**页面说明：**

从首页的左侧，进入项目详情页

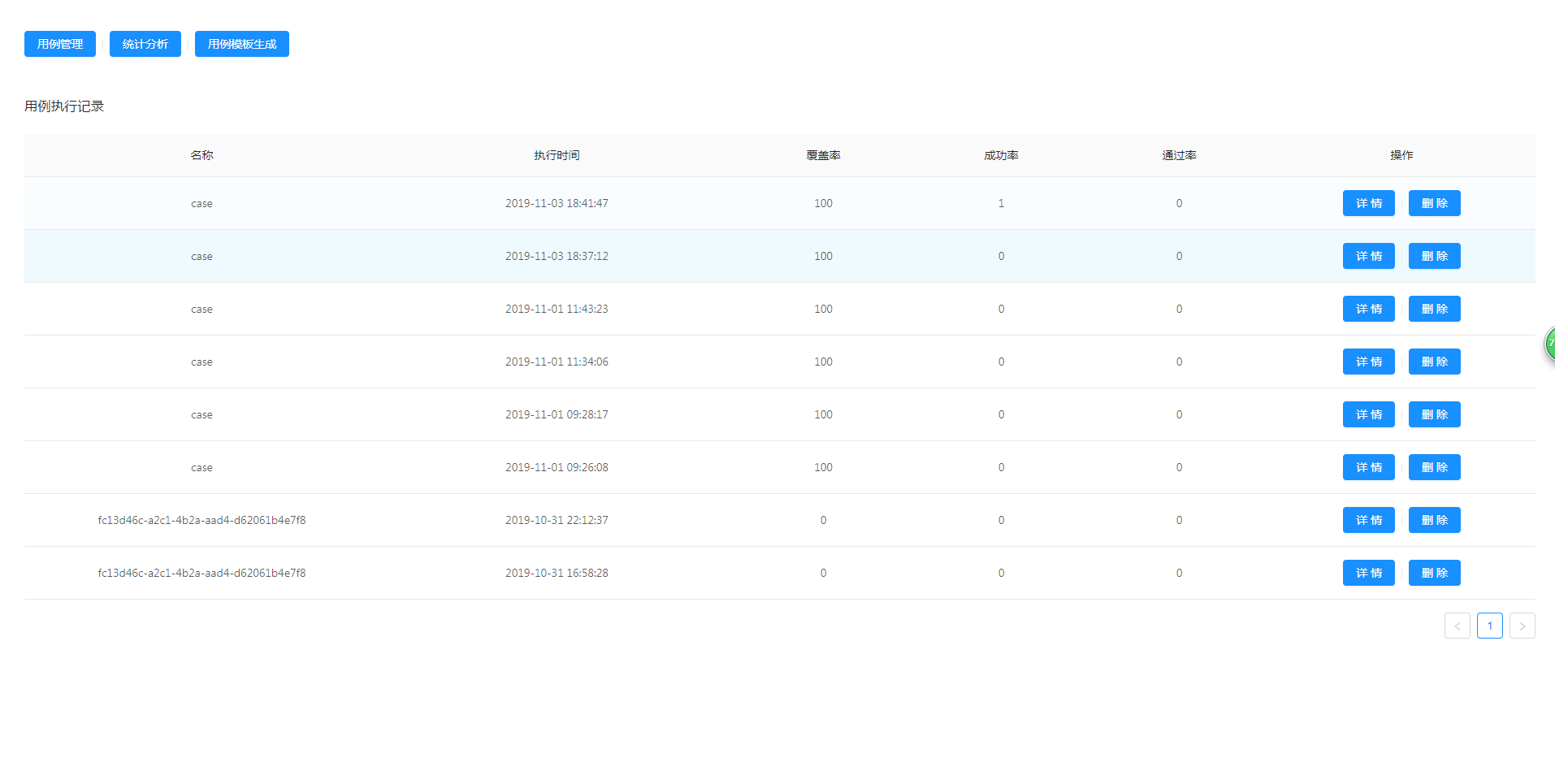
**新增项目请联系管理员**

**试用时可以选择使用【测试项目】，该项目为公用**

