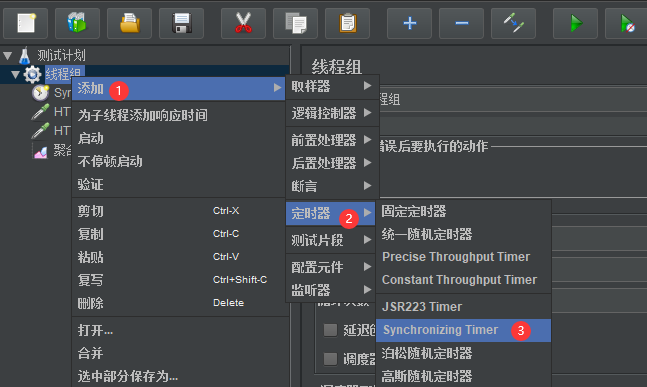
# Jmeter性能组件

### 1、同步定时器（Synchronizing Timer）

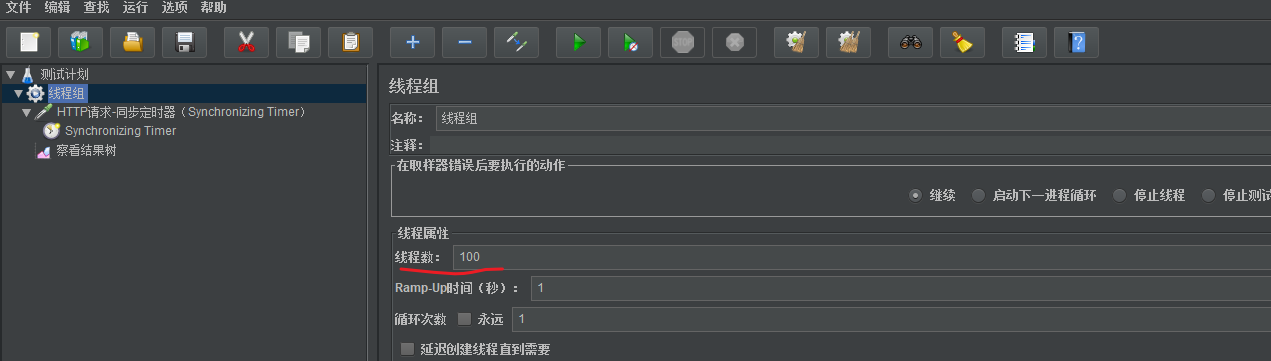
作用：实现多个用户（线程）并发访问

位置：



应用：实现100个用户同时访问请求

设置线程数100，创建100个用户



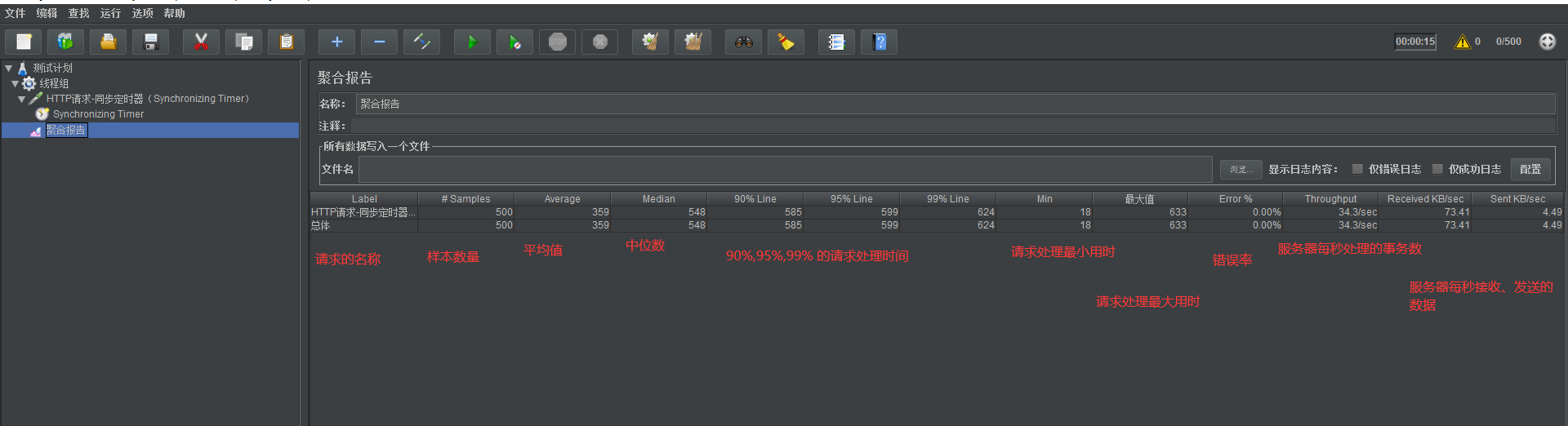
添加同步定时器，设置每组数量为100，100个用户创建完毕后同时执行请求



属性：



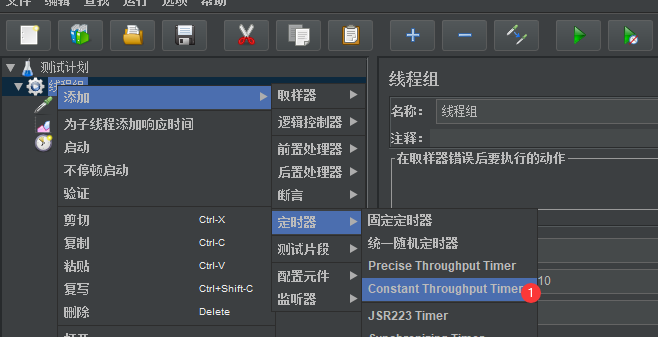
查看聚合报告：



### 2、常量吞吐器

作用： 实现一个用户频繁的访问服务器

位置：



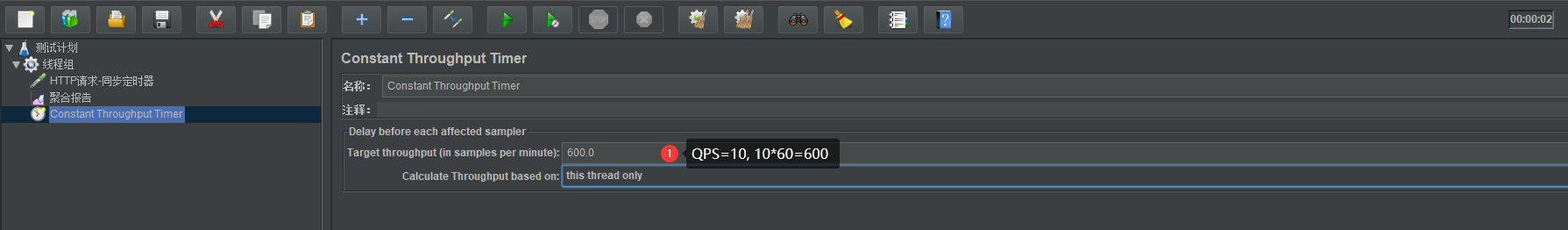
应用：一个用户以10QPS（每秒10次）的频率访问接口，持续20s

QPS： query per seconds 每秒查询服务器的次数

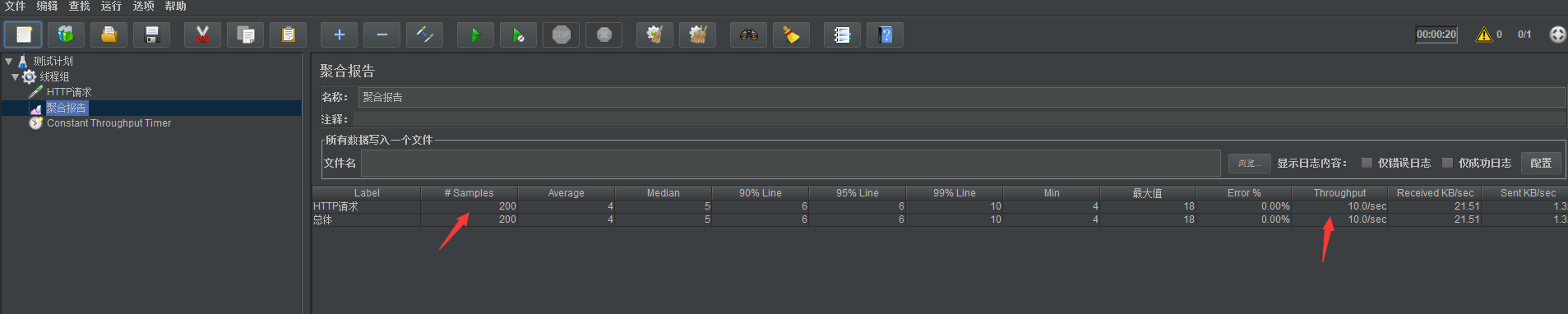
设置1个线程（代表一个用户）， 循环访问次数为 20s \* 10 =200 次



设置每分钟的吞吐量为： 10QPS \* 60s=600



查看聚合报告，访问次数为200，服务器每秒钟处理请求10个



### 3、分布式环境搭建（模拟）

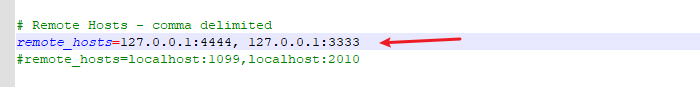
#### 3台机器，执行分布式



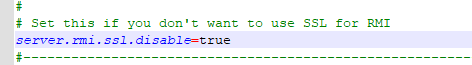
配置主机和执行机

主机配置端口和远程访问属性：

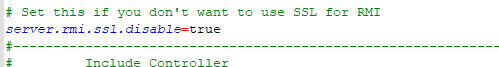
remote\_hosts



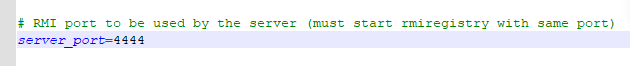
server.rmi.ssl.disable



A机器

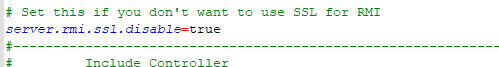


Server\_port

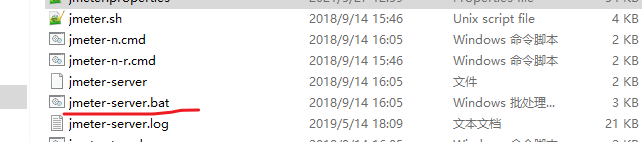


B机器





开启执行机：



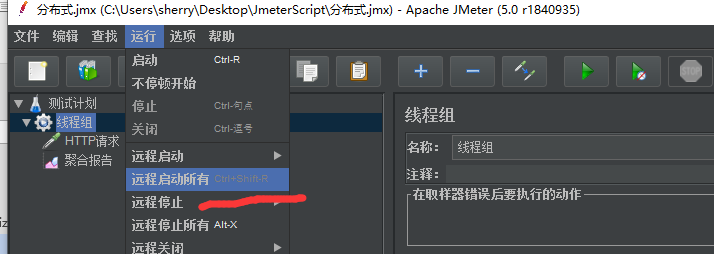
开启主机，选择图形化的方式，因为在主机上编写代码，执行机不需要



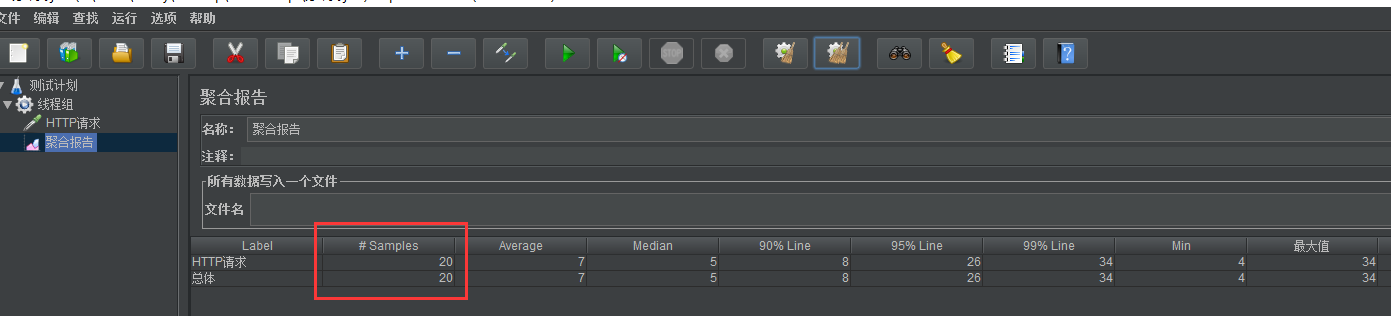
执行20次，2台机器分别执行10次； 主机设置循环10次

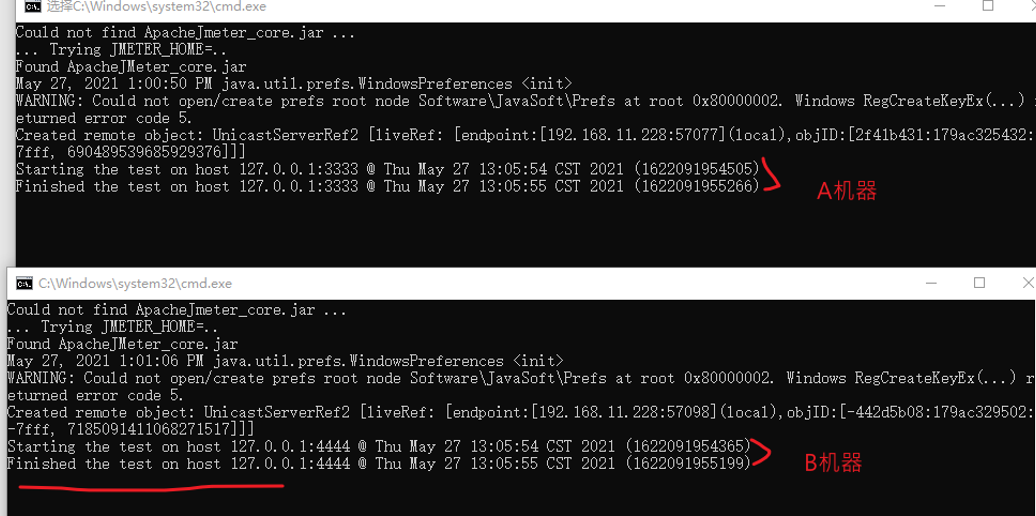


远程启动所有：

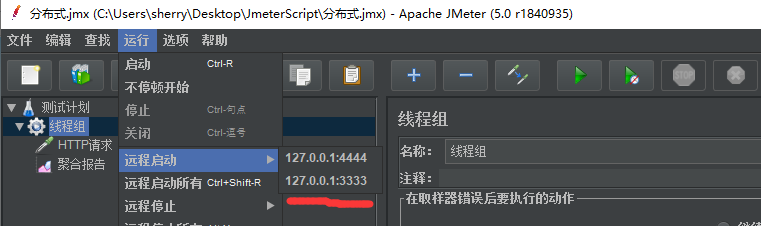


查看结果:



