

鲁班大数据开发平台

用户指南

v1.1

目录

! " # \$ % & ' #
* + # #
, - # #
. / # #
O1 # #
3" # 4567 # #
9: ! ; < = > ? @ABC # #
D/ # #
E F G H # #
IJKL#M; NOPQRST UV # #
IJKL#W; XYZ F [# #
IJKL#]; ^ _ Z F [# #
IJKL#`; a b c d R e f g h # #
IJKL#); i j k l # #
9: 3; mn <= o p q r s B C # #
D/ # #
E F G H # #
IJKL#M; NOPQRST UV # #
IJKL#W; XYZ F [# #
IJKL#]; Y u Y I v Z F [w # #
IJKL#`; y z P Q R d { u l | # #
IJKL#); } ~ P Q • Z [€ # #
IJKL#2; a b c d R e f g h # #
IJKL#); i j k l # #
• " #, f „ ... # # x #
Z F [# # x #
† ≠ ^ % o # # x #
„ ... [€ # # x #
IJKL#M; Š 6 Z F [< DE # # x #
IJKL#W; ^ _ Z F [• Ž # # W #
IJKL#]; • • Z F [• Ž # #] #
P Q ' ' " • -- ~ TM Š o # #] #
† ≠ ^ % o # #] #
„ ... [€ # #] #
IJKL#M; ž • • • ... q # #` #
IJKL#W(... q Y i j # #` #
IJKL#]; E ☒ ¥ | • • ... q ; # #` #
IJKL#` # > § P Q .. ☒ a # #` # 2 #
IJKL#); > Y c d ; # #` # 8 #
IJKL#2# > § P Q « - # #` # 8 #
IJKL#8# > § P Q .. ☒ # #` # \ #
IJKL#W§ • • Ž ; # #` # x #



W(# # ŠK™Í ~J~õM" ŠK™Í ~J~õW" ŠK™Í ~J~õ] Ó Õ "] Ö Ö Ö ÷ Ø • [(B) #
] (# # ÙKúüüÍ ~ü Ó Ô #[(B8#
` (# # ÙKšùKúüüÝ KKþ Ó Ô #[(B8#
) (# # →äKýÍ MüøqÉ Ó Ô ; #[(Bt#

6. mûøqföDDF 代码: #[(Bx#
ñ ð W; mn < = opqr s BC9: Ó Õ #[(AM#

M#JÍ üCM 节点代码 #[(AM#

W(I PÜRKML 代码 #[(AN#

] JÍ ucw 代码 #[(A#

`(JÍ uc] 代码 #[(A2#

#

#

一、产品简介

概述

#

鲁班是天翼云大数据平台类产品，基于 Hadoop 技术，为您提供从数据采集、数据存储、数据计算、数据开发、资源调度、数据治理、数据服务到可视化展现的一站式解决方案，并提供平台自动化部署、运维和安全保障，以及多租户、分权分域等能力。

鲁班能够轻松处理多种数据，并进行**离线计算、实时计算、交互式查询**等，同时，可以**发布数据 API**，轻松开放数据能力。

利用鲁班能够在线编写数据处理程序，从而完成数据的清洗、加密、脱敏、打标、元数据抽取等多方面的数据治理任务，让您摆脱繁杂的系统运维工作，专注于数据本身

#

优势

技术领先：采用最新稳定版本的大数据技术，并与国内领先的大数据团队共同维护，快速提升数据处理效率

开箱即用：产品内包含的通信、金融企业级的大数据处理模板工具，让您在自己的数据系统中快速使用我们多年沉淀的数据处理经验及方法

可视化开发：鲁班提供了开发 IDE 套件，支持在线编写多种语言脚本，并一键提交到大数据集群执行，返回执行状态和结果；支持页面拖拽式的工作流编辑与发布；丰富的数据可视化组件，拖拽式配置大屏或 BI 展示且支持 UDF、数、资源 和能 等企业级

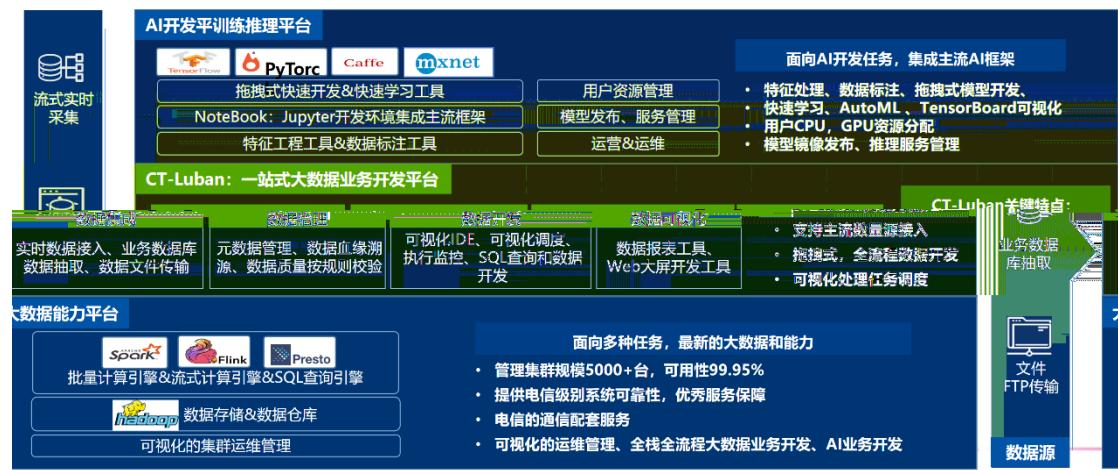
场景

1： 信行业大数据分 利用大数据技术实现 户离 分 ，及时 户离 ， 台 户

2： 行业大数据分 利用 大数据分 ， 站， 站， ， 人 流信 ，
，并提供人流 流， 价等决 信 支

3 : 工业大数据分利用工业大数据提升业平,包产品障与进产工化产程能、产计与程

架构



数据开发的流程 下：

- 数据产生:** 业务系统 天产大数据, 存储在业务系统 的数据 中, 包括 化和 结合化数据
- 数据集成:** 您 同 不同业务系统的数据 鲁班中, 方可通 鲁班的 数据存储与处理能力分 的数据。鲁班提供数 据集成服务, 可以支持多种数据源类 , 据 的调度 同 业务系统的数据。
- 数据开发:** 完成数据的同 , 可以 数据进行加工、分 与 (数据开发) 等处理, 从而发现 价 。
- 数据使用:** 分 与处理 的结果数据, 可以供业务人 使用 分 的价 , 并可 数据治理。
- 数据展现与发布:** 处理成 , 可以通 、大屏服务展示大数据分 、处理 的成果; 也可发布数据 api 开放数据能力。

#

#

二、快速入门

案例一：公司新闻舆情分析

景

分 行业内 个 的企业新 数据，为 类新 打标（ 面、中 、 面）， 分 行业 面新
。新 数据 下：

ID	TITLE	DATE	COMP_NAME	NEWS_CODE	SENTIMENT
记录 ID	新闻标题	新闻日期	企业名字	新闻编号	新闻标签

为 成 个时 的企业 面新 数数据：

DATE	COMP_NAME	week1	week2	week3	week4	week5	week6	week7	week8
日期	企业名字				往前	第 N 周	负面	新闻	总数

操作步

Step 1：上 数据 鲁班平台

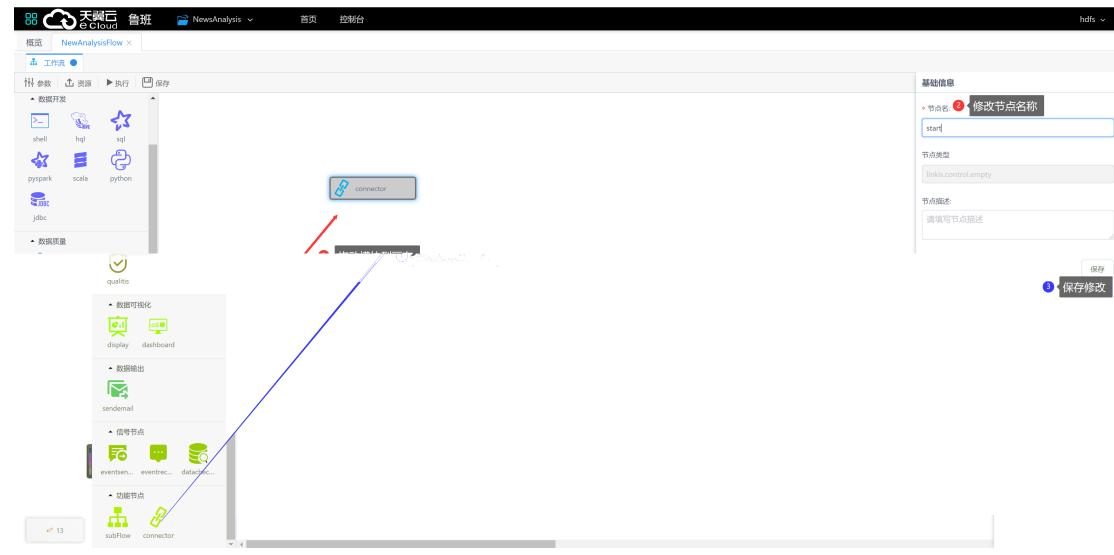
The screenshot shows the RuBian (LuBan) platform interface. On the left, there's a sidebar with '工作空间' (Workspace), '数据仓库' (Data Warehouse), 'UDF函数' (UDF Functions), and '方法函数' (Method Functions). The main area shows a tree view of an 'hdfs' directory containing various dates from 20200701 to 20200801. A context menu is open over the '20200801' folder, with option ① '选择HDFS' (Select HDFS) highlighted. Another context menu is open over a 'hive' folder, with option ② '创建文件夹, 上传sample2020.csv文件' (Create folder, upload sample2020.csv file) highlighted. A message at the bottom says '文件上传成功, 并复制文件路径' (File uploaded successfully, and copied file path). To the right, there's a '帮助中心' (Help Center) icon and a status bar showing '15'.

1. 工作 → 数据分 → scripts-> HDFS
2. 键 hdfs , 件 NewAnalysis
3. 键 NewAnalysis 上 件
4. 键 件, 件 , 键 件 , 件

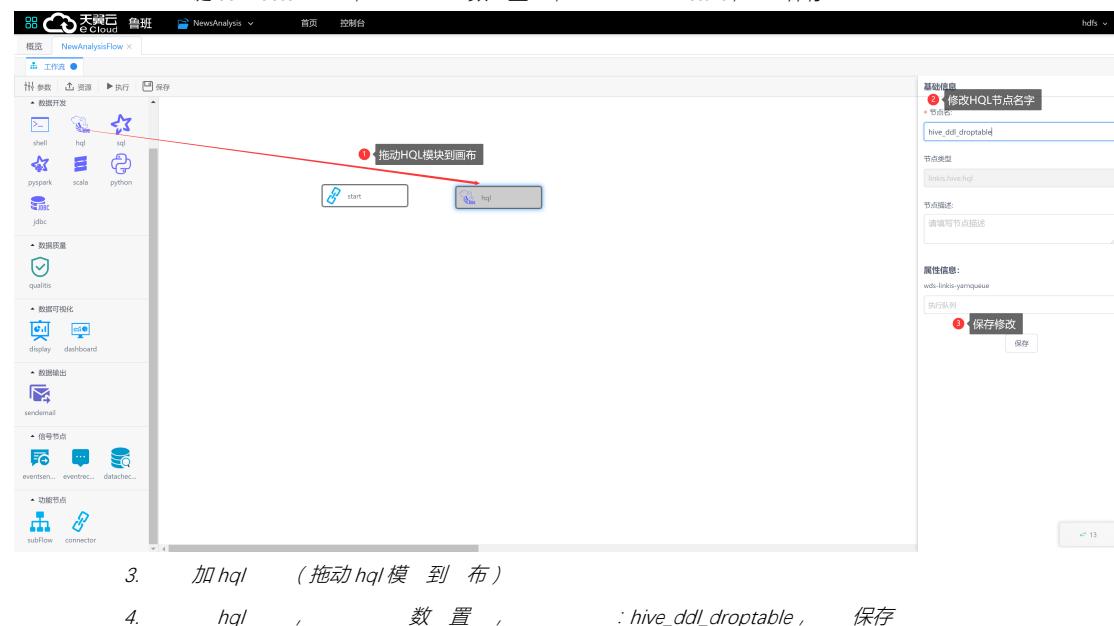
Step 2 : 工作流

在工作流中，【用开发】标签进入【工作流开发】页面，【工程】，工程信息，【工作流】，工作流信息，成功的工作流，进入工作流开发页面。

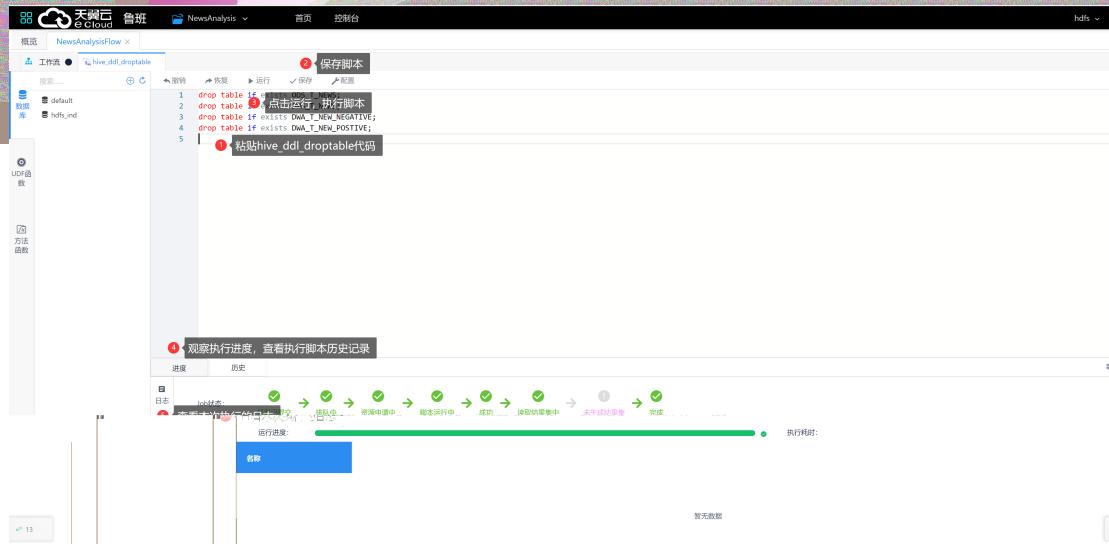
Step 3 : 编辑工作流



1. 加 connector (拖动 connector 模块到布)
2. 键 connector , 数置 , : start , 保存



3. 加 hql (拖动 hql 模块到布)
4. hql , 数置 , : hive_ddl_droppable , 保存



5. 进 编辑 面, 下代

```
drop table if exists ODS_T_NEWS;
drop table if exists DWS_T_NEWS;
drop table if exists DWA_T_NEW_NEGATIVE;
drop table if exists DWA_T_NEW_POSITIVE;
```

6. 保存, 保存脚本 , 运行, 请

以上 分 加 代 _____):

1. 1 个 Hql , 为 hive_ddl_createtable
2. 4 个 Scala : , 为 genData_1、NewByDay、NegNewByWeek、hiveToMysql
3. 1 个 jdbc , 为 mysql ddl
4. 1 个 qualitis , 为 qualitis

, genData_1 , 分 , 为 genData_2 和 genData_3

中, qualitis 配置 下:

1. 拖动 qualitis 标 布
2. qualitis , 进 数据 开发 面, 并配置 下

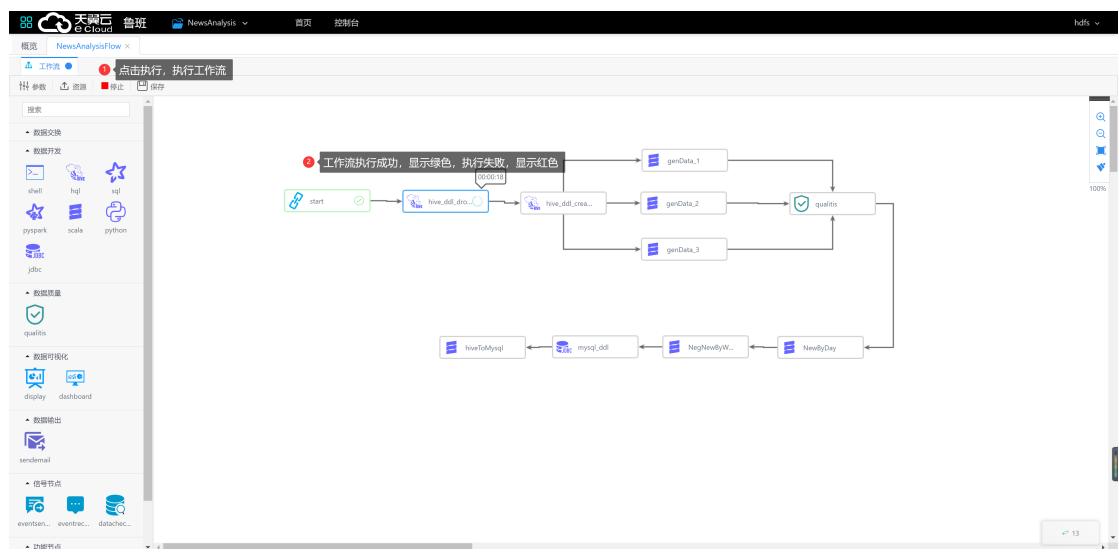
添加校验规则NullValueCheck

② 编辑校验规则。通过下拉框选择相关内容

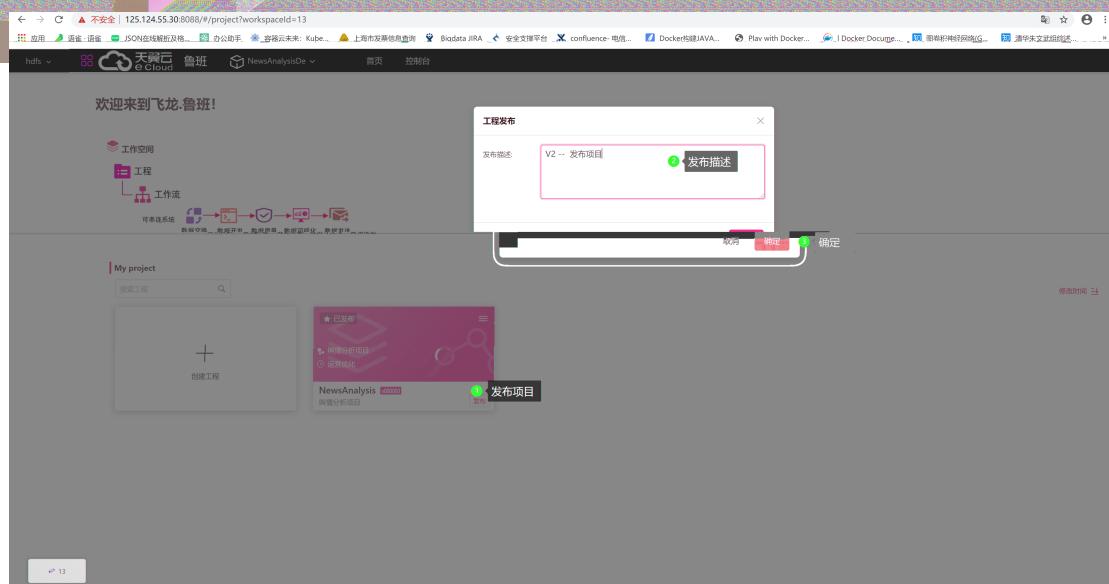
① 过滤条件编写

③ 校验配置（从工作空间-> 数据质量进入，这个指标会议监控形式展示出来）

最，加的系，下，使用标线即可：



Step 4 : 发布 调度系统



1. 工作 → 用开发→ 工作流开发-> HDFS

2. 发布, 发布



3. 进【工作】→【产运维】→【Schedulis】能中，到

4. 【执行工作流】，开 运行 工作流

Step 5：定

1. 进 【工作】->【数据分】->【visualis】 进 与 同 的可视化

2. 加数据源 -> 编辑数据源 , 键入 下

jdbc:mysql: 192.168.0.38:3306 用户 / 密 : root / Bigdata2@20

3. 视 , 新视 (上)

COMPNAME	WEEK1COUNT
荣安德控股有限公司	7
上海得联电子商务股份有限公司	4
软控股份有限公司	1
深圳得利电子科技股份有限公司	5
通鼎互联信息股份有限公司	4
贵州圣济堂医药产业股份有限公司	5

