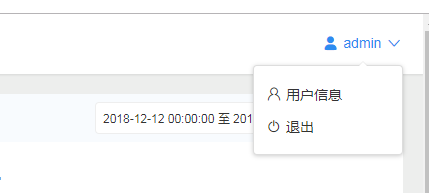
# 登录

1. 输入<http://192.168.220.204:8888/view/login/index.html> 网址，输入用户名，密码登录EasyScheduler



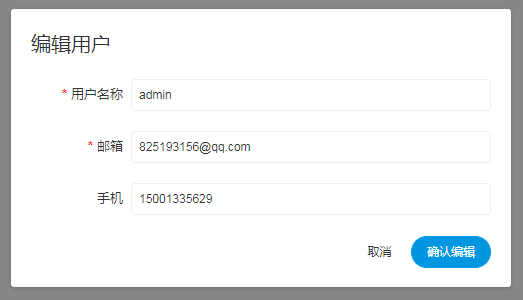
1. 登录之后每个页面的右上角都有用户的身份标识。点击下拉箭头包含用户信息和退出两个按钮



* 点击“用户信息”按钮，如下图：



点击” 修改”按钮，修改用户信息



* 点击退出按钮则退出系统，返回登录页面

# 安全中心

只有管理员才有安全中心，安全中心的主要功能是给管理员提供管理普通用户的功能。管理员可以有多个，管理员是功能上的管理，不参与具体的业务。**也就是说管理员是不能执行具体任务的**。