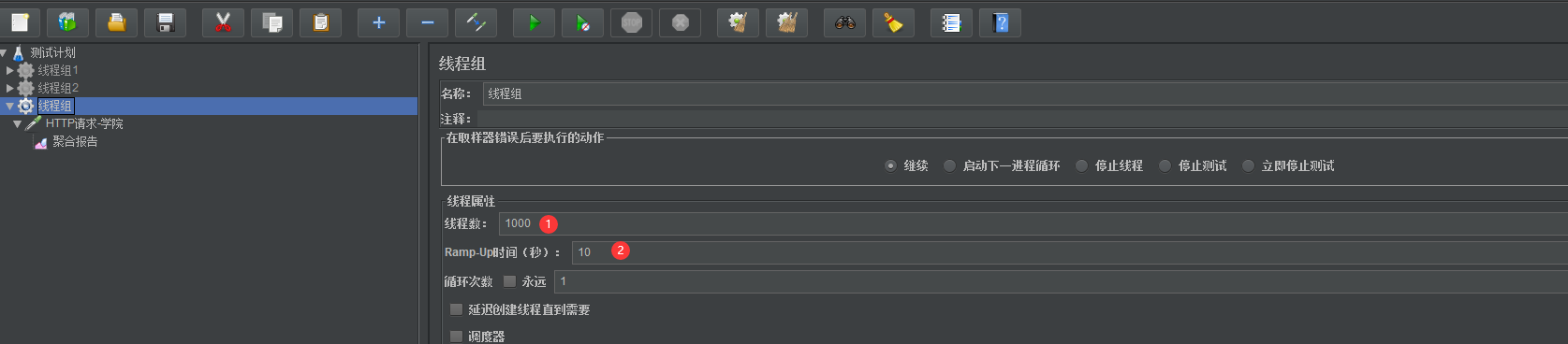


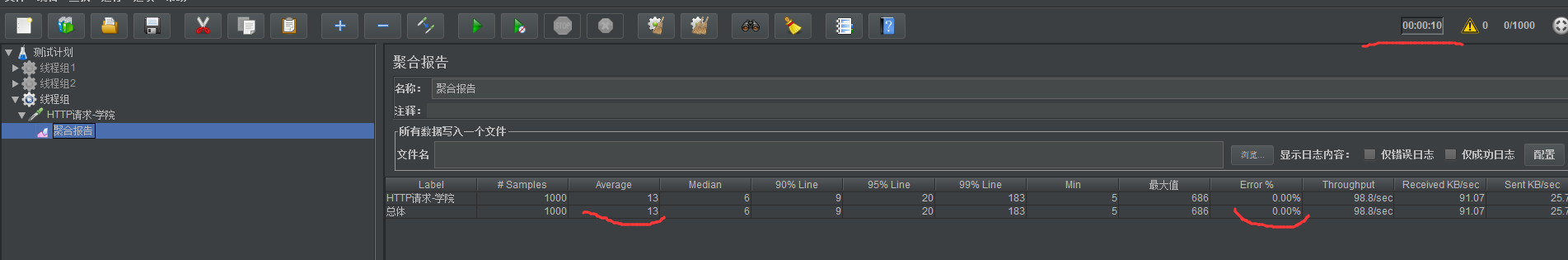
也可以分别启动，指定某个执行机运行：



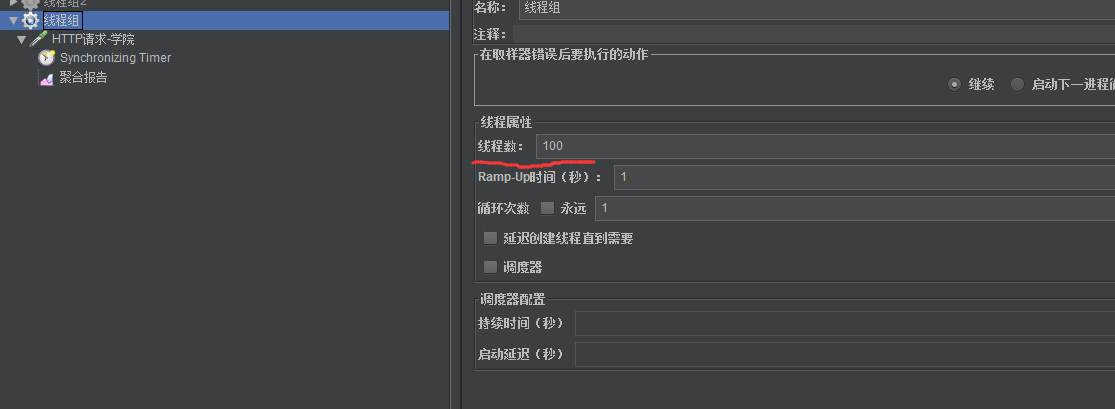
### 4、高并发：

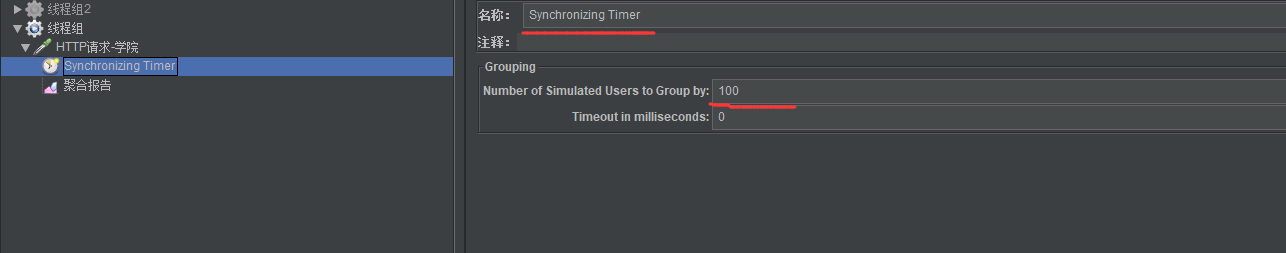
10s内1000个用户访问服务器资源





100个用户同时访问服务器

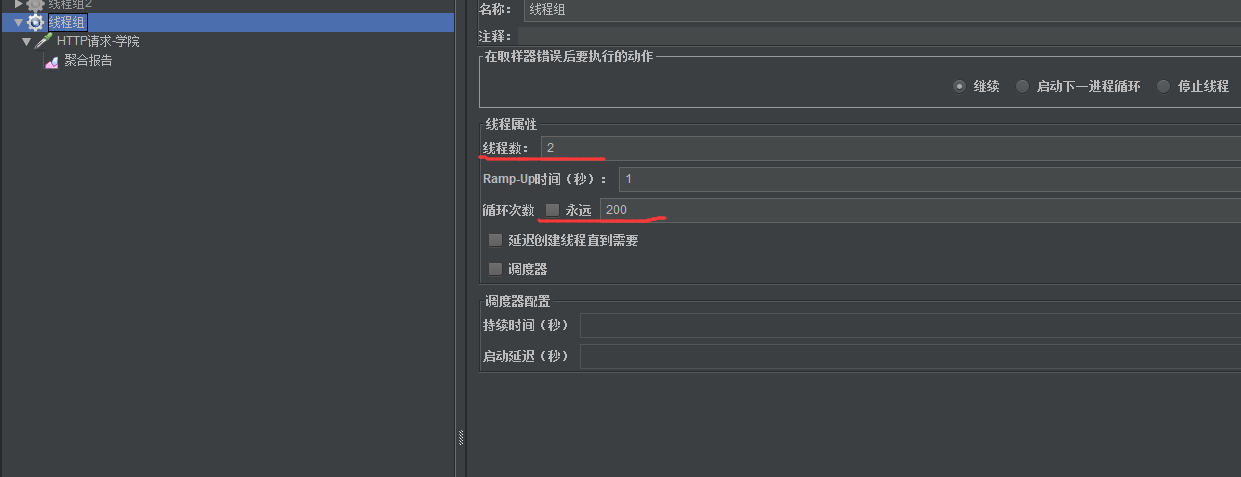


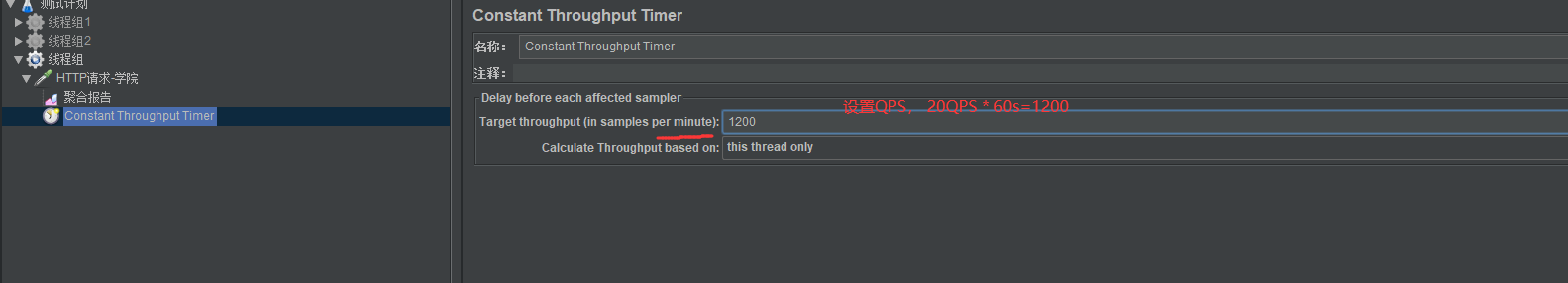


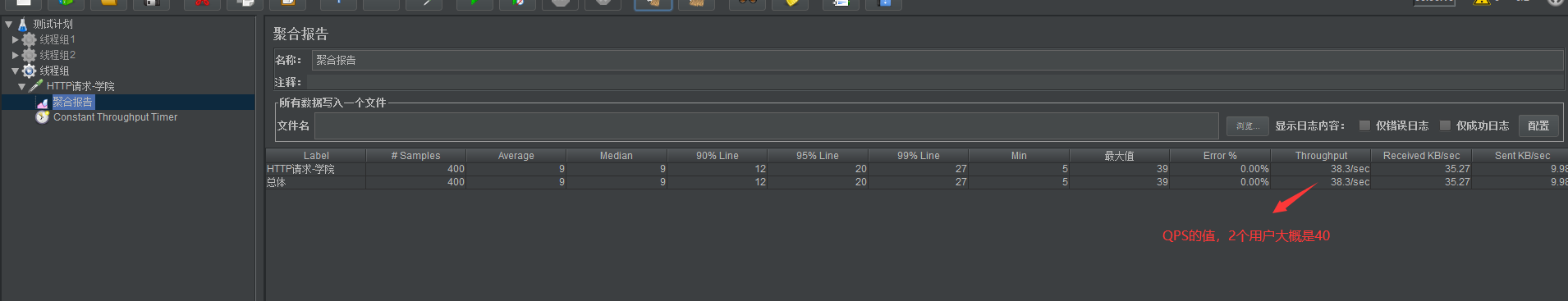


2个用户以20QPS的频率访问服务器，持续10s

20\*10=200次

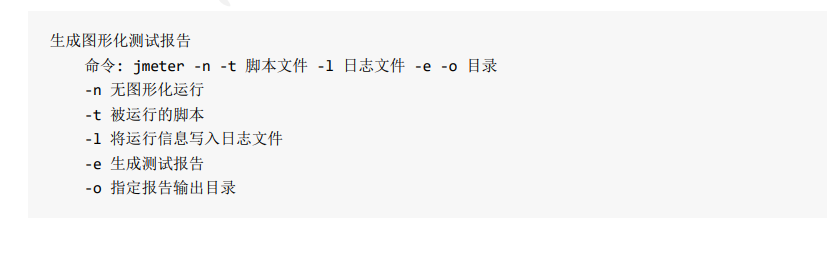


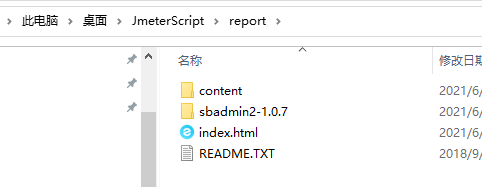




### 5、图形化测试报告

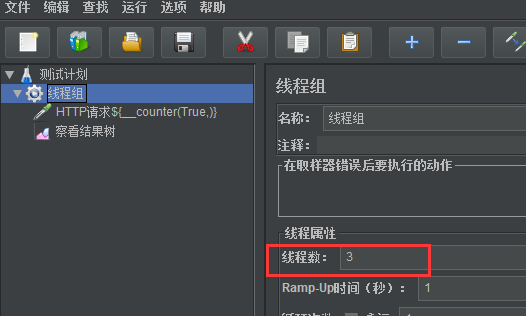
配置jmeter环境变量，生成html图形化测试报告





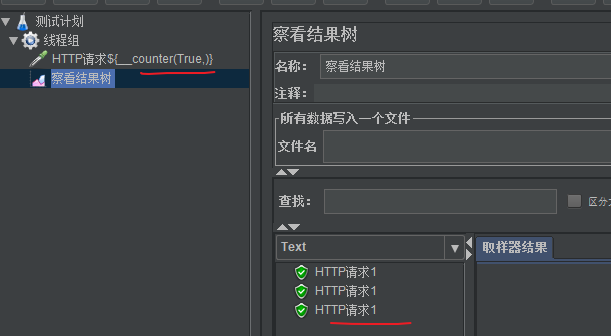
# Jmeter基本组件

### 1、线程组

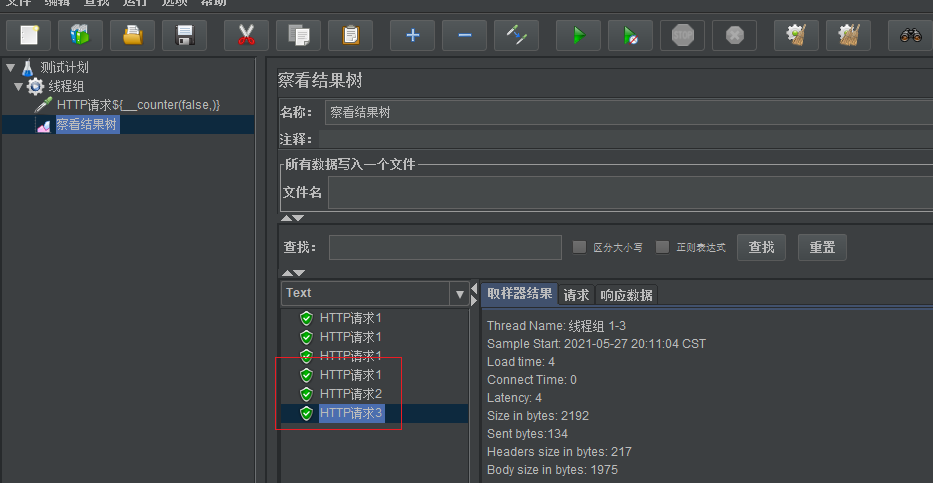


线程数： 可以指定组内线程数

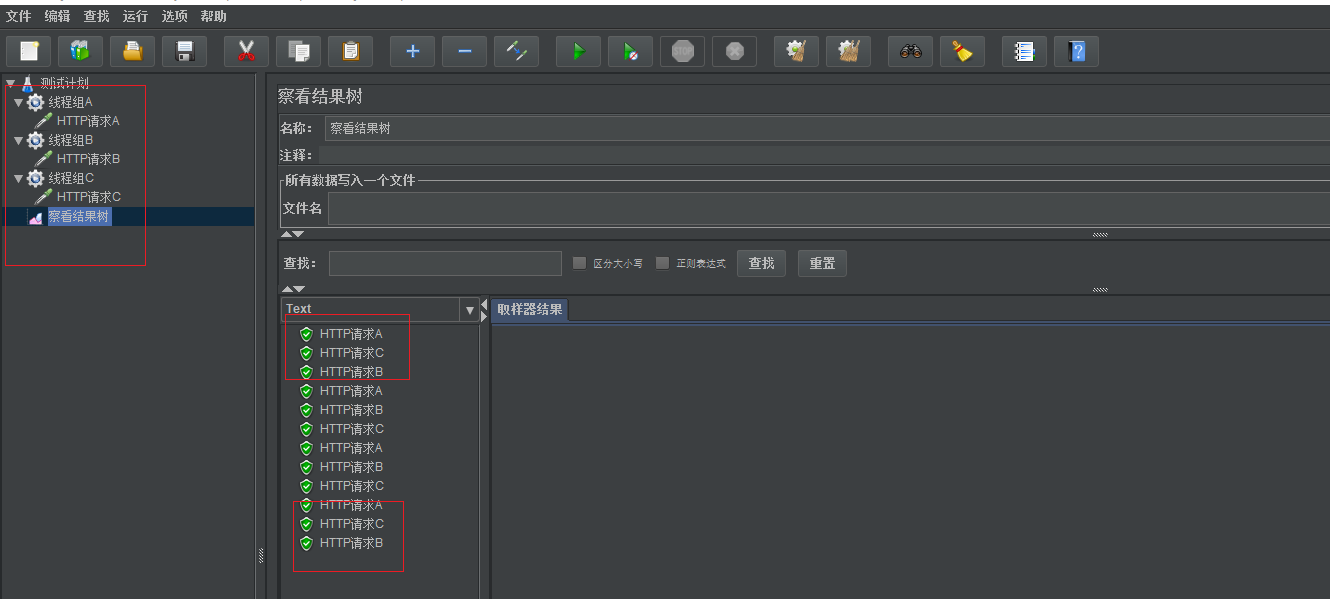
设置了每个线程都有自己的计数器



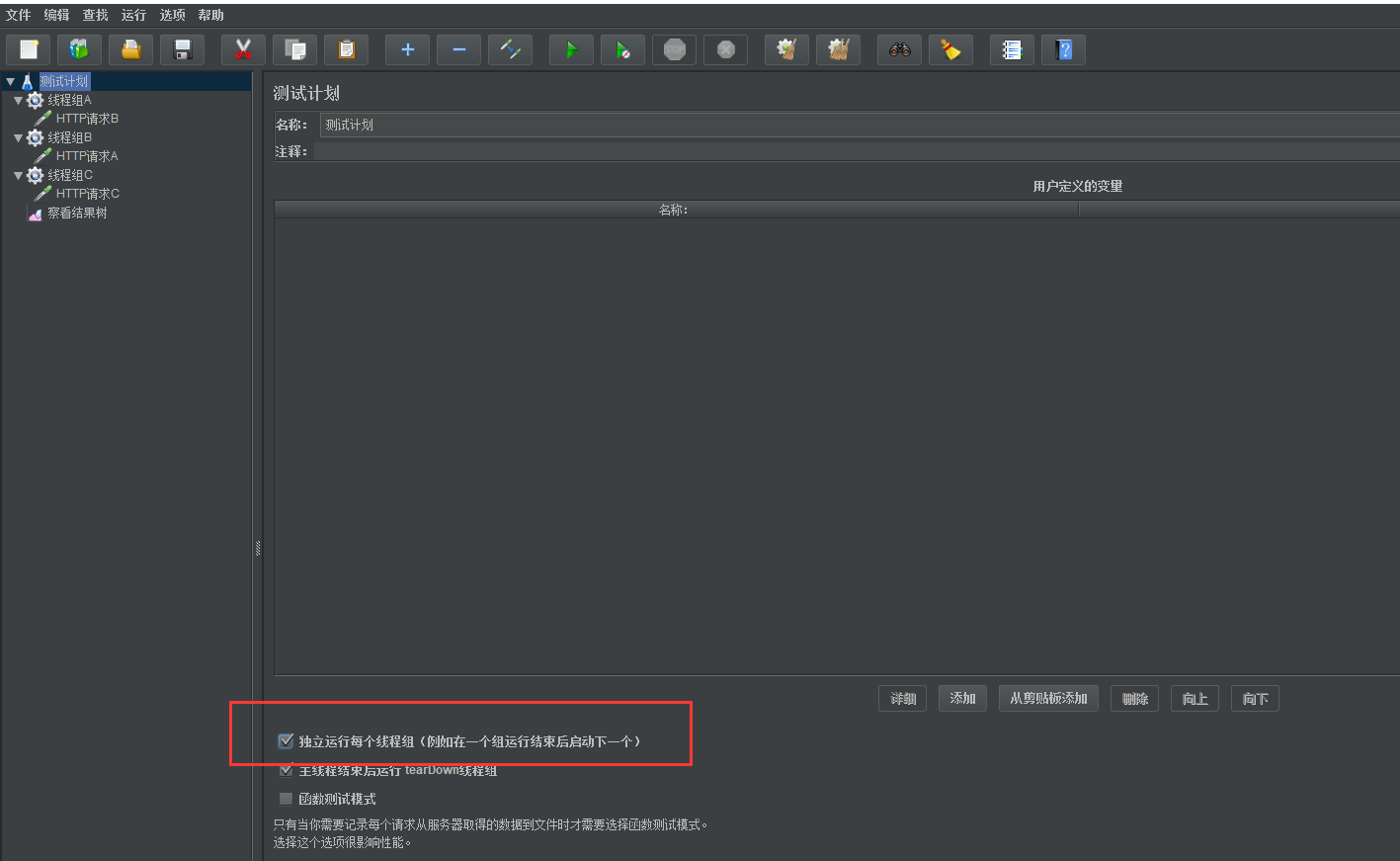
借助函数标志 ${\_\_counter(false,)}，会显示1，2，3

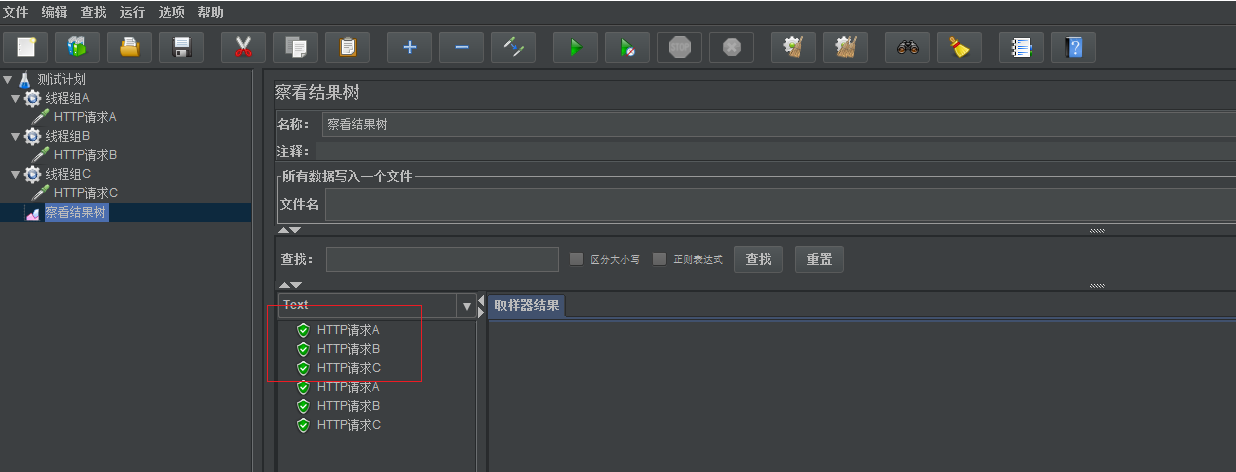


线程组的运行顺序：

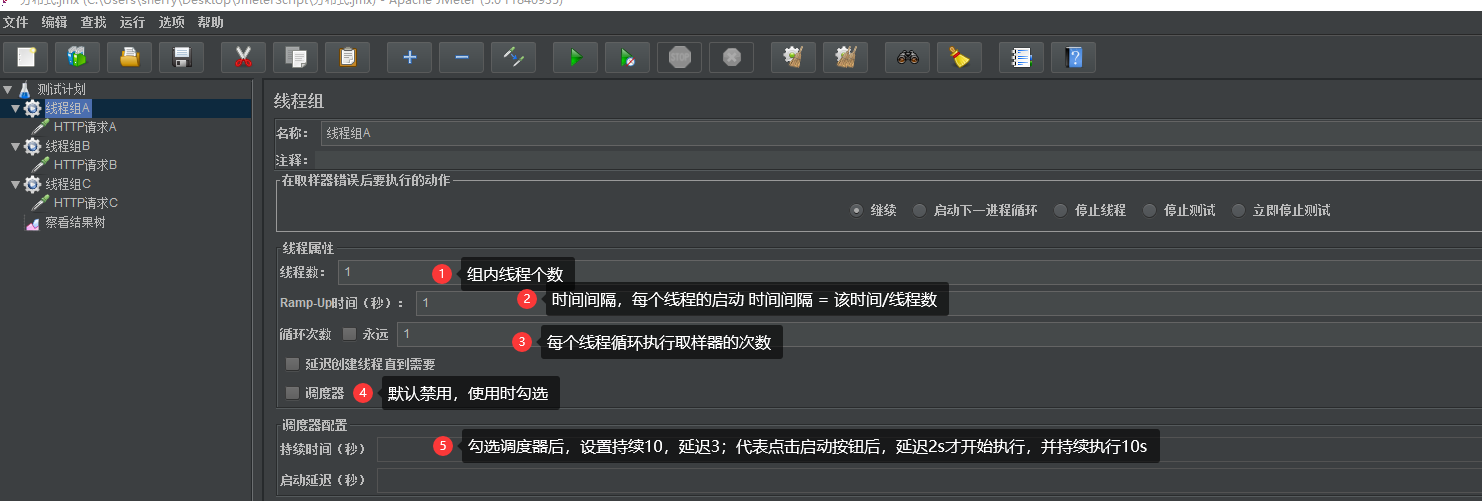


勾选下面测试计划选项后的执行顺序

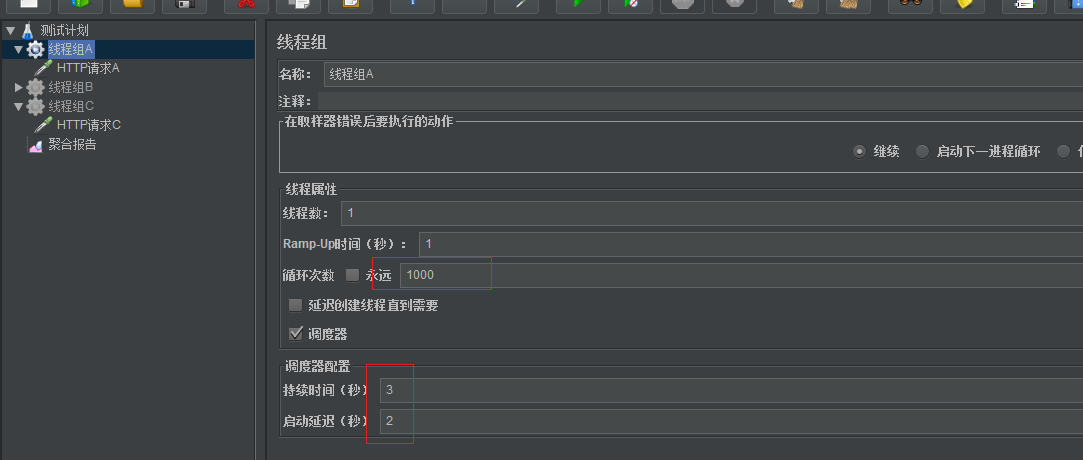




属性归纳：