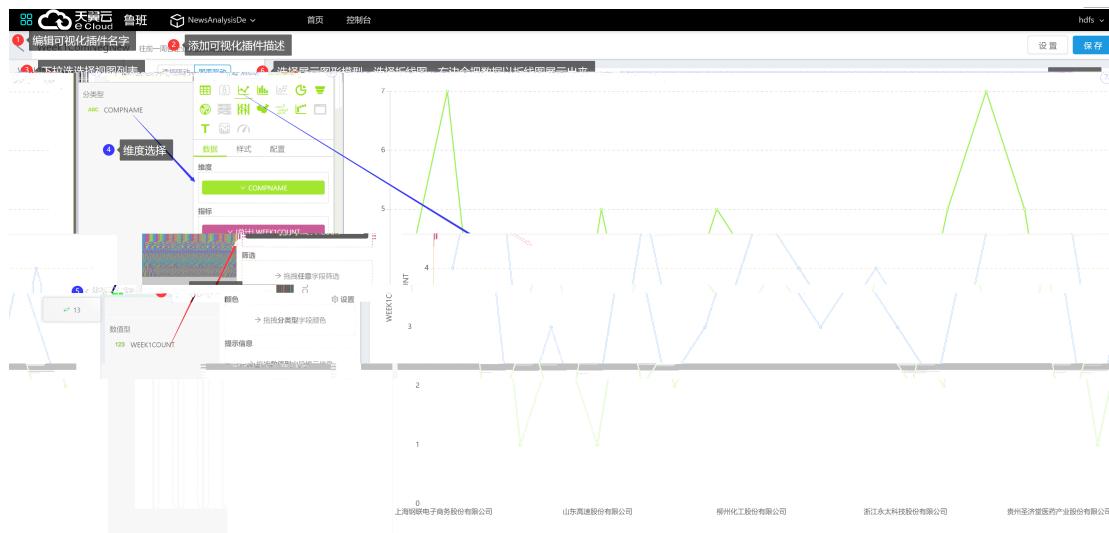
4. 编辑视 , 视 , 及 [SQL 语]

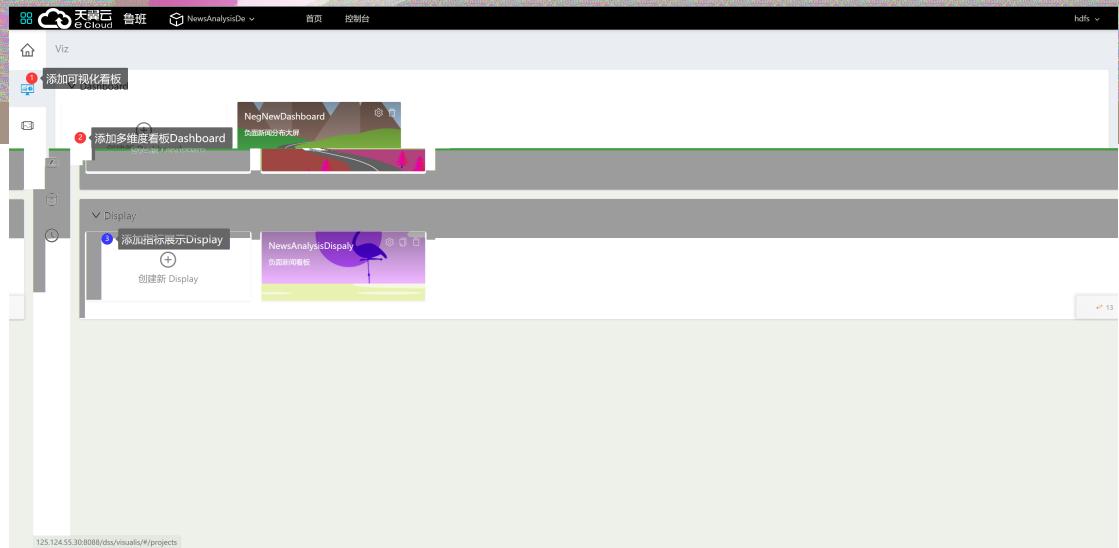
- Week1NegCount 的 SQL 语句 : `select COMPNAME, WEEK1COUNT FROM news.DWA_T_NEW limit 20;`
 - 8WeeksNegNewCount 的 SQL 语句 :
- ```
select COMPNAME, WEEK1COUNT, WEEK2COUNT, WEEK3COUNT,
WEEK4COUNT, WEEK5COUNT, WEEK6COUNT, WEEK7COUNT, WEEK8COUNT,
(WEEK1COUNT + WEEK2COUNT+ WEEK3COUNT+ WEEK4COUNT + WEEK5COUNT + WEEK6COUNT + WEEK7COUNT +
WEEK8COUNT) totalCount FROM news.DWA_T_NEW order by totalCount desc LIMIT 20;
```

Widget List

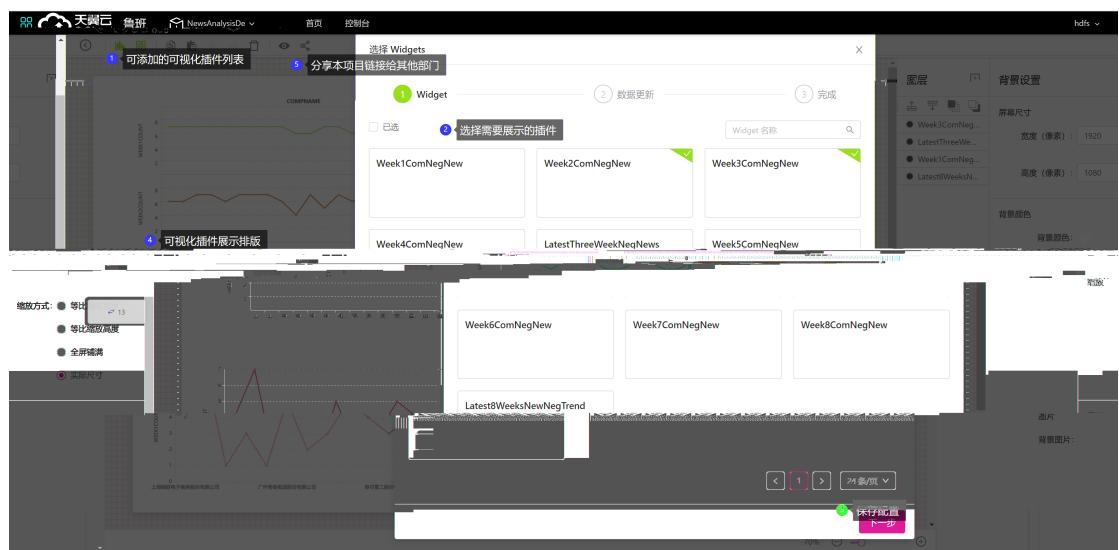
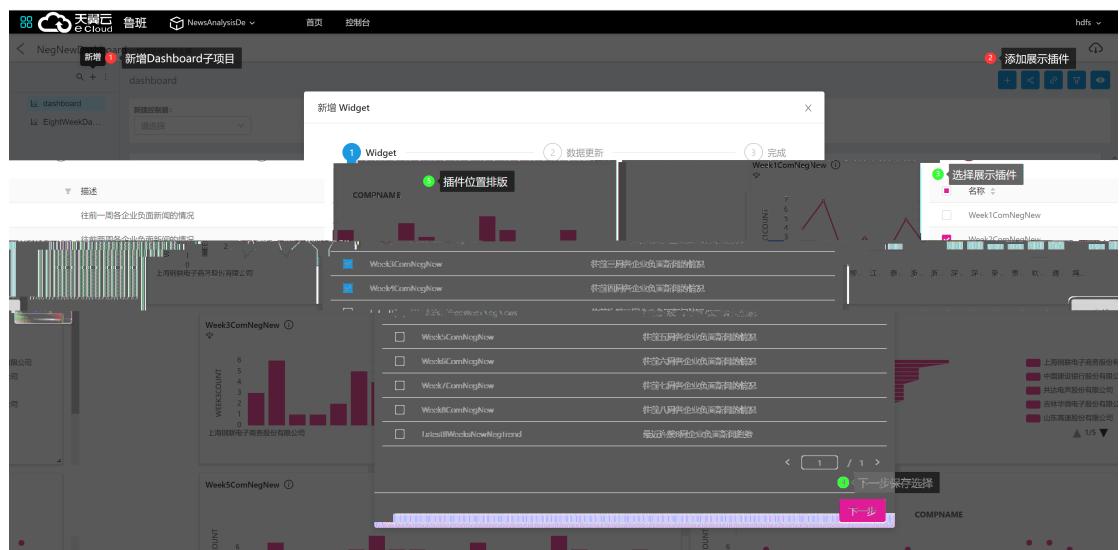
| 名称                     | 描述             | 操作 |
|------------------------|----------------|----|
| Week1NegCount          | 往前一周各企业负面新闻的情况 |    |
| Week2ComNegNew         | 往前两周各企业负面新闻的情况 |    |
| Week3ComNegNew         | 往前三周各企业负面新闻的情况 |    |
| Week4ComNegNew         | 往前四周各企业负面新闻的情况 |    |
| LatestThreeWeekNegNews | 往前连续三周企业负面新闻情况 |    |
| Week5ComNegNew         | 往前五周各企业负面新闻的情况 |    |
| Week6ComNegNew         | 往前六周各企业负面新闻的情况 |    |
| Week7ComNegNew         | 往前七周各企业负面新闻的情况 |    |
| Week8ComNegNew         | 往前八周各企业负面新闻的情况 |    |
| test8WeeksNewNegTrend  | 最近连续8周企业负面新闻趋势 |    |

< > 20条/页

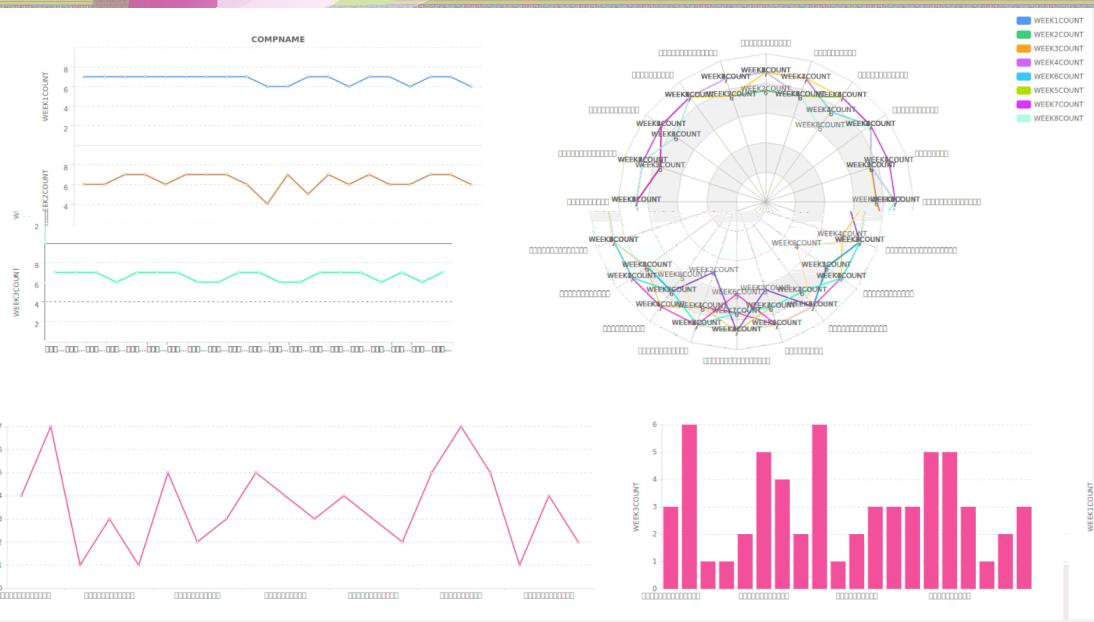




### 1. 加可视化 板 Dashboard 和 display



### 1. 加可视化 件



最可到上的分结果

## 案例二：航空公司的客户价值分析

景

本案的标是户价，即通过客户数据，户的价。使用3个标（最近时、频率率和金额）——RFM模进行户分。本案基于模进行化计，下示：

| 模型              | L [单位：月]             | R [单位：月]               | F [单位：次]          | M[单位：公里]        | C                        |
|-----------------|----------------------|------------------------|-------------------|-----------------|--------------------------|
| 航空公司<br>LRFMC模型 | 会员入会时间<br>距观测窗口结束的月数 | 客户最近一次乘坐公司飞机距观测窗口结束的月数 | 客户在观测窗口内乘坐公司飞机的次数 | 客户在观测窗口内累计的飞行里程 | 客户在观测窗口内乘坐舱位所对应的折扣系数的平均值 |

| 字段中文    | 字段英文              |
|---------|-------------------|
| 会员卡号    | MEMBER_NO         |
| 入会时间    | FFP_DATE          |
| 第一次飞行日期 | FIRST_FLIGHT_DATE |
| 性别      | GENDER            |

会员卡级别

FFP\_TIER

工作地城市

WORK\_CITY

工作地所在省份

WORK\_PROVINCE

工作地所在国家

WORK\_COUNTRY

年龄

age

观测窗口的结束时间

LOAD\_TIME

飞行次数

FLIGHT\_COUNT

观测窗口总基本积分

BP\_SUM

第一年精英资格积分

EP\_SUM\_YR\_1

第二年精英资格积分

EP\_SUM\_YR\_2

第一年总票价

SUM\_YR\_1

第二年总票价

SUM\_YR\_2

观测窗口总飞行公里数

SEG\_KM\_SUM

观测窗口总加权飞行公里数 ( $\Sigma$  舱位折扣 $\times$ 航段距离)

WEIGHTED\_SEG\_KM

末次飞行日期

LAST\_FLIGHT\_DATE

观测窗口季度平均飞行次数

AVG\_FLIGHT\_COUNT

观测窗口季度平均基本积分累积

AVG\_BP\_SUM

观察窗口内第一次乘机时间至 MAX (观察窗口始端, 入会

时间) 时长

最后一次乘机时间至观察窗口末端时长

LAST\_TO\_END

平均乘机时间间隔

AVG\_INTERVAL

观察窗口内最大乘机间隔

MAX\_INTERVAL

观测窗口中第 1 年其他积分 (合作伙伴、促销、外航转入等)

ADD\_POINTS\_SUM\_YR\_1

观测窗口中第 2 年其他积分 (合作伙伴、促销、外航转入等)

ADD\_POINTS\_SUM\_YR\_2

积分兑换次数

EXCHANGE\_COUNT

平均折扣率

avg\_discount

第 1 年乘机次数

P1Y\_Flight\_Count

第 2 年乘机次数

L1Y\_Flight\_Count

第 1 年里程积分

P1Y\_BP\_SUM

第 2 年里程积分

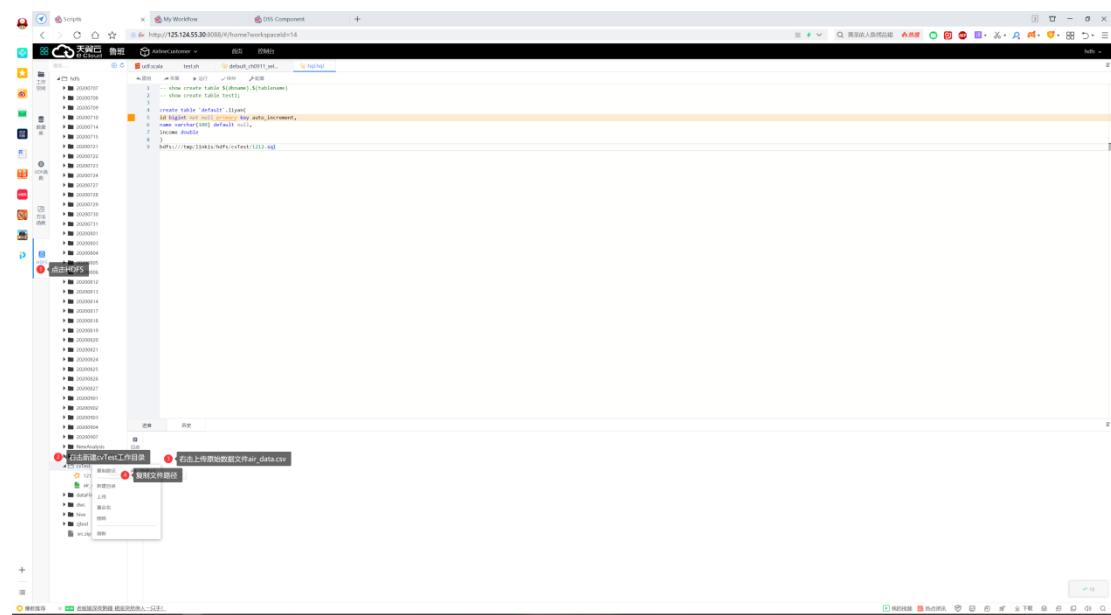
L1Y\_BP\_SUM



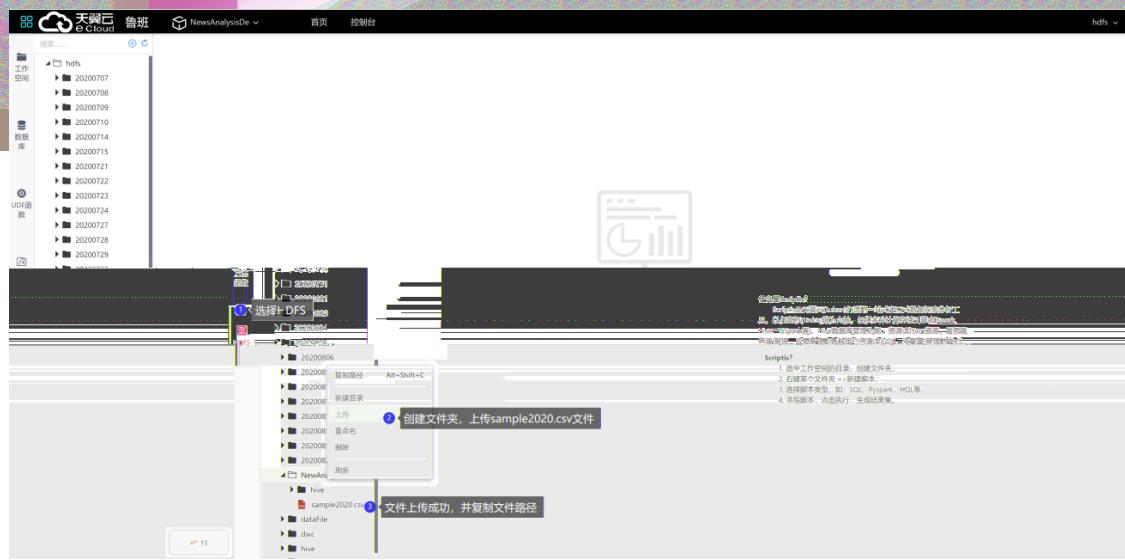
|                          |                         |
|--------------------------|-------------------------|
| 观测窗口总精英积分                | EP_SUM                  |
| 观测窗口中其他积分（合作伙伴、促销、外航转入等） | ADD_Point_SUM           |
| 非乘机积分总和                  | Eli_Add_Point_Sum       |
| 第2年非乘机积分总和               | L1Y_ELi_Add_Points      |
| 总累计积分                    | Points_Sum              |
| 第2年观测窗口总累计积分             | L1Y_Points_Sum          |
| 第2年的乘机次数比率               | Ration_L1Y_Flight_Count |
| 第1年的乘机次数比率               | Ration_P1Y_Flight_Count |
| 第1年里程积分占最近两年积分比例         | Ration_P1Y_BPS          |
| 第2年里程积分占最近两年积分比例         | Ration_L1Y_BPS          |
| 非乘机的积分变动次数               | Point_NotFlight         |

操作步

## Step 1：上 数据 鲁班平台

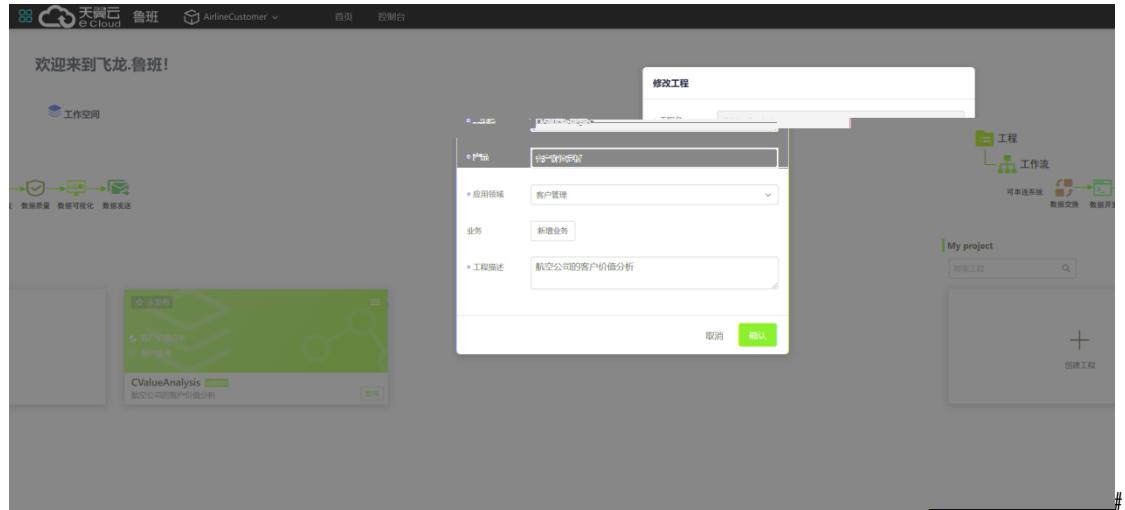


5. 工作 → 数据分 → scripts -> HDFS
6. 键 hdfs , 件 cvTest
7. 键 cvTest 上 数据 件 ( 件 ) air\_data.csv
8. 键 件 , 件 , 键 件 , 件