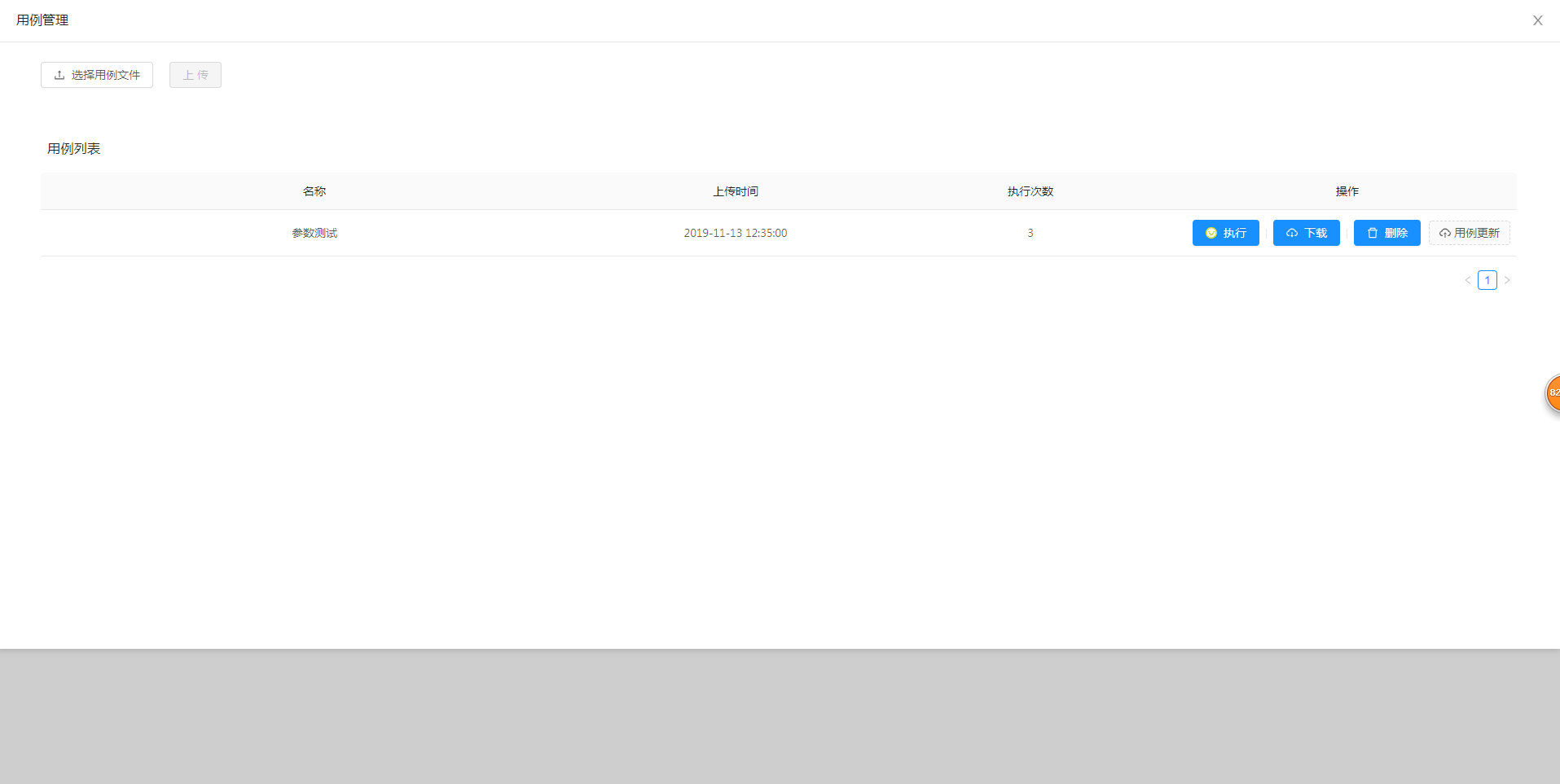
**由于目前未集成账户管理功能，请使用者切勿擅自更新、删除和执行他人项目内的用例，请尊总他人劳动成果。**