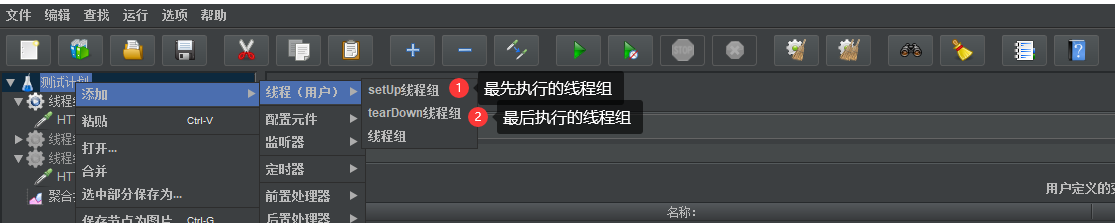


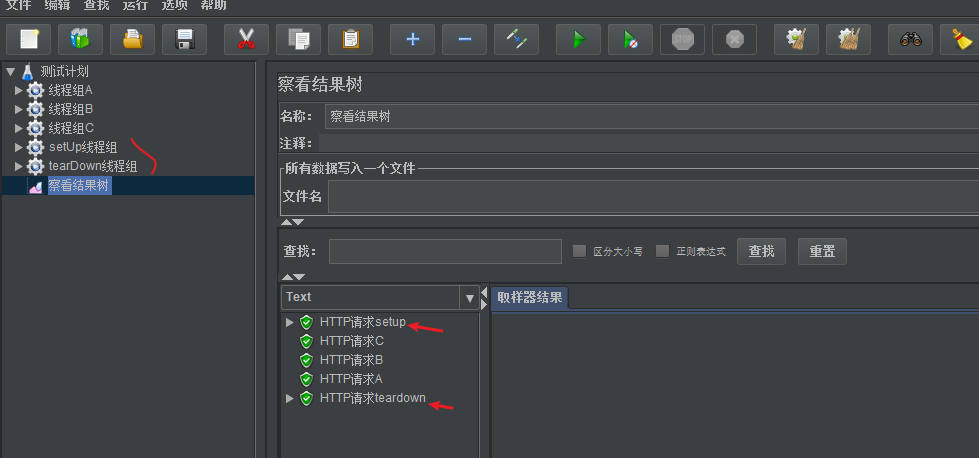
持续时间包括启动延迟的时间，到达持续时间后，就终止执行任务了，没有执行完的请求也终止了



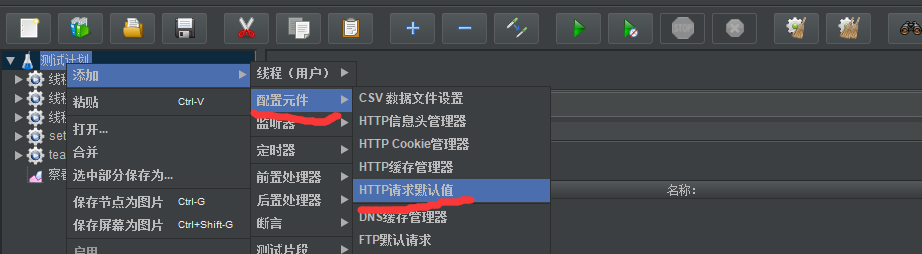


Setup和teardown线程组：

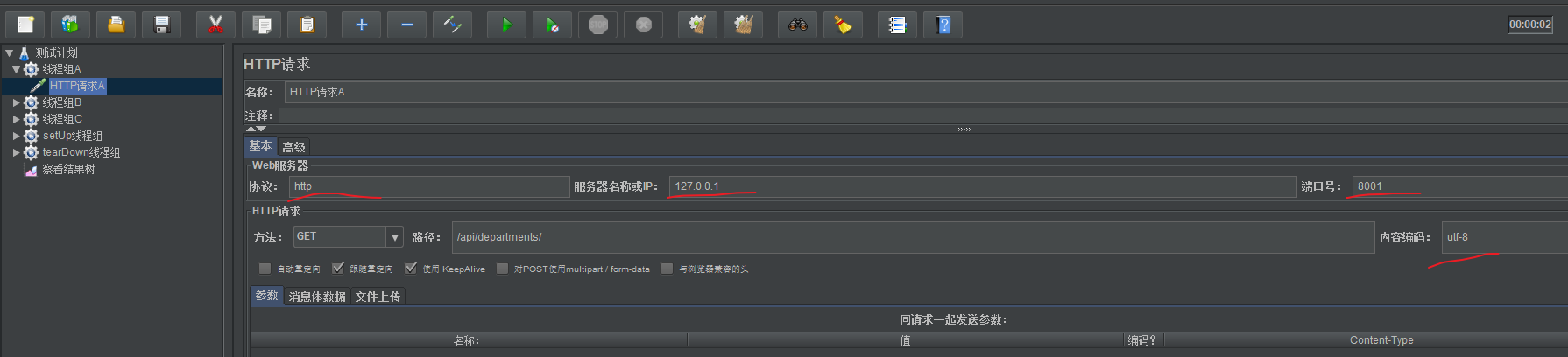




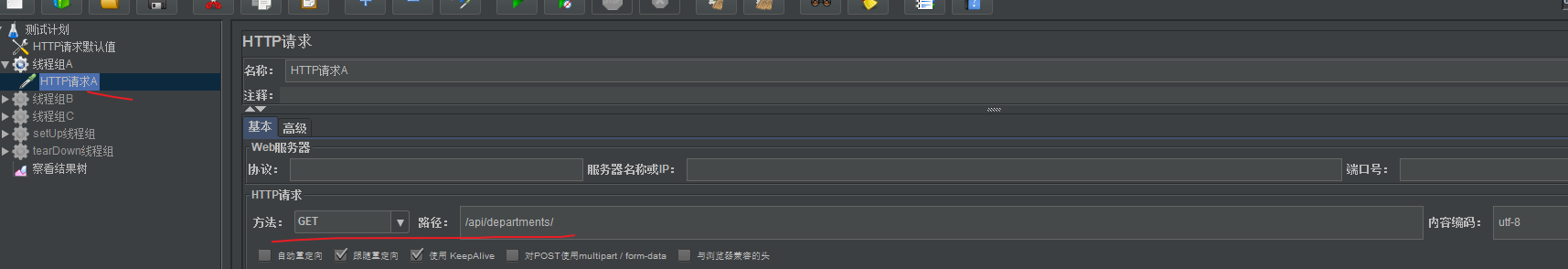
### 2、Http请求默认值



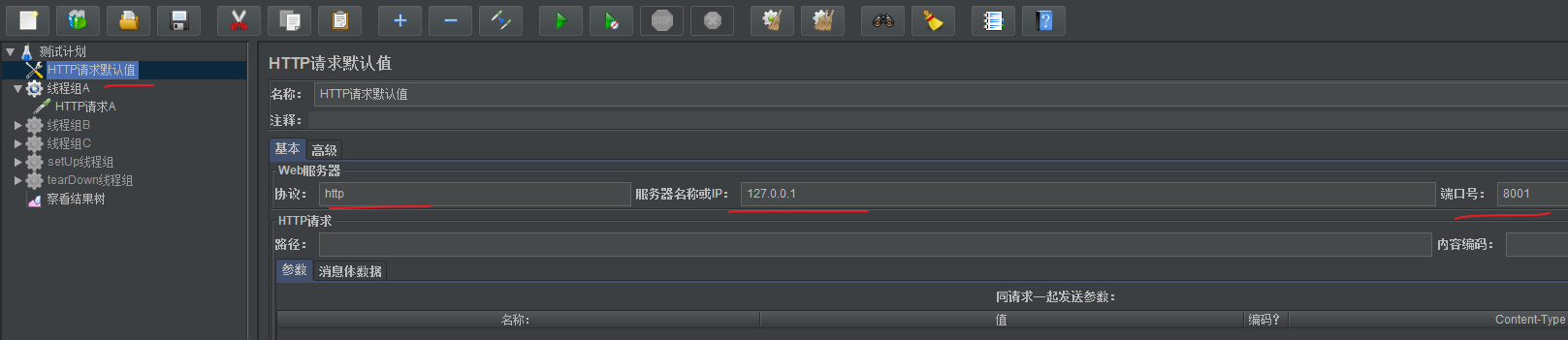
将被重复使用的字段封装进组件

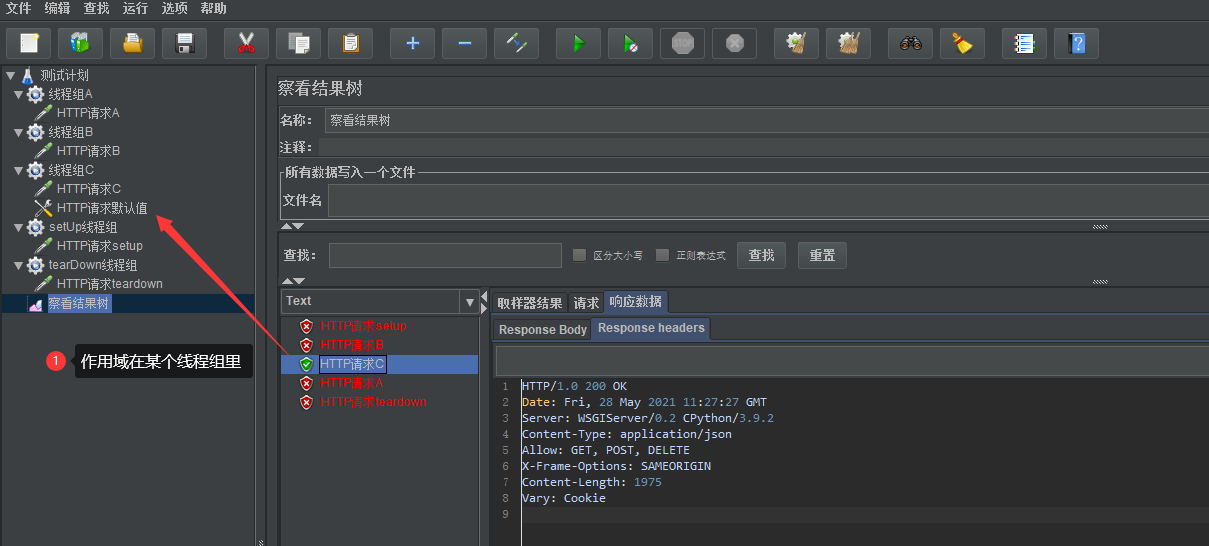


A请求中的数据，删除协议等信息



在http请求默认值的组件中配置信息，同样可以正常执行A请求

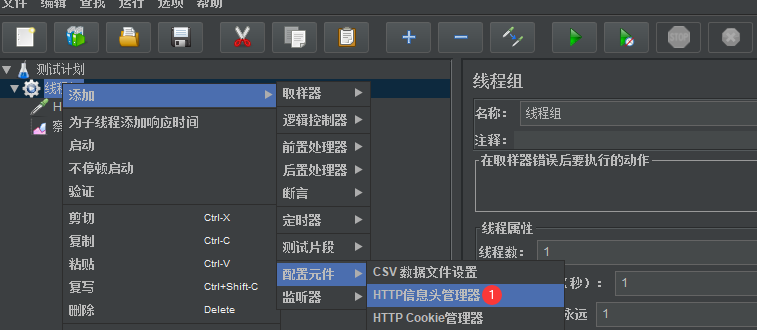




### 3、http信息头管理器

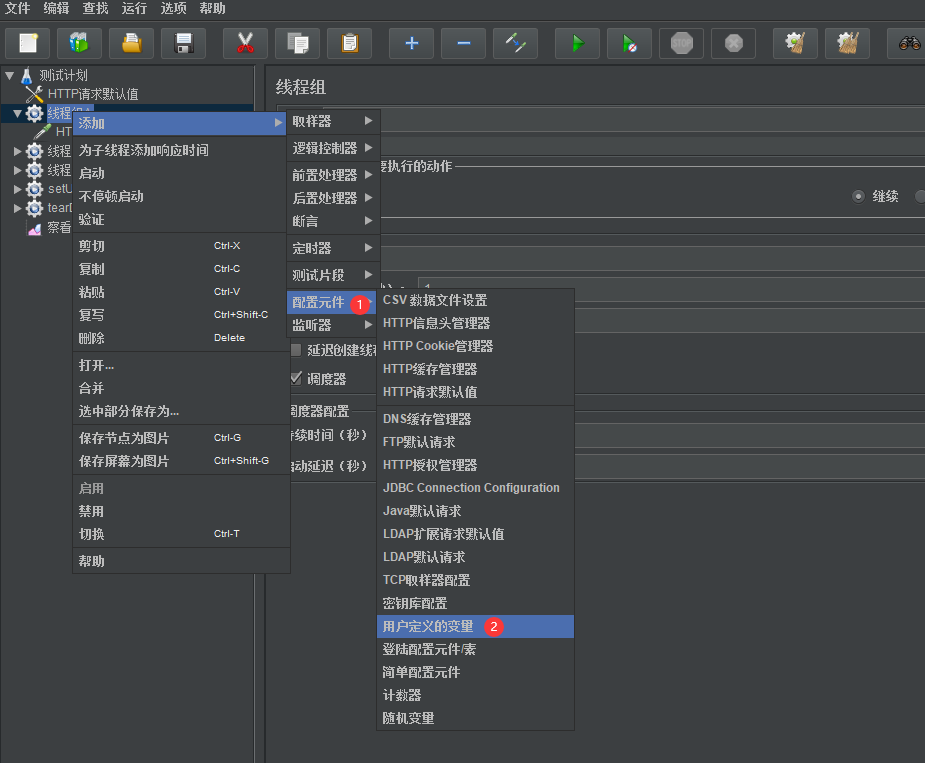
新增修改实现时提交的数据是JSON格式的，需要声明提交的数据的内容类型

Content-Type application/json;charset=utf-8

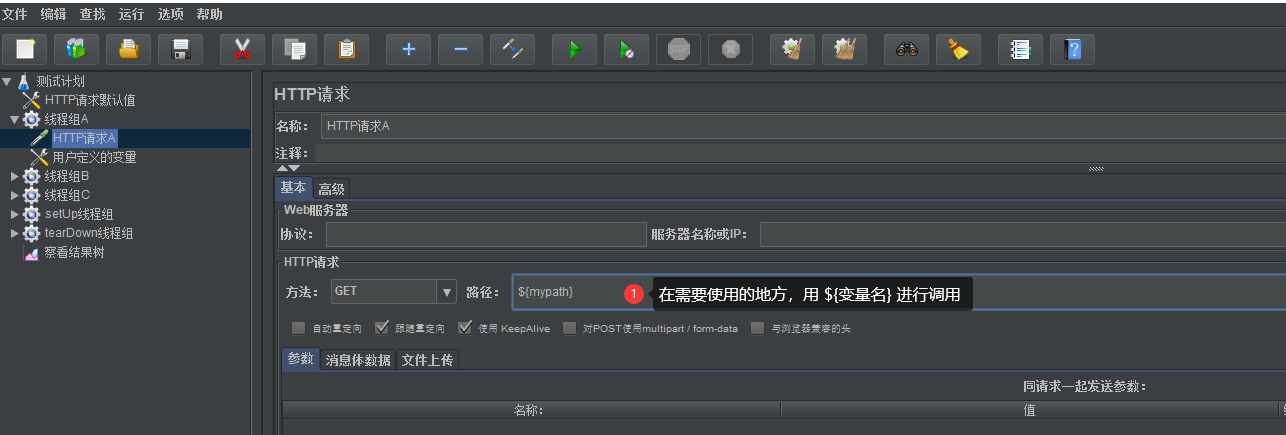




### 4、参数化-用户定义的变量





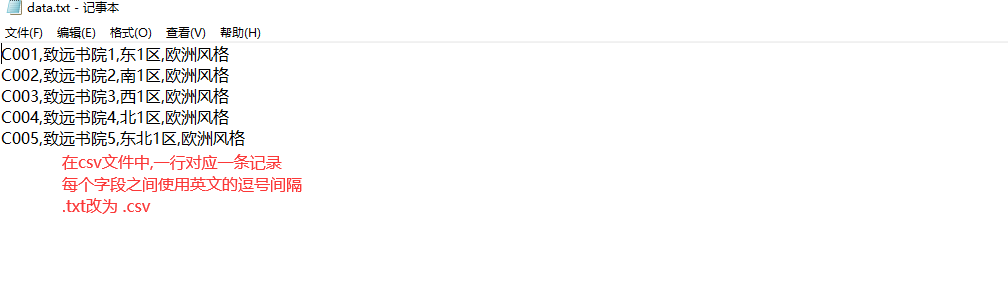


### 5、参数化-CSV数据文件

完成一个新增的脚本，新增数据成功

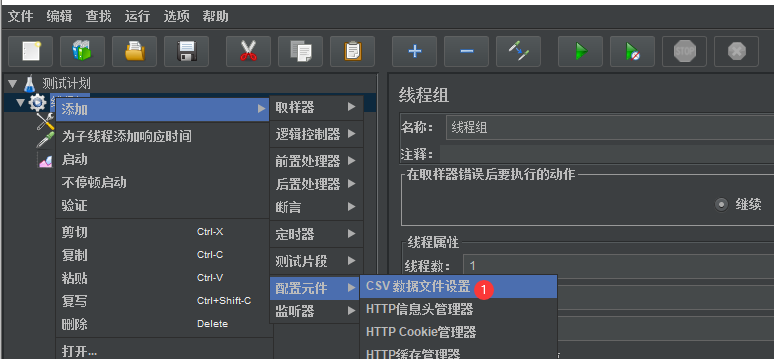


创建CSV文件



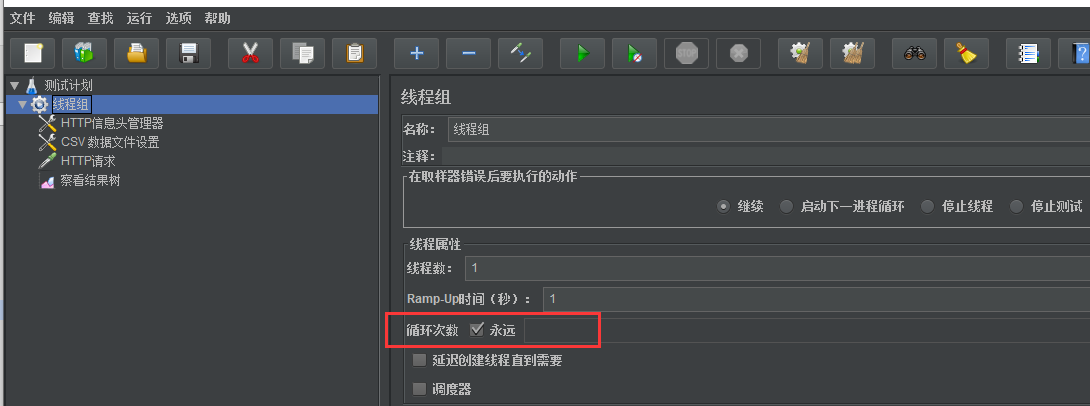
关联脚本与数据：

配置csv数据文件设置





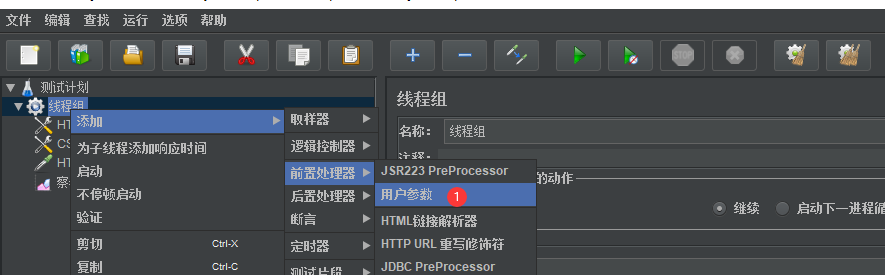
上面设置为4和5，线程数循环设置为永远（因为4和5会终止循环，并不会无线的循环下去），执行请求的时候，文件有几条记录就会自动执行几次请求，不需要手动设置循环次数