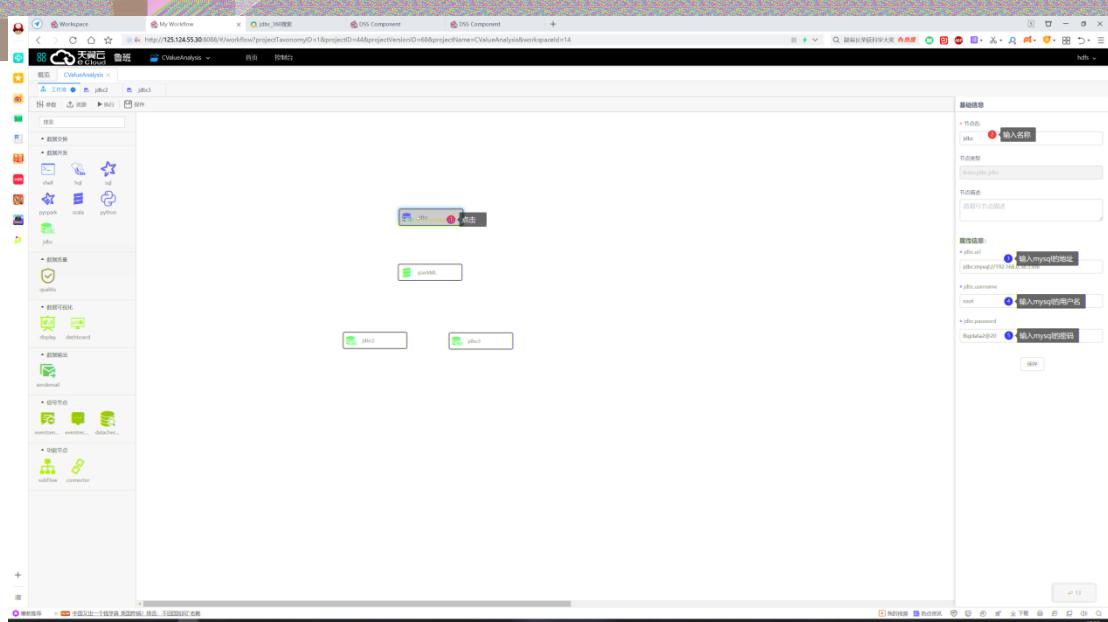
## Step 2 : 工作流

在工作 中，【用开发】标，进【工作流开发】页面，【工程】，工程信，【工作流】，工作流信，成的工作流，进 工作流开发页面

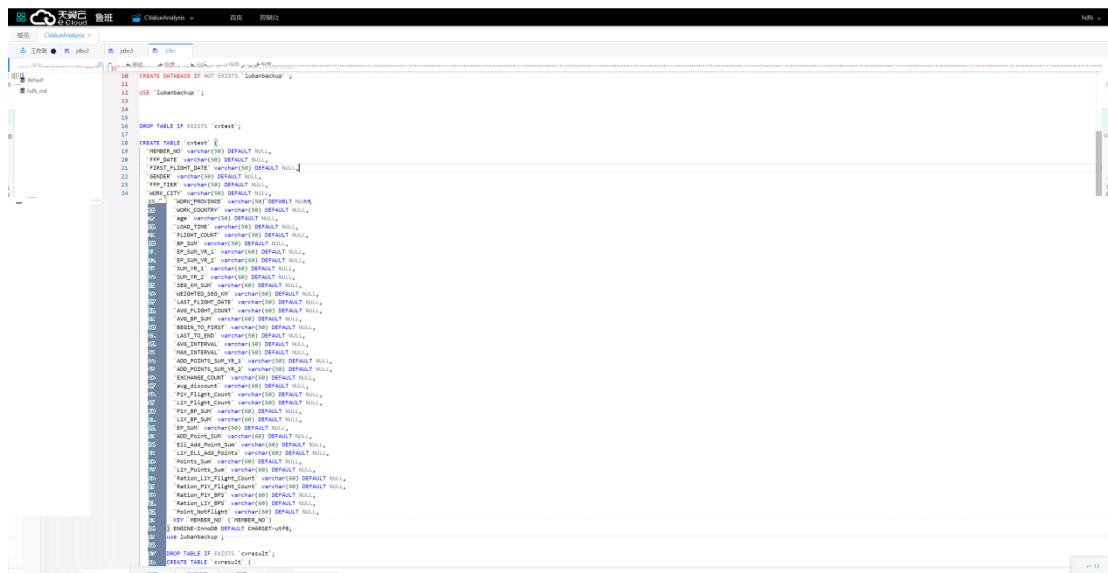


### Step 3 :

( 工作流 )

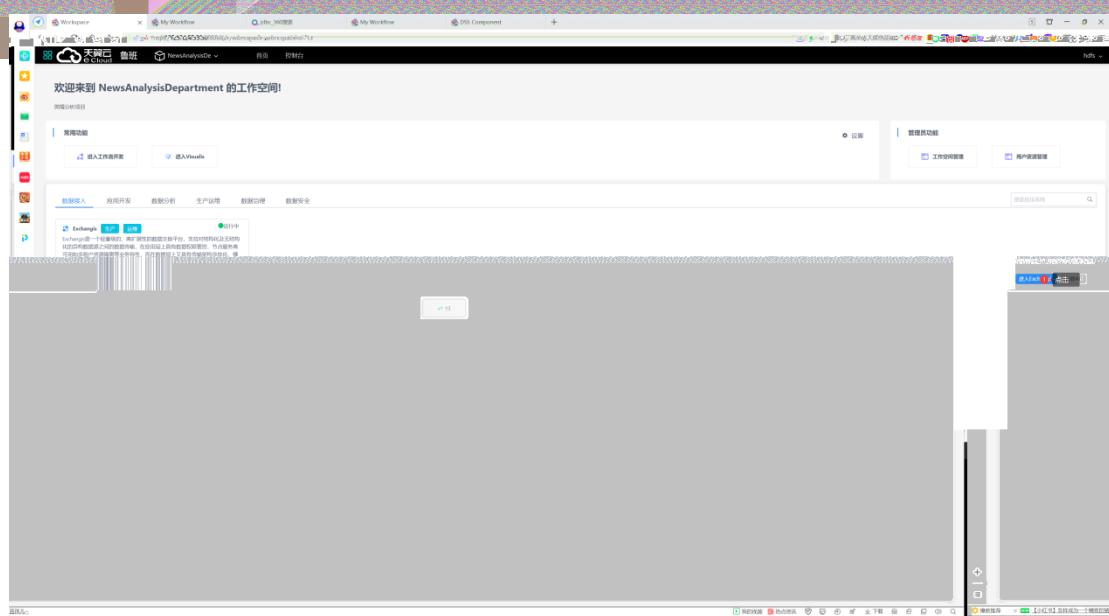


1. 加 JDBC (拖动 JDBC 模块到布)
  2. 键 JDBC , 数置 , : JDBC , mysql 的 URL、用户和密等信 , 保存

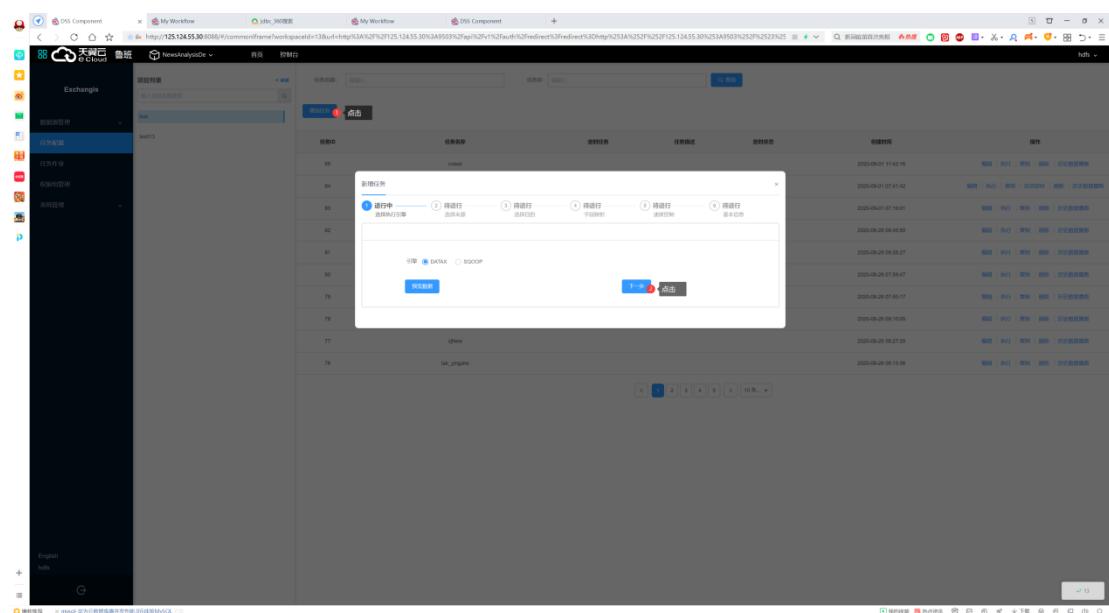


3. 编辑 JDBC , 工程 数据 ( *lubanbackup* ) 数据 ( *cvttest* ) 及结果数据 ( *cvresult* )

## Step 4 : 数据 标 中



1. 工作 → 数据 → 进 Exchangeis



2. 任务配置→任务配置→ 加任务

#

The screenshot shows the Oceansoft Cloud Data Pipeline interface. A modal window titled "新增任务" (Add Task) is open, specifically for step 2: "选择执行引擎" (Select Engine). The engine selected is "hdfs". The "选择来源" (Select Source) tab is active. The configuration details include:

- 数据源类型: hdfs
- 数据源: /tmp/linkis/hdfs/cvTest/air\_data.csv
- 路径: /tmp/linkis/hdfs/cvTest/air\_data.csv
- 压缩类型: 请选择压缩类型
- 编码: UTF-8
- 传输方式: 记录
- 文件类型: csv
- 字段分隔符: ,

At the bottom right of the modal is a button labeled "下一步" (Next Step) with a red circle containing the number 6 and the text "点击" (Click).

### 3. 配置数据 源

The screenshot shows the Oceansoft Cloud Data Pipeline interface. A modal window titled "新增任务" (Add Task) is open, specifically for step 3: "选择目的" (Select Destination). The destination selected is "mysql". The "选择来源" (Select Source) tab is active. The configuration details include:

- 数据源类型: mysql
- 数据源: lubanbackup
- 库名: lubanbackup
- 表名: cvtest
- 写入方式: insert
- 批量大小: 请输入批量大小值

At the bottom right of the modal is a button labeled "下一步" (Next Step) with a red circle containing the number 6 and the text "点击" (Click).

### 4. 配置数据 目的

cvtest

新增任务

1 已完成 选择执行引擎    2 已完成 选择来源    3 已完成 选择目的    4 进行中 速度控制    5 待进行 基本信息

速度控制

\* 作业速率限制: 10 Mb/s

\* 作业记录数限制: 10000 条/s

\* 错误记录数超过: 1 条, 任务失败

高级

上一条 下一步 ① 点击

5. 配置 JOB 数

cvtest

新增任务

1 已完成 选择执行引擎    2 已完成 选择来源    3 已完成 选择目的    4 已完成 速度控制    5 进行中 基本信息

保存预览

任务名称: cvtest 提醒人: 请输入提醒人

定时: corn表达式 任务描述: 请输入任务描述

每00 分钟

执行用户: hdfs 执行节点: 192.168.0.154:9501

超时时间: 43200 S 同步方式: 全量(FULL) 增量(INCR)

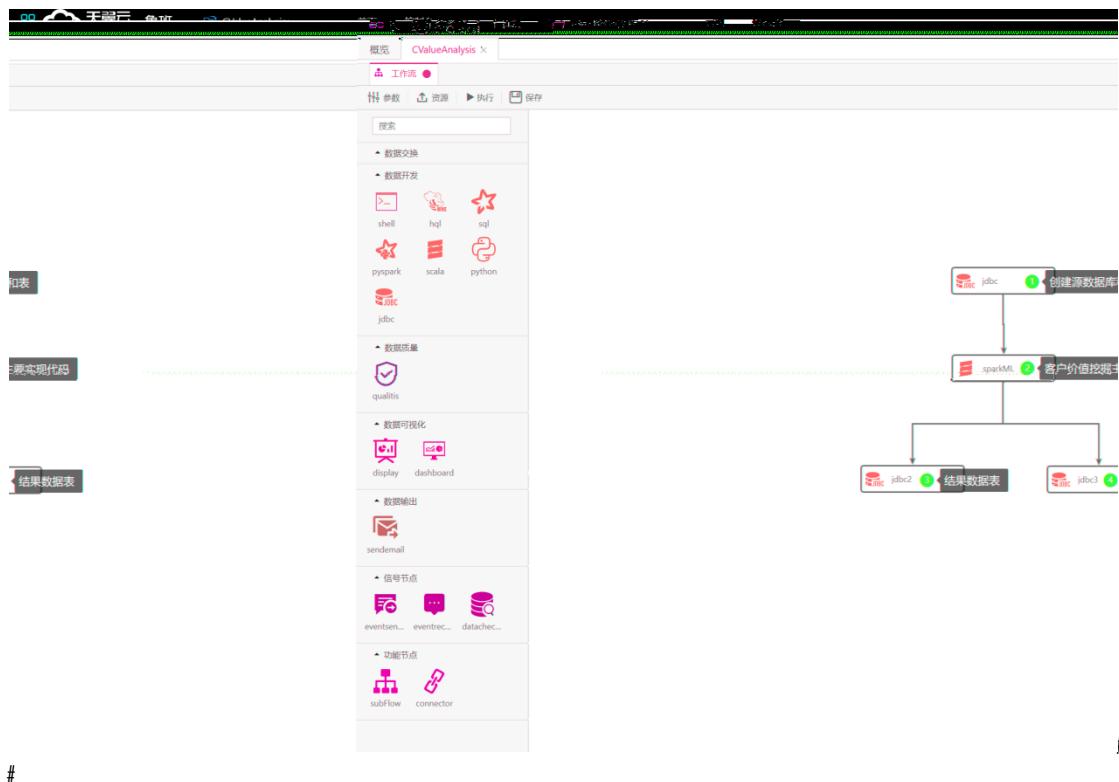
上一步 ① 点击 保存

6. 配置 JOB 数

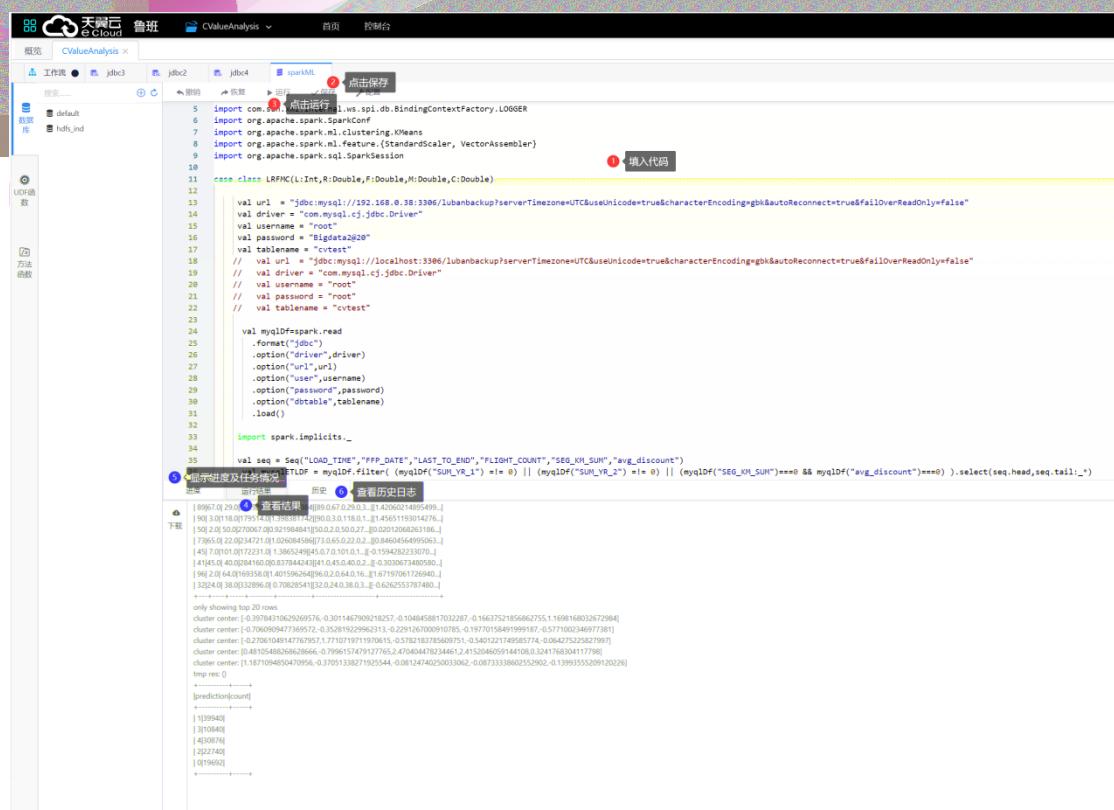
| 任务ID | 任务名称                      | 定时任务     | 任务描述 | 定时状态   | 创建时间                | 操作                                                                                                 |
|------|---------------------------|----------|------|--------|---------------------|----------------------------------------------------------------------------------------------------|
| 85   | cctest                    |          |      |        | 2020-09-01 11:42:16 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 84   | tar_schedule_mysql_to_ftp | 0.01 *** |      | PAUSED | 2020-09-01 07:41:42 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 83   | test122                   |          |      |        | 2020-09-01 07:19:01 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 82   | tar_join                  |          |      |        | 2020-08-28 08:49:50 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 81   | tar_split                 |          |      |        | 2020-08-28 08:26:27 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 80   | tar_replace               |          |      |        | 2020-08-28 07:59:47 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 79   | tar_filter_by_id          |          |      |        | 2020-08-28 07:50:17 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 78   | newcustomer               |          |      |        | 2020-08-28 08:10:05 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 77   | zjNew                     |          |      |        | 2020-08-28 08:27:28 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |
| 76   | tar_yinghe                |          |      |        | 2020-08-28 08:13:36 | <a href="#">编辑</a> <a href="#">执行</a> <a href="#">暂停</a> <a href="#">恢复</a> <a href="#">历史数据查询</a> |

7. 执行 JOB, 源数据 到 的 (mysql)

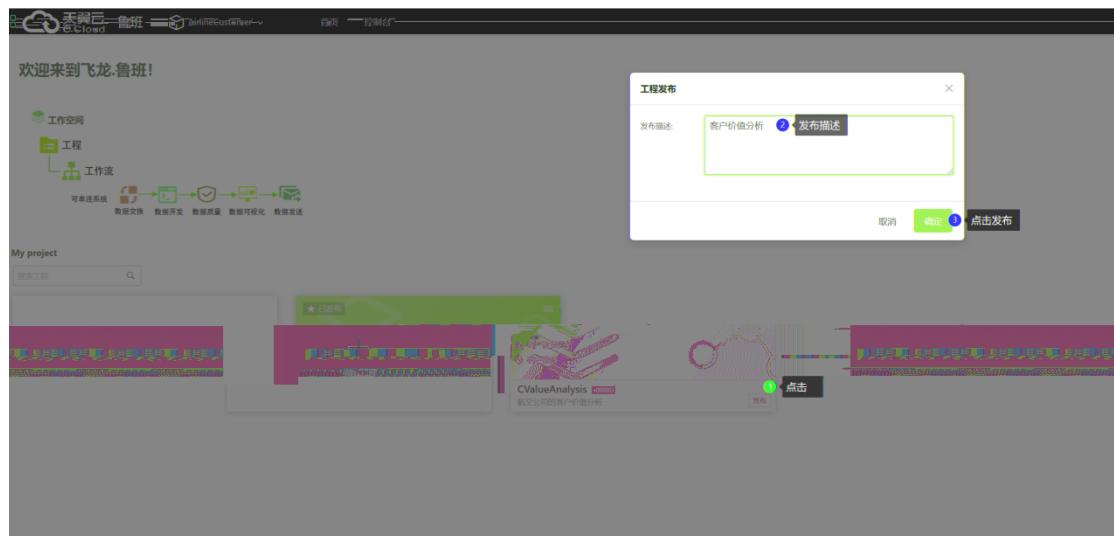
### Step 5 : 配置数据加工流程



1. 拖动一个 spark ( 为 sparkML ): 户价 分 的主 实现代
2. 两个jdbc ( 分 为 jdbc2 和 jdbc3 ) 布上 : 存储分 结果的
3. 自的实现代



## Step 6：发布 调度系统



1. 工作 → 用开发 → 工作流开发
  2. 发布,发布

The screenshot shows the Schedulis application interface. At the top, there's a navigation bar with tabs like '首页' (Home), '定时调度' (定时调度), '正在运行' (正在运行), '执行历史' (执行历史), '用户参数' (User Parameters), and '系统管理' (System Management). Below the navigation is a search bar with placeholder text '请输入...' and a '搜索' (Search) button. The main area displays a table of projects with columns for '项目名' (Project Name), '描述' (Description), '状态' (Status), and '操作' (Operations). One project, 'CValueAnalysis', is highlighted in blue. The status column shows '待完成' (Pending) and '在 2020-08-08 09:38:31' (At 2020-08-08 09:38:31). The operations column includes links for '查看' (View), '编辑' (Edit), '删除' (Delete), and '暂停' (Pause).