**项目详情长这样**



**点击【用例管理】**



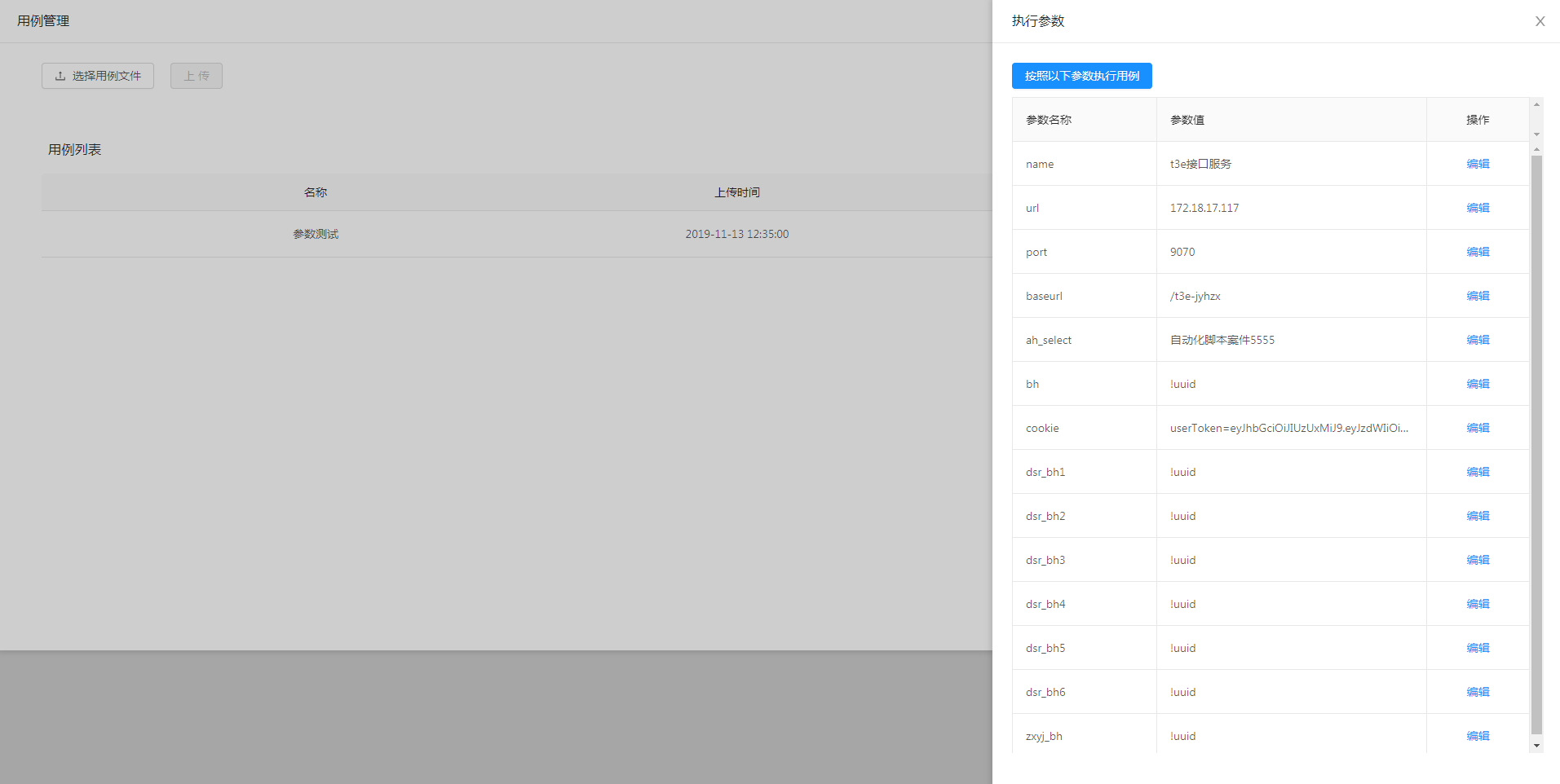
**可以上传用例文件**

**可以对用例内容进行更新（重新上传的形式）**

**执行时可以在页面编辑参数**

**用例会根据页面内编辑的参数去执行**

**编辑时目前是可以同步修改参数名称的，实际参数名称会在用例内进行引用，不建议修改，更新用例时，会将用例内的参数重新提取并保存，也就是在每次用例更新上传后，参数都会变为和用例中一致。**



**内建函数说明：**

**【!uuid】 当参数的值为!uuid时，参数在生效时会自动替换为32位的uuid，一般用于数据库参数，创建测试数据时使用**

**11.18追加自定义函数如下：**

****

**以上自定义函数可以用在自定义参数内或用例参数内，可以和其他内容或函数进行拼接**

**例如随机案号可以如下产生**

**({{!randint(2000,2019)}})湘执{{!randint(1,5000)}}号**

**会生成(2015)湘执1922号，这样的案号**

**—————持续增加中————**

**用例说明**：

用例分为三部分，也就是三个sheet页，分别是

[**keywords**],[**caselist**],[**db\_config**]

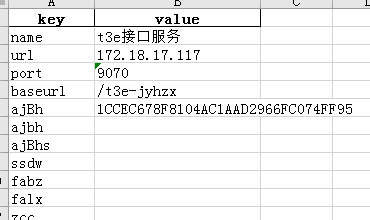


**用例在执行过程中，只会读取以上三个sheet页的内容，所以你可以新增sheet页来保存你想保存的一切，思路、想法、说明等等，但是，不要重名，也不要修改这三个sheet的名称。**

+++++++++++++++++++分隔线+++++++++++++++++++

[**keywords**]