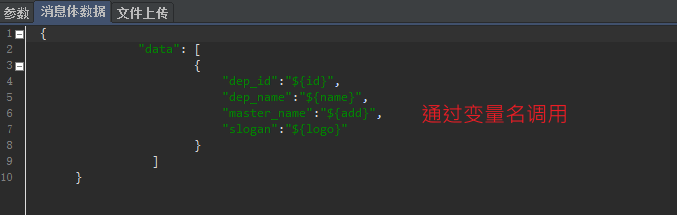
调用数据：

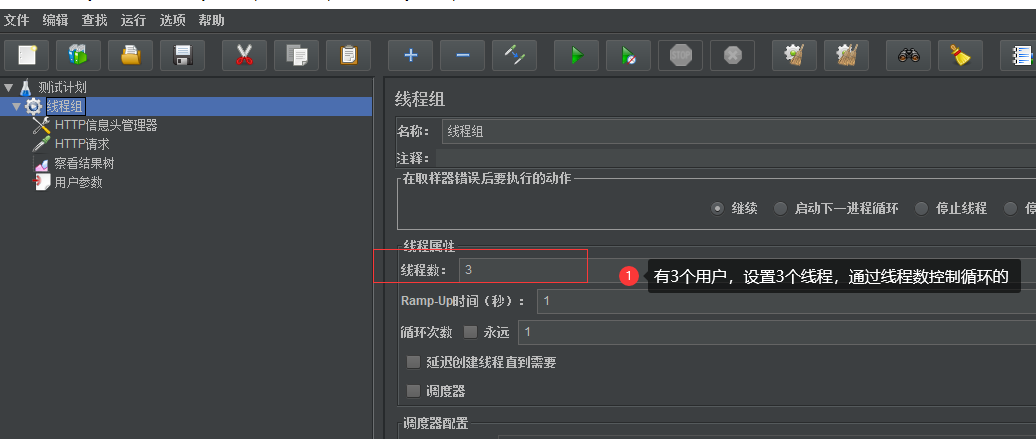


### 6、参数化-用户参数





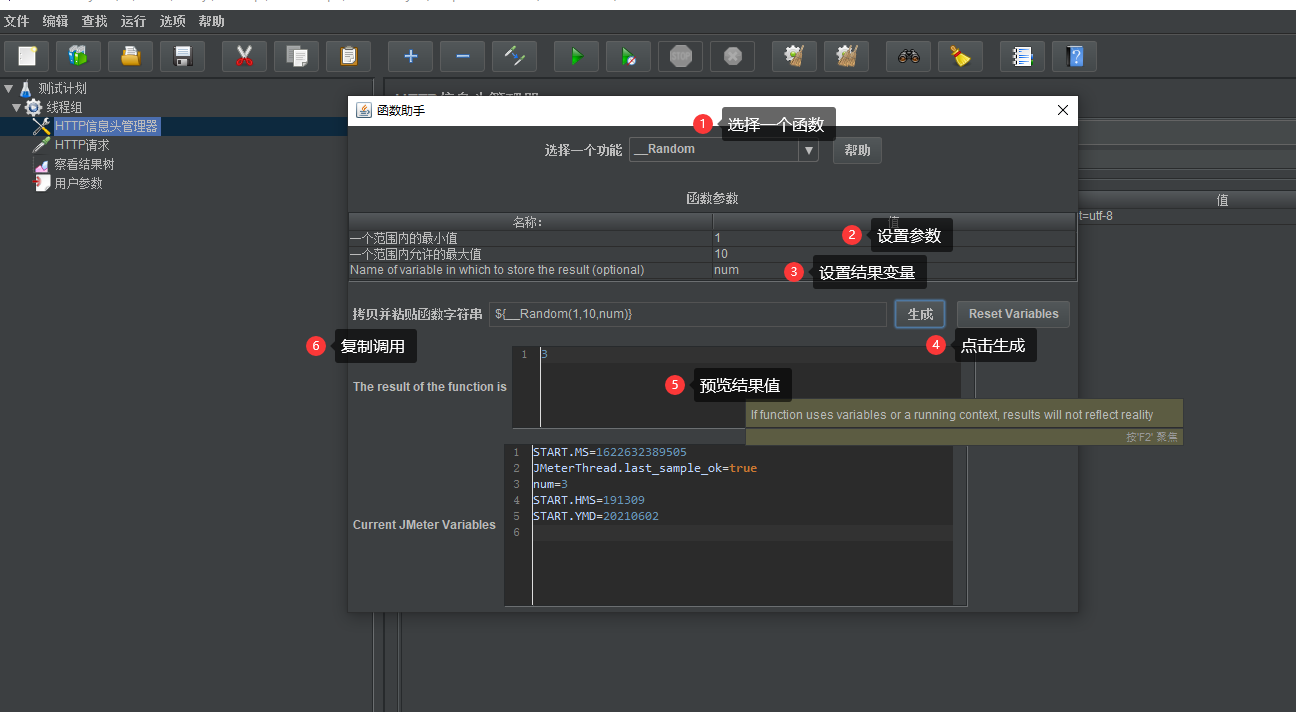


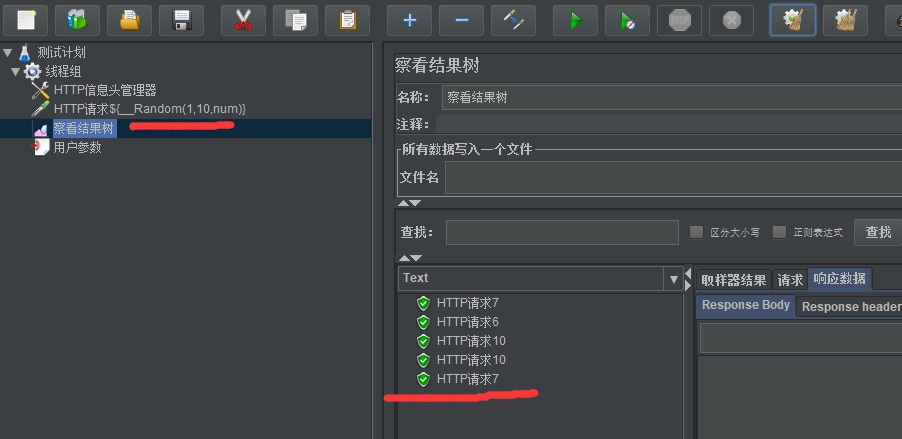


设置的线程数，如果大于用户数，会从头开始遍历用户，并赋值执行

### 7、参数化-函数



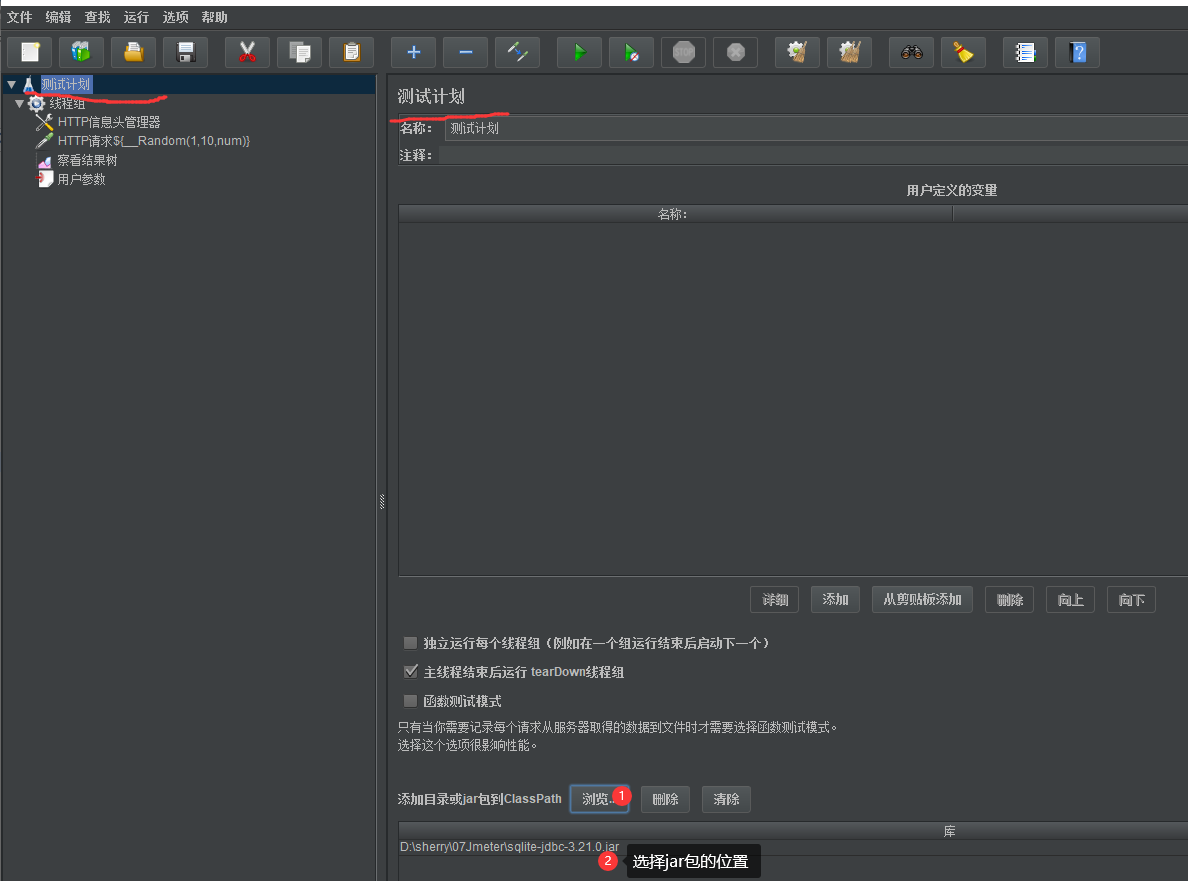


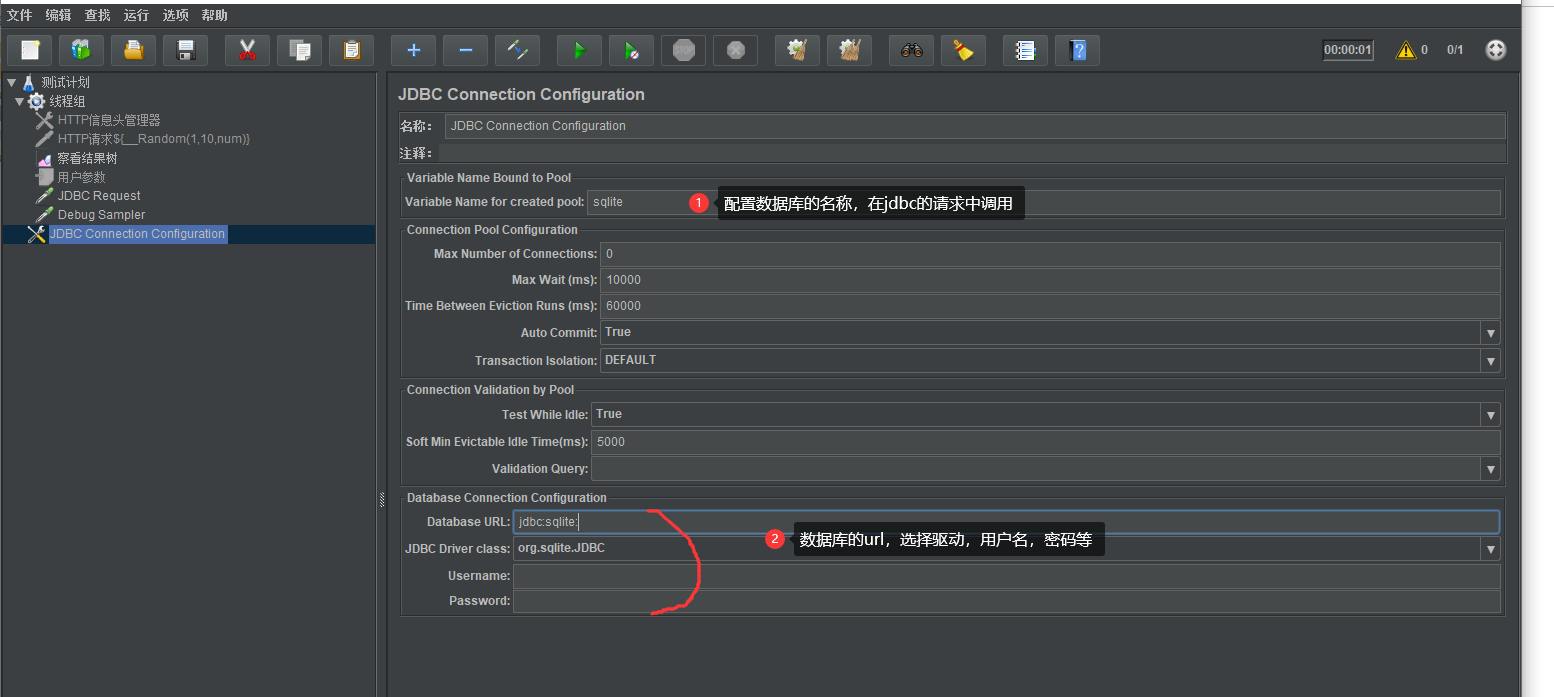


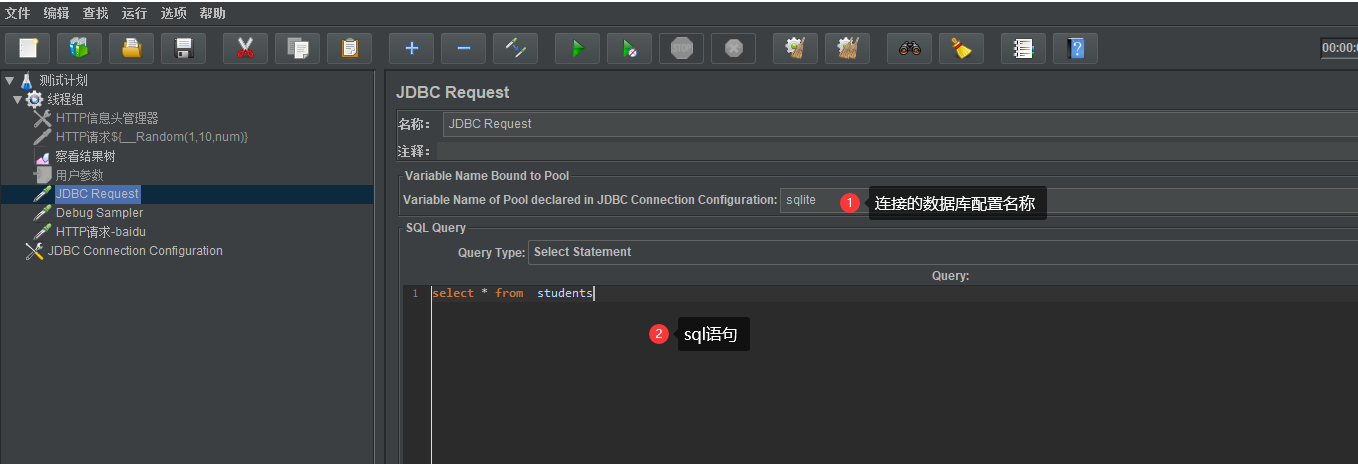
Counter计数器函数true（每个用户都有自己的计数器），false所有用户共用一个计数器