## 租户管理

租户是Linux上的用户，用于作业的提交。

* 创建、编辑租户



租户编码：**租户编码是Linux上的用户**，唯一，不能重复

租户名称：租户的名称

队列：**租户对应的YARN上的队列**，在数据库 **t\_escheduler\_queue** 中设置

描述：租户的描述信息

## 用户管理

用户是EasyScheduler上的用户，用于EasyScheduler上的**功能操作**。

* 创建、编辑用户



用户名称：用户的名称，唯一，不能重复

密码：设置用户的登录密码

租户：设置该用户所属的租户

邮箱：输入用户的邮箱，用来邮件发送和任务告警

手机：输入用户的手机号

注意：**如果该用户切换了租户，则该用户所在租户下所有资源将复制到切换的新租户下**

* 授权

**管理员**可以对**普通用户**进行**非其创建的**项目、资源、数据源和UDF函数进行授权。因为项目、资源、数据源和UDF函数授权方式都是一样的，所以以**项目授权**为例介绍。

1. 点击指定人的授权按钮，如下图：



1. 选中**项目**按钮**，**进行项目授权



项目列表：是该用户未授权的项目

已选项目：是该用户已授权的项目。

**特别注意：对于用户自己创建的项目，该用户拥有所有的权限。则项目列表和已选项目列表中不会体现。**

## 告警组管理

告警组是告警用户抽象出来的组，使用告警组来管理用户。

* 新建、编辑邮件组



组名称：输入组的名称

组类型：支持**邮件/短信**两种

备注：输入告警组的备注信息

* 管理用户



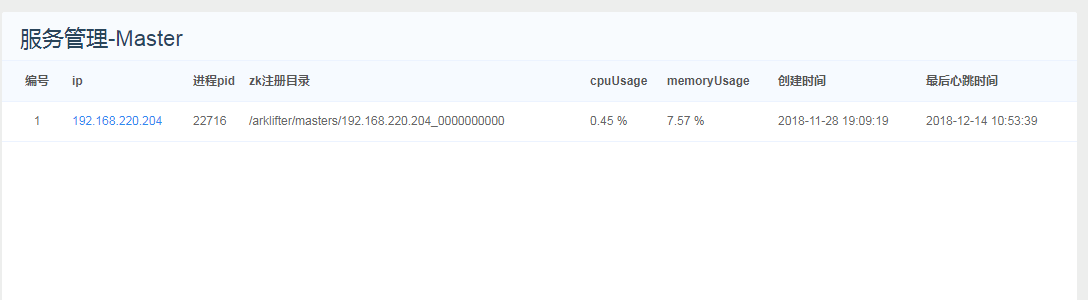
管理用户列表：是未添加到该组的用户列表

已选管理用户：是已添加到该组的用户列表

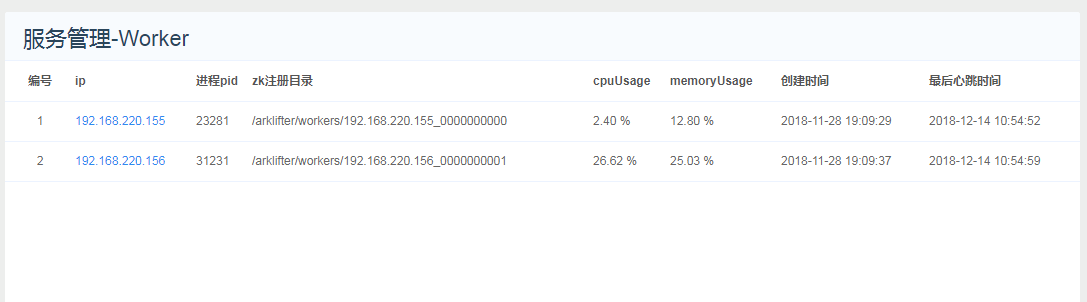
## 服务管理

服务管理是对EasyScheduler的Master、Worker的任务监控

* Master



* Worker



# 资源中心

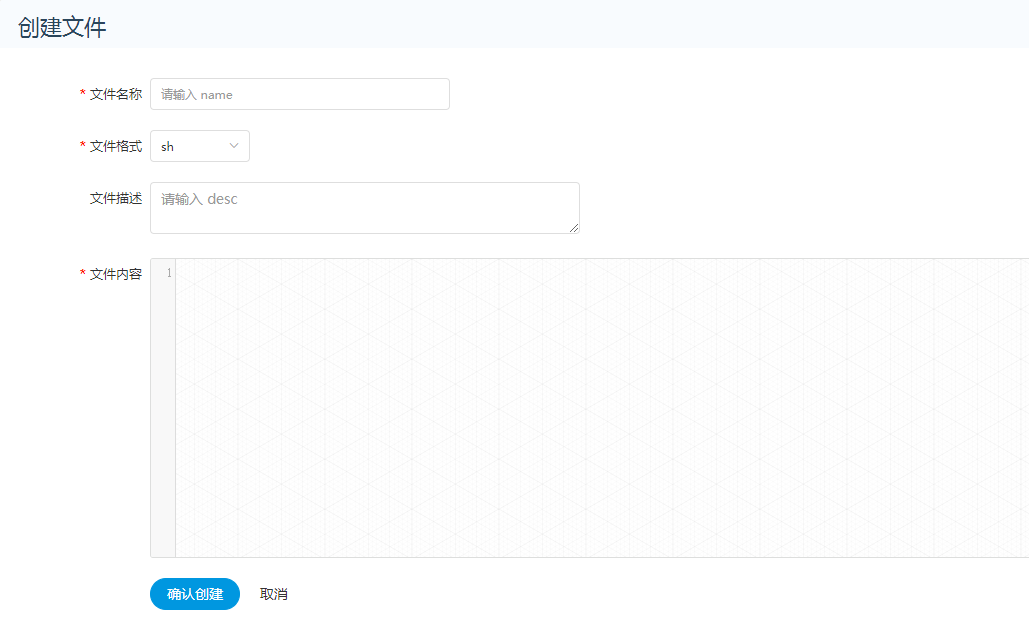
资源中心主要分为文件管理和UDF函数管理。

文件管理：主要是用户的程序，脚本和配置文件需要上传到HDFS进行统一管理

UDF函数管理：对用户创建的UDF进行管理

## 文件管理

* 创建文件



文件格式支持以下几种类型：

txt、log、sh、conf、cfg、py、java、sql、xml、hql

* 上传文件



文件名**：**输入文件的名称

描述：输入文件的描述信息

上传文件：**点击上传**按钮进行上传，将文件拖拽到上传区域，**文件名**会自动以上传的文件名称补全

* 文件查看

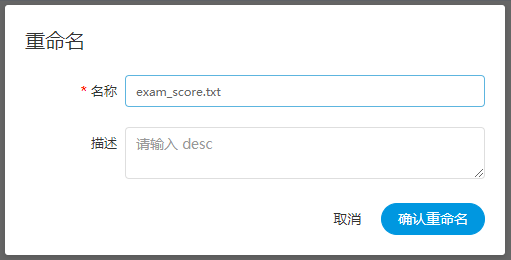
对可查看的文件类型，点击 **文件名称** 可以查看文件详情



* 下载文件

可以在 **文件详情** 中点击右上角下载按钮下载文件，或者在文件列表后的**下载**按钮下载文件

* 文件重命名



* 删除

文件列表，点击 **删除** 按钮，删除文件

## UDF管理

### 资源管理

资源管理和文件管理功能类似，不同之处是资源管理是上传的UDF函数，文件管理上传的是用户程序，脚本及配置文件

### 函数管理

* 创建、编辑UDF函数



**目前只支持HIVE的临时UDF函数**

UDF函数名称：输入UDF函数时的名称

包名类名：输入UDF函数的全路径

参数：用来标注函数的输入参数

数据库名：预留字段，用于创建永久UDF函数

UDF资源：设置创建的UDF对应的资源文件

使用说明：输入描述UDF函数的使用

# 数据源中心

数据源中心支持MySQL、POSTGRESQL、HIVE及Spark数据源

* 创建、编辑MySQL数据源



数据源：选择MYSQL

数据源名称：输入数据源的名称

描述：输入数据源的描述

IP/主机名：输入连接MySQL的IP

端口：输入连接MySQL的端口

用户名：设置连接MySQL的用户名

密码：设置连接MySQL的密码

数据库名：输入连接MySQL的数据库名称

Jdbc连接参数：用于MySQL连接的参数设置，以JSON形式填写

* 创建、编辑POSTGRESQL数据源



数据源：选择POSTGRESQL

数据源名称：输入数据源的名称

描述：输入数据源的描述

IP/主机名：输入连接POSTGRESQL的IP

端口：输入连接POSTGRESQL的端口

用户名：设置连接POSTGRESQL的用户名

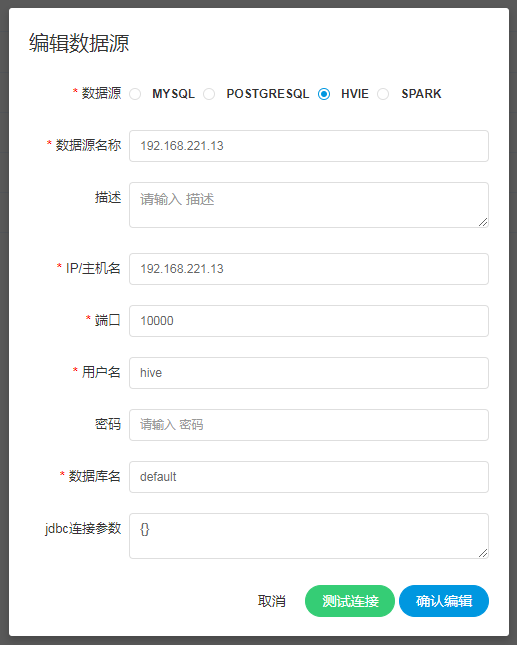
密码：设置连接POSTGRESQL的密码

数据库名：输入连接POSTGRESQL的数据库名称

Jdbc连接参数：用于POSTGRESQL连接的参数设置，以JSON形式填写

* 创建、编辑HIVE数据源

1. 使用**HiveServer2**方式连接



数据源：选择HIVE

数据源名称：输入数据源的名称

描述：输入数据源的描述

IP/主机名：输入连接HIVE的IP

端口：输入连接HIVE的端口

用户名：设置连接HIVE的用户名

密码：设置连接HIVE的密码

数据库名：输入连接HIVE的数据库名称

Jdbc连接参数：用于HIVE连接的参数设置，以JSON形式填写

1. 使用**HiveServer2 HA Zookeeper**方式连接



数据源：选择HIVE

数据源名称：输入数据源的名称

描述：输入数据源的描述

IP/主机名：输入连接Zookeeper的集群

端口：输入连接Zookeeper的端口

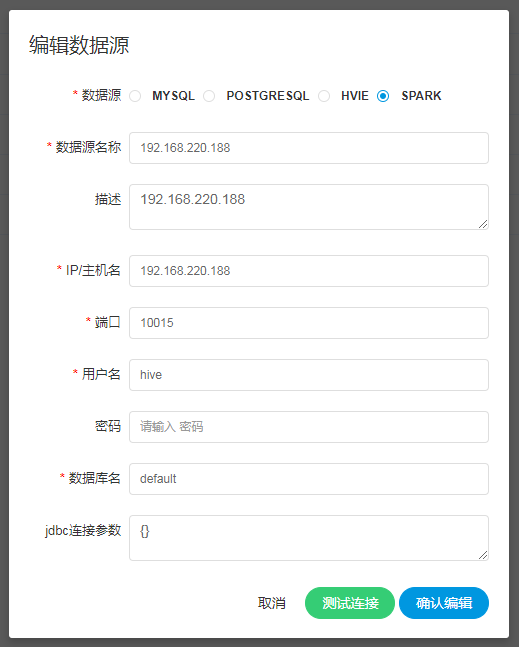
用户名：设置连接HIVE的用户名

密码：设置连接HIVE的密码

数据库名：输入连接HIVE的数据库名称

Jdbc连接参数：用于Zookeeper连接的参数设置，以JSON形式填写

* 创建、编辑Spark数据源



数据源：选择Spark

数据源名称：输入数据源的名称

描述：输入数据源的描述

IP/主机名：输入连接Spark的IP

端口：输入连接Spark的端口

用户名：设置连接Spark的用户名

密码：设置连接Spark的密码

数据库名：输入连接Spark的数据库名称

Jdbc连接参数：用于Spark连接的参数设置，以JSON形式填写