这里是用来编写参数值的，用法类似jmeter的参数控制，不过目前版本不带函数调用，后续版本会加……吧……



左侧是变量的名称，右侧是对应的值，程序在处理参数时，会将用例中引用的变量自动替换为参数中的值，

引用方式为【 {{key-name}} 】

以图中ajbh为例，当你想用于请求参数时，可以如下处理

{“ajbh”: {{ajbh}}

如果你想用于数据库中的参数，可以如下处理

Select \* from xxx.xxxx where c\_ajbh = {{ajbh}}

\*\*\*\*\*敲黑板\*\*\*\*\*

**参数名称不支持大写字母，请使用小写字母并做好区分**

**重名的话，会被最后的赋值所取代，请一定注意**

\*\*\*\*完\*\*\*\*

+++++++++++++++++++分隔线+++++++++++++++++++

[**caselist**]

这里就是用例的执行主体了，主体中的列名代表不同的作用，以下会展开说，先说执行思路

在caselist中的数据，会从上至下，从左至右，按照**固定**的顺序执行

所以，在用例设计时，要考虑接口在**正常业务中的调用顺序**

[**name**]

接口名称，根据swagger内的名称自动生成，可以任意修改

[**method**]

请求方法，根据swagger内的名称自动生成，不建议修改，除非swagger维护错了

[**url**]

请求地址，根据swagger内的名称自动生成，自带参数，不建议修改

[**headers**]

请求头，默认生成，包含cookie，可以根据项目实际情况进行调整

[**cs**]

请求参数，**非常重要**，默认值由swagger内读取，并加入备注信息，用例编排前一定要将数据进行合理的、符合项目的进行参数化

[**sfzx**]

是否执行，默认为否，编排用例后标记为【是】后，即可执行

[**wait\_time**]

等待时间，单位是【秒】，支持两位小数

一般用于有较长处理间隔，但是又要查询处理结果时使用，可以在一定程度上保证查询的结果是有效的，不等待空着即可，如果输入内容不符合规范，比如输入了汉字等，也视为不等待

[**front\_sql**]

前置sql，主要目的是为了方便生成主体数据，因为某些服务不会自己生产数据，所以需要手动进行数据插入，或者，在用例执行前，需要对数据进行简单的处理或跳过某些环节导致的数据扶正需要，**总之写在这里的sql，是在该条用例的执行之前，完成的操作的**。

[**sql\_res**]

查询结果sql，该sql是在用例执行完请求后，在对结果验证之前，执行的sql

一般为查询sql，目的是为了对请求结果做校验或对请求操作的数据库结果做校验

查询结果会自动转换为json格式

查询sql需要对每个返回字段进行重命名操作，如果不重命名，会保留字段名称

例如

Select c\_bh as ajbh from t\_zx\_aj where xxxxx

返回结果为

{ajbh: xxxxxxxxxxxxxxxxxxxxx}

如果去掉as ajbh

返回结果为

{c\_bh: xxxxxxxxxxxxxxxxxxxxx}

这里的重命名，如果是用来验证接口返回结果的，一定要和接口返回结果中的验证字段保持名称一致，否自会验证失败

对于返回结果中的多层嵌套情况，比如下面这种

{

"key1": "value1",

"key2": [{

"key3": "value3",

"key4": "value4"

}]

}

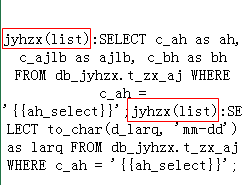
Json中多层嵌套，而我们要验证的部分在key2的value中

此时我们也要以同样的方式封装我们的sql\_res

封装的方式为

【数据库连接名】（封装的key名称）：【具体sql】

例子如下



封装名称一致的sql，查询结果会封装到一个key下

[**yzfs**]

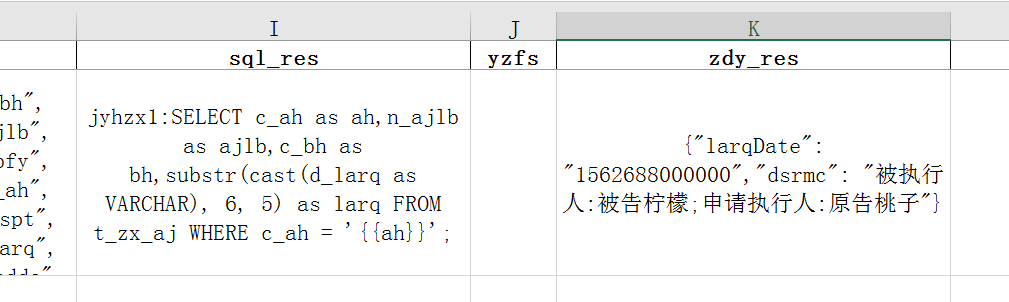
验证方式

目前提供两种验证方式

**第一种：**

Yzfs填写为空

【Sql\_res的查询结果】 + 【zdy\_res中的内容】 拼为一个json，与接口返回json做比较



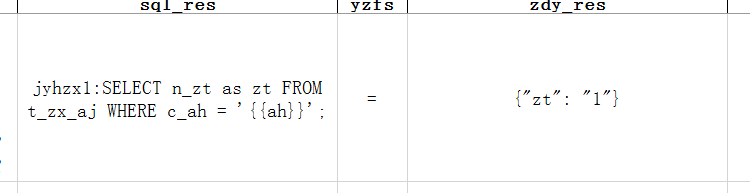
**第二种：**

比较【Sql\_res的查询结果】和【zdy\_res中的内容】

这种一般用在接口不返回实际处理数据（多为操作接口），需要验证接口在数据库中的操作是否符合预期

比如，某接口在接收参数后，会返回【操作成功】，并更新aj表的n\_zt字段为1

验证时可以写



这时会比较sql\_res中的查询结果json是否和zdy\_res中的json一致

[**zdy\_res**]

自定义验证json

这个一般用在某些不会被操作，但是仍会返回的字段

比如案由字段，该字段一般不会被修改，但是很多查询接口会返回，此时就可以写在这里进行验证，使用方式详见上面的字段解释

[**back\_sql**]

后置sql

这里一般用来清理数据，一般在最后一条用例执行完毕后，对过程数据进行清理

可选操作，如果觉得没必要，可以空着，方便在页面检查整体的流程数据

填写在这里的sql，会在用例执行完毕后进行操作，放入什么操作，可以自由选择

[**res\_for\_key**]

将接口返回值，转化为参数，用于后续的接口参数，格式如下：

【预期的参数名称】：【key\_name1】.【key\_name2】.【key\_name3】

例如，接口返回为{‘a’: ‘b’, ‘c’: ‘d’, ‘e’: {‘f’: ‘g’, ‘h’: ‘f’}}

此时需要将值【f】提取出来，并命名为【ajbs】，可以写成如下格式

ajbs:e.h

后续接口可以通过{{ajbs}}来引用该提取参数

如果是全部提取，可以写成如下格式

ajbs:\*

不进行提取，空着即可

\*\*\*\*\*敲黑板\*\*\*\*\*

**所有的sql，在编写时，要遵循一个规则**

**那就是数据库连接名（db\_config中的name字段）一定要有，否则会报错**

**【数据库连接名】：【sql主体】**

**多sql用【；】隔开**

**【数据库连接名】：【sql主体】；【数据库连接名】：【sql主体】；**

**多个sql在执行完毕后，会合并为一个json，所以sql是可以分开写的**

**但是要注意，所有sql的返回结果，一定要是一条**

**一般查询接口在验证时，一定要保证接口返回结果为唯一数据**

**暂时不支持多条返回结果的匹配验证，请不要这么写**

\*\*\*\*完\*\*\*\*

+++++++++++++++++++分隔线+++++++++++++++++++

[**db\_config**]

初始化为空，这里主要是放脚本对应的服务的数据库的连接信息的

一句话，数据库链接要和脚本对应

否则，你懂的

就算你不需要sql操作，也要写，而且要写个对的

因为没校验这里，不填或连不上，就会执行失败，就是这么任性

以后会改……



[**name**]

数据库连接名，随便起，但是写sql之前一定要带上它，参见上面的caselist中的说明

[**db\_type**]

数据库类型，现在只支持abase，就直接写abase就行

[**db\_ip**]

数据库地址，你懂的

[**db\_port**]

数据库端口，你懂的

[**db\_user**]

用户名，你懂的

[**db\_password**]

用户密码，你懂的

[**db\_name**]

数据库名称，就是你存数据的库的名字

\*\*\*\*\*敲黑板\*\*\*\*\*

**这里支持多库的处理，如果你的服务~~非常不幸的~~连接了多个数据库**

**可以在这里配置多个数据源，用name区分，在用例中写sql时**

**前缀放上不同的名字即可**

\*\*\*\*完\*\*\*\*

**验证结果的说明：**

**当验证方式为自己组装的数据与接口返回值进行比较时，**

**我们的验证字段可能会比实际返回值的字段要少**

**此时，**

**接口返回值中的字段比查询字段多时，视为默认正确，即验证通过**

**接口返回值中的字段比查询字段少时，视为接口有问题，即验证不通过**

**相同的字段名中，数值不一致，视为接口有问题，即验证不通过**