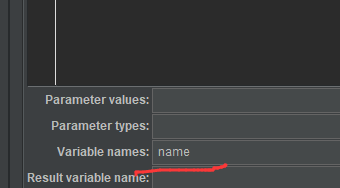
### 8、连接数据库

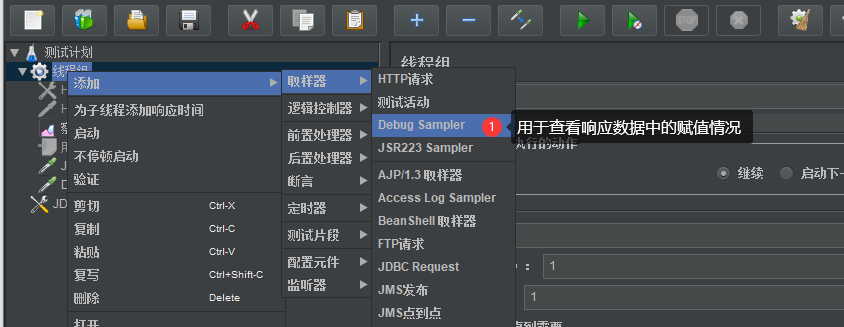
整合第三方jar包，选择mysql，oracle，sqlite等





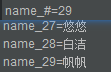


给查询到的结果命名

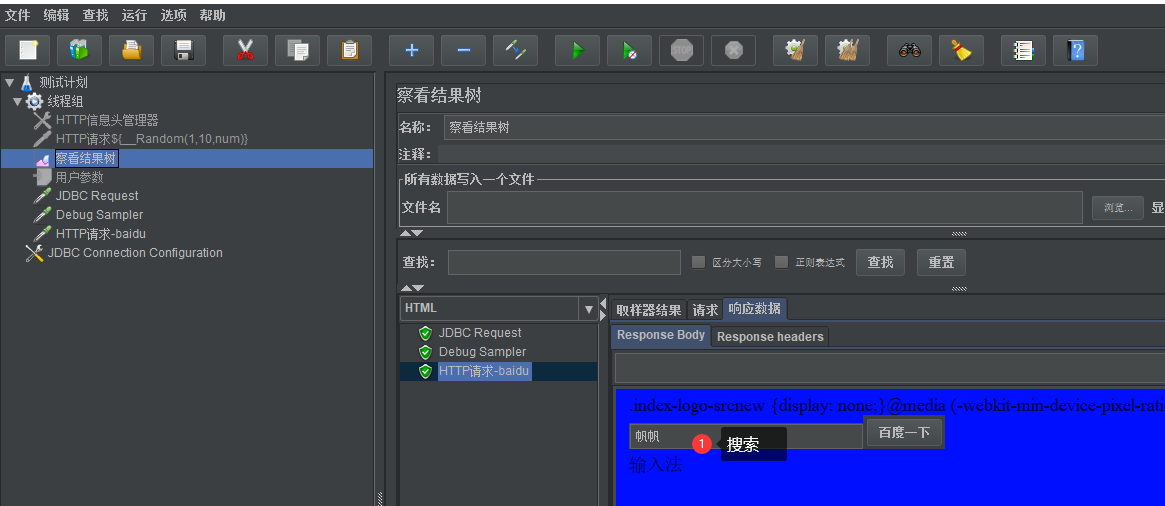


上面的组件可以查看结果数据的赋值规则

数据接收规则 变量名\_N=第N个结果， 变量名\_#=结果总数

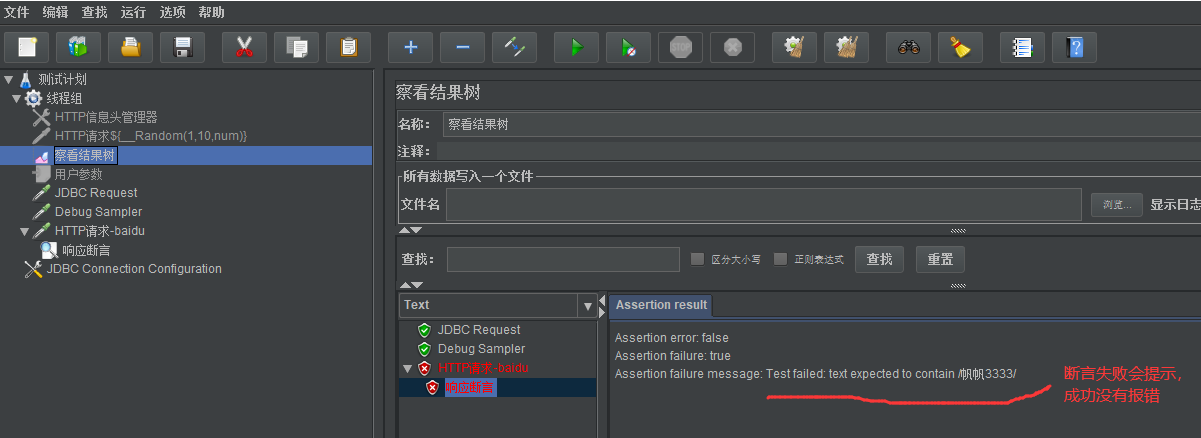




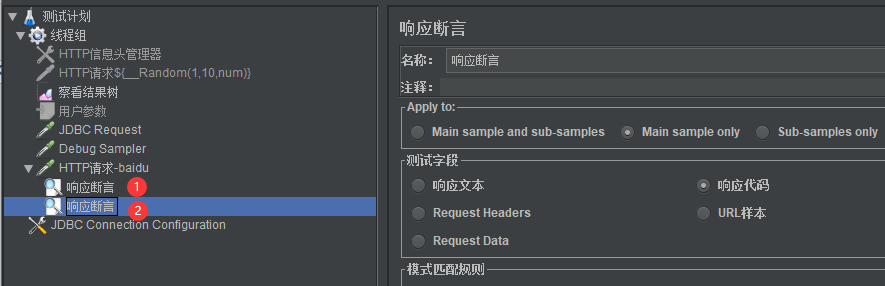


### 9、响应断言

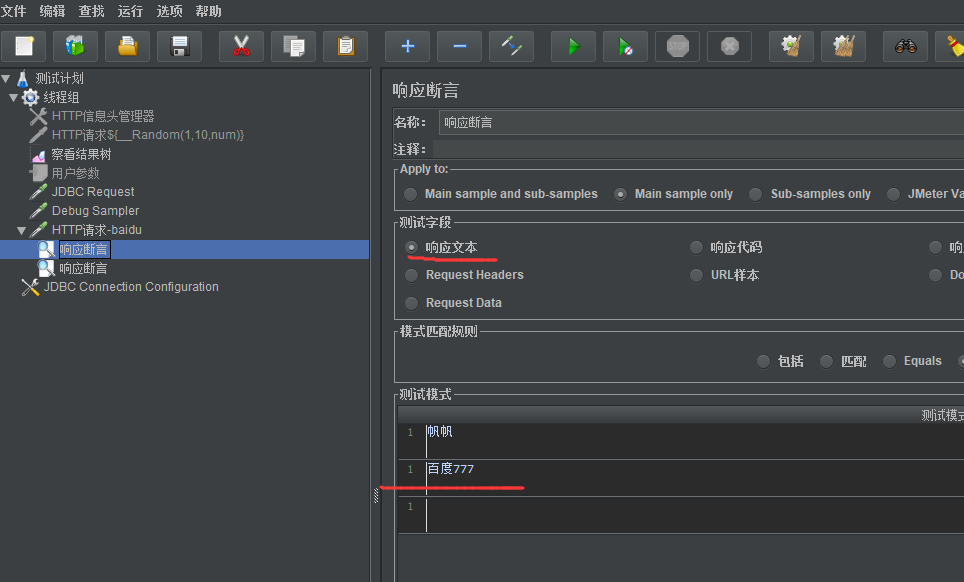


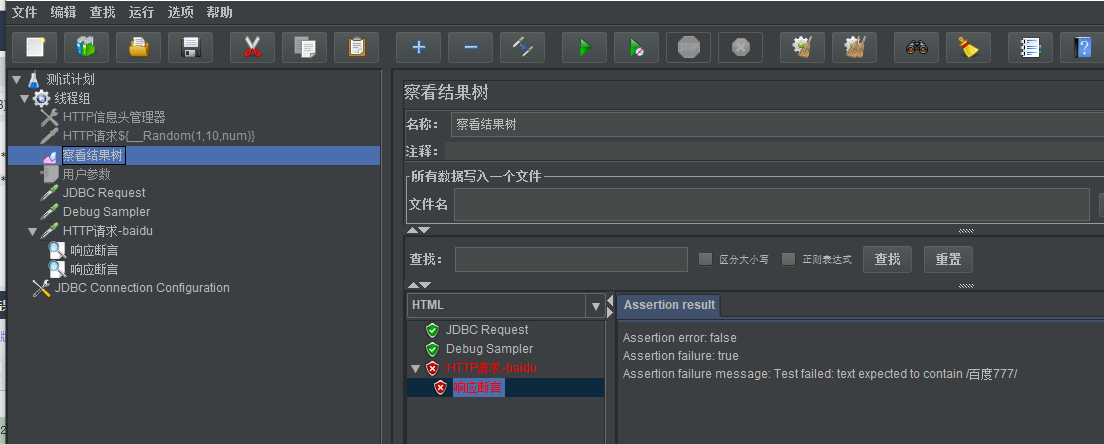


可以对一个请求添加多个响应断言

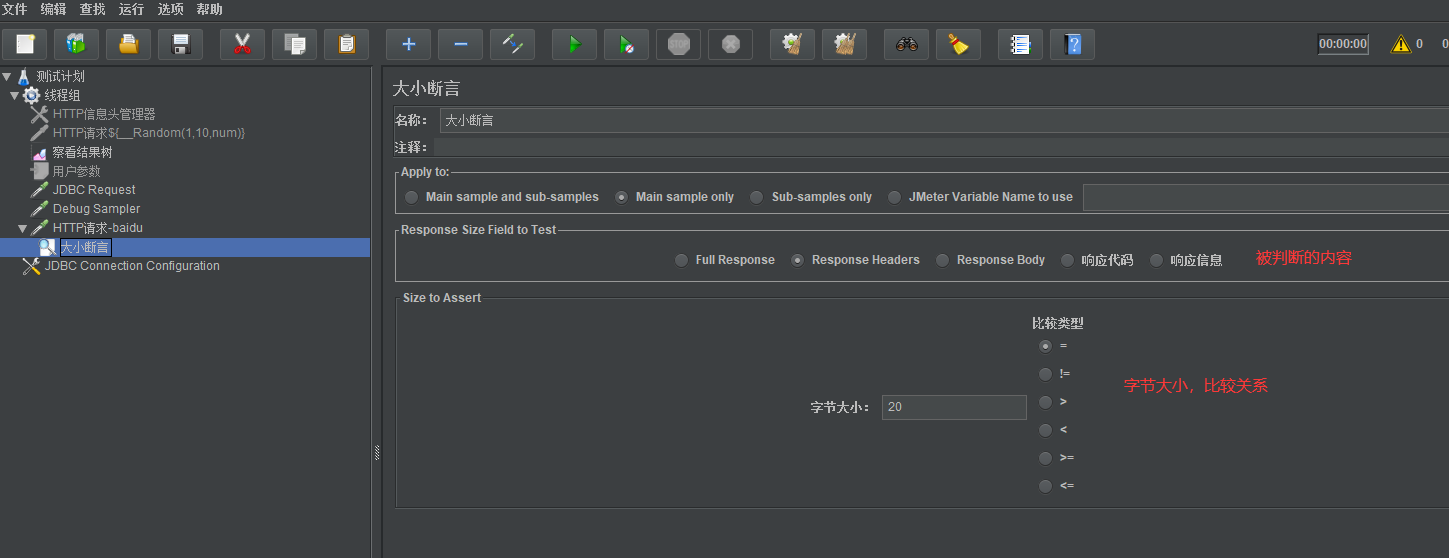


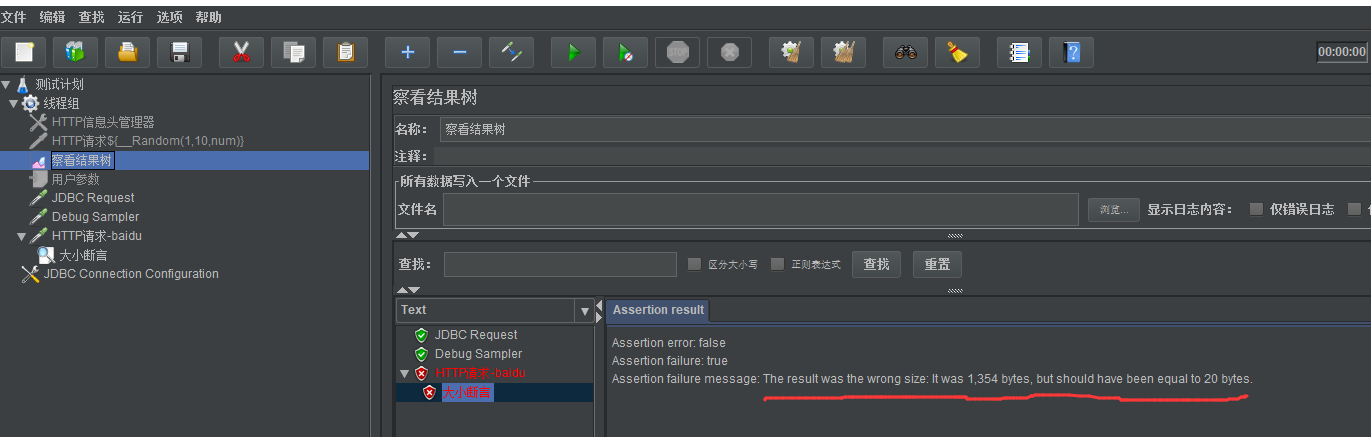
断言的内容可以添加多条



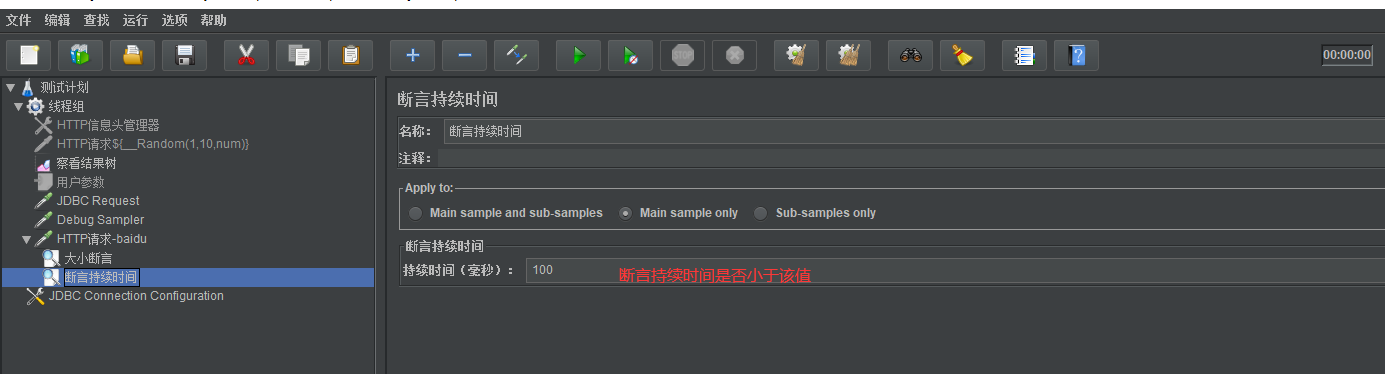


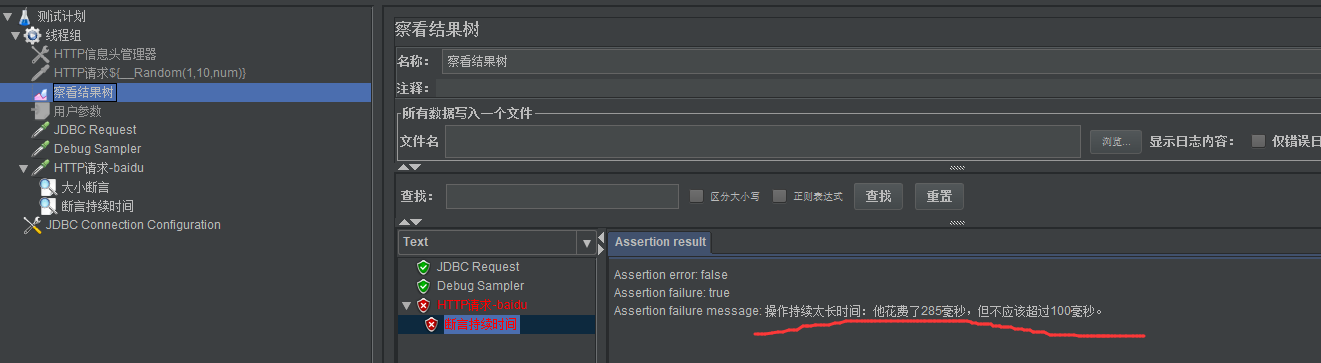
### 10、大小断言





### 11、断言持续时间



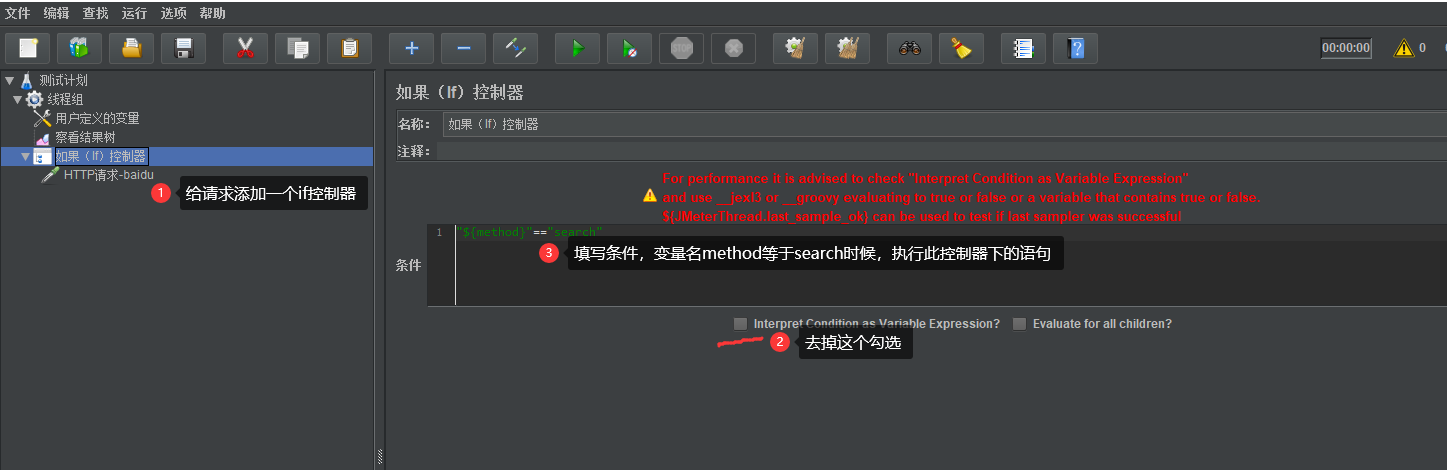


### 逻辑控制器

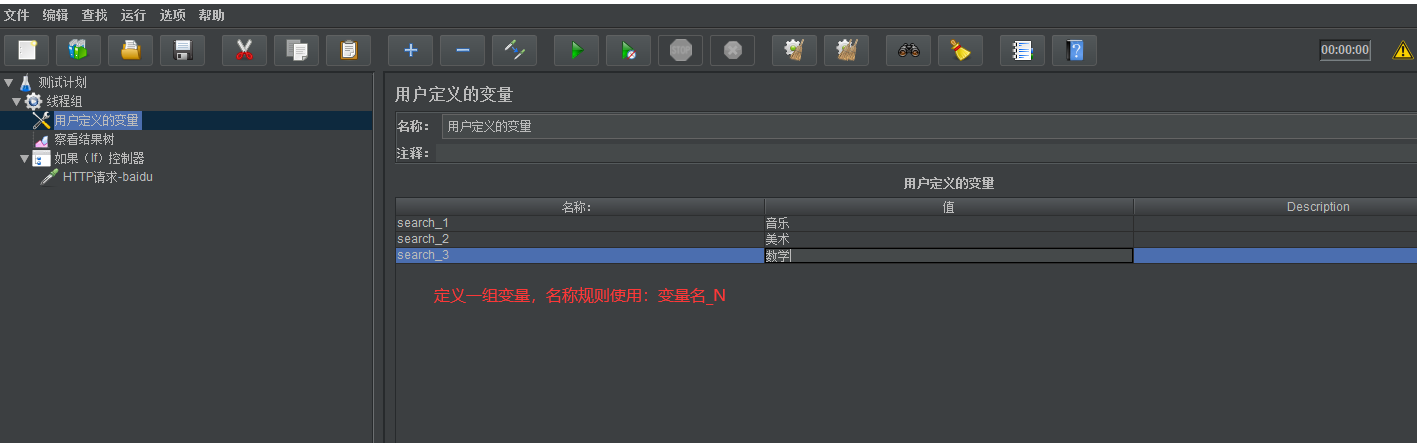