# 项目管理

项目是调度对用户流程定义DAG分组的一个抽象

* 创建、编辑项目



项目名称：输入项目的名称

描述：输入项目的描述

## 项目首页

点击项目列表中的**项目名称**，可以跳转到指定的项目首页，如下图：



项目首页其中包含四个部分，任务状态统计，流程状态统计、流程定义统计及统计的时间范围

任务状态统计：是指在指定时间范围内，统计任务实例中的待运行、失败、运行中、完成、成功的个数

流程状态统计：是指在指定时间范围内，统计流程实例中的待运行、失败、运行中、完成、成功的个数

流程定义统计：是统计该用户创建的流程定义及管理员授予该用户的流程定义

**可以点击图，或者数量跳转到相应的任务实例，流程实例和流程定义列表**

## 工作流

工作流下分为流程定义、流程实例和任务实例三个功能模块

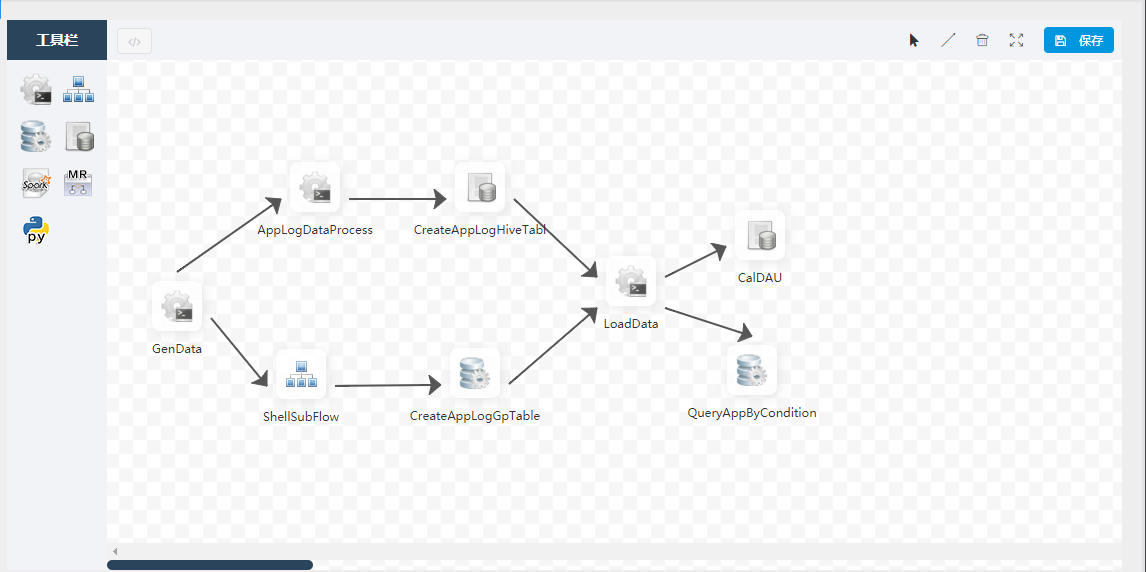
流程定义：是可视化拖拽成的DAG的统称，它是静态的，没有状态

流程实例：对流程定义的每次实例化会生成一个流程实例，是动态的，是有状态的

任务实例：流程实例DAG中每个Task称为任务实例，是动态的，是有状态的

### 流程定义

* 创建工作流



**左侧工具栏**是目前调度支持的任务类型，当前调度支持SHELL、子流程、存储过程、SQL、MR、Spark和Python七种任务类型

**右上角图标**分别是拖动节点和选中项、选择线条连线、删除选中的线或节点、全屏和流程定义保持，其主要功能是DAG的绘制所用

1. 创建 **SHELL**节点

拖动**工具栏**中的任务节点到画板中，双击任务节点，如下图：



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

失败重试次数：任务失败重新提交的次数，支持下拉和手填

失败重试间隔：任务失败重新提交任务的时间间隔，支持下拉和手填

脚本：用户开发的SHELL程序

资源：是指脚本中需要调用的资源文件列表

自定义参数：是SHELL局部的用户自定义参数，会替换脚本中以${变量}的内容

任务依赖：**请参考任务依赖**讲解

1. 创建 **子流程** 节点

拖动**工具栏**中的任务节点到画板中，双击任务节点，如下图：



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

子节点：是选择**子流程**的流程定义，右上角**进入该子节点**可以跳转到所选子流程的流程定义

任务依赖：**请参考任务依赖**讲解

1. 创建 **存储过程** 节点

拖动**工具栏**中的任务节点到画板中，双击任务节点，如下图：



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

失败重试次数：任务失败重新提交的次数，支持下拉和手填

失败重试间隔：任务失败重新提交任务的时间间隔，支持下拉和手填

数据源：存储过程的数据源类型支持MySQL和POSTGRESQL两种，选择对应的数据源

方法：是存储过程的方法名称

自定义参数：存储过程的自定义参数类型支持IN、OUT两种，数据类型支持VARCHAR、INTEGER、LONG、FLOAT、DOUBLE、DATE、TIME、TIMESTAMP、BOOLEAN九种数据类型

任务依赖：**请参考任务依赖**讲解

1. 创建 **SQL** 节点

拖动**工具栏**中的任务节点到画板中，双击任务节点，如下图：



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

失败重试次数：任务失败重新提交的次数，支持下拉和手填

失败重试间隔：任务失败重新提交任务的时间间隔，支持下拉和手填

数据源：SQL数据源支持MySQL、POSTGRESQL、HIVE和Spark四中数据源类型，选择对应的数据源

sql类型：支持查询和非查询两种，查询是**select类型**的查询，是有结果集返回的，可以指定邮件通知为表格、附件或表格附件三种模板。非查询是没有结果集返回的，是针对**update、delete、insert**三种类型的操作

sql参数：输入参数格式为key1=value1;key2=value2…

sql语句：SQL语句

UDF函数：对于HIVE类型的数据源，可以引用资源中心中创建的UDF函数,其他类型的数据源暂不支持UDF函数

自定义参数：SQL任务类型自定义参数类型和数据类型同存储过程任务类型一样。**区别在于SQL任务类型自定义参数会替换sql语句中${变量}，而存储过程是自定义参数顺序的给方法设置值**

任务依赖：**请参考任务依赖**讲解

1. 创建 **MR** 节点

拖动**工具栏**中的任务节点到画板中，双击任务节点，如下图：

* JAVA程序



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

失败重试次数：任务失败重新提交的次数，支持下拉和手填

失败重试间隔：任务失败重新提交任务的时间间隔，支持下拉和手填

主函数的class：是MR程序的入口Main Class的全路径

程序类型：选择JAVA语言

主jar包：是MR的jar包

命令行参数：是设置MR程序的输入参数，支持自定义参数变量的替换

其他参数：支持 –D、-files、-libjars、-archives格式

资源： **如果其他参数中引用了资源文件，需要在资源中选择指定**

自定义参数：是MR局部的用户自定义参数，会替换脚本中以${变量}的内容

任务依赖：**请参考任务依赖**讲解

* Python程序



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

失败重试次数：任务失败重新提交的次数，支持下拉和手填

失败重试间隔：任务失败重新提交任务的时间间隔，支持下拉和手填

程序类型：选择Python语言

主jar包：是运行MR的Python jar包

其他参数：支持 –D、-mapper、-reducer、-input -output格式，这里可以设置用户自定义参数的输入，比如：

-mapper **"mapper.py 1"** -file mapper.py -reducer reducer.py -file reducer.py –input /journey/words.txt -output /journey/out/mr/${currentTimeMillis}

