': {'aCode': '005', 'aName': '费城'}, 'favorites': {'books': ['权利的游戏', '飘'], 'cites': ['芝加哥', '芝加哥', '洛杉矶']}, 'age': 22, 'height': 1.81}
```

练习13:

删除所有country在USA或者年龄小于等于26岁的用户

解答13:

In [49]:
```
condition = {
'$or':[
{'country':'USA'},
{'age':{'$lte':26}}
]}
users_col.delete_many(condition)
result = users_col.find({},{'_id':0})
for each in result:
    print(each) #
```

```
{'name': 'xiaobu', 'country': 'China', 'address': {'aCode': '001', 'aName': '北京'}, 'favorites': {'books': ['西游记', '红楼梦', '三国演义', '水浒传'], 'cites': ['韶关', '深圳', '佛山']}, 'age': 27, 'height': 1.7, 'city': '上海'}
```

删除所有数据

In [50]:
```
users_col.delete_many({})
```

Out[50]:
```
DeleteResult({'n': 1, 'ok': 1.0}, acknowledged=True)
```

6. Python Object与Mongodb Document的转换

ODM(Object-Document Mapper)是指将python中的对象和MongoDb中文档之间进行映射，将文档和对文档数据的操作封装成一个python类。
用户在操作文档数据的时候，只需关心类的定义与使用方法即可，无需关系与mongodb的交互细节。

如下是一个ODM操作示例：

由于环境中没有安装mongoengine，示例仅使用普通的类和操作方法做演示。有兴趣的同学可以参考网上关于mongoengine的资料，并在本地实验。
创建一个userTable类，python_to_mongo函数将类对象转化为文档插入mongodb，mongo_to_python函数 根据查询条件从mongodb中查询出相应的文档并将其转化为python类对象

In [51]:
```
class userTable(object):
    def __init__(self, _id, name, age, country, height, favorites):
        self._id = _id
        self.name = name
        self.age = age
        self.country = country
        self.height = height
        self.favorites = favorites

# 插入一条文档
def insert_one_to_mongo(user):
    users_col.insert_one(user.__dict__)

# 根据条件从mongodb中查询文档
def query_from_mongo(condition):
    result = users_col.find(condition)
    user_list = []
    for doc in result:
        temp = userTable(doc['_id'],doc['name'],doc['age'],doc['country'],doc['h
        user_list.append(temp)
    return user_list

if __name__=='__main__':

    #首先删除users_col全部的数据
    users_col.delete_many({})

    #构造两条数据
    user1 = userTable('001', 'lilei', 28, 'China', 1.81, {'books':['谁动了我的奶
    user2 = userTable('002', 'hanmeimei', 30, 'China',1.70, {'movies':['头号玩家
    user3 = userTable('003', 'zhangsan', 21, 'China',1.76, {'books':['人类简史',

    #插入三条数据
    insert_one_to_mongo(user1)
    insert_one_to_mongo(user2)
    insert_one_to_mongo(user3)

    #根据查询条件查询文档（查询喜欢书籍的用户）
    user_list = query_from_mongo({'favorites.books':{'$exists':True}})

    #输出结果
    for s in user_list:
        print(s.__dict__) #Lilei & zhangsan
```

```
{'_id': '001', 'name': 'lilei', 'age': 28, 'country': 'China', 'height': 1.81, 'f
avorites': {'books': ['谁动了我的奶酪', '飘']}}
{'_id': '003', 'name': 'zhangsan', 'age': 21, 'country': 'China', 'height': 1.76,
'favorites': {'books': ['人类简史', 'CASPP']}}
```

练习14：

根据示例程序，完成下述程序待填充的部分。
两个函数：update_to_mongo、delete_many_to_mongo
四个CRUD操作：（#待填充 语句）

In [ ]:

```
class userTable(object):
    def __init__(self, _id, name, age, country, height, favorites):
        self._id = _id
        self.name = name
        self.age = age
        self.country = country
        self.height = height
        self.favorites = favorites

# 插入一条文档
def insert_one_to_mongo(user):
    users_col.insert_one(user.__dict__)

# 根据条件从mongodb中查询文档
def query_from_mongo(condition):
    #查询
    result = users_col.find(condition)
    #将查询结果转换为pyhton对象
    user_list = []
    for doc in result:
        temp = userTable(doc['_id'],doc['name'],doc['age'],doc['country'],doc['h
        user_list.append(temp)
    return user_list

#根据条件和更新操作，对文档进行更新
def update_to_mongo(condition, operation):
    #待填充

#根据条件删除文档
def delete_many_to_mongo(condition):
    #待填充

if __name__=='__main__':

    #首先删除users_col全部的数据
    delete_many_to_mongo({})