#### 12、If

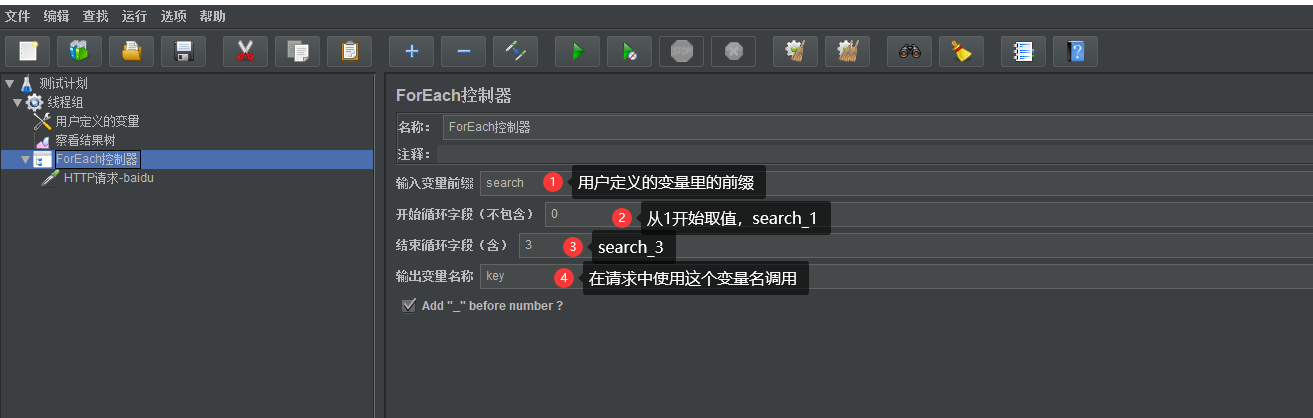
定义一个用户变量，如果是 search，则访问baidu

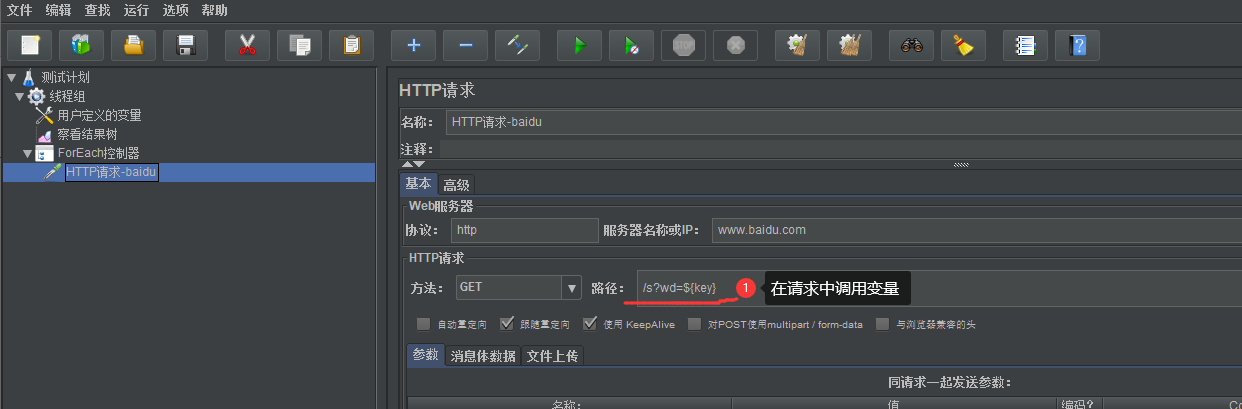




#### 13、Foreach

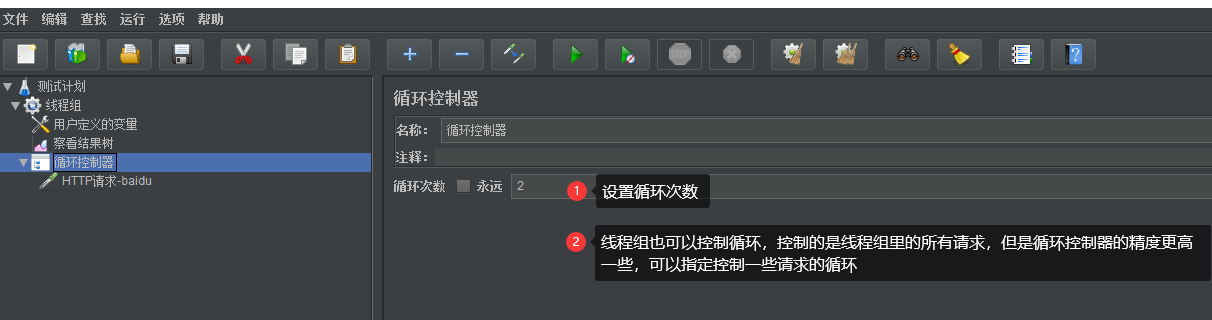








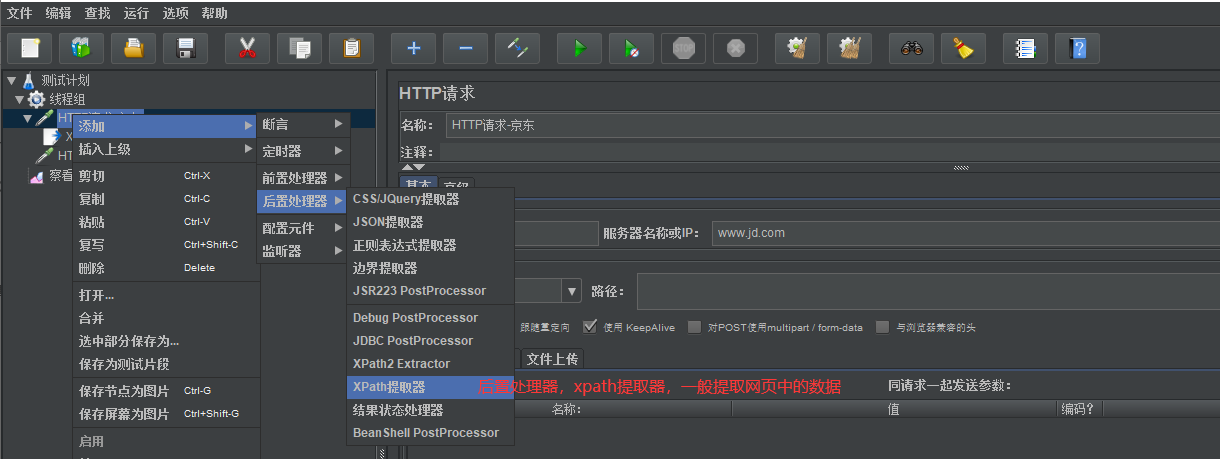
#### 14、循环控制器（while）



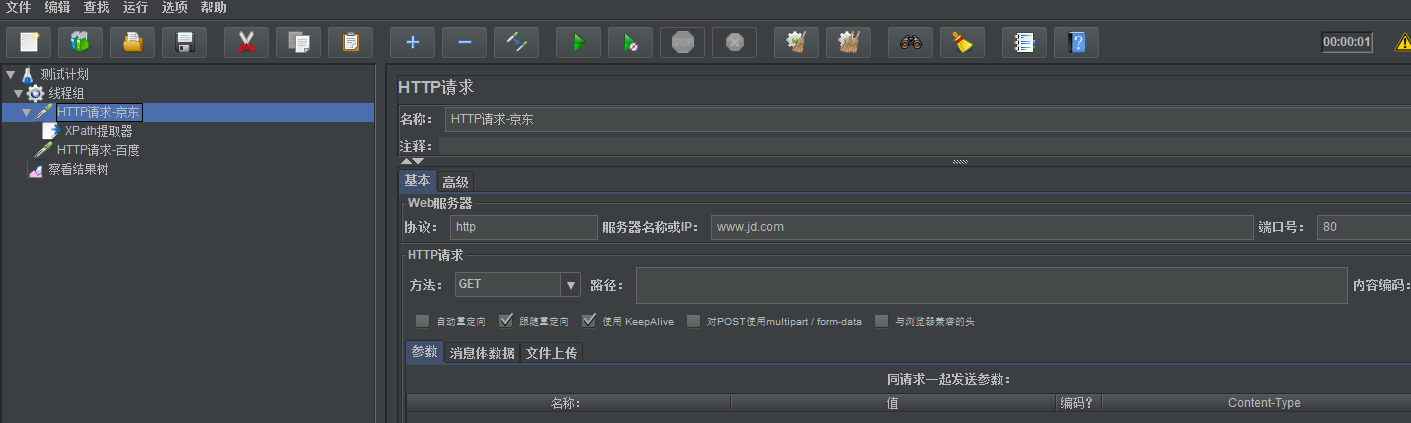
### 线程组内关联

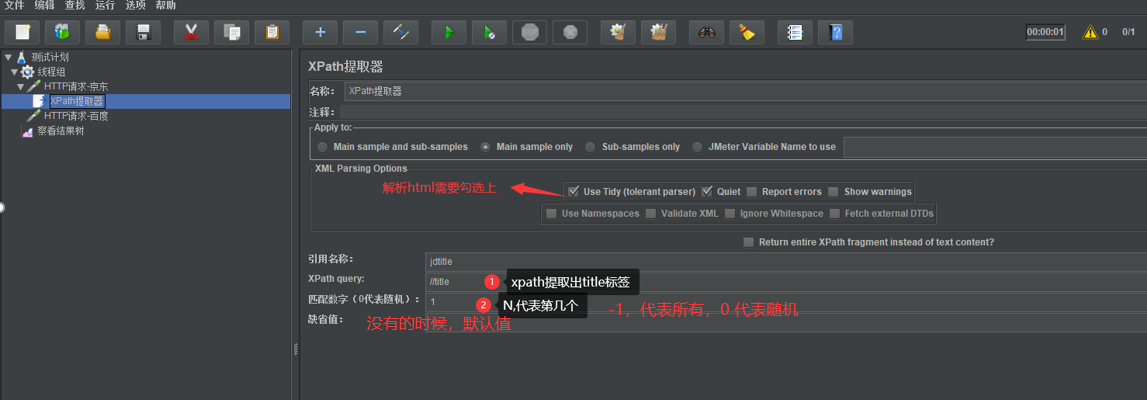
上一个请求返回的数据作为下一个请求的参数，会使用到关联

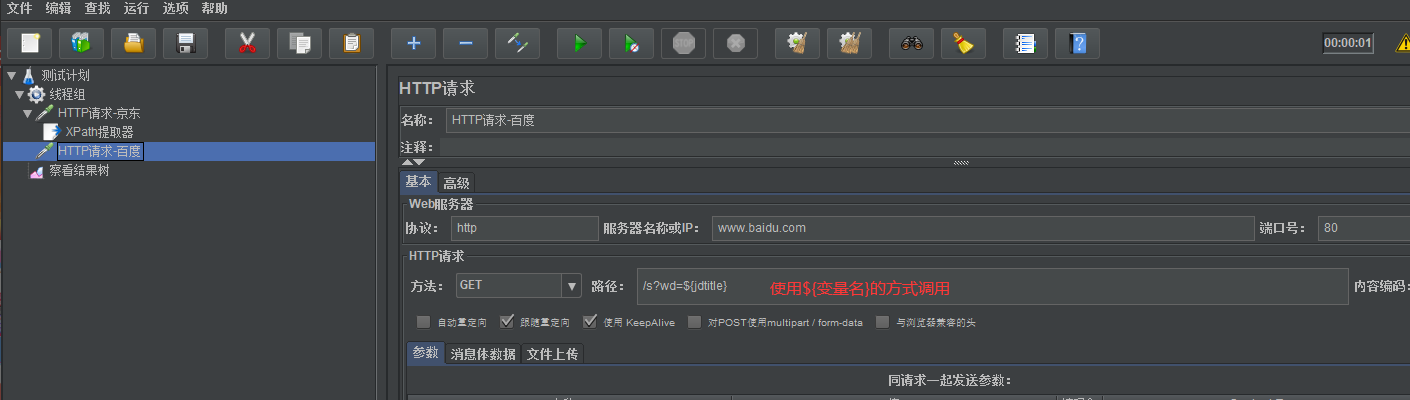
#### 15、后置处理器-xpath



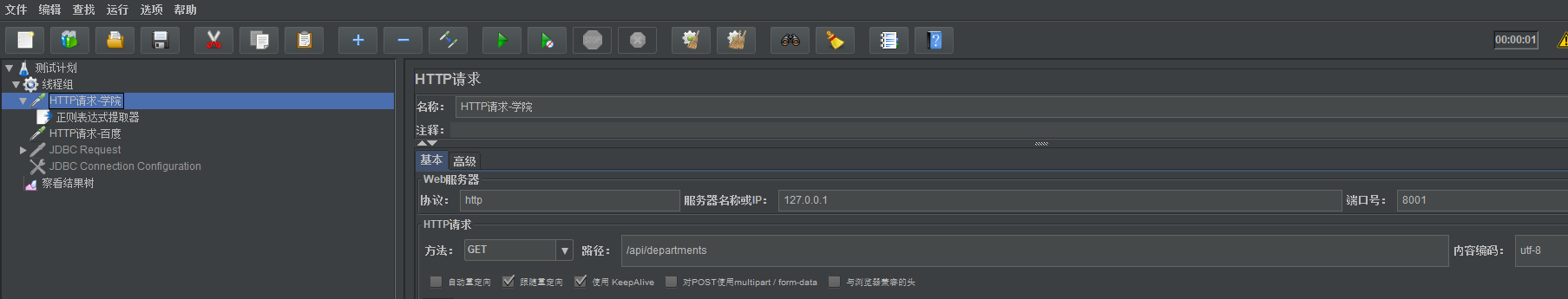
访问京东，并将京东的title值传递到百度搜索





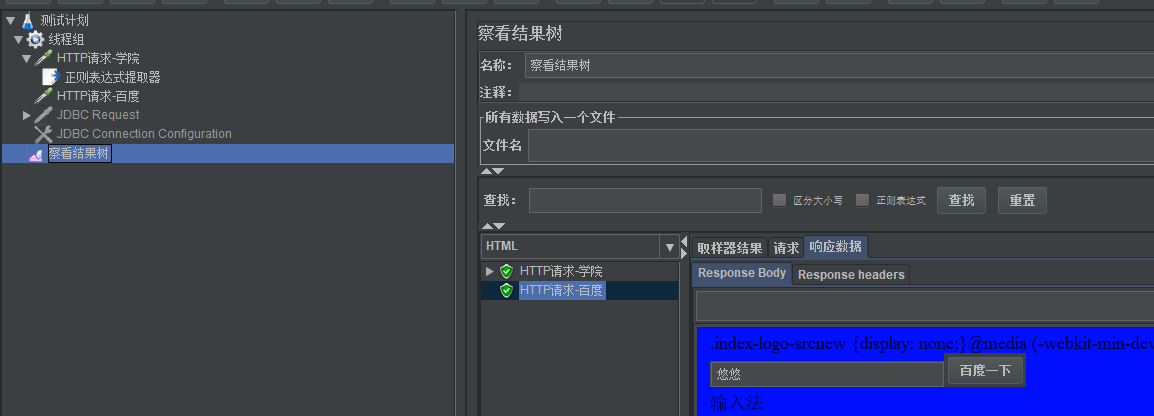


#### 16、后置处理器-正则表达式



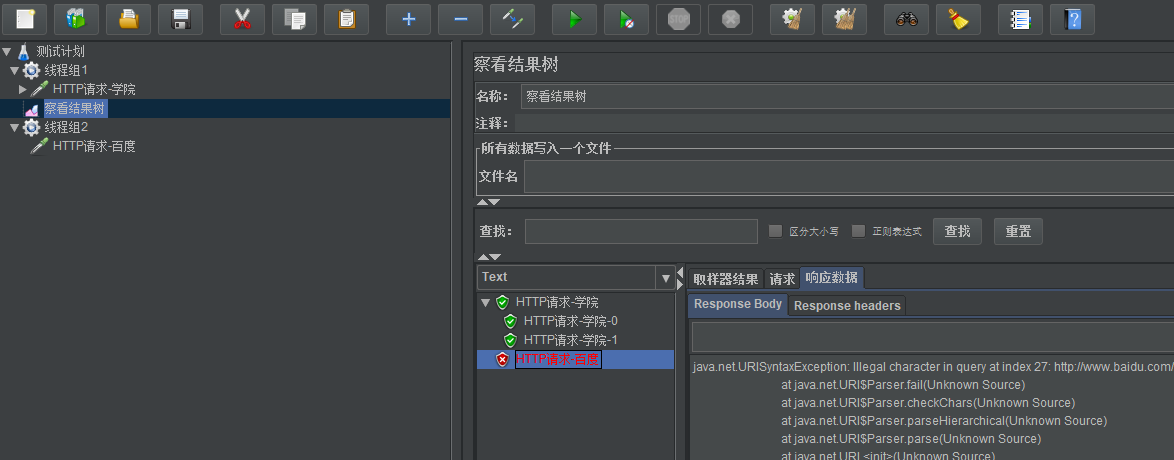
如果从数据库中请求的数据，直接使用变量赋值即可，见基本组件-连接数据库操作





#### 17、跨线程组关联

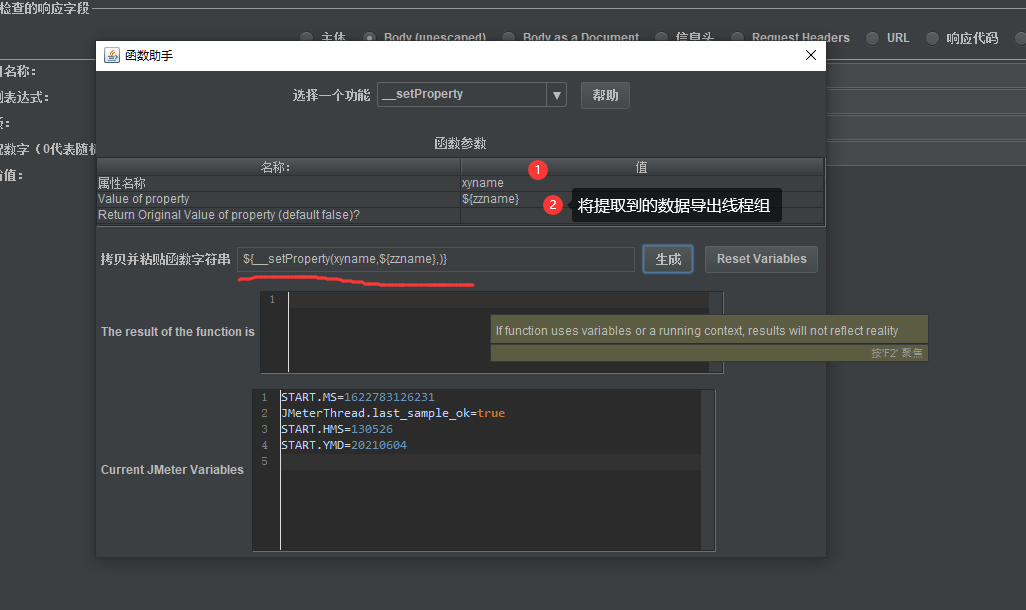
将以上2个请求放到2个线程组内，且独立运行每个线程组，保证第一个请求先运行，但是报错了