3. 进入【工作】->【产运维】->【schedulis】能中，到

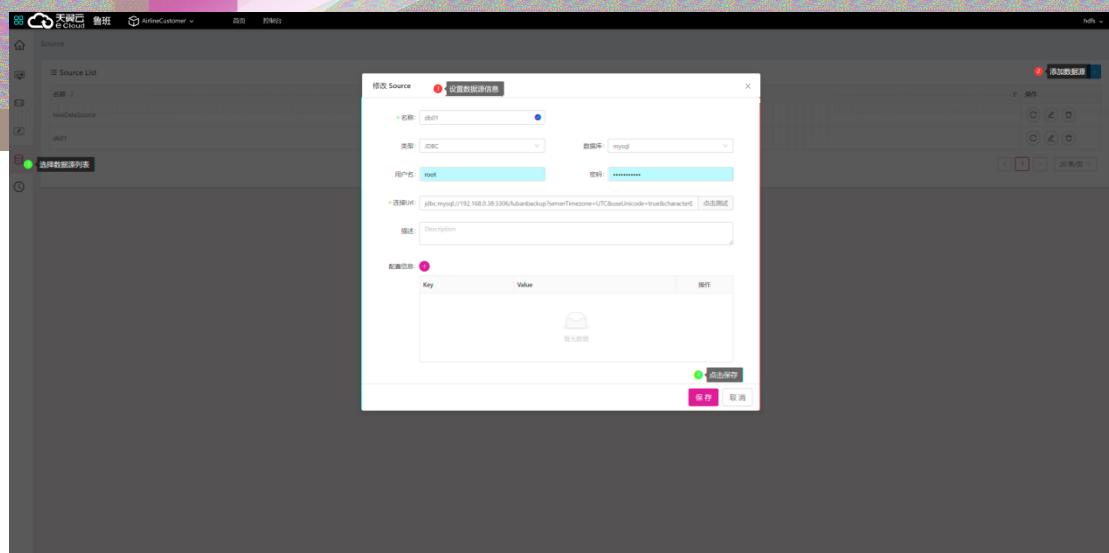
This screenshot shows a detailed view of the 'CValueAnalysis' project within the Schedulis application. On the left, there's a sidebar with project names like 'CValueAnalysis', 'ChiakeAnalysis...', 'jdb...', 'jdbcl...', 'jdbcl...', 'sparkML...', and 'sparkSQL...'. The main panel has tabs for '工作流' (Workflow), '调度关系' (Scheduling Relations), and '调度历史' (Scheduling History). The '调度关系' tab is active, showing a grid of scheduling relations between various components. A right-hand sidebar provides project details: '创建时间: 2020-08-01 14:48:41', '最后修改时间: 2020-08-08 09:38:31', '项目创建人: hdfs', and '你有权: ADMIN'.

4. 【执行工作流】，开运行 工作流

Step 5：定

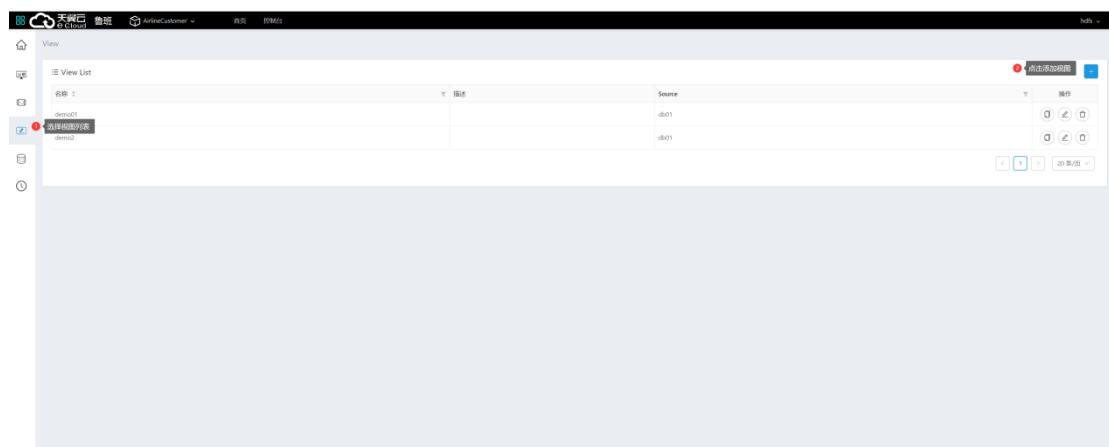
This screenshot shows the AirlineCustomer application interface. It features a dashboard with multiple project cards arranged in a grid. Each card contains a small image, a project name, and some status indicators. A specific card for 'AirlineCustomer-MavenA...' is highlighted with a red border. Below the dashboard, there's a section titled '我收藏的项目' (My Favorites) with a single item: 'test12'.

1. 进入【工作】->【数据分】->【visualis】进与同的可视化

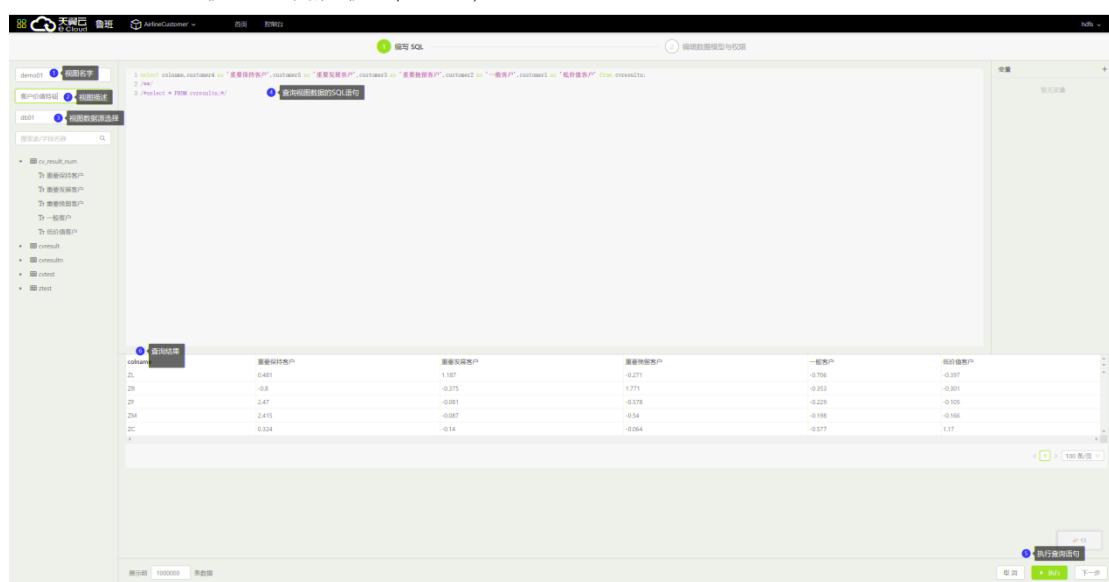


2. 加数据源 -> 编辑数据源 , 键入 下

`jdbc:mysql: 192.168.0.38:3306 用户 /密 : root /Bigdata2@20`

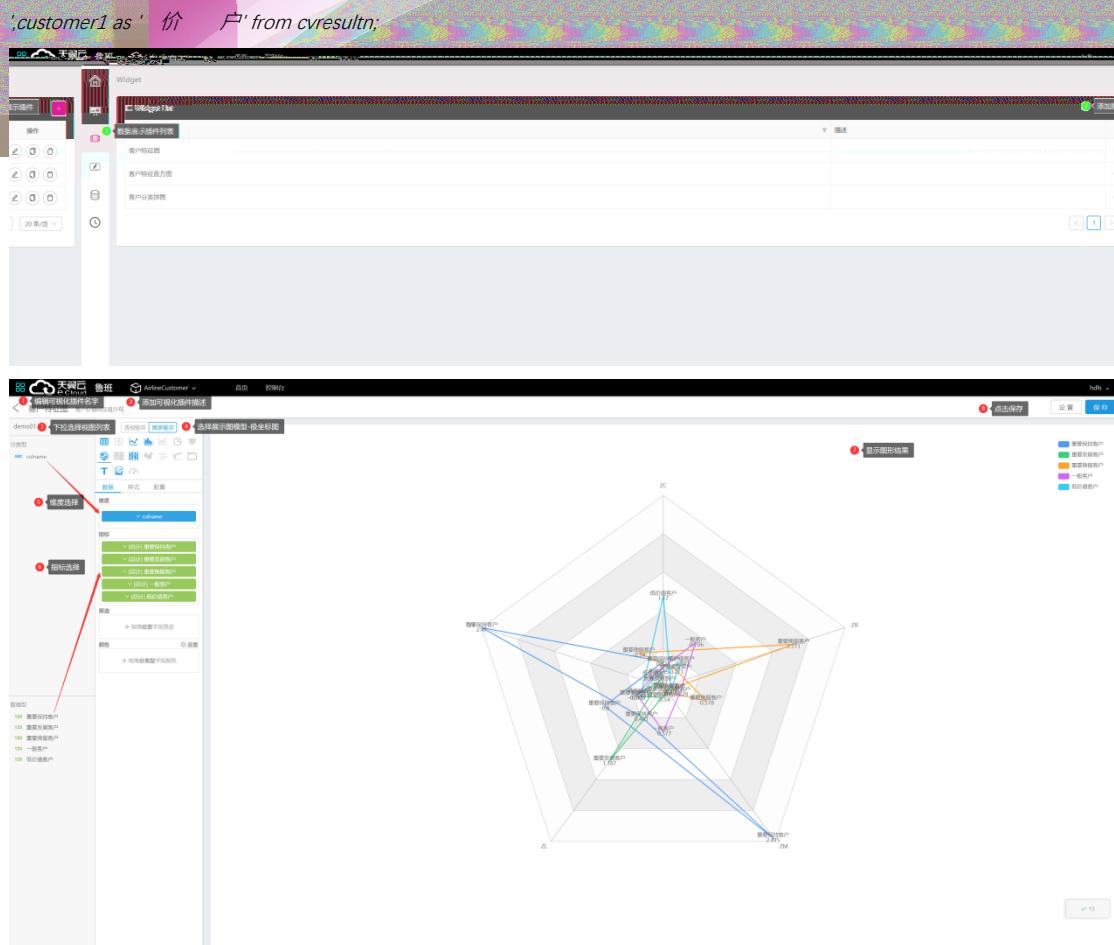


3. 视 , 新视 (上)

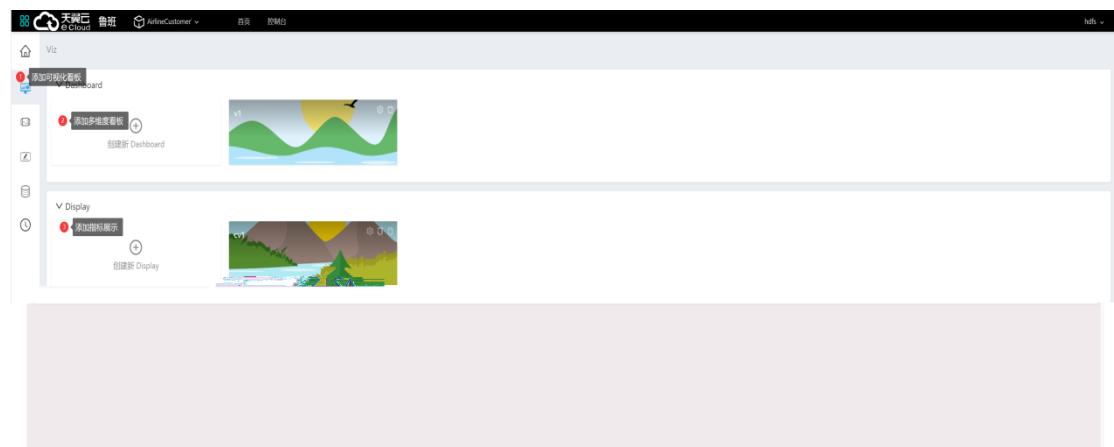


4. 编辑视 , 【视】及【SQL语】

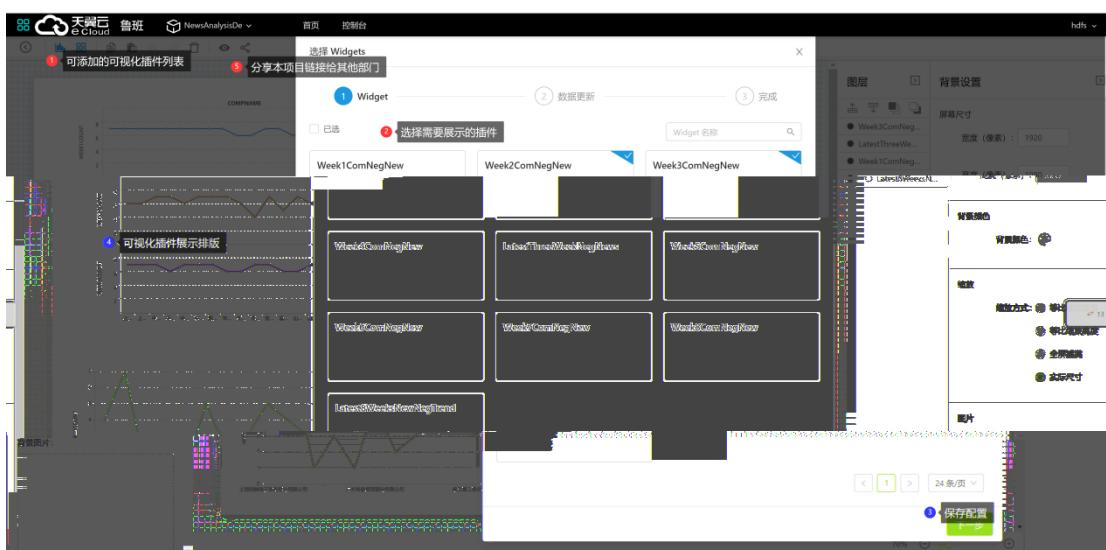
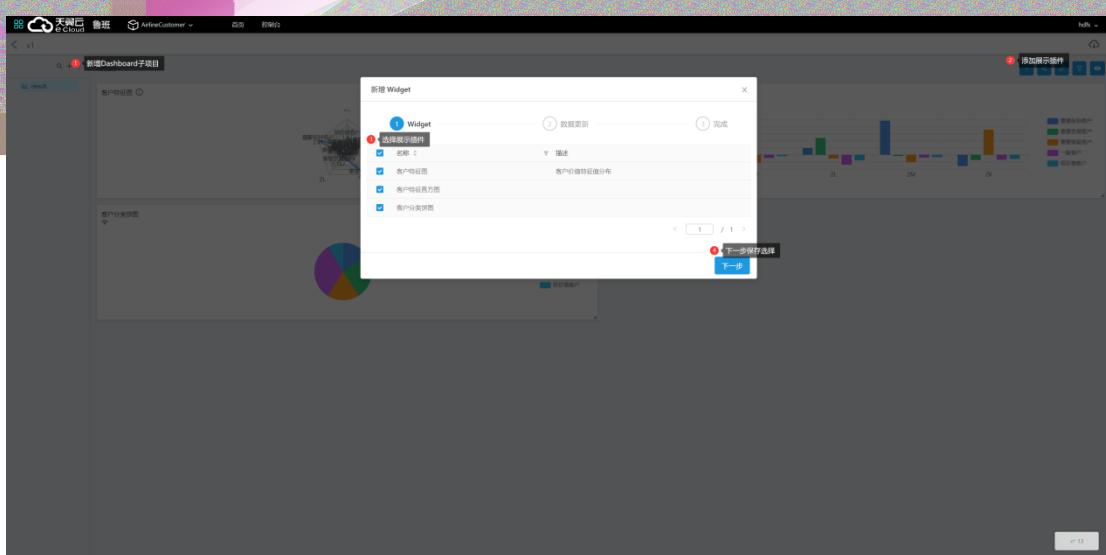
`select colname, customer4 as '重要维持客户', customer5 as '重要发展客户', customer1 as '重要新客户', customer2 as '一般客户', customer3 as '低价值客户'`



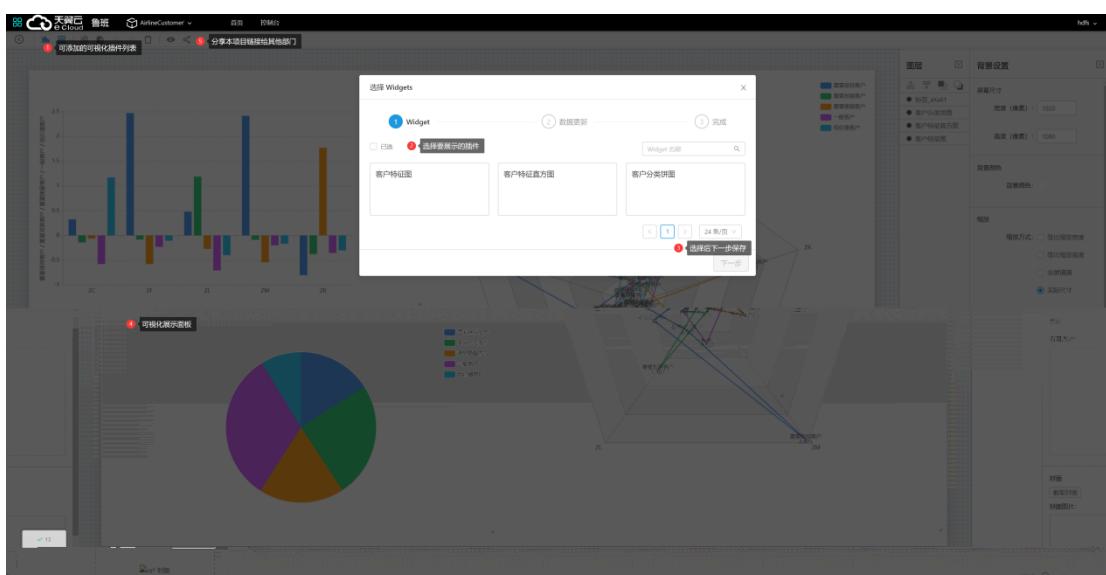
2. 加展示件，加件



2. 加可视化板 Dashboard 和 display



## 2. 加可视化组件



最 可 到 上 的 分 结果

### 三、详细使用

#### 工作流

#### 功    明

工作流提供 大的任务调度 能。使用工作流，可   类任务 一定的 序           ，系统 自动   一执行，   方

。

#

#### 使用流

##### Step 1：进 工作流页面

The screenshot shows the WeDataSphere workbench interface. At the top, there is a navigation bar with the logo, workspace name 'test', and links for '首页' (Home) and '控制台' (Console). Below the navigation bar, a banner says '欢迎来到 test 的工作空间!' (Welcome to test's workspace!). The main area is divided into several sections:

- 常用功能** (Common Functions): Includes '进入工作流开发' (Enter Workflow Development), which is highlighted with a red arrow.
- 应用开发** (Application Development): Contains tabs for '工作流' (Workflow) (selected), '数据开发' (Data Development), and '数合开发' (Data Integration). A detailed description for '工作流开发' states it is based on Linkis and designed for workflow tasks.
- StreamSQL开发** (StreamSQL Development): Contains tabs for '流式' (Streaming) and '实时' (Real-time). It is described as a real-time solution using Flink SQL.
- 数据服务开发** (Data Service Development): Contains tabs for 'API' and '数据服务'. It is described as a service-oriented architecture (SOA) solution.