其中 -mapper 后的 **mapper.py 1是两个参数，第一个参数是mapper.py，第二个参数是1**

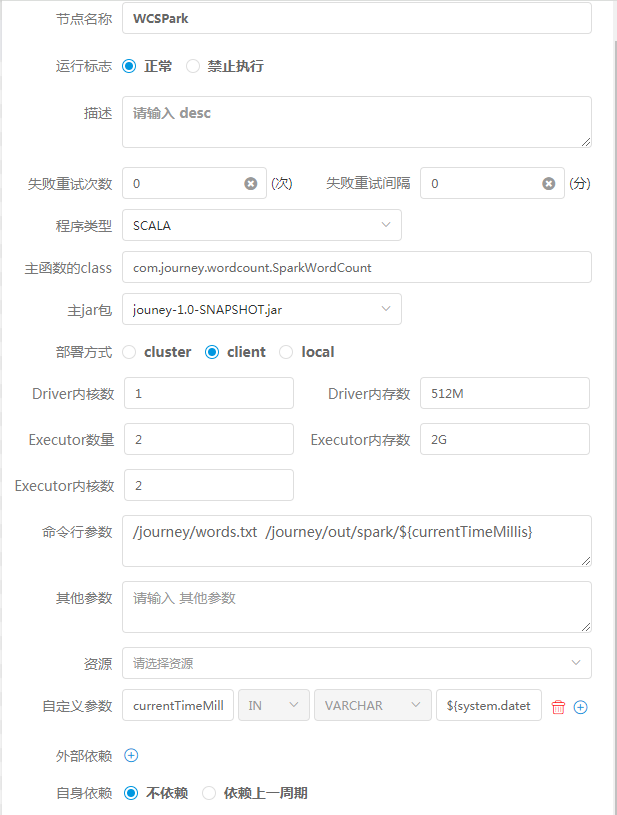
资源： **如果其他参数中引用了资源文件，需要在资源中选择指定**

自定义参数：是MR局部的用户自定义参数，会替换脚本中以${变量}的内容

任务依赖：**请参考任务依赖**讲解

1. 创建 **Spark** 节点

拖动**工具栏**中的任务节点到画板中，双击任务节点，如下图：



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

失败重试次数：任务失败重新提交的次数，支持下拉和手填

失败重试间隔：任务失败重新提交任务的时间间隔，支持下拉和手填

程序类型：支持JAVA、Scala和Python三种语言

主函数的class：是Spark程序的入口Main Class的全路径

主jar包：是Spark的jar包

部署方式：支持yarn-cluster、yarn-client、和local三种模式

Driver内核数：可以设置Driver内核数及内存数

Executor数量：可以设置Executor数量、Executor内存数和Executor内核数

命令行参数：是设置Spark程序的输入参数，支持自定义参数变量的替换。

其他参数：支持 --jars、--files、--archives、--conf格式

资源：**如果其他参数中引用了资源文件，需要在资源中选择指定**

自定义参数：是MR局部的用户自定义参数，会替换脚本中以${变量}的内容

任务依赖：**请参考任务依赖**讲解

注意：JAVA和Scala只是用来标识，没有区别，如果是Python开发的Spark则没有主函数的class，其他都是一样

1. 创建 **Python** 节点

拖动**工具栏**中的任务节点到画板中，双击任务节点，如下图：



节点名称：一个流程定义中的节点名称是唯一的

运行标志：标识这个节点是否能正常调度

描述信息：描述该节点的功能

失败重试次数：任务失败重新提交的次数，支持下拉和手填

失败重试间隔：任务失败重新提交任务的时间间隔，支持下拉和手填

脚本：用户开发的Python程序

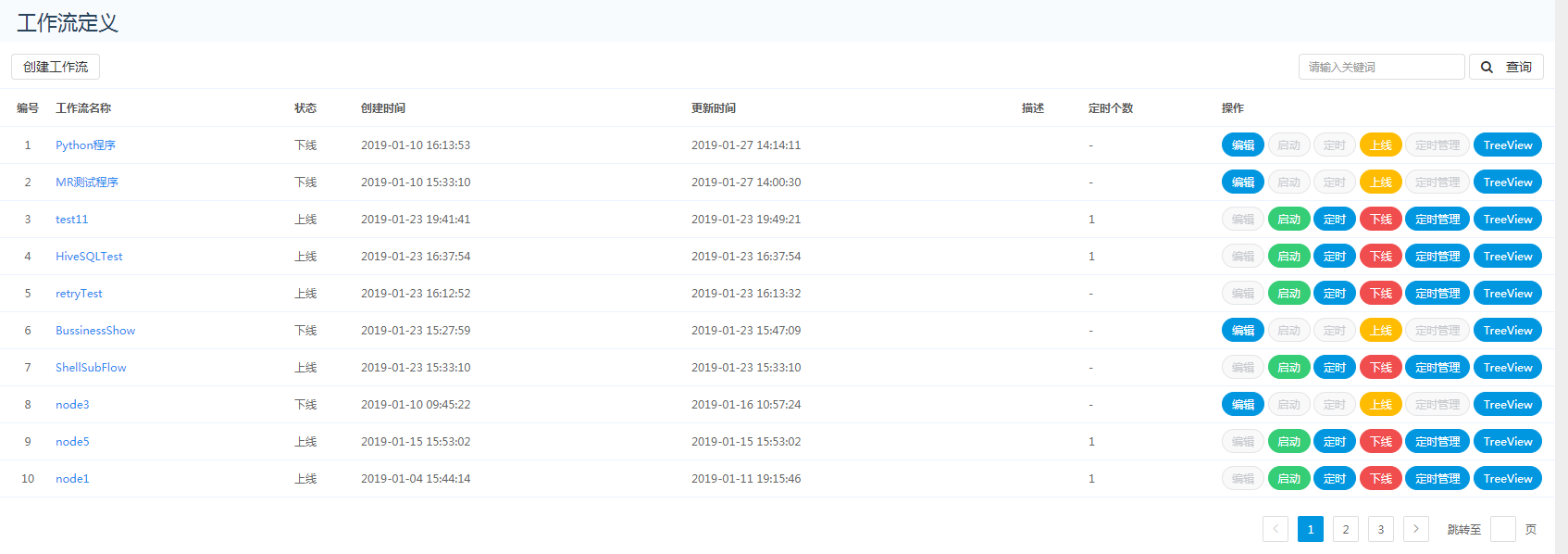