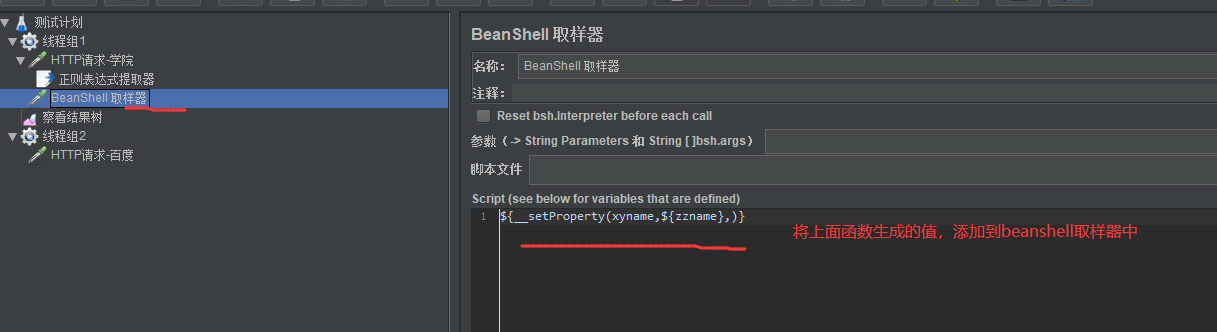


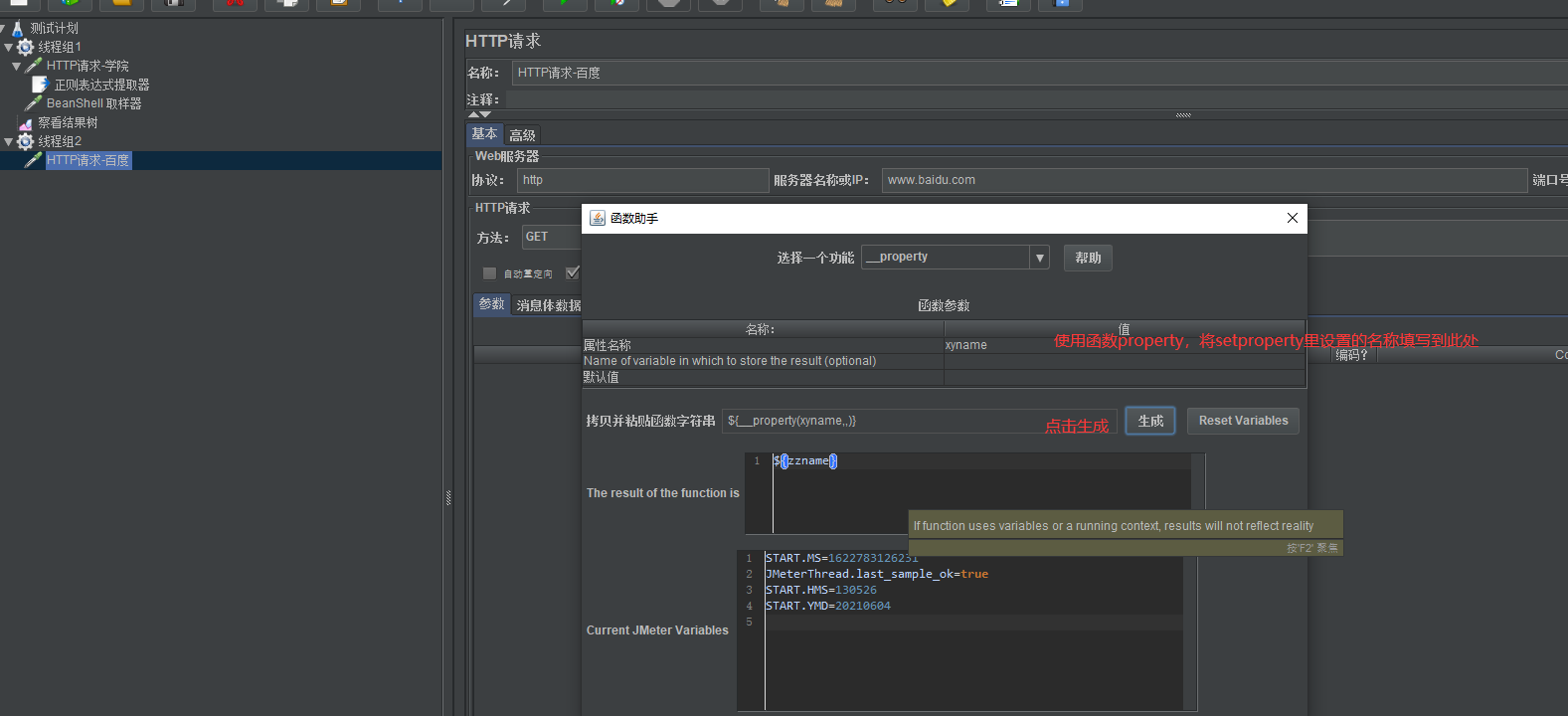
提取器提取出的变量，作用域只限于当前线程组内

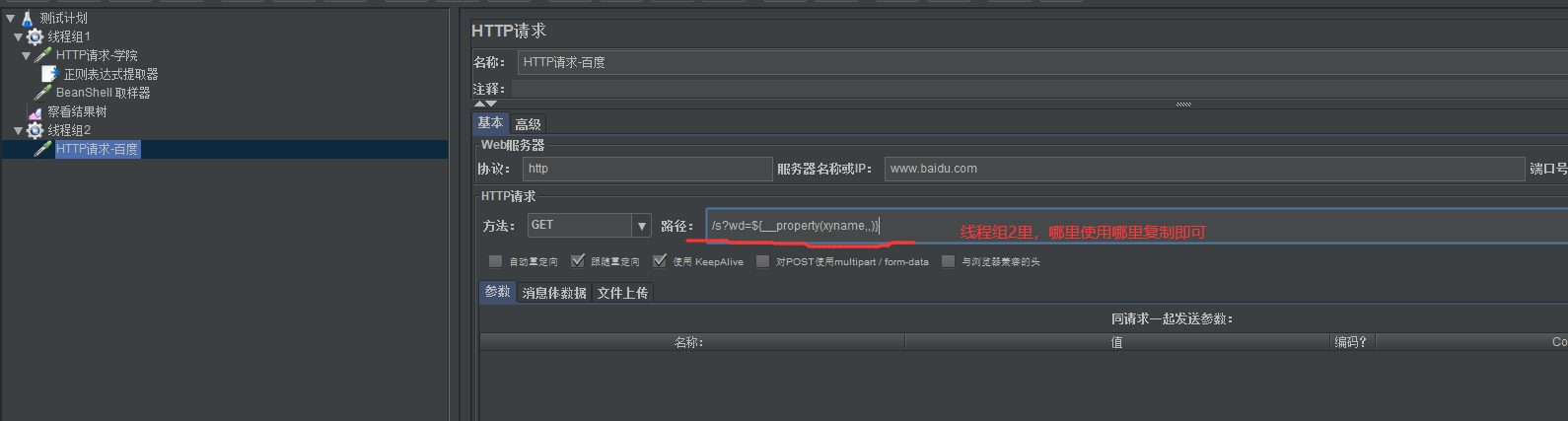


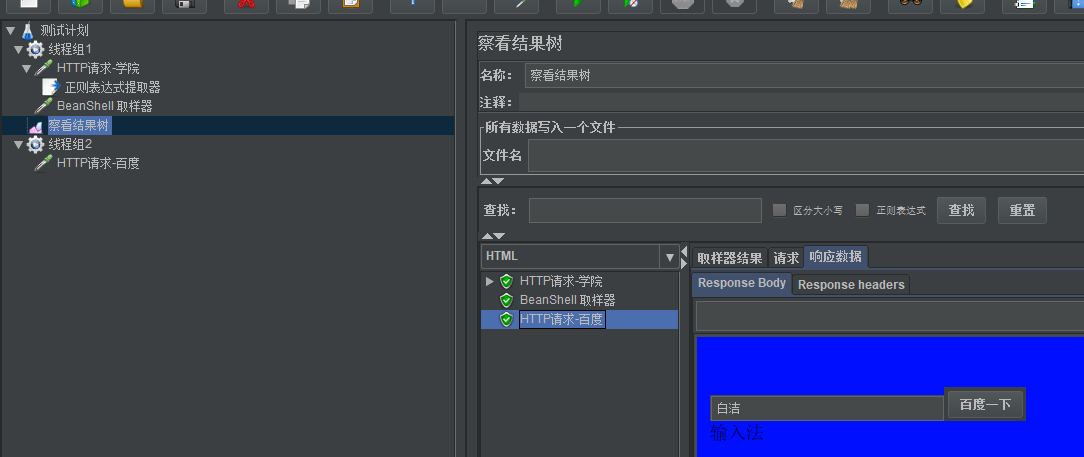
将提取的字段导出线程组，导入到公共的空间中







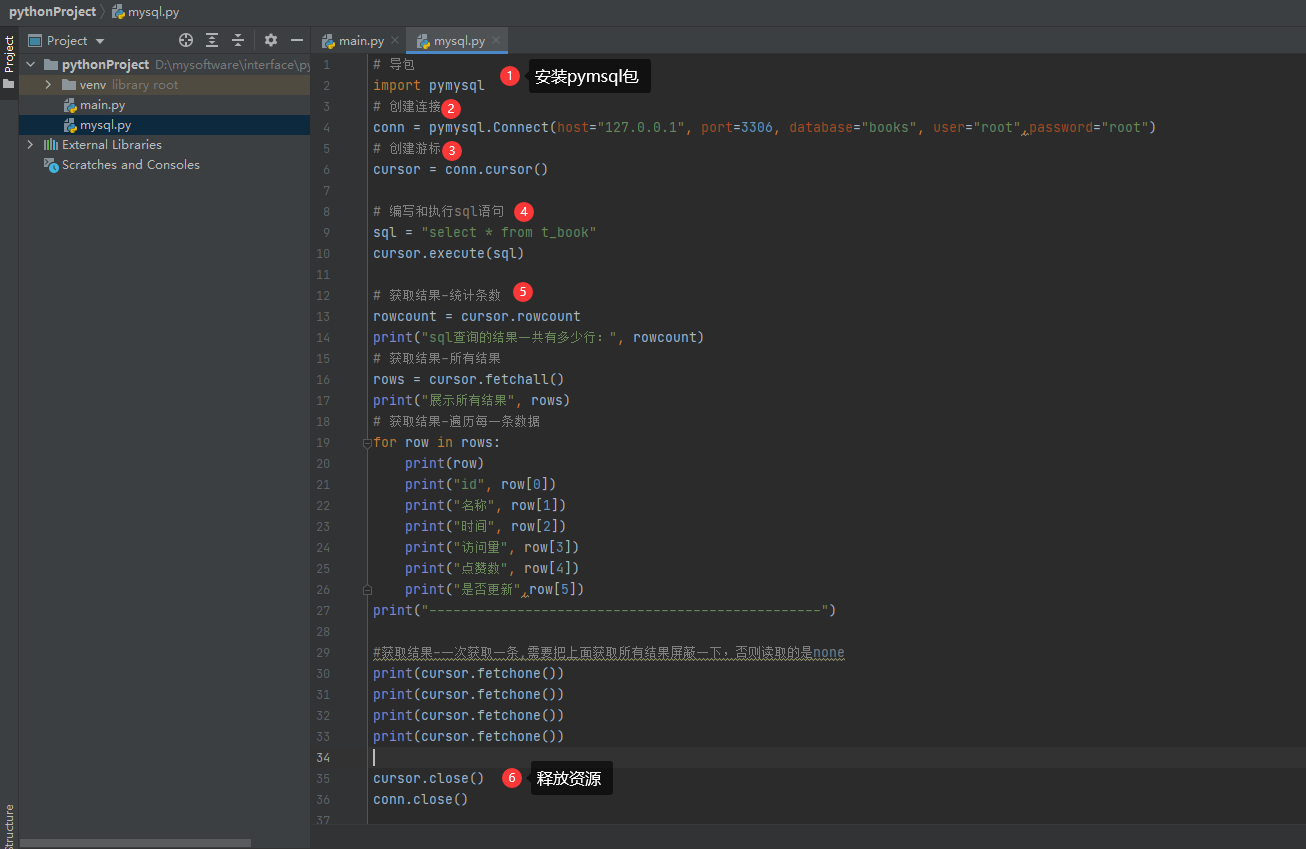




# PyMySQL的使用

## 基本使用步骤如图

### 查询



### 新增



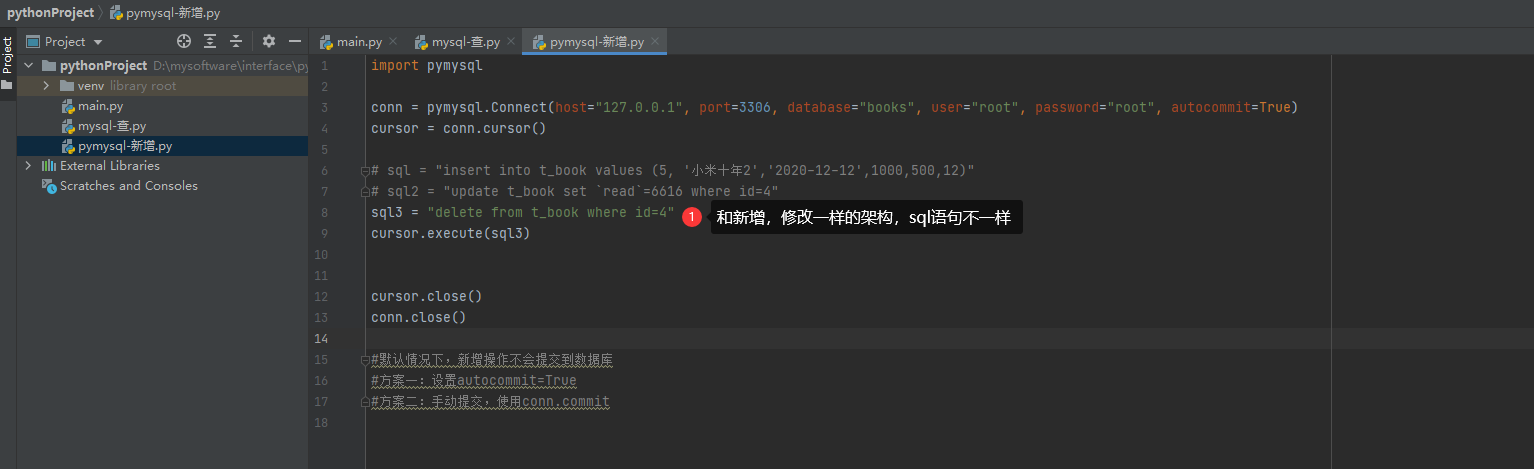
### 修改



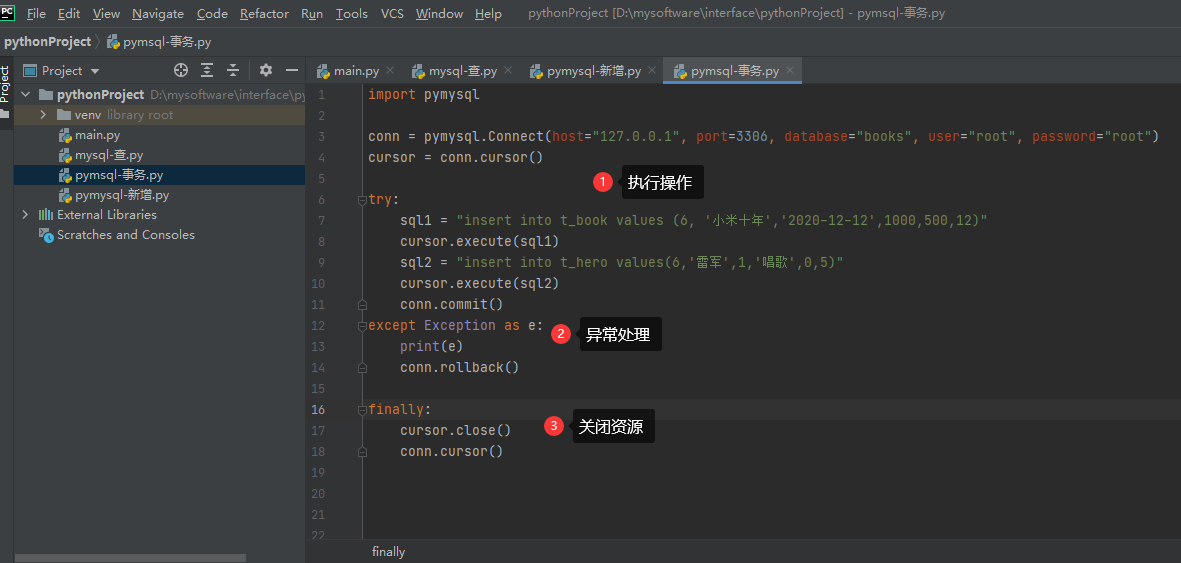
sql2 = "update t\_book set `read`=6616 where id=4"