    #构造两条数据
    user1 = userTable('001', 'lilei', 28, 'China', 1.81, {'books':['谁动了我的奶
    user2 = userTable('002', 'hanmeimei', 30, 'China',1.70, {'movies':['头号玩家
    user3 = userTable('003', 'zhangsan', 21, 'China',1.76, {'books':['人类简史',

    #插入两条数据
    insert_one_to_mongo(user1)
    insert_one_to_mongo(user2)
    insert_one_to_mongo(user3)

    #根据查询条件查询文档（查询年龄大于25岁且身高高于1.75的用户）
    user_list = query_from_mongo({})   #待填充

    #输出结果
    print('修改之前\n')
```

```
    for s in user_list:
        print(s.__dict__) #lilei

    #将hanmeimei的身高改为1.76。
    update_to_mongo({}, {})            #待填充

    #根据查询条件查询文档（查询年龄大于25岁且身高高于1.75的用户）
    user_list = query_from_mongo({})   #待填充

    print('\n修改之后\n')
    #输出结果
    for s in user_list:
        print(s.__dict__) #lilei & hanmeimei

    #删除喜欢书籍的用户
    delete_many_to_mongo({})           #待填充

    #查询剩余全部用户
    user_list = query_from_mongo({})

    print('\n删除之后\n')
    #输出结果
    for s in user_list:
        print(s.__dict__) #hanmeimei
```

解答14:

In [54]:

```
class userTable(object):
    def __init__(self, _id, name, age, country, height, favorites):
        self._id = _id
        self.name = name
        self.age = age
        self.country = country
        self.height = height
        self.favorites = favorites

# 插入一条文档
def insert_one_to_mongo(user):
    users_col.insert_one(user.__dict__)

# 根据条件从mongodb中查询文档
def query_from_mongo(condition):
    #查询
    result = users_col.find(condition)
    #将查询结果转换为pyhton对象
    user_list = []
    for doc in result:
        temp = userTable(doc['_id'],doc['name'],doc['age'],doc['country'],doc['h
        user_list.append(temp)
    return user_list

#根据条件和更新操作，对文档进行更新
def update_to_mongo(condition, operation):
    #待填充
    result = users_col.update_many(condition,operation)
    return result
21
##result = users_col.update_many(condition, {'$inc': {'age': 1}})
##result = users_col.find(condition)
```

```
#根据条件删除文档
def delete_many_to_mongo(condition):
    #待填充
    result = users_col.delete_many(condition)
    return result

if __name__=='__main__':

    #首先删除users_col全部的数据
    delete_many_to_mongo({})

    #构造两条数据
    user1 = userTable('001', 'lilei', 28, 'China', 1.81, {'books':['谁动了我的奶
    user2 = userTable('002', 'hanmeimei', 30, 'China',1.70, {'movies':['头号玩家
    user3 = userTable('003', 'zhangsan', 21, 'China',1.76, {'books':['人类简史',

    #插入两条数据
    insert_one_to_mongo(user1)
    insert_one_to_mongo(user2)
    insert_one_to_mongo(user3)

    #根据查询条件查询文档（查询年龄大于25岁且身高高于1.75的用户）
    user_list = query_from_mongo({'age':{'$gt':25},'height':{'$gt':1.75}})   #待

    #输出结果
    print('修改之前\n')
    for s in user_list:
        print(s.__dict__) #lilei

    #将hanmeimei的身高改为1.76。
    update_to_mongo({'name':'hanmeimei'}, {'$set':{'height':1.76}})            #

    #根据查询条件查询文档（查询年龄大于25岁且身高高于1.75的用户）
    user_list = query_from_mongo({'age':{'$gt':25},'height':{'$gt':1.75}})   #待

    print('\n修改之后\n')
    #输出结果
    for s in user_list:
        print(s.__dict__) #lilei & hanmeimei

    #删除喜欢书籍的用户
    delete_many_to_mongo({'favorites.books':{'$exists':True}})           #待填充

    #查询剩余全部用户
    user_list = query_from_mongo({})

    print('\n删除之后\n')
    #输出结果
    for s in user_list:
        print(s.__dict__) #hanmeimei
```

修改之前

```
{'_id': '001', 'name': 'lilei', 'age': 28, 'country': 'China', 'height': 1.81, 'f
avorites': {'books': ['谁动了我的奶酪', '飘']}}
```

修改之后

```
{'_id': '001', 'name': 'lilei', 'age': 28, 'country': 'China', 'height': 1.81, 'f
avorites': {'books': ['谁动了我的奶酪', '飘']}}
{'_id': '002', 'name': 'hanmeimei', 'age': 30, 'country': 'China', 'height': 1.7
6, 'favorites': {'movies': ['头号玩家', '肖申克的救赎']}}
```

删除之后

```
{'_id': '002', 'name': 'hanmeimei', 'age': 30, 'country': 'China', 'height': 1.7
6, 'favorites': {'movies': ['头号玩家', '肖申克的救赎']}}
```

In [ ]: