# 场景联动表TB4045\_SCENEREACTION

**列信息：**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 列描述 | 数据类型 | 使用介绍 |
| F1001\_GROUPCODE | 组 | char(8) |  |
| F4045\_REACTIONCODE(主键) | 场景代码 | char(24) |  |
| F4045\_REACTIONDESC | 场景描述 | char(40) |  |
| F1000\_AOJ | 权限区域 | unsigned int |  |
| F4045\_INHIBIT | 是否禁止 | unsigned char | 1：是、0：否 |
| F4045\_TRIGGERTYPE | 触发器类型 | unsigned char | 0：状态量变位、  1：火警信号、  2：手动模式(界面使用) |
| F4045\_TRIGGERPARA | 触发器参数 | char(512) | 请看以下介绍 |

**字段‘触发器参数’：**

当‘触发器类型’值是0时，此处需要填写触发的点代码和触发的值，多个触发点写多对，格式为"pointcode1:value1;pointcode2:value2;"，若不填写触发值表示只要此点发生状态变位就触发，例如点代码pointcode3的写法：" pointcode1:value1;pointcode2:value2; pointcode3;"

当‘触发器类型’值是1时，待定；

当‘触发器类型’值是2时，此字段填空；

# 场景步骤表TB4046\_SCENESTEP

**列信息：**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 列描述 | 数据类型 | 使用介绍 |
| F4046\_STEPDESC | 步骤描述 | char(128) |  |
| F4046\_STEPNO (主键) | 步骤序号 | unsigned short |  |
| F4046\_SUBSTEPNO(主键) | 子步骤序号 | unsigned short |  |
| F4045\_REACTIONCODE(主键) | 场景代码 | char(24) |  |
| F4046\_MODE | 步骤模式 | unsigned char | 1：自动、  2：半自动、  3：手动 |
| F4046\_STEPTYPE | 步骤类型 | unsigned char | 离散控制、模拟量控制、序控、通用命令 |
| F4046\_HASSUBOBJ | 是否有子步骤 | unsigned char | 1：是、0：否 |
| F4046\_CTLPERMSVPT | 控制许可脚本 | char(64) | 脚本名称，判断本步骤是否满足执行条件 |
| F4046\_GRAPH | 推图 | char(255) | 填写图名称、屏号等，UI界面操作打开此图； |
| F4046\_CTRLOBJ | 控制对象 | char(64) | 填写AO、DO、序控代码，本步骤下发通用命令时此处为空； |
| F4046\_VALUE | 控制值 | float | 填写AO、DO控制值 |
| F4046\_PARA | 控制参数 | char(1024) | 本步骤下发通用命令时此处填写加密的json包 |
| F4046\_FBACKSTRATEGY | 反馈策略 | unsigned char | 0：无、  1：DI反馈、  2：管道结果、  3：执行完成延时等待 |
| F4046\_FBACKPARA | 反馈参数 | char(512) | 请看下面字段介绍 |
| F4046\_TIMEOUT | 超时时间 | unsigned short | 获取反馈结果超时等待时间，单位秒 |
| F4046\_FAILSTRATEGY | 执行失败策略 | unsigned char | 0：无、  1：人工介入(用户决定继续或退出)、  2：退出场景联动 |

**字段‘反馈参数’：**

当‘反馈策略’值是0时填空；

当‘反馈策略’值是1时，此处需要填写反馈点代码和反馈成功值，多个反馈点写多对，格式为"pointcode1:value1;pointcode2:value2;"，若不填写反馈成功值此点无效(即冒号后面的值)；

当‘反馈策略’值是2时，填空；

当‘反馈策略’值是3时，此字段填写延时时间秒；

# 步骤配置信息、执行流程和注意事项



**配置信息如上图：**

‘步骤序号’表示一级步骤（父步骤），‘子步骤序号’表示二级步骤（子步骤），最多支持两级步骤，‘子步骤序号’是从1开始的步骤设置‘是否有子步骤’为‘否’；

‘步骤序号’从0开始配置，‘子步骤序号’在同一个‘步骤序号’下亦从0开始配置；

上图中‘步骤序号’为0，‘子步骤序号’为0的列‘是否有子步骤’设为‘是’，则此条步骤作为父步骤，有效字段分别是：‘步骤描述’、‘场景代码’、‘步骤模式’、‘是否有子步骤’，步骤描述’是对所拥有子步骤的整体描述，例如此步骤描述是‘启动电扶梯’，下面的‘子步骤序号’分别为1，2，3的子步骤为分别执行不同的电扶梯的步骤；

执行时上图‘步骤序号’为0的步骤有子步骤时跳过子步骤序号是0的步骤，从1号子步骤开始执行；‘步骤序号’为1的步骤无子步骤，直接开始执行；

**执行流程：**

第一步：半自动或手动模式时等待操作员跳过或确认执行此条步骤，自动模式直接执行，此外界面获取到本步骤执行时需要根据推图字段打开对应的工程图；

第二步：执行时有控制许可脚本就执行此脚本判断是否满足执行条件；

第三步：执行控制信息，

第四步：若设置的‘反馈策略’是‘DI反馈’、‘管道结果’，则在设置的超时时间内读取结果；若设置的‘反馈策略’是‘延时等待’，则等待配置的时间（反馈参数中配置此时间）后执行下一个步骤；若是配置‘无’，则直接执行下一个步骤；

第五步：设置的‘反馈策略’是‘DI反馈’、‘管道结果’，获取反馈结果超时或者反馈结果表示执行失败时，需要根据配置的‘执行失败后策略’执行，配置是‘无’，直接执行下一个步骤，配置是‘自动退出’，则退出整个场景联动；配置是‘人工介入’，则与界面交互，等待用户手动确认：继续往下执行、跳过此父步骤未执行的子步骤、退出本场景联动几种选择；

接着就是重复执行‘第一步’到‘第五步’；

**注意事项：**

自动，半自动、手动区别：自动步骤直接执行，半自动步骤让用户手动确认执行或者跳过本步骤；

# 界面和后台服务交互

界面和后台服务通过redis进行交互信息，请配合下一标题‘redis结构’信息来了解交互过程。

**交互流程：**

BS界面未打开场景联动界面时，需要不断获取正在执行的场景，根据redis list型数据的key：“scene:running:list”，获取正在执行场景的场景代码scenecode，提醒用户有被触发的场景；

接着当用户打开场景联动界面时，根据场景代码组成场景信息hash结构的key：“scene: scenecode:hash”，获取正在执行场景的状态信息，解析对于json包，更新界面中此场景的状态；json包格式请在下面的标题‘json包格式’中查看；

场景界面每个场景块需要有一个紧急停止的按钮，用户点击后能把命令写入redis中，后台服务停止相应的正在执行的场景；

手动执行场景时，界面下发一个命令，走ICE发送到后台服务；

# redis结构：

**场景信息hash：**

key：scene: scenecode:hash

feilds：aoj-权限区域id，info-整个场景信息json包，格式查看下文

**正在执行场景列表list：**

key：scene:running:list

队列：scenecode（场景代码）

根据scenecode可以组成key：scene: scenecode:hash

**人工介入列表list（界面设置此项）**

key：scene:manual:list

列表：scene:manual:scenecode:hostname:string

**人工介入信息获取（界面设置此项）**

key：scene:manual:scenecode: hostname:string

value：json包

# json包格式：

## 场景信息hash对应json

|  |  |  |
| --- | --- | --- |
| key | scene: scenecode:hash |  |
| 域：info | {  "code": "", #场景代码  "group":"", #组代码  "ismanual":false, #去除  "fstepcount": 10, #父步骤总数  "fsteprunno":10, #正在执行父步骤序号，-1表示没有正在执行的步骤  "csteprunno":1, 正在执行子步骤序号  "begin":, #场景执行时间戳  "end":, #场景结束时间戳  "0":[ #序号0父步骤  {  "cstep":0, #子步骤序号  "desc":"", #步骤描述  "mode":"自动",#步骤模式  "graph":"", #界面要主动打开图  "status":1,  "ret":1;  "retstr":""  }，  {  "cstep":1, #子步骤序号  "desc":"", #步骤描述  "mode":"自动",#步骤模式  "graph":"", #界面要主动打开图  "status":1,  "ret":1;  "retstr":""  }  ],  "1":[],  ...  "9":[]  } | ismanual：true手动操作，false自动触发  status：步骤状态量  0-初始状态，未触发  1-正在执行  2-等待执行  3-等待用户确认  4-等待反馈结果  5-执行后反馈的结果失败或超时，由用户确认跳过当前父步骤或者继续执行或者退出当前场景联动；  6-跳过 （状态5时用户选择跳过的步骤的状态都是跳过）  7-退出执行（状态5时用户选择退出场景联动时后续步骤状态都是退出执行）  8-禁止控制  9-执行完毕（步骤真正执行完毕，接着要执行下一步）  ret：步骤执行后的结果  1-成功  2-失败  3-获取结果超时  4-跳过  5-退出此步骤  6-拒控  retstr：对于本步骤执行结果的描述（包括失败的反馈点）  json结构中根据父步骤总数，从0开始的字符串作为key，对应一个数组，若此数组只有一个对象，那么表示此父步骤没有子对象； |
| 域：aoj | 权限区域id | 判断用户是否拥有显示和控制权限 |

## 手动执行场景json包

|  |  |  |
| --- | --- | --- |
| key | scene:manual:scenecode:hostname:string | scenecode：场景代码  hostname：主机名，防止多台机器都下发控制重名 |
| value | {  "user":1, #用户名  "role":1; #角色名  "opt":1,  "code":"" #场景代码  确定的步骤序号，本步骤，父步骤  } | opt：1-手动执行整个场景 #去除  2-人工确认继续往下执行  3-人工选择跳过本步骤所属父步骤  4-人工紧急停止整个场景 |
|  |  |  |