注意read使用的不是英文引号，是这个符号~上的键盘标识

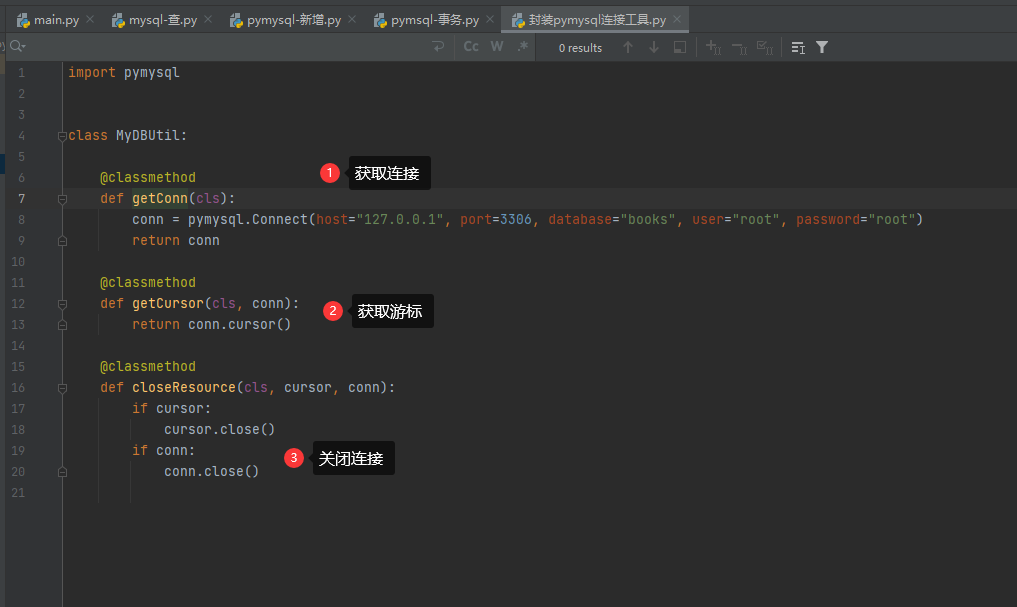
### 删除



事务：



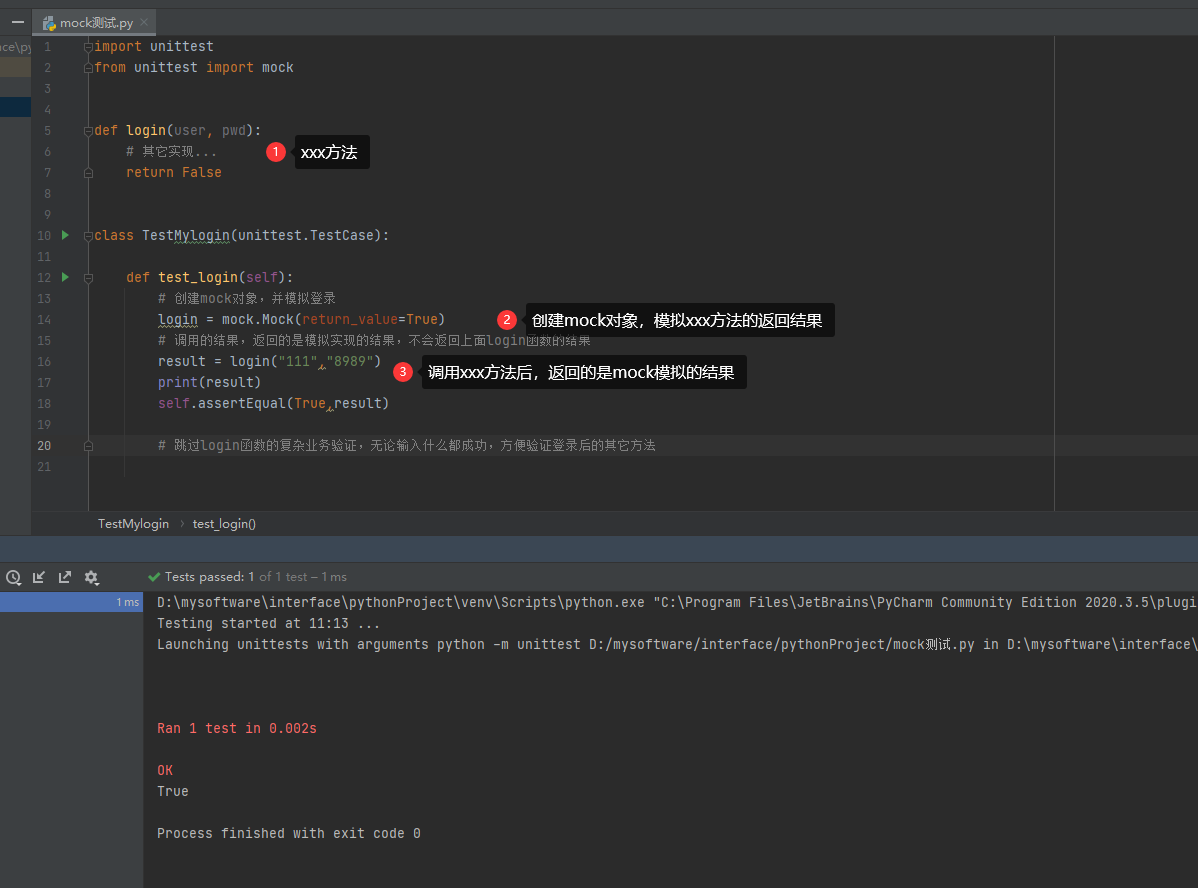
## 封装pymysq的连接



调用



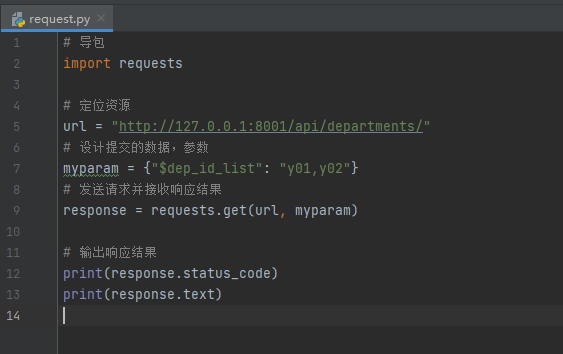
Mock的使用



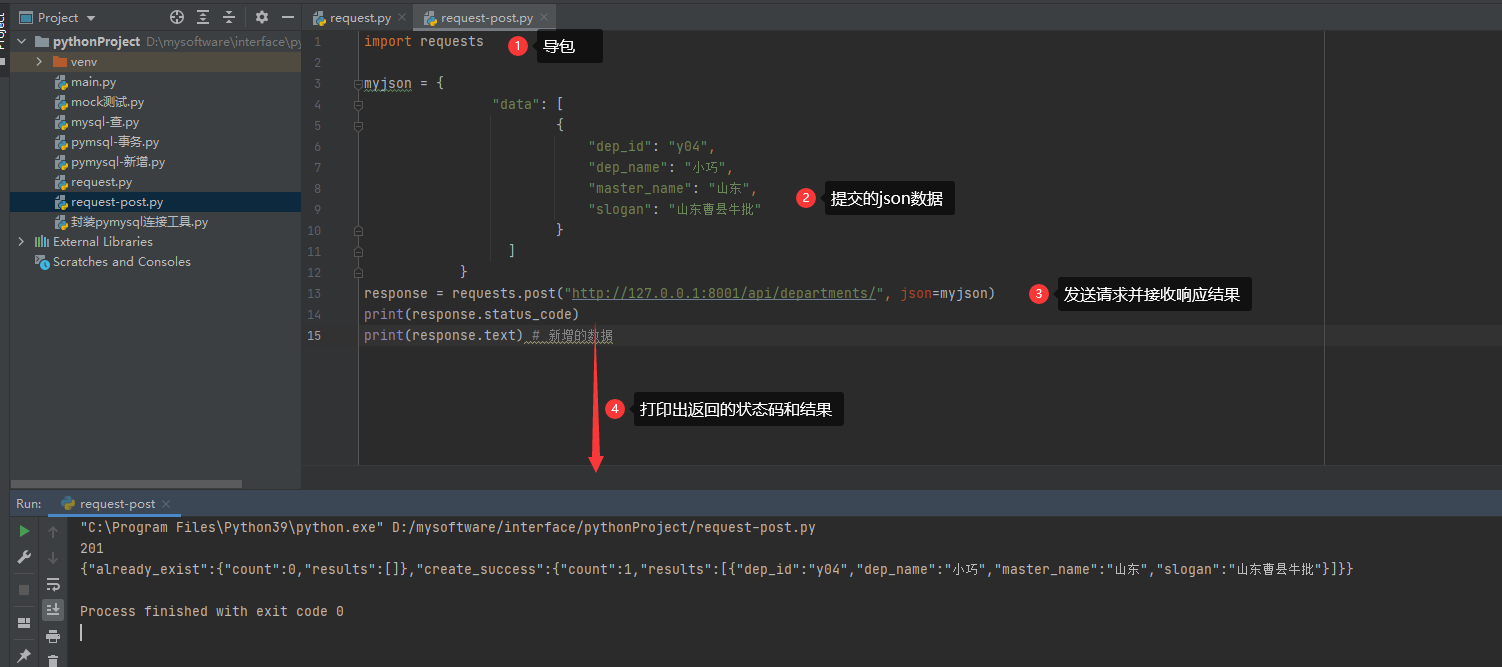
# Requests库的使用

## 基本使用如图

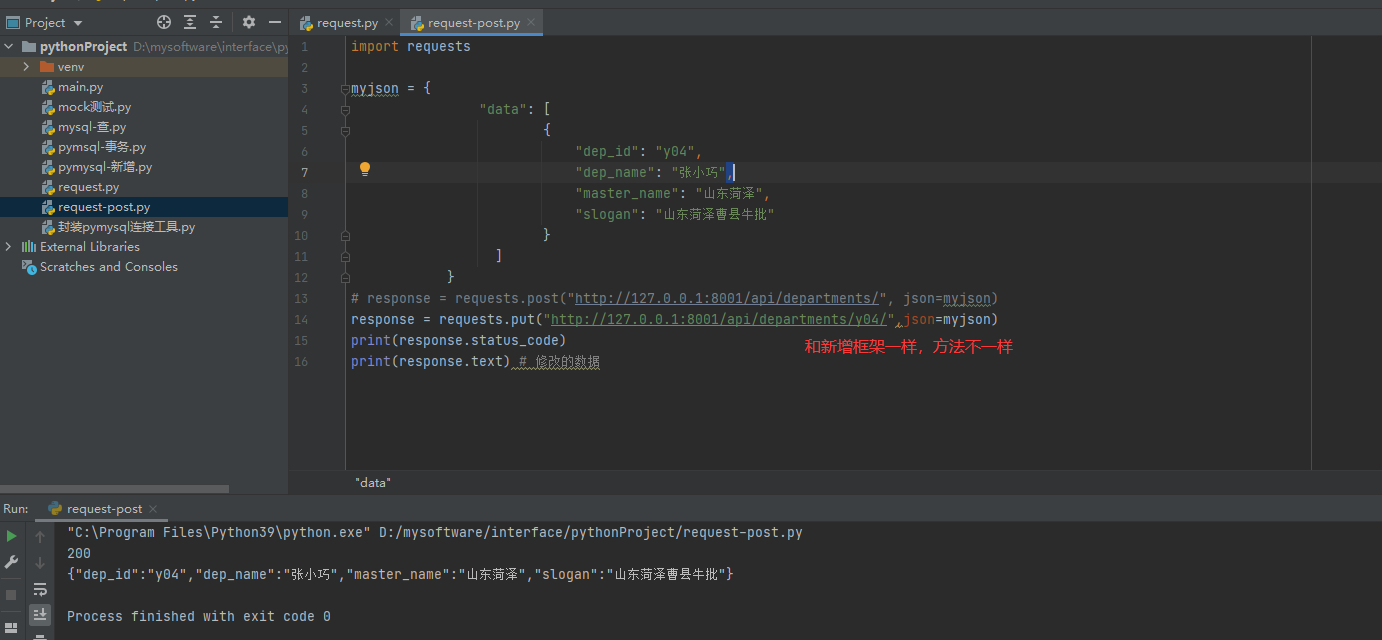
### 请求-get



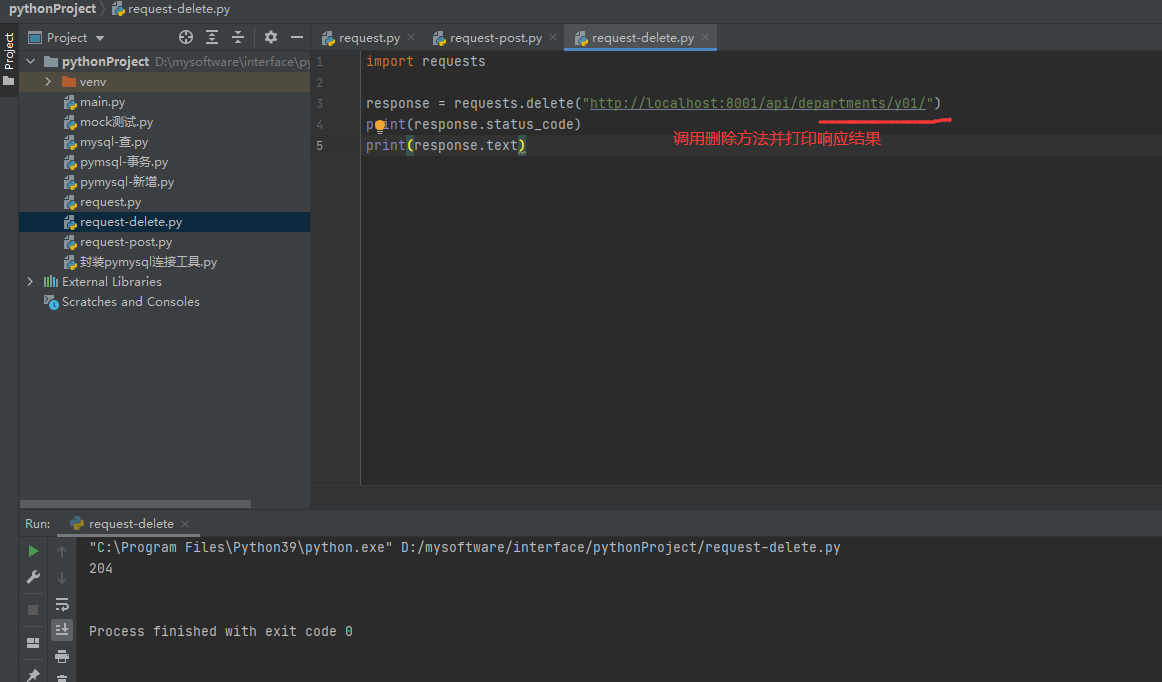
### 请求-post



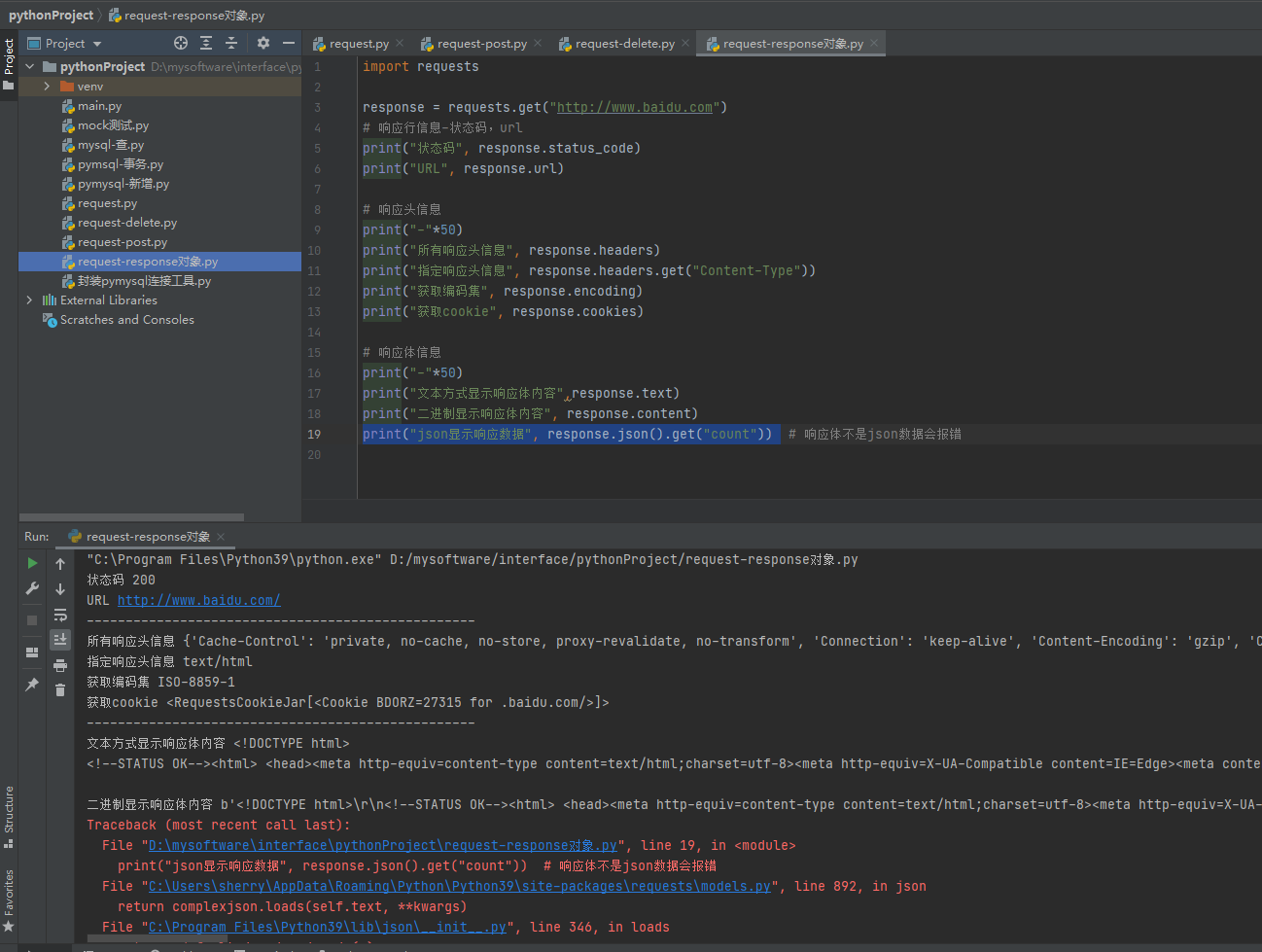
### 请求-put

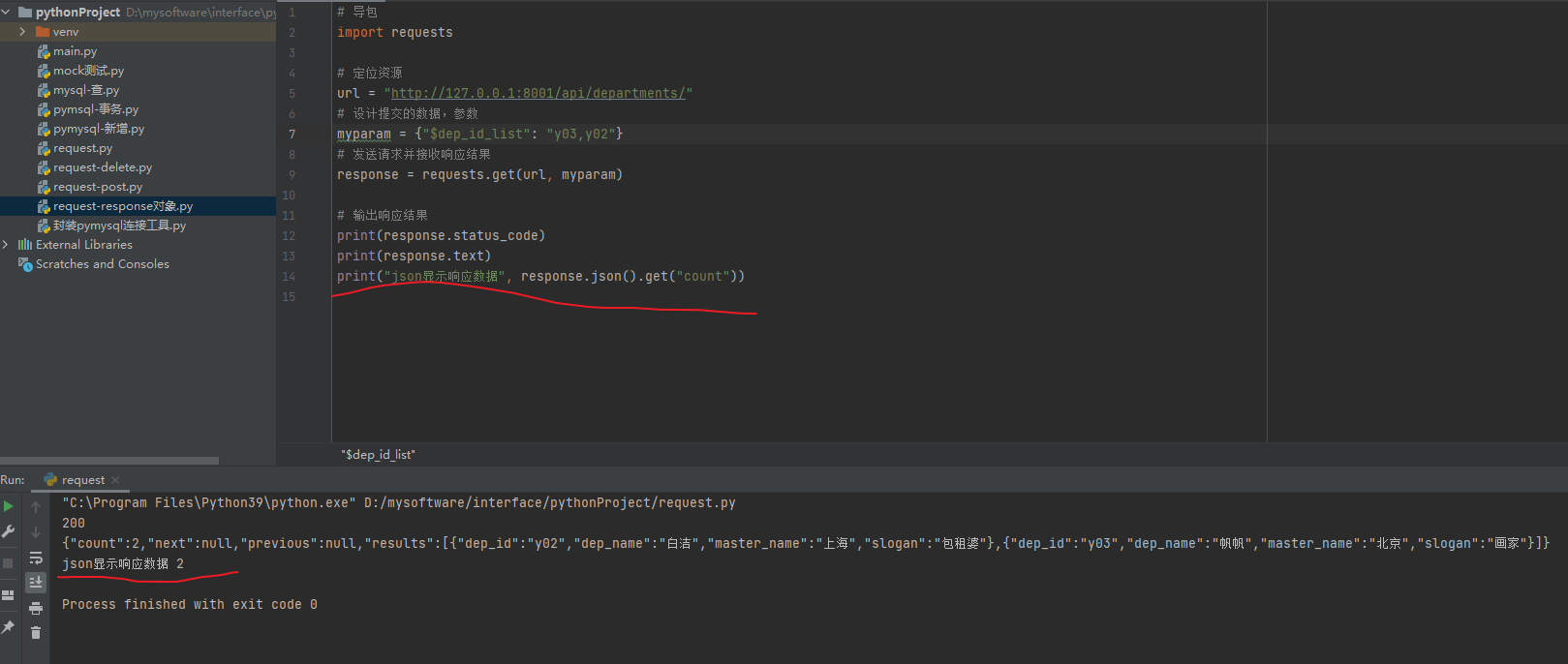


### 请求-delete

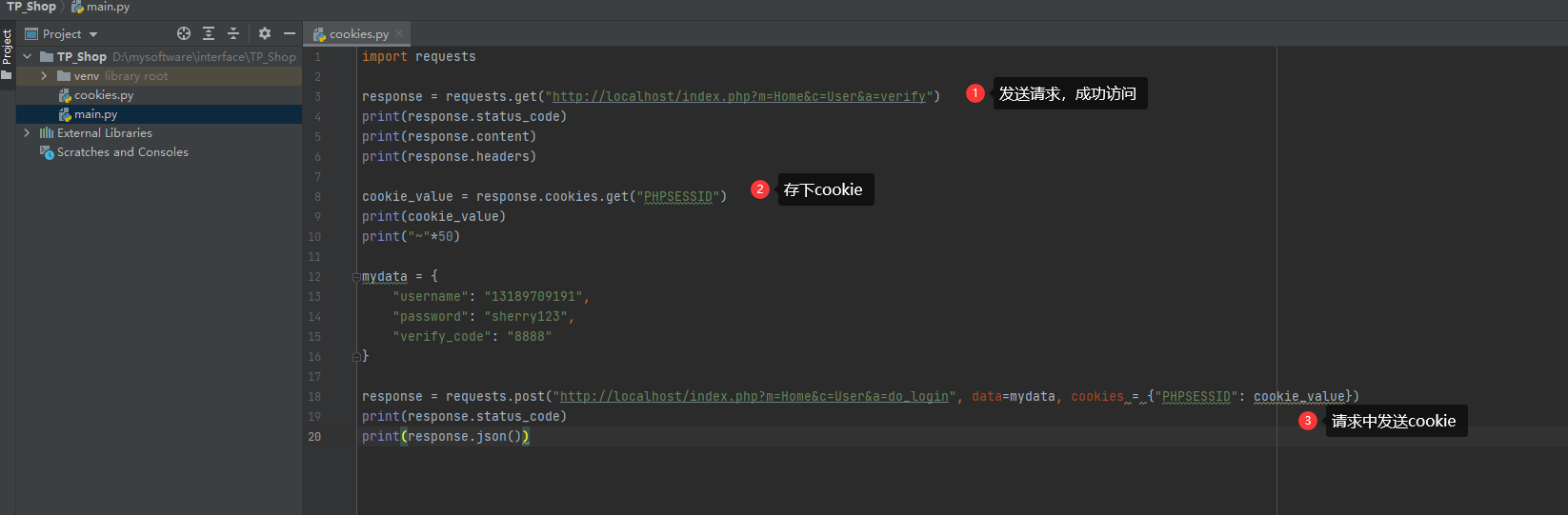


## Response对象可以获取的结果

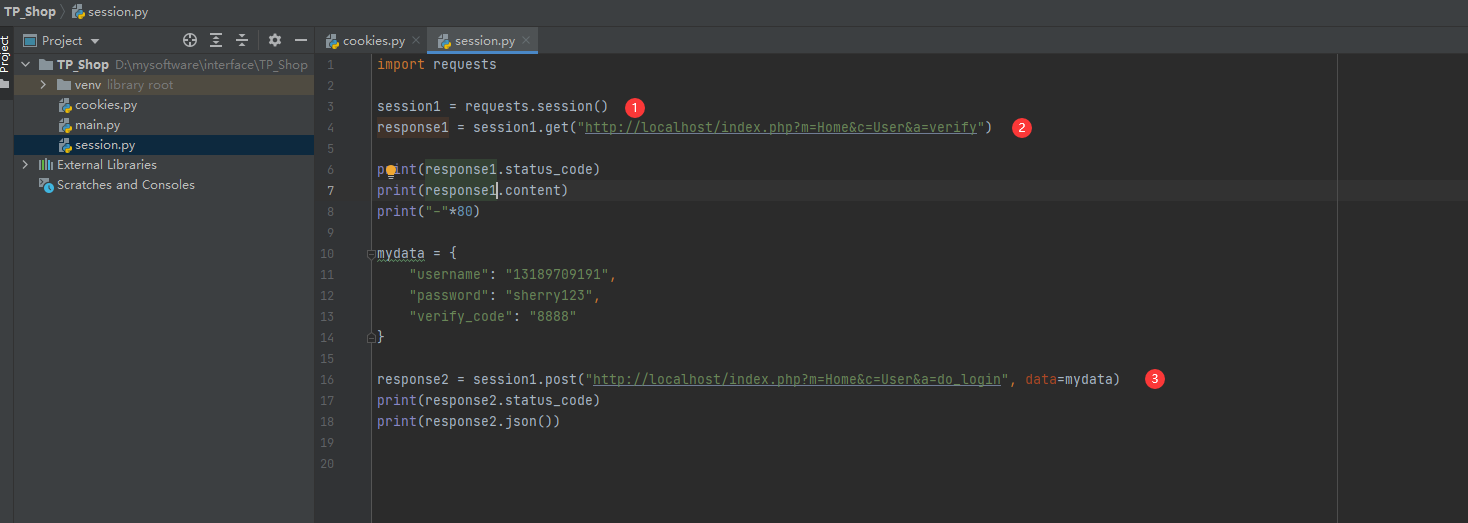




### Cookie



### Session



### Requests and unittest