At the bottom of the interface, there is a footer with the text '1.进 一工作 , “用开发”下的“进 工作流开发”，上 。系统 工作流开发 页' (1. Enter one work, "Using development" under "Enter Workflow Development", up . System Workflow Development page).

#

欢迎来到工作流开发首页!



欢迎来到工作流开发首页!

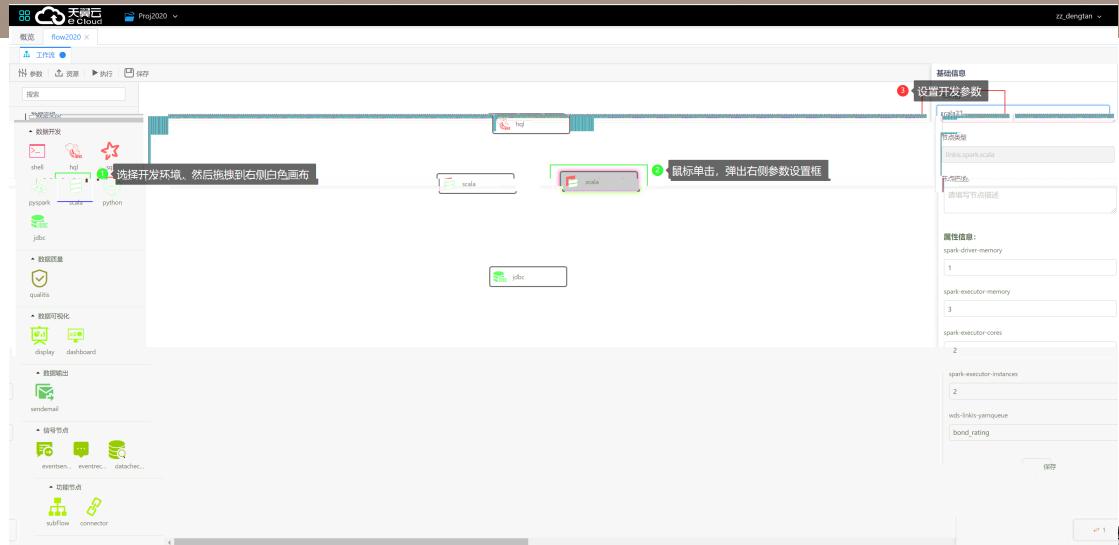


2. “工程”，工程，3，写完信，系统工作流页。



3. “工作流”，并在的示的中写信，系统即工作流编辑页面。

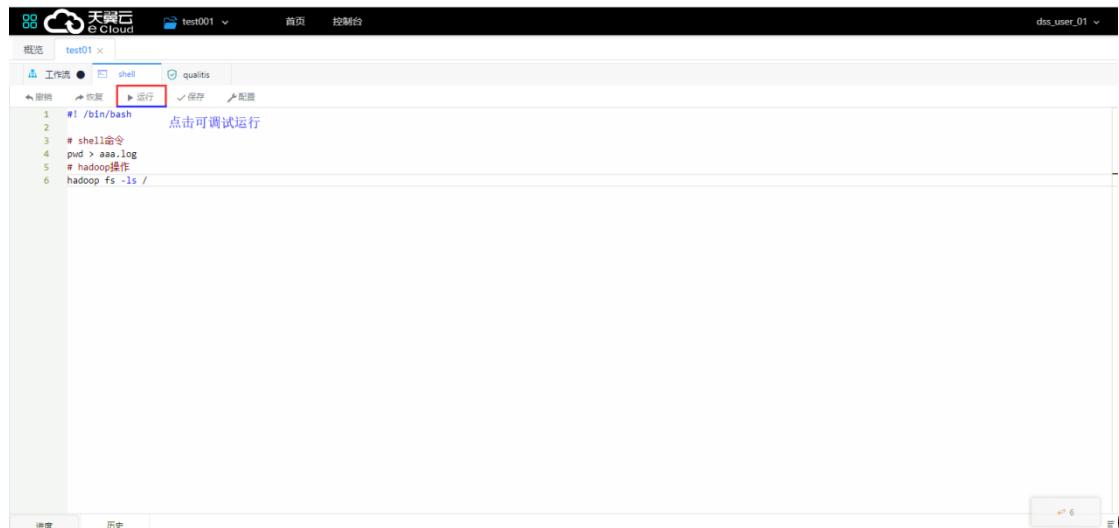
## Step 2：编辑工作流任务



1. 根据开发的代码，选择模版，拖拽模版到画布，按键模可以在设置数，标进模的开发页面，模可以以及脚本等操作。

#

#

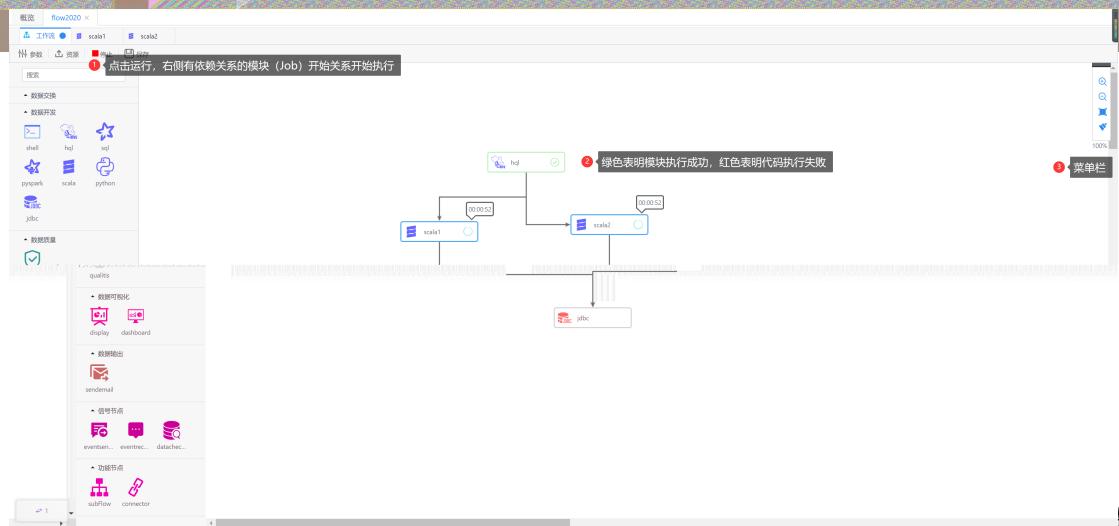


2. 中任任务，编辑，可置基本信息，等；部分，可代页面或置页面，可在中编写脚本或置信息。

#

#

### Step 3 : 执行工作流任务



3. 在工作流编辑主页上，‘执行’，即可自动保存并运行 工作流。执行成 与 的 实时展示在工作流的 个 上。

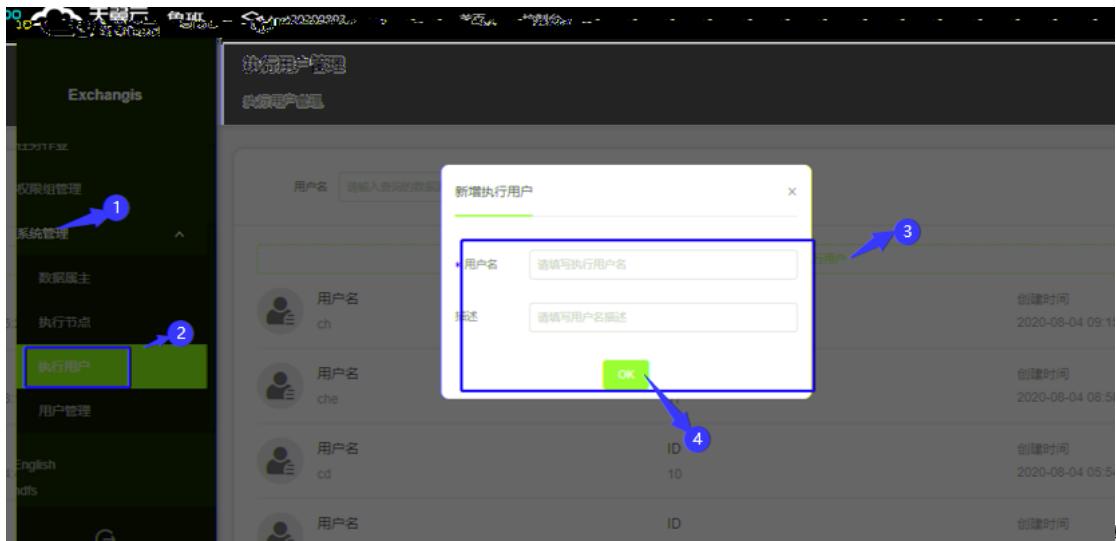
## 数据交换(Exchangis)

### 功 明

Exchangis 是一个轻 级的、 展 的数据交 平台，支持 结 化及 结 化的数据源 的数据 ，在 用 上具 数据权 、 服务 可用和多租户资源 离等业务 ，而在数据 上具 多 化、模 件化和组 件 等 。#

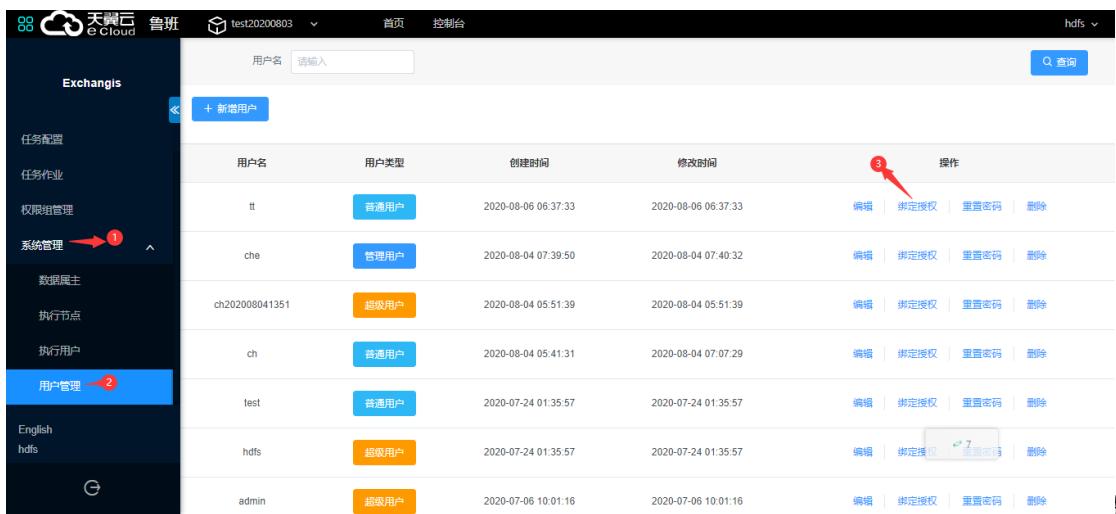
## 使用流

### Step 1：加执行用户

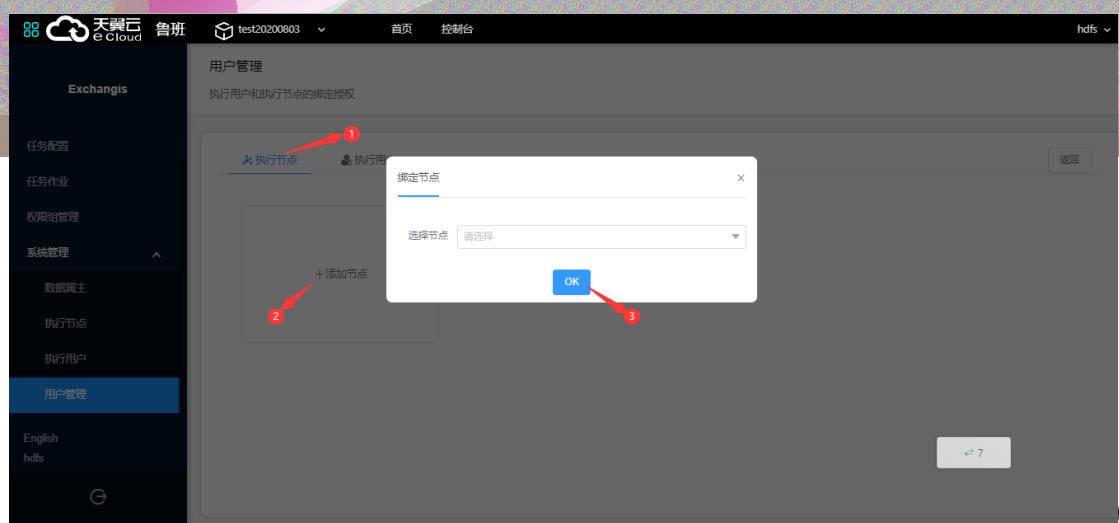


1. 在系统的 理下，进入 执行用户，进 页面。
2. 页面中的新 执行用户，在 中 写用户名 ( )，按 OK。

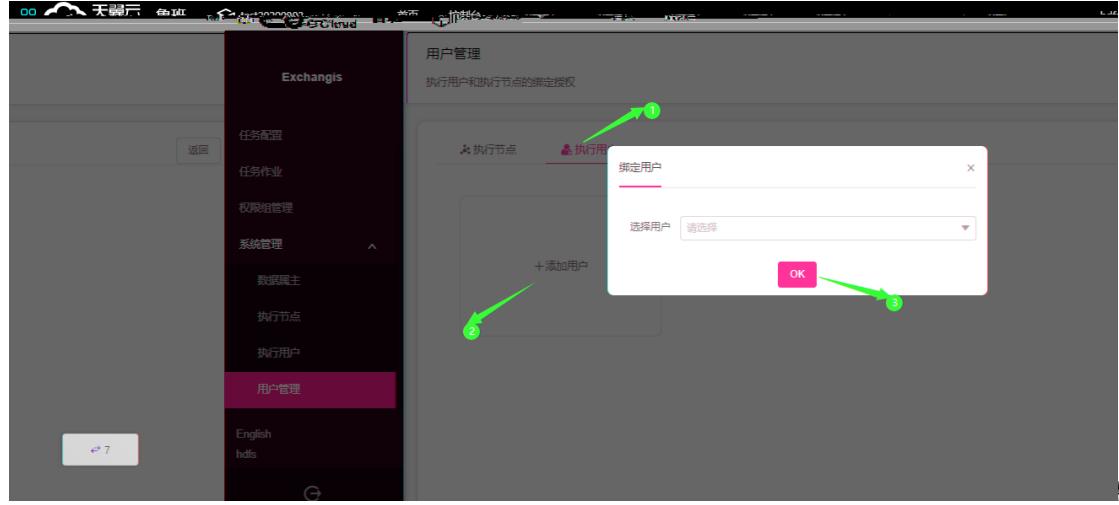
### Step 2. 用户 定 权:



3. 在系统的 理下，进入 用户 理进 页面，用户的 定 权。



4. 在 **页面中** 加 **，在** 中 **的**， **OK**。



5. **页面中的执行用户，在** **页面中** **加用户，在** 中 **的用户，** **OK**。

Step3 执行用户：

1. 在系统的管理下，执行进入页面，执行的理，进入页面。

2. 页面中的加，在用户以及定类，OK。

#### Step 4 新数据源主

1. 在系统的管理下，执行进入页面，执行的理，进入页面。

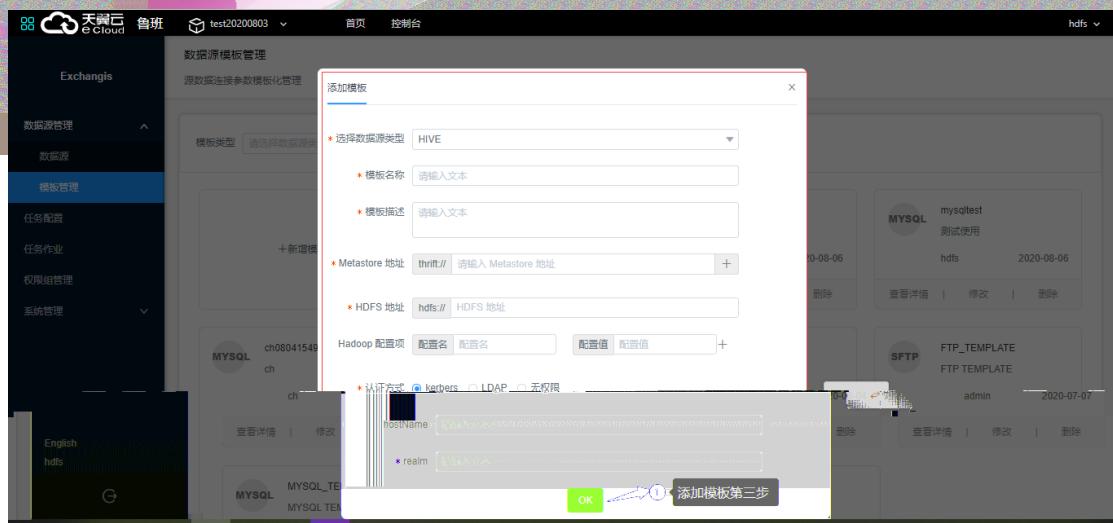
1. 中的系统 理下 , 新 数据 主, 进 页面。
2. 页面中的新 数据源 主, 在 中 写 主 - ( ), 提交。

### Step 5 新 :

1. 下的任务配置, 进 页面。
2. 页面中的新 , 在 中 写 主 - ( ), 提交。

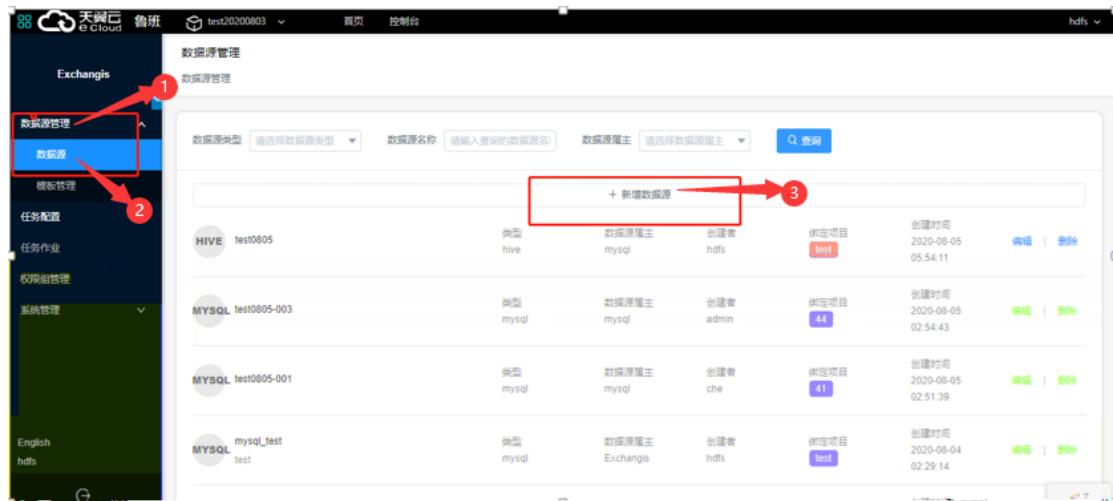
### Step 6 新 数据模板

1. 数据源 理模 下的模板 理进 页面, 在页面中 新 模板。



2. 在 中 数据源类 为 HIVE- 写 -模板 -Metastore -HDFS -Hadoop 配置 ( ) - 方式， OK。

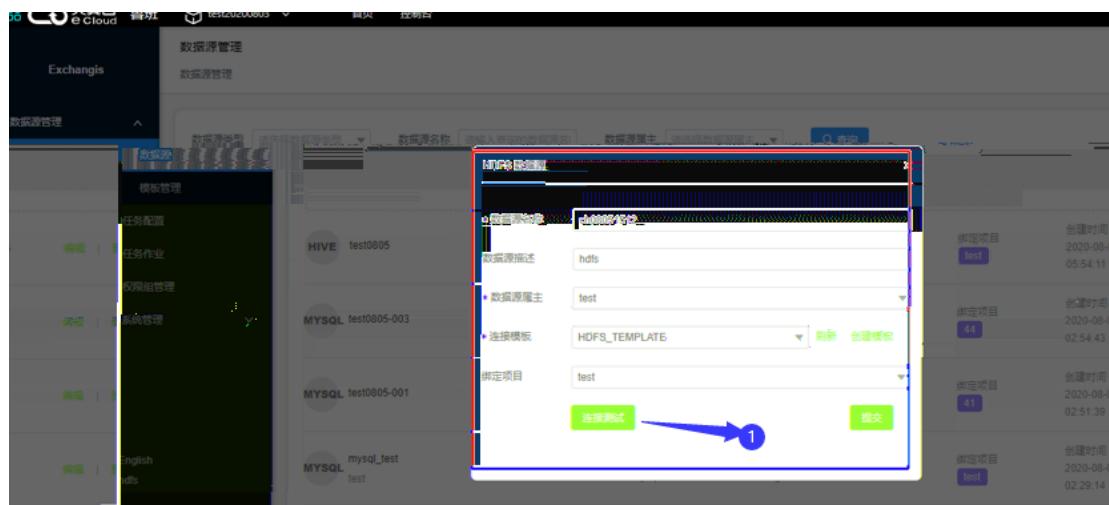
#### Step 7 新 数据源:



1. 数据源 理下 ， 数据源进 页面， 页面中新 数据源。



2. 在 中可以 选 择 数据 ( MySQL 数据源 ), 大 数据 存 储 ( HIVE 数据 , HDFS ), 结 化 数据 存 储 ( SFTP 服务 , ElasticSearch ) 的数据源 hdfs , 。



3. 在 写 数据 源 - 数据 源 ( ) - 数据 源 主 - 数据 源 模板 - 定 的

4. 页面提示 系数据源成功，提交。

#### Step 8 加任务：

1. 数据或数据源(以 csv 本 数据为 )。

1. 数据或数据源(以 csv 本 数据为 )。

2. 标动到页面上能模，数据分中的scriptis进页面。

The screenshot shows the Tianyi Cloud Ruanban interface. A context menu is open over a file named 'ch\_test.sql' in the 'hdfs' directory. The menu items are: 撤销 (Undo), 恢复 (Redo), 运行 (Run), 保存 (Save), 配置 (Configure). The '上传' (Upload) option is highlighted with a red box and a blue arrow labeled '3'. The left sidebar shows other files like 'shell.sh', 'sparkSql.sql', 'test.scala', etc. The right panel shows a progress bar and a log table.