资源：是指脚本中需要调用的资源文件列表

自定义参数：是Python局部的用户自定义参数，会替换脚本中以${变量}的内容

任务依赖：**请参考任务依赖**讲解

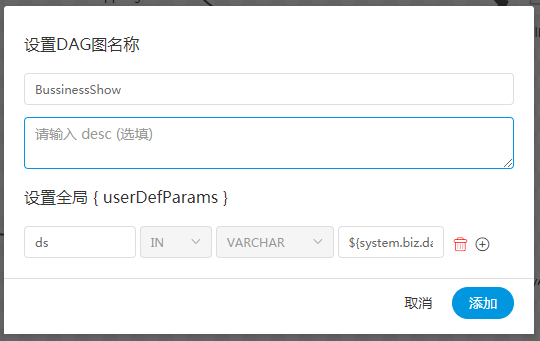
* 流程定义操作

流程定义列表



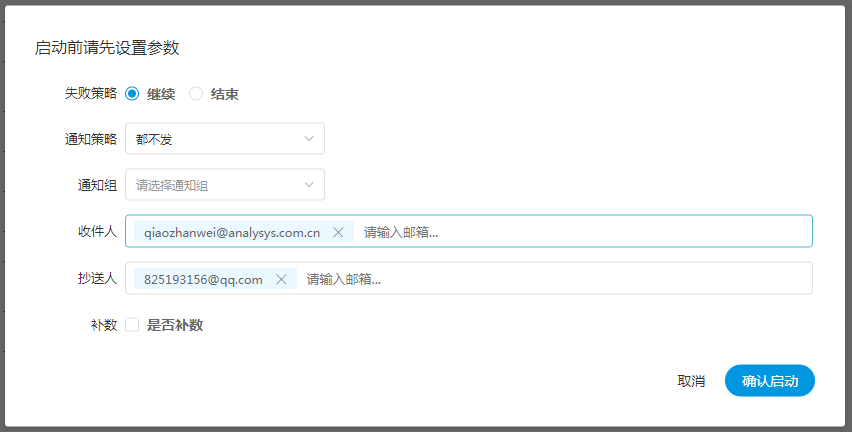
1. 编辑/保存

对**下线**状态的流程定义，用户可对DAG进行编辑



在**保存**流程定义的时候可以设置全局参数，局部参数可以使用${变量}的方式引用全局参数

1. 启动



失败策略：**继续**是如果有失败的任务节点，其他并行的任务节点将继续执行。**结束**是如果有失的败节点，其他并行任的务节点将被Kill掉

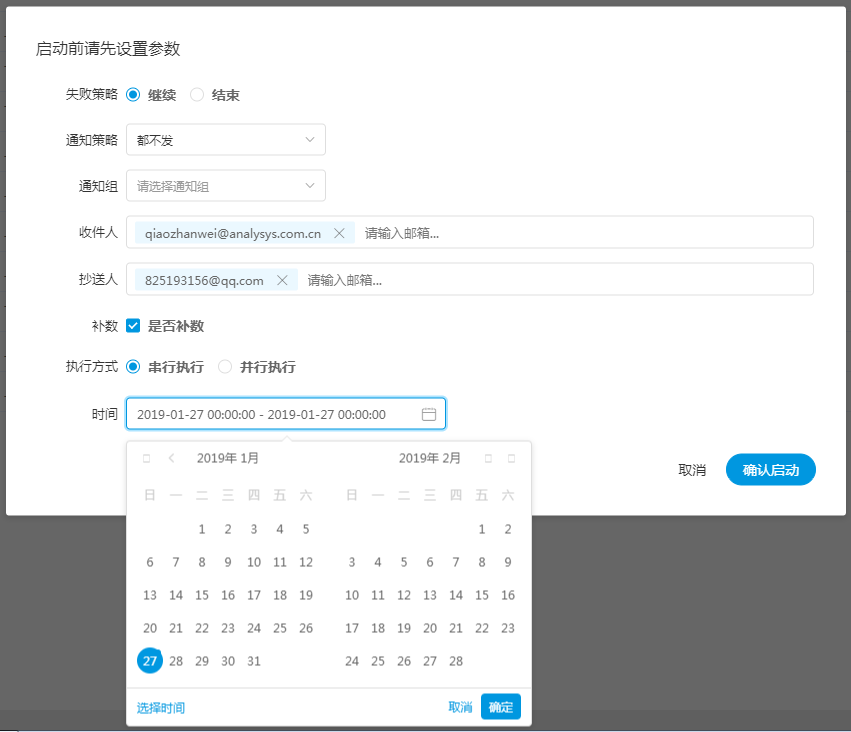
通知策略：都不发、成功发、失败发、成功或失败都发四种策略

通知组：是通知用户所属的组

收件人：邮件收件人列表

抄送人：邮件抄送人列表

补数：是**对流程进行历史的补数，支持区间补数和某一天的补数**，如下图：

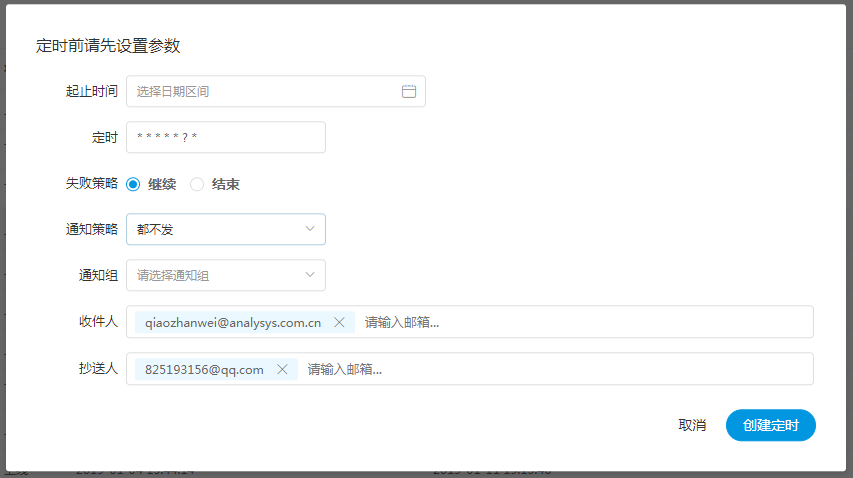


执行方式：补数可以以串行执行或并行执行，串行执行是流程实例列表以队列的形式逐个提交执行，并行执行是流程实例列表并发形式的提交执行

时间：**选择指定的补数时间区间**，**此时业务时间为补数时间**，如果起始时间和结束时间相等，则为某一天的补数，否则为区间补数。关于系统参数，请参考系统参数的讲解

1. 定时

流程定义可以设置定时，周期性的跑作业。如下图：



起止时间：选择该定时的**有效时间范围**

定时：可视化定时，设置相应的定时



失败策略：**继续**是如果有失败的任务节点，其他并行的任务节点将继续执行。**结束**是如果有失的败节点，其他并行任的务节点将被Kill掉

通知策略：都不发、成功发、失败发、成功或失败都发四种策略

通知组：是通知用户所属的组

收件人：邮件收件人列表

抄送人：邮件抄送人列表

1. 上线/下线

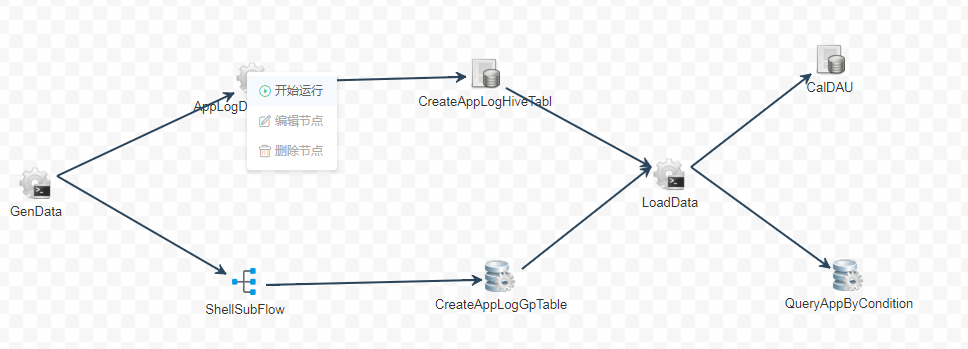
流程定义有两个状态：上线和下线。只有上线的流程才能启动或调度，下线的流程用来编辑