3. 键本件中，上件。

The screenshot shows the Tianyi Cloud Ruanban interface. A context menu is open over a file named 'copy.sh' in the 'hdfs' directory. The '新建脚本文件' (New Script File) option is highlighted with a red box and a blue arrow labeled '1'. A new dialog box titled '新建脚本文件' is displayed. It contains fields: 名称: 'copy' (highlighted with a red box and a blue arrow labeled '2'), 脚本类型: 'Shell' (highlighted with a red box and a blue arrow labeled '2'), and a '创建路径' (Create Path) dropdown with the value '/usr/local/linkis/workspace/hdfs/测试' (highlighted with a red box and a blue arrow labeled '3'). The background shows a progress bar and a log table.

4. CSV件上成，键件新一个xshell脚本。



5. 脚本，  
hadoop fs -put /usr/local/linkis/workspace/hdfs/ /csv.csv hdfs:///tmp/linkis/hdfs/20200804  
csv 文件 copy 到 hdfs 中。

| 任务ID | 任务名称           | 定时任务 | 任务描述 | 定时状态 | 创建时间                | 操作               |
|------|----------------|------|------|------|---------------------|------------------|
| 49   | test           |      |      |      | 2020-08-06 02:00:15 | 编辑 执行 复制 删除 历史数据 |
| 48   | test23         |      |      |      | 2020-08-06 01:48:19 | 编辑 执行 复制 删除 历史数据 |
| 31   | mysql2hive     |      |      |      | 2020-08-04 02:43:17 | 编辑 执行 复制 删除 历史数据 |
| 28   | mysql-hive     |      |      |      | 2020-08-01 01:50:22 | 编辑 执行 复制 删除 历史数据 |
| 21   | 字符串类型转换        |      |      |      | 2020-07-29 02:22:21 | 编辑 执行 复制 删除 历史数据 |
| 19   | ftp2mysql      |      |      |      | 2020-07-28 06:52:51 | 编辑 执行 复制 删除 历史数据 |
| 18   | mysql2ftp      |      |      |      | 2020-07-28 06:47:16 | 编辑 执行 复制 删除 历史数据 |
| 17   | mysql-localfs  |      |      |      | 2020-07-27 09:30:32 | 编辑 执行 复制 删除 历史数据 |
| 16   | hivetolocal_fs |      |      |      | 2020-07-27 09:22:34 | 编辑 执行 复制 删除 历史数据 |

6. 在 hdfs 集群中 csv 文件。

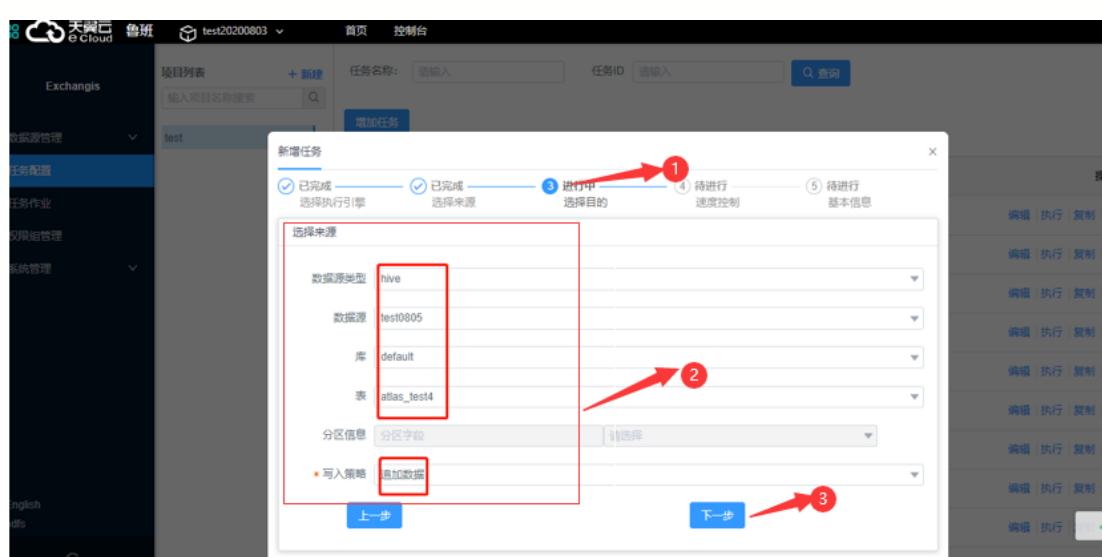
7. 返回 Exchangeis 页面，中的任务配置，进 页面。

8. 数据源 定的 ，进 页面。

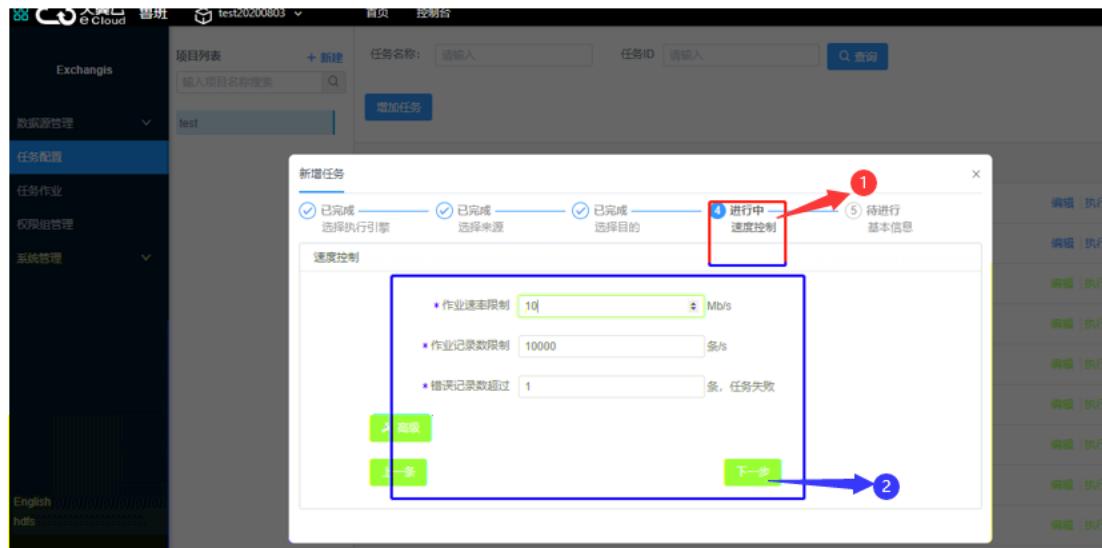
9. 加任务，在 执行的 (DATAx, SQOOP) 下一。

10. 源 中 写数据源类 为 hdfs, 数据源为 的数据源- 为 hdfs 集群中 的 - 类

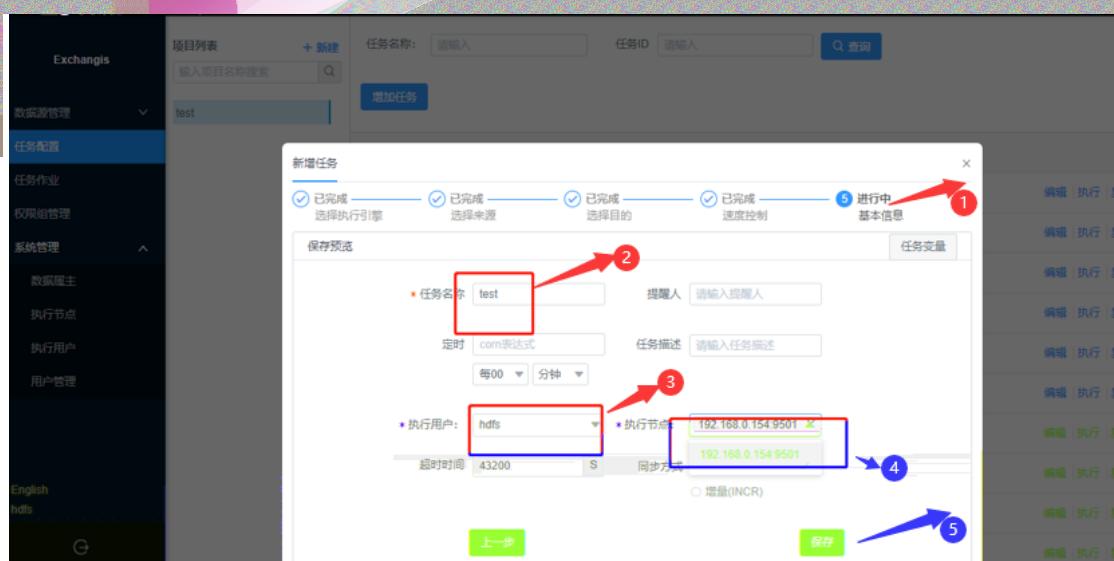
( ) - 编 ( ) - 方式为 - 件类 为 csv- 分 ( ), 下一。



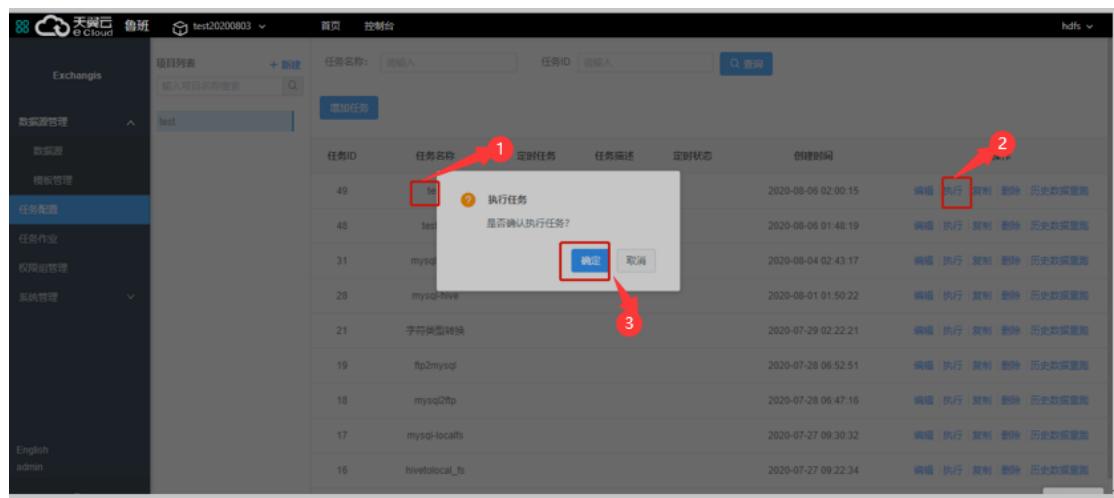
11. 的 写数据源类 - 数据源- - - 写 , 下一。



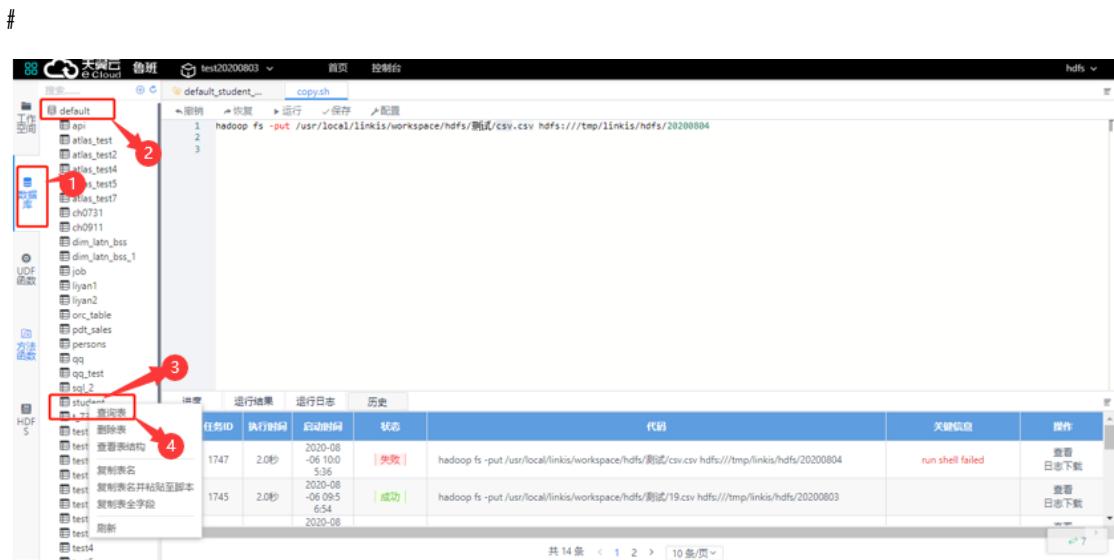
12. 速度 - 作业速率 - 作业 数 - 数 或 加 级 置 , 下一。



13. 基本信息 -写任务 -提 人( ) -定时( ) -任务 ( ) -执行用户-执行 - 时时  
( ) -同 方式( ), 保存。



14. 任务新 成 , 任务 的执行 , 在 中 , 页面提示任务开 执行。



The screenshot shows the eCloud interface with a database table named 'student'. The table has columns: id, name, sex, address, and age. There are three rows of data:

|   | id | name | sex | address | age |
|---|----|------|-----|---------|-----|
| 1 | 10 | 小程   | 男   | 安徽      | 29  |
| 2 | 10 | C*** | **  | ****    | 29  |
|   | 2  | 黎明   | 男   | 安徽      | 20  |

15. 任务状态为成功，返回 scriptis 页面，在数据中到 default，进入到 student，键查询，在中可以到上 的数据。

## Scriptis

### 功 明

Scriptis 是一个高效的在线开发 IDE，提供语法、代码自动全、能和语法提示等功能，并支持在线开发、在线调用、多人同编辑、一键发布 UDF 资源和数鲁班。

### 使用流

#### Step 1 进 Scriptis

The screenshot shows the Scriptis interface with a sidebar containing several sections:

- 文件系统 (File System)
- 数据仓库 (Data Warehouse)
- UDF 函数 (UDF Functions)
- 方法函数 (Methods)
- HDFS

A red arrow points to the '方法函数' section. To the right of the sidebar, there is a '什么是Scriptis?' section with the following text:

Scriptis是天翼云(Luban)打造的一站式交互式数据探索分析工具，以任意环境(Linux)做为内核，提供多种计算存储引擎(HDFS, Hive, TiSpark等)，Hive数据管理功能，资源(Yarn)资源，服务器资源管理、应用管理和各种用户资源(如UDF, 变量等)管理的能力。

Scriptis?

1. 选中工作空间的目录，创建文件夹；
2. 右键某个文件夹 =>新建脚本；
3. 选择脚本类型，如：SQL, PySpark, HQL等；
4. 书写的脚本，点击执行，生成结果集。

1. 页面布 分为“文件系统”、“数据”、“UDF 函数”、“方法 函数”和“HDFS”标，可一打开查

, 也可通过编辑脚本的方式进行资源的操作。

#

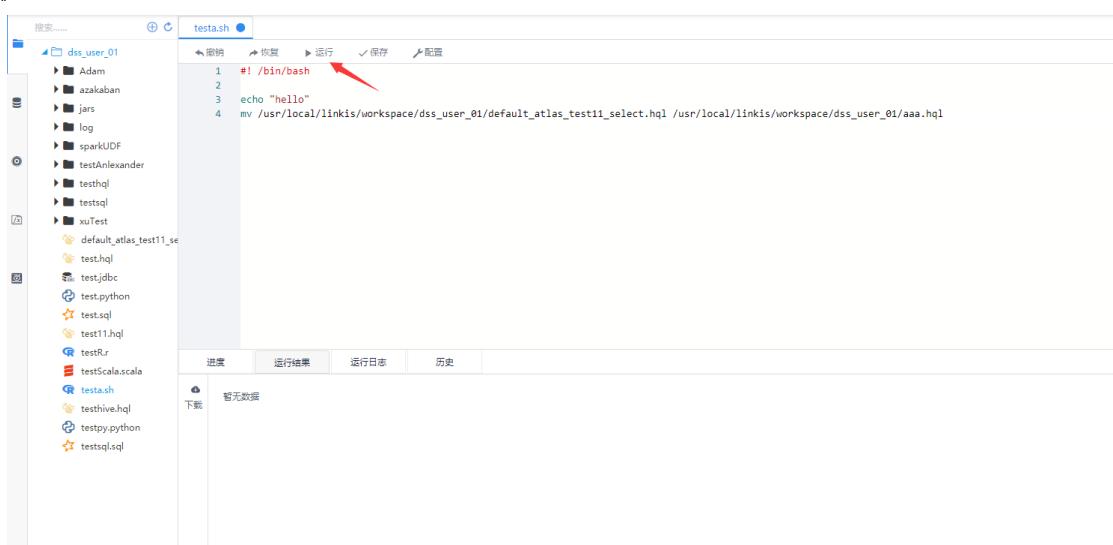
## Step 2 编写、执行脚本



1. 键入“新建”，可在的下多级

2. 并键入“新脚本”可 shell、python 等多种语言脚本

#



3. 处以 shell 脚本为。脚本件中的脚本类 shell，完成，自动打开脚本编辑页面，并脚本内。

4. 【运行】，执行脚本，执行程中执行的实时展示，以及结的结果展示

#

## Step 3 查 执行结果

可在下方的“进”、“”、“执行结果”等标签页查看信。

注

1. 脚本中可不加 而 使用系统 部署的 spark context ( sc ) 等 数。 下：

The screenshot shows a code editor interface with two tabs: 'testHangGuang.py' (Python) and 'test.scala' (Scala). The left sidebar shows a file tree with a 'xuTest' folder containing 'TestScala.scala', 'hangHuangCase.py', and 'rtest.r'. The right sidebar shows a list of files including 'test.scala', 'test.sql', 'testHangGuang.py', 'testZainiCase.scala', 'xuPythonSpark.py', 'default\_atlas\_test11\_s', 'test.hql', 'test.jdbc', 'test.python', 'test.sql', 'test11.hql', 'testRr', and 'testScala.scala'. The 'testHangGuang.py' tab contains the following Scala code:

```
1 k = sc.parallelize([("spark", 1), ("hadoop", 4)])
2 y = sc.parallelize([("spark", 2), ("hadoop", 5)])
3 joined = x.join(y)
4 final = joined.collect()
5 print "Join RDD -> %s" % (final)
6
7
```

#

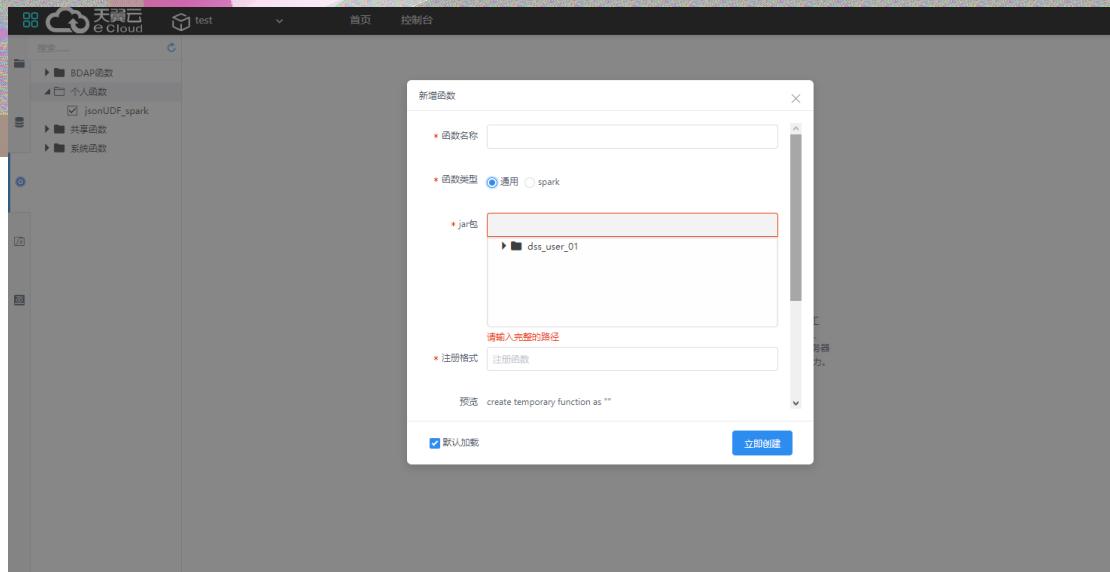
2. 数 能的使用。

The screenshot shows a code editor interface with a single 'test.scala' tab. The left sidebar shows a file tree with a 'xuTest' folder containing 'TestScala.scala', 'hangHuangCase.py', 'rtest.r', and 'test.scala'. The right sidebar shows a list of files including 'test.scala', 'test.sql', 'testHangGuang.py', 'testZainiCase.scala', 'xuPythonSpark.py', 'default\_atlas\_test11\_s', 'test.hql', 'test.jdbc', 'test.python', 'test.sql', 'test11.hql', 'testRr', and 'testScala.scala'. The 'test.scala' tab contains the following Scala code:

```
1 def printString(str:String):Unit={
2 | println(str)
3 }
4
```

3. 先 语言的脚本，在脚本中，定义一个 数。以 scala 为 ， 一个 scala 脚本，并在 中定义一个数 printString，并保存。

#



4. 在“UDF 数”->“个人 数”，键“新 数”，可 新 数。数（本中的 printString() 等信，件系统中 的脚本，即可完成 数与脚本的 定。完 成，系统自动加 数，并可在编写 脚本 件中不加 用 使用 数。

5. 用户可 配置 自定义 数，并可在脚本代 中 的 成 \${ 数 } 式，可成 执行。

## 天翼视屏

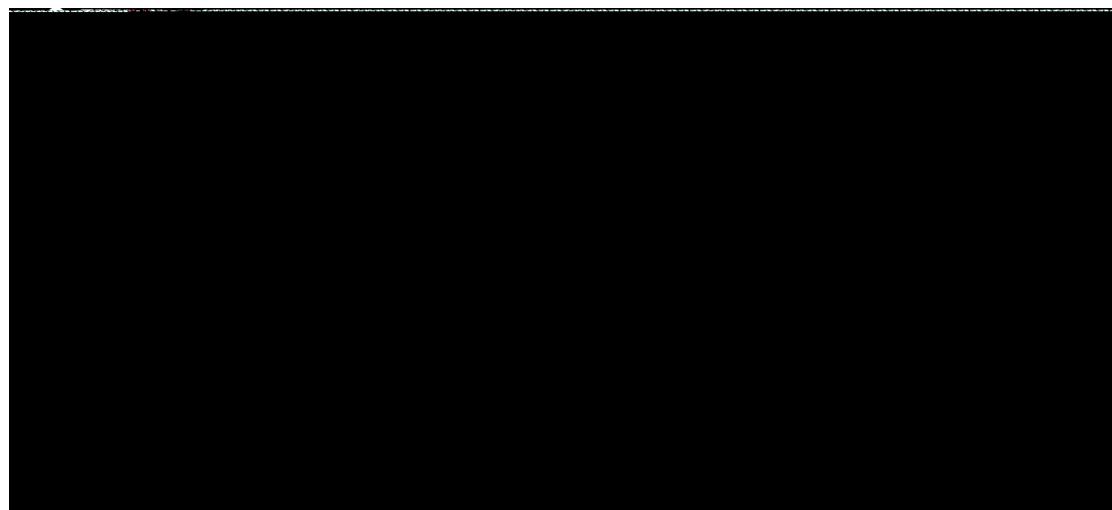
### 功 明

天翼视屏用于数据可视化展示，提供丰富的组件和模板，您 展、业务、理信 分等多种业务的展示

#

使用流

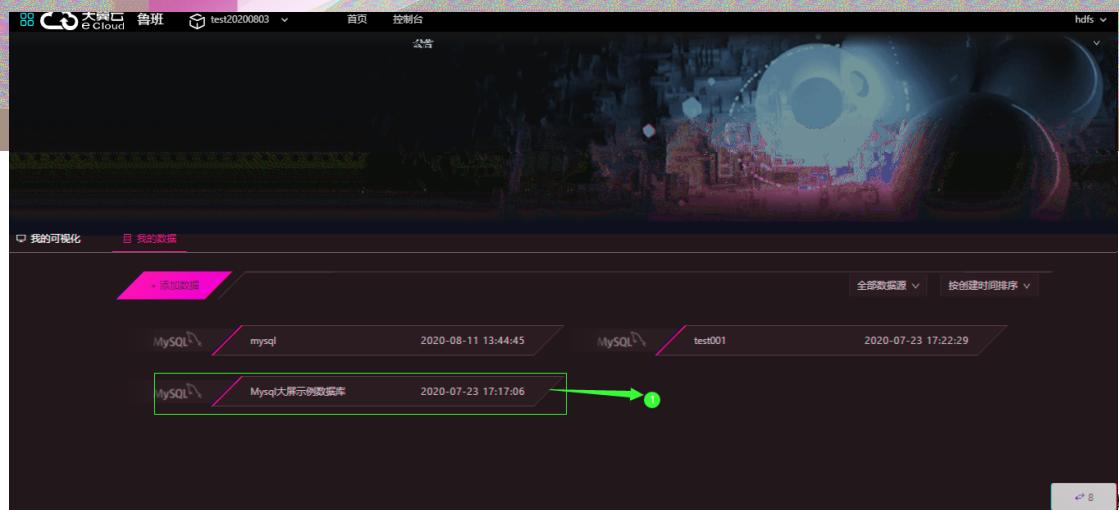
Step 1. 数据源



M( 在可视化 页 , 在 页面中“我的数据”。#

The screenshot shows the Tianchi Cloud interface with the 'My Data' tab selected. A modal window titled 'Add Data' is open, showing a dropdown for 'Type' set to 'MySQL database'. The main area displays a list of existing MySQL databases: 'mysql' and 'test001'. Three numbered arrows point to specific elements: ① points to the 'My Data' tab; ② points to the '+ Add Data' button; ③ points to the '确定' (Confirm) button in the modal window.

2. 在页面中 加数据 , 在 中 据 可以 写数据类 , 在 据不同类 写不同信 。

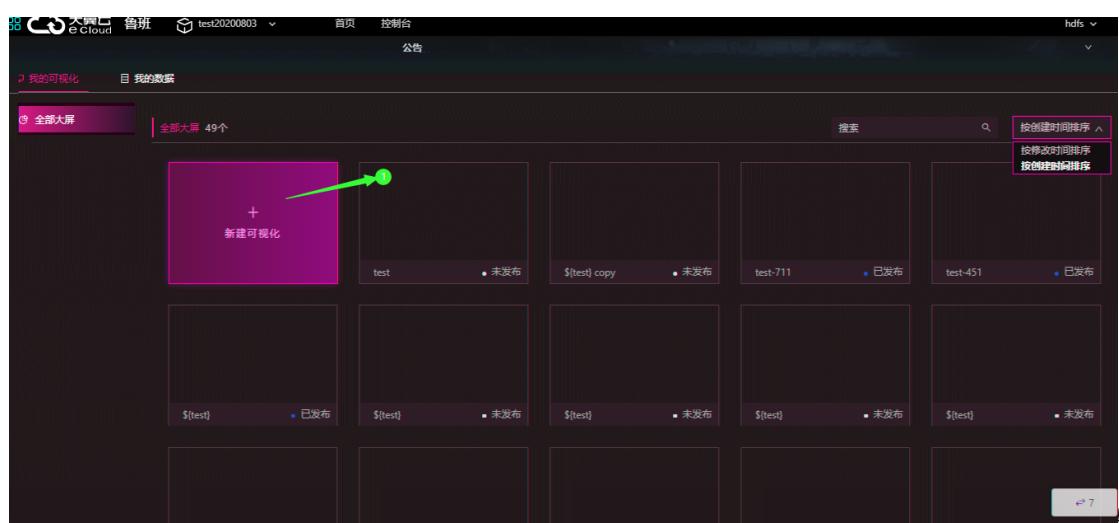


3. 完成，并通过，数据源中 的数据源。

#

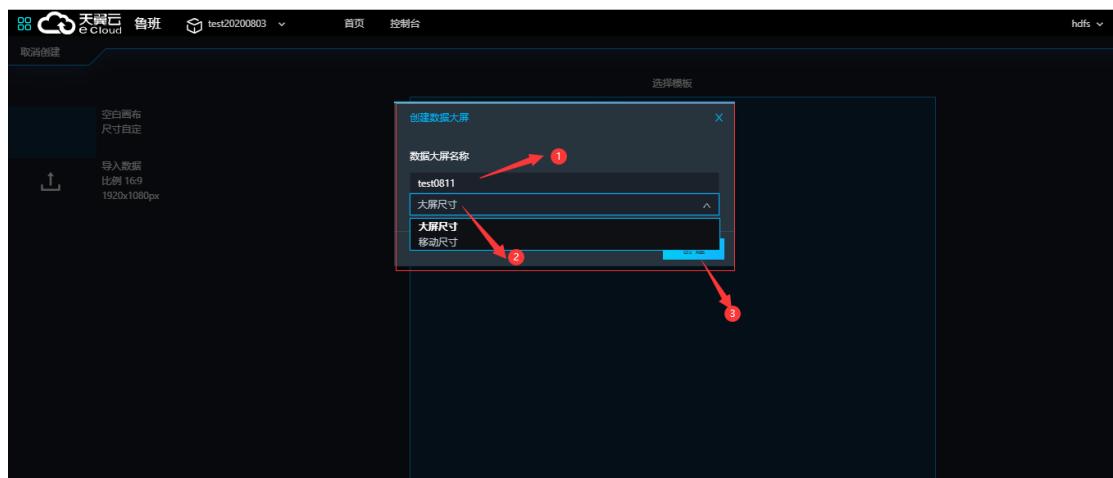
## Step 2. 布

#

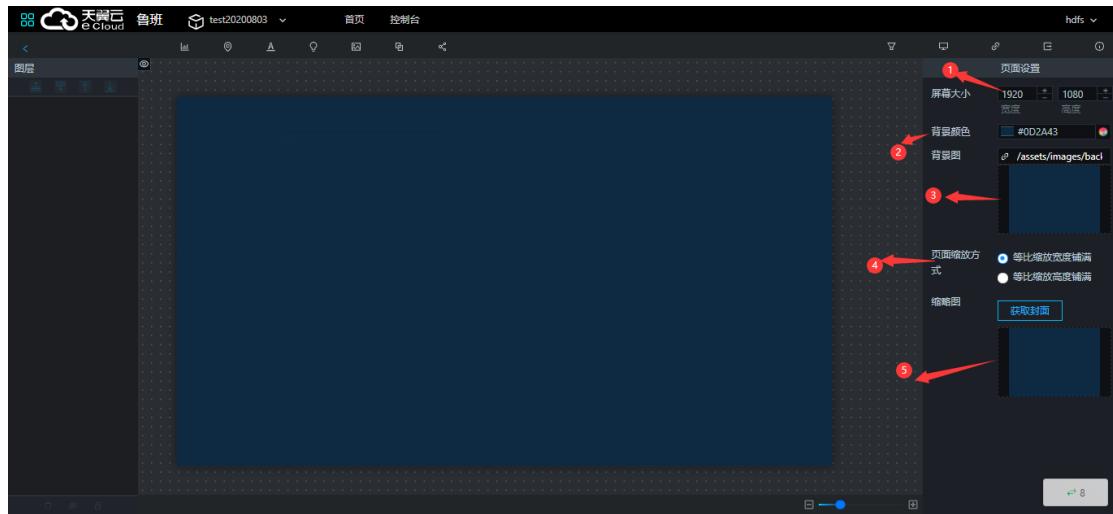


1. 进入天翼视屏 页面新 可视化

#



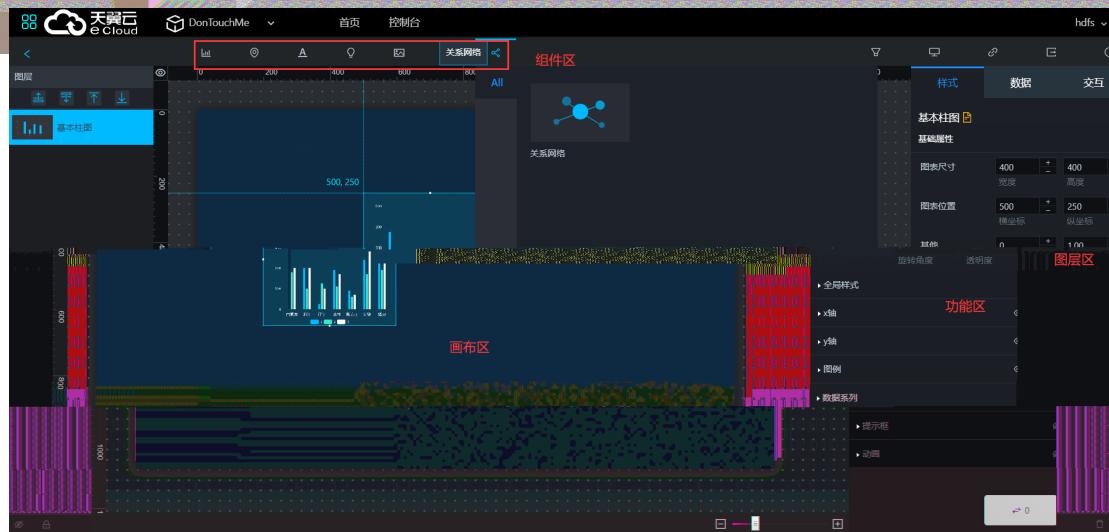
2. 进入页面，在其中写大屏，以及大屏下，。



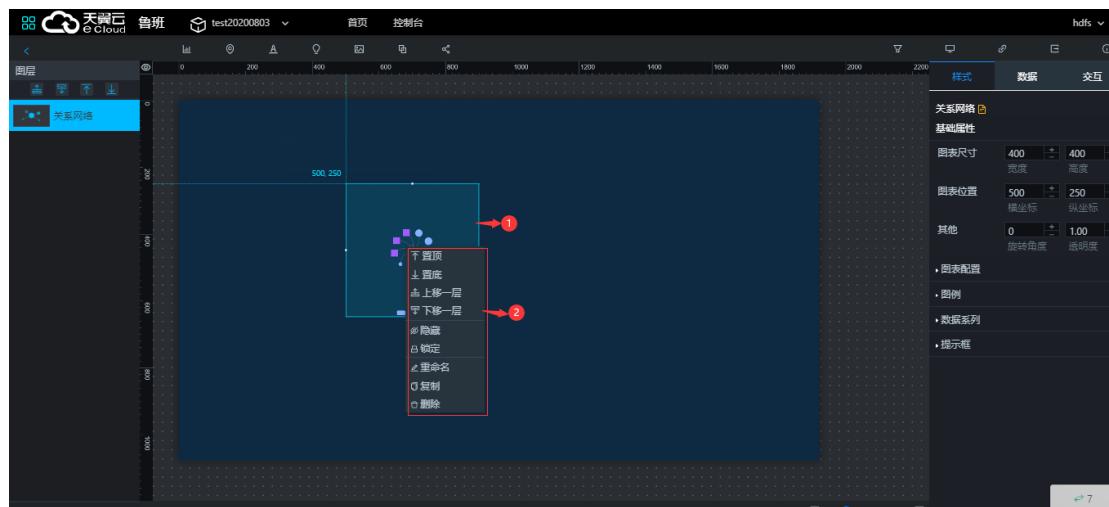
3. 成功进入页面，在页面中可以1. 计屏大，2. , 3. , 4. 页面方式，5. 查。

#

### Step 3 加可视化组件



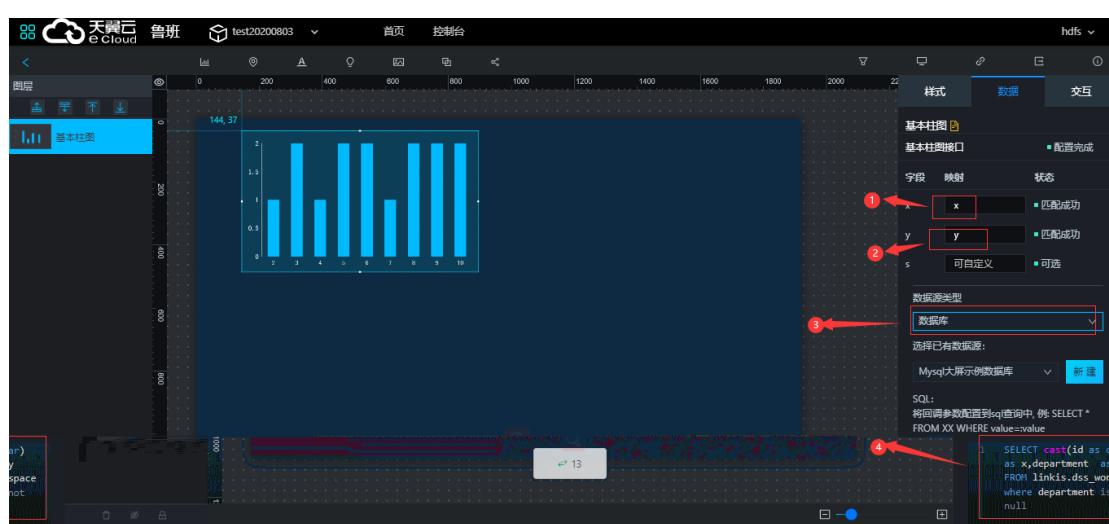
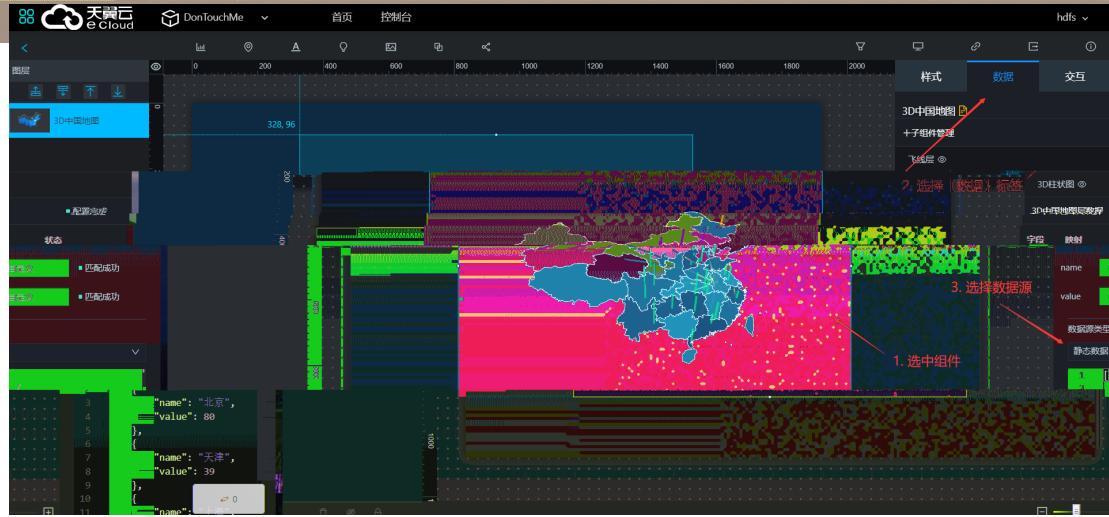
1. 组件的组件，现在布，可用的组件包种类统计、标等

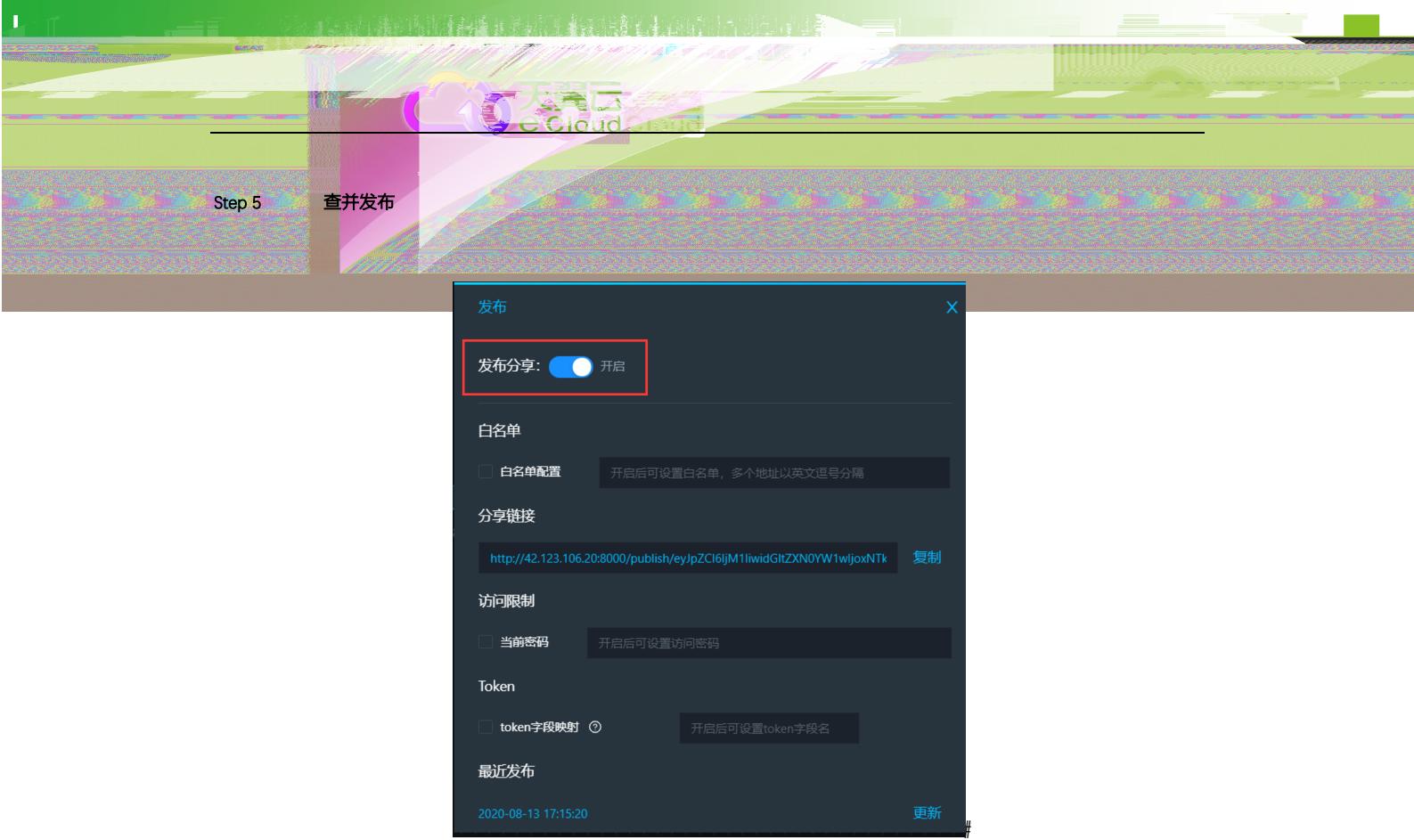


2. 调键可以进行置，置，上一，下一等操作。到视效

#

#### Step 4 数据源





1. 在‘查并发布’，勾选‘发布，开启发布开关’，即可通过‘进行分享’，并进行权限设置。

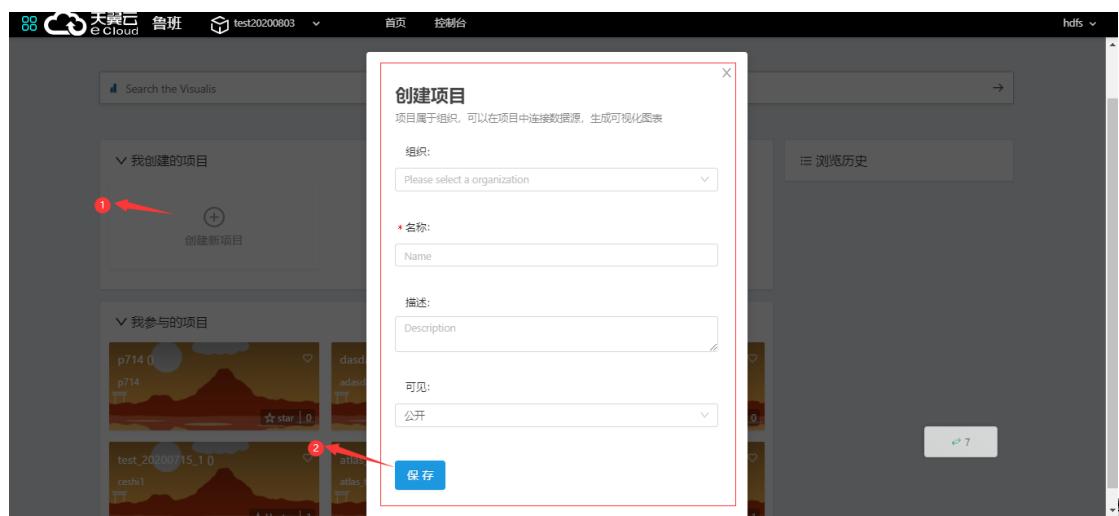
## Visualis

### 功    能

Visualis 支持拖拽式数据定义、动态取数、全维分析、多维分页、实时查询等数据开发的分布式模式，并且支持金融级的数据验证。

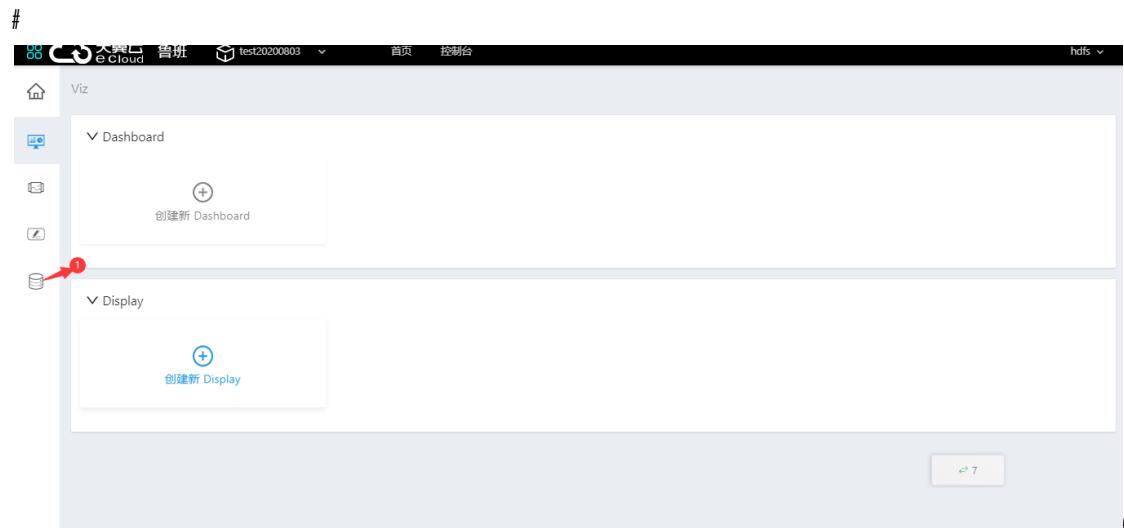
## 使用流

### Step 1

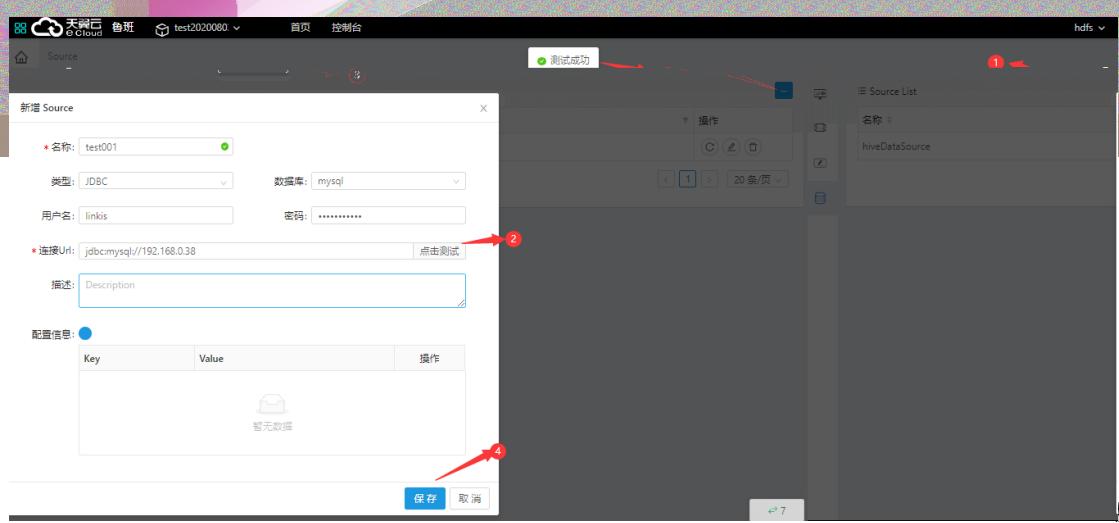


1. 进 Visualis 页面, 新 在 中 组 ( )- - ( )- 可 , 保存。

### Step 2 数据源



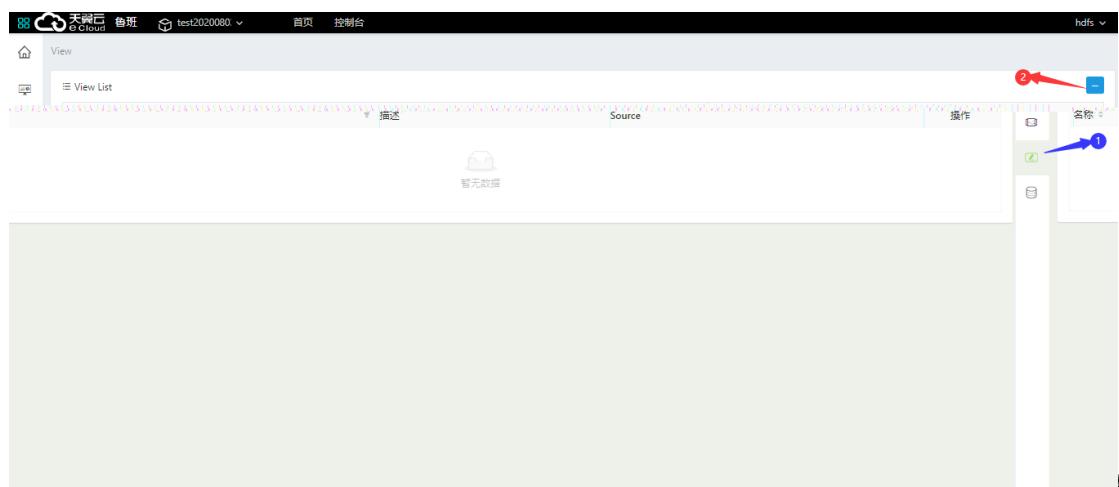
2. 进 页面, source 标。



3. 页面新写信，上提示成，保存。

#

### Step 3 视



1. View 标，在页面 新。

#

The screenshot shows the Tianyun Cloud Data Workbench interface. At the top, there's a navigation bar with '首页' (Home) and '控制台' (Console). Below it is a search bar with 'test001'. A red arrow labeled 1 points to the search bar. Another red arrow labeled 2 points to the 'SQL' tab. A third red arrow labeled 3 points to the 'Execute' button. On the right, a success message box is open with '执行成功' (Execution successful) and a green checkmark icon. Below the search bar, there's a dropdown menu with 'test001' selected. Underneath, there's a table preview with columns 'id' and 'department'. The table contains five rows of data. At the bottom right of the table preview, there are 'Cancel', 'Execute' (highlighted with a red arrow), 'Next Step', and 'Direct Save' buttons.

2. 在页面中写，以及 source，写 SQL 语句，执行

This screenshot shows the 'Model' configuration page for a data source named 'test001'. It includes sections for 'Model' and 'Auth'. In the 'Model' section, there are two rows: one for 'id' (set to '维度' - Dimension and '数字' - Number) and one for 'department' (set to '维度' - Dimension and '字符' - Character). A red arrow labeled 1 points to the 'id' row. A red arrow labeled 2 points to the 'Auth' tab at the top right. At the bottom right, there are 'Previous Step', 'Cancel', and 'Save' (highlighted with a red arrow) buttons. A green success message box is also visible.

3. 下一 编写数据模与权限，保存。

This screenshot shows the 'View' configuration page for the 'test001' view. It has a 'View List' table with one entry: '名称' (Name) is 'test001', '描述' (Description) is empty, and 'Source' is 'test001'. At the top right, there's a '修改' (Modify) button. Below the table, there are three red arrows labeled 1, 2, and 3 pointing to the 'Delete' (trash can), 'Edit' (pencil), and 'List' (list icon) buttons respectively. At the bottom right, there are 'Previous Step', 'Cancel', and 'Save' buttons.

#### Step 4 视 齐 组件

The screenshot shows the 'Widget' section of the dashboard. A red arrow labeled '1' points to the 'New' button (新建) in the top-left corner of the main content area. Another red arrow labeled '2' points to the 'Save' button (保存) in the top-right corner of the same area. The URL at the bottom of the browser window is 125.124.55.30:8088/dss/visualis/#/project/33/vizs.

1. widget 标进页面，新 widget。

#

The screenshot shows the configuration panel for a widget named 'test001'. A red arrow labeled '1' points to the 'Edit' button (编辑) in the top-left corner of the configuration panel. A blue arrow labeled '2' points to the 'Category' dropdown menu (分类型). A green arrow labeled '3' points to the 'Dimensions' (维度) section. A blue arrow labeled '4' points to the 'Indicator' (指标) section. A blue arrow labeled '5' points to the 'Search' (查询) button. A blue arrow labeled '6' points to the 'Save' (保存) button in the top-right corner of the configuration panel. The URL at the bottom of the browser window is 125.124.55.30:8088/dss/visualis/#/project/33/vizs.

2. 页面写widget，View，拖 维度 标 执行成，保存。

#

#

#### Step 4 显示页面

The screenshot shows the eCloud interface with the title 'Step 4 显示页面'. In the top navigation bar, there are icons for '天翼云' (Cloud), 'Viz', 'Dashboard', and 'Display'. The 'Display' tab is selected. A modal window titled '新增 Display' (Create New Display) is open, containing fields for 'Name' (必填) and 'Description', and a checkbox for '是否发布' (Is Published) with options '发布' (Published) and '编辑' (Edited). Red numbered arrows point to: 1. The 'Display' tab in the navigation bar; 2. The 'Create New Display' button in the 'Display' section; 3. The 'Name' input field in the modal; and 4. The 'Save' button in the modal. Below the modal, the 'Display' section shows a thumbnail of a display named 'test001'.

1. Viz 标进 页面, Display 在 中写 - ( ) - 是发布, 保存。

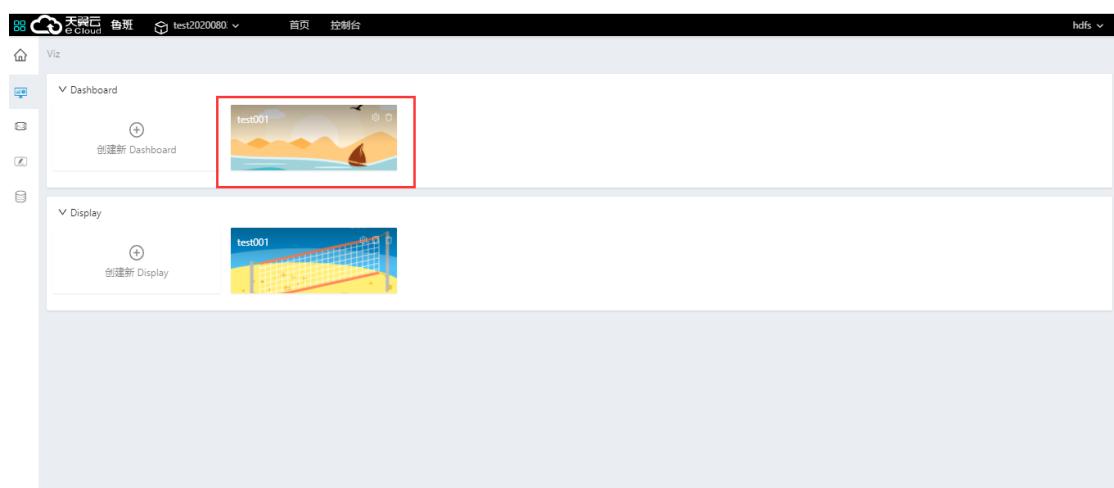
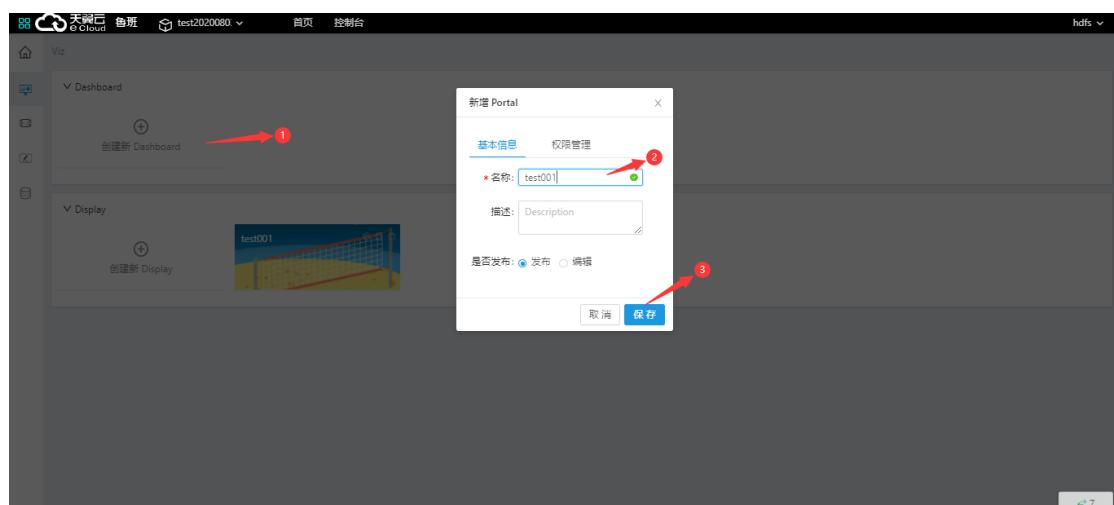
The screenshot shows the eCloud interface with the title 'Step 4 显示页面'. In the top navigation bar, there are icons for '天翼云' (Cloud), 'Viz', 'Dashboard', and 'Display'. The 'Display' tab is selected. A modal window titled '选择 Widgets' (Select Widgets) is open, showing a list of widgets with 'test001' selected. Red numbered arrows point to: 1. The 'Widgets' icon in the navigation bar; 2. The 'Widget' selection in the modal; and 3. The 'Next Step' button in the modal. To the right of the modal, there is a sidebar with sections for '图层' (Layers), '背景设置' (Background Settings), '屏幕尺寸' (Screen Size) (Width: 1920, Height: 1080), '背景颜色' (Background Color), '缩放' (Zoom) (Fit to Screen), and '图片' (Image).



2. 进入 Display 页面，在中选择的 widget

#

#### Step 5 Dashboard (可选)



1. 在中写 - ( )一是发布，保存。

#

## 数据质量 ( Qualitis )

### 功 明

Qualitis 是一套金融级、一站式的数  
理平台，提供了数据  
并用一套统一的流程 定义，数据  
使用流程是：新  
加技术  
任务执行  
任务查询

### 使用流

#### Step 1：新

| 项目ID | 项目名称        | 操作        |
|------|-------------|-----------|
| 19   | 测试          | 删除   执行任务 |
| 20   | test_0728_1 | 删除   执行任务 |
| 22   | 测试1         | 删除   执行任务 |
| 24   | 测试2         | 删除   执行任务 |
| 25   | test001     | 删除   执行任务 |
| 27   | test009     | 删除   执行任务 |
| 28   | 新的项目        | 删除   执行任务 |
| 29   | test002     | 删除   执行任务 |
| 32   | test3312    | 删除   执行任务 |
| 41   | bug_fix     | 删除   执行任务 |
| 46   | test0811    | 删除   执行任务 |

1. 在 **配置**，**我的**。
2. **主页面上的“新”**。

新增普通项目

项目名称: test0811

项目介绍: 测试

确定

3. **、**，**定**

4. 中示新

## Step 2：加技术

1. step1 中的新Id, 进上示面

3. 表行数检测，是校验

4. 在数据源、库、表件

5. 在SQL语句中我们可以配置的SQL语句

6. 结果集进行校验，是校验

7. 校验、模板、结果集进行校验

8. 定

#####

9. 技术新成

#

### Step 3 : 任务执行

The screenshot shows the Tianyi Cloud Qualitis interface for task execution. The main panel displays project details: Project Name (test\_0728\_1), Project ID (20), and Project Description (single table). Below these are tabs for '附加技术规则' (Additional Technical Rules), '导入