1. 定时管理

一个流程定义可以添加1个定时，定时管理是对一个流程定义上的定时的修改，只有**上线**的定时才是**有效**的定时

1. 指定节点启动

对流程定义可以选择节点右击 **指定节点启动**，如下图：



启动前可以设置若干参数，如下图：



失败策略：**继续**是如果有失败的任务节点，其他并行的任务节点将继续执行。**结束**是如果有失的败节点，其他并行任的务节点将被Kill掉

节点执行：节点执行模式有三种，分别为向后执行、向前执行和仅执行当前节点

向后执行是从指定节点向后执行，一直到后续节点不能继续执行为止。向前执行是从开始节点到指定的节点区间执行。仅执行当前节点是只执行指定的节点

通知策略：都不发、成功发、失败发、成功或失败都发四种策略

通知组：是通知用户所属的组

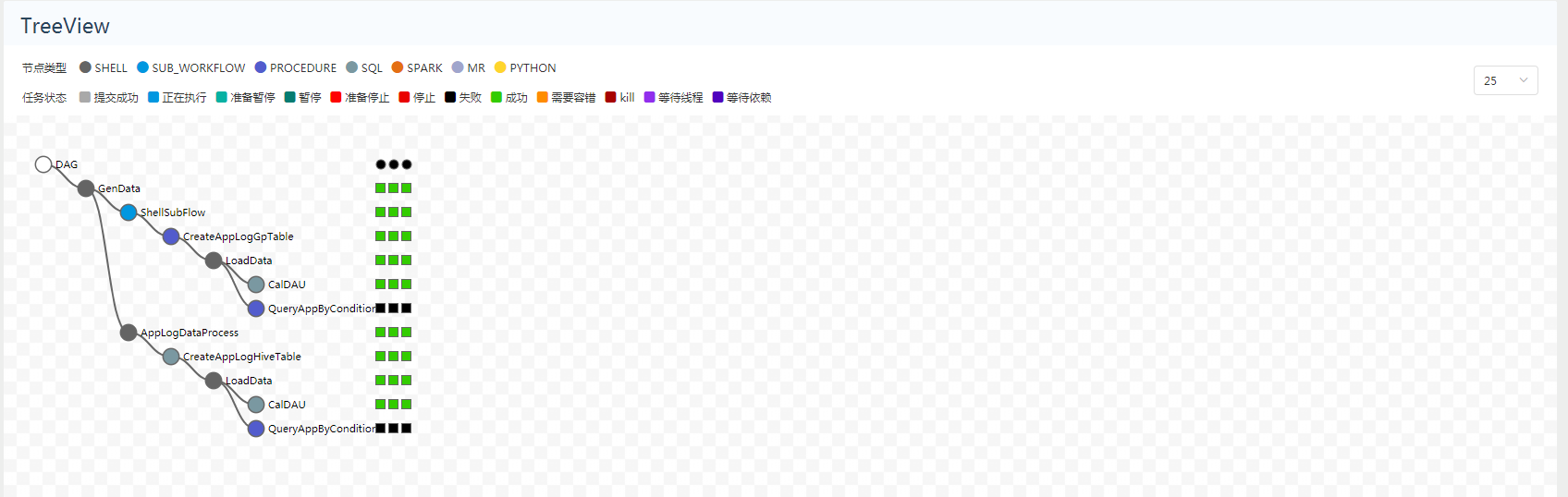
收件人：邮件收件人列表

抄送人：邮件抄送人列表

补数：见流程定义 **启动**

1. TreeView

用来显示某个流程定义跑过的流程实例及任务实例的状态

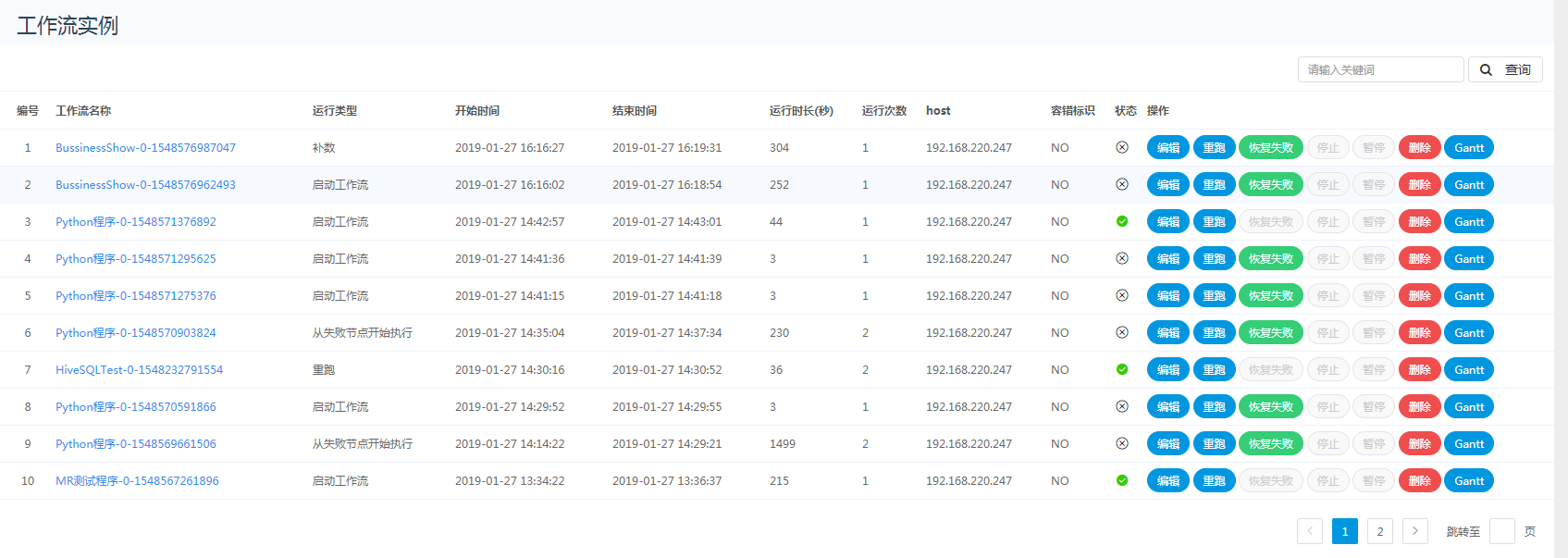


**左侧是流程定义 DAG Tree的展示**，非叶子节点有展开和收缩的功能。**右边是流程实例的集合**，每一列都是流程定义的一次调度，圆点表示的是流程实例的状态，方块表示的是任务实例的状态。如果是子流程，可以通过任务实例跳转到相应的子流程

### 流程实例

流程实例是流程定义的实例化

1. 流程实例列表



工作流名称：是根据流程定义名称外加一个随机数

运行类型：包括启动工作流、从当前节点开始执行、恢复被容错的工作流、恢复暂停流程、从失败节点开始执行、补数、调度执行、重跑、暂停、停止、恢复等待线程的流程

开始时间：流程实例运行的开始时间

结束时间：流程实例运行的结束时间

运行时长(秒)：结束时间减去开始时间

运行次数：流程实例运行的次数 ，对于重跑流程实例的记录

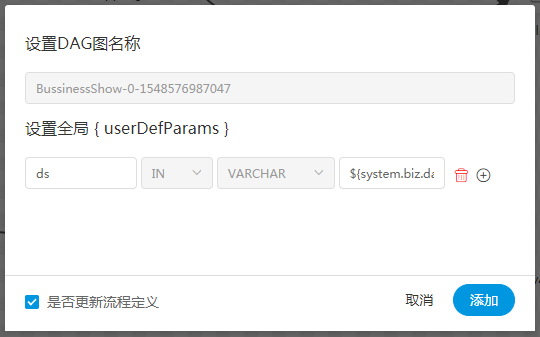
host：标识流程实例调度的机器ip

容错：流程实例是否被容错过

状态：流程实例最终运行的状态

1. 编辑/保存

对于**终态**的流程实例，可以进行编辑保存。**终态**是流程实例最终的状态为成功、失败、暂停或停止四个状态。



是否更新流程定义： 如果checkbox勾选，**表示流程实例的修改会更新流程定义**

如果不勾选**，表示只编辑本次流程实例**

1. 重跑

重跑是在不生成新的流程实例情况下，重新跑任务实例

特别注意：重跑时**系统参数**已经固化了，系统参数不会随着时间改变而改变

1. 恢复失败

点击 **恢复失败** 可以让流程实例从失败节点开始重新执行

1. 停止

点击 **停止** 将Kill掉正在运行的任务节点

特别注意：**如果任务运行周期比较短或者正好在Kill任务的时候，任务已经结束了，此时将Kill不掉正在运行的任务，继而会把下一个任务设置为Kill状态**

1. 暂停

对 **正在运行** 的流程实例，点击 **暂停** 可以让 **正在运行任务的后继节点** 设置为暂停

1. 恢复暂停

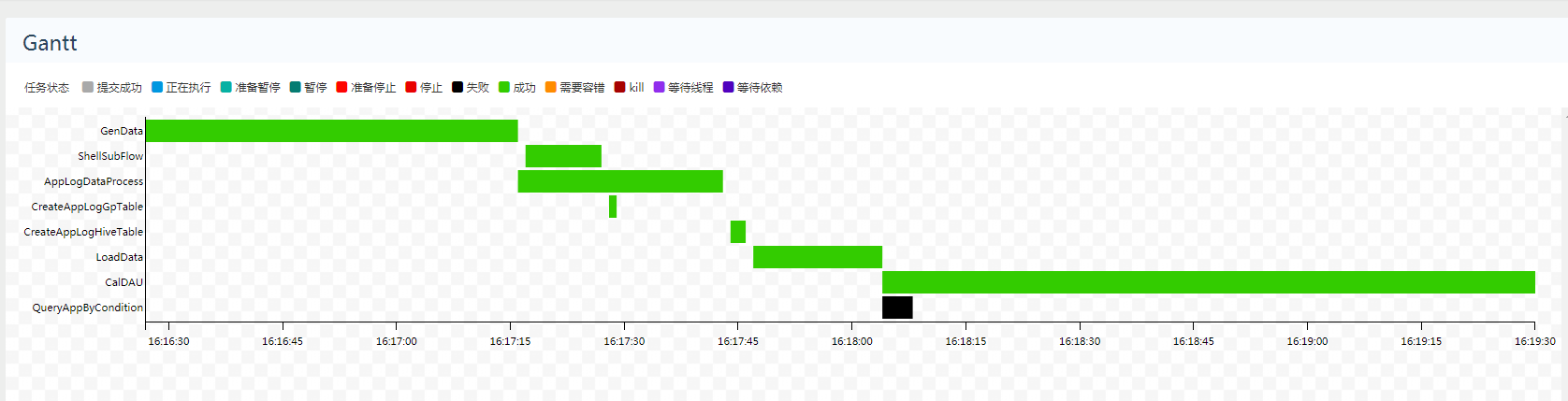
对于 **暂停** 的流程实例，点击 **恢复暂停** 可以让流程实例从暂停位置继续执行

1. 删除

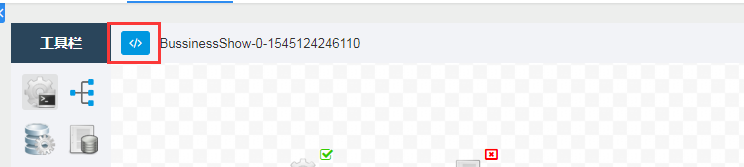
删除流程实例及流程实例下的任务实例

1. Gantt

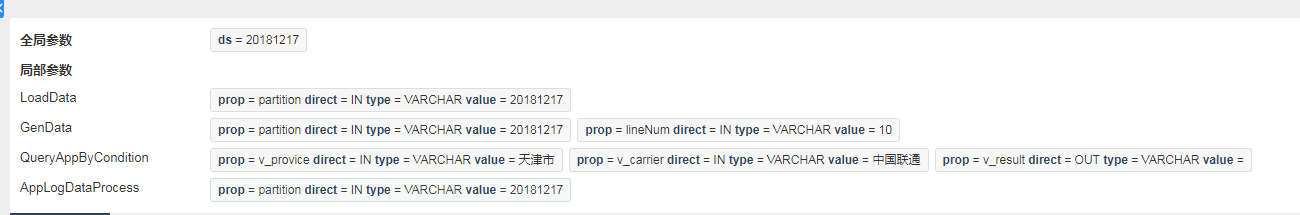
Gantt图纵轴是某个流程实例下的任务实例的拓扑排序，横轴是任务实例的运行时间



1. 查看流程实例运行变量



点击隐藏按钮，查看流程实例运行变量。如下图：



**点击变量是对变量的复制**

1. 查看历史任务实例

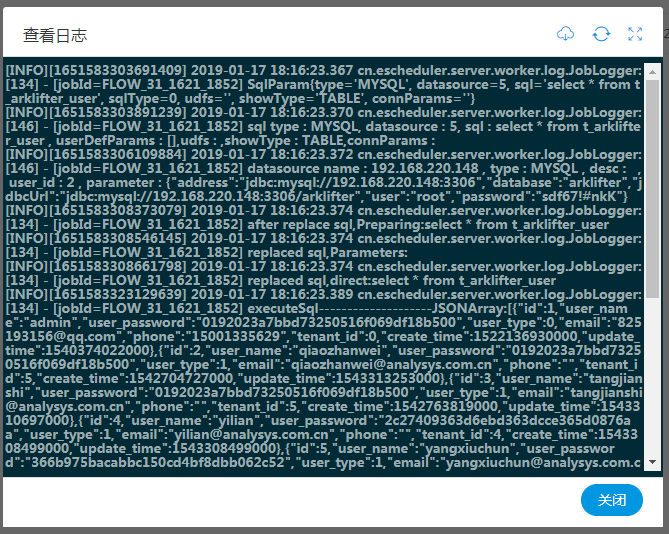


点击任务实例节点，点击 **查看历史，**可以查看该流程实例运行的该任务实例列表

1. 查看日志



点击任务实例节点，点击 **查看日志，**可以查看该任务实例运行的日志，如下图：

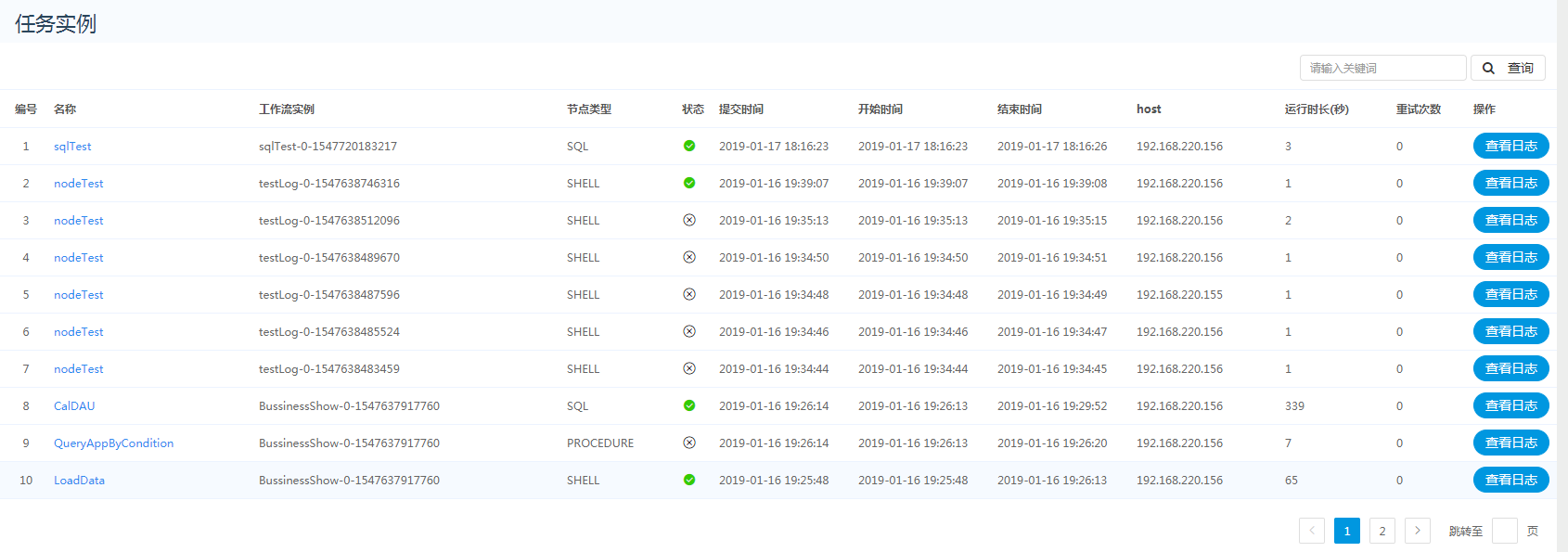


右上角是**下载日志**、**刷新日志**和**放大/缩小按钮**

**注意：日志查看是分片的查看，上下滚动查看**

### 任务实例

任务实例是流程实例任务节点的列表



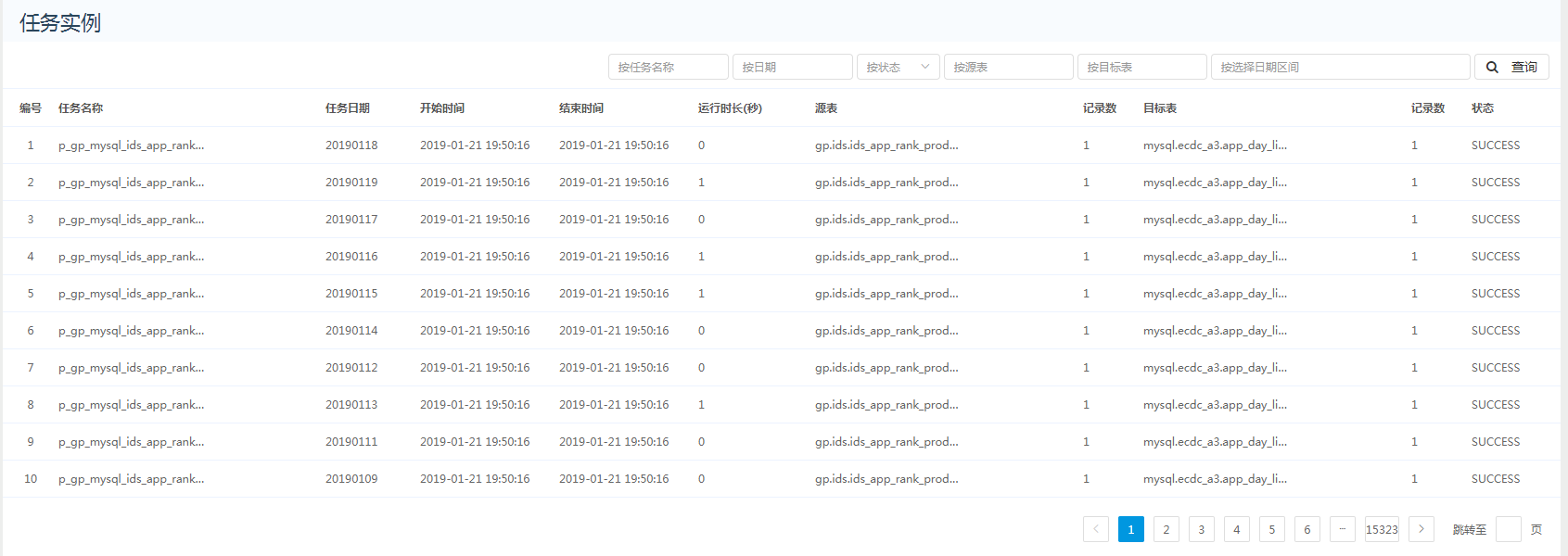
两种方式查看任务实例：

* 第一种是通过流程实例任务节点 **查看历史**，这时查看的是此流程实例的任务实例 **重跑的列表**
* 第二种是通过点击 **流程实例** 导航栏，调转到流程实例列表，这时查看的是**所有流程实例的任务实例列表**

查看日志：点击 **查看日志** 按钮，可下载和查看日志

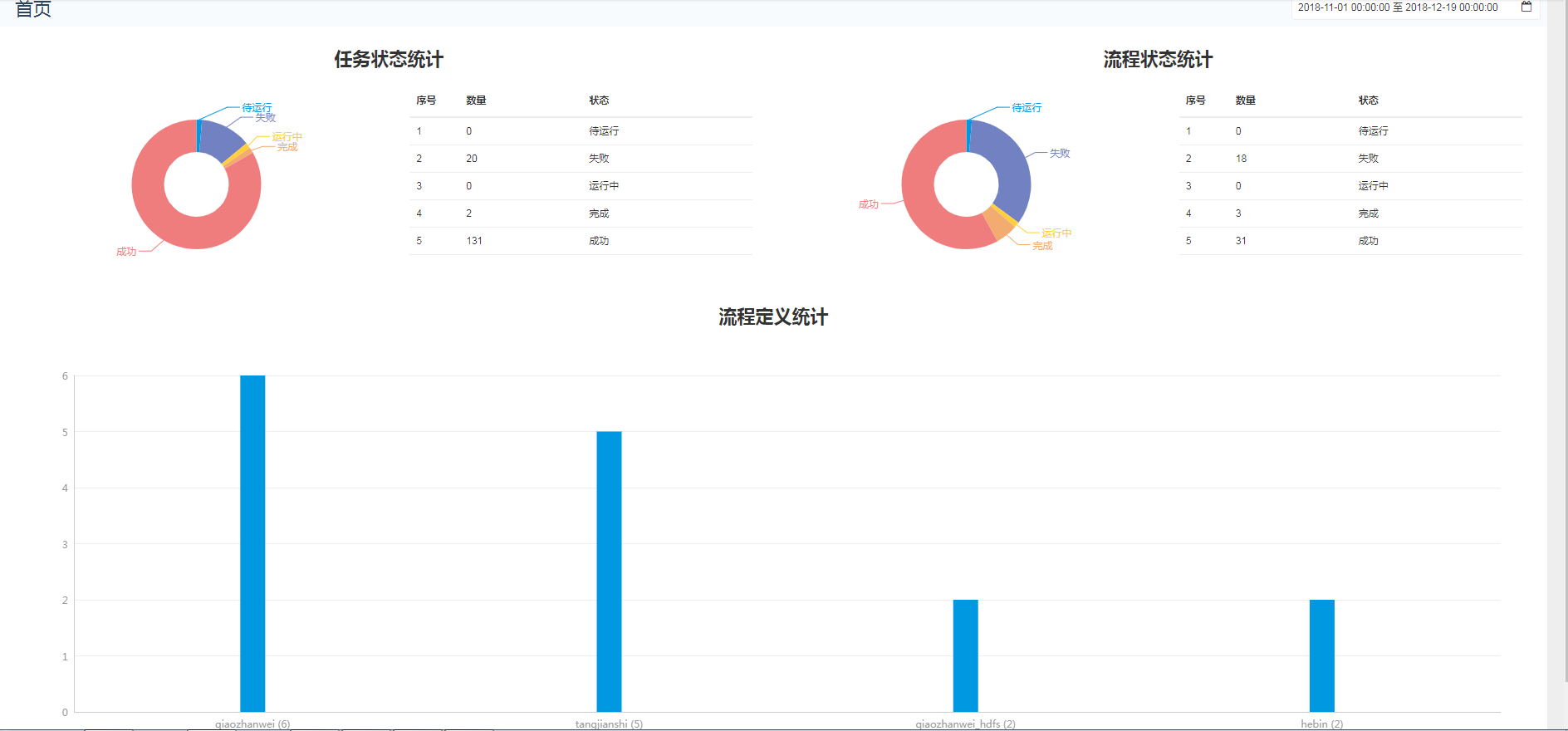
### 任务记录

任务记录是千帆的数据稽核



可以按照**任务名称、日期、状态、原表、目标表、日期区间**查询

# 首页



首页是对所有项目在指定时间范围内的任务状态、流程状态和流程定义的统计。

首页和项目首页的主要区别在于：

* 首页中的图表是没有链接的，项目首页中图表是有链接的
* 首页统计的是所有的项目，项目首页统计的是某一个项目

# 任务依赖

任务依赖分为**水平依赖**和**垂直依赖**

* 水平依赖就是指DAG图的有向依赖，是同一个流程实例任务节点的前驱，后继之间的依赖关系
* 垂直依赖是流程实例之间的任务依赖，**基于定时的依赖**。垂直依赖又分为**自依赖**和**外部依赖**。

**自依赖**：是同一个流程实例的上一周期的 **任务** 依赖，如图：



**外部依赖**：是对多个外部流程流程实例的任务实例的依赖

外部依赖策略存在依赖策略，分别有自定义、全部成功、全部失败、任一成功、任一失败

注意：自定义是可以自定义外部流程任务是成功或失败

依赖任务：



定时依赖图解： A流程定义某任务依赖B流程定义某任务，TA 为流程定义A的定时器，TB为流程定义B的定时器，TBStart到TBEnd为TB的周期，周期分别支持小时、天、月

* TA > TB 且 TA > TBEnd

TBStart TB TBEnd TA

只有在TB到TBEnd区间范围内符合任务实例依赖条件的，才能依赖成功。否则依赖失败

* TA > TB 且 TBEnd > TA

TBStart TB TA TBEnd

只有在 TB 到 TA区间范围内符合任务实例依赖条件的，才能依赖成功。否则依赖失败

# 系统参数

* 系统参数

|  |  |
| --- | --- |
| 变量 | 含义 |
| ${system.biz.date} | 日常调度实例定时的定时时间前一天，格式为 yyyyMMdd，补数据时，该日期 +1 |
| ${system.biz.curdate} | 日常调度实例定时的定时时间，格式为 yyyyMMdd，补数据时，该日期 +1 |
| ${system.datetime} | 日常调度实例定时的定时时间，格式为 yyyyMMddHHmmss，补数据时，该日期 +1 |

* 时间自定义参数

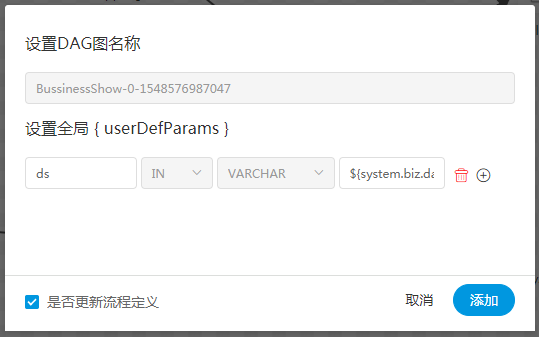
支持代码中自定义变量名，声明方式：${变量名}。可以是引用 "系统参数" 或指定 "常量"。

注意，我们定义这种基准变量为 $[...] 格式的，$[yyyyMMddHHmmss] 是可以任意分解组合的，比如：$[yyyyMMdd], $[HHmmss], $[yyyy-MM-dd], $[HH:mm:ss] 等

也可以这样：

* 后 N 年：$[add\_months(yyyyMMdd,12\*N)]
* 前 N 年：$[add\_months(yyyyMMdd,-12\*N)]
* 后 N 月：$[add\_months(yyyyMMdd,N)]
* 前 N 月：$[add\_months(yyyyMMdd,-N)]
* 后 N 周：$[yyyyMMdd+7\*N]
* 前 N 周：$[yyyyMMdd-7\*N]
* 后 N 天：$[yyyyMMdd+N]
* 前 N 天：$[yyyyMMdd-N]
* 后 N 小时：$[HHmmss+N/24]
* 前 N 小时：$[HHmmss-N/24]
* 后 N 分钟：$[HHmmss+N/24/60]
* 前 N 分钟：$[HHmmss-N/24/60]
* 用户自定义参数

用户自定义参数分为**全局参数**和**局部参数**。全局参数是保存流程定义和流程实例的时候传递的全局参数，全局参数可以在整个流程中的任何一个任务节点的局部参数引用。例如：



ds为全局参数，引用的是系统参数。



任务中partition通过${ds}来引用全局参数，对于脚本可以通过${partition}来引用变量partition的值，或通过JDBC直接将partition的值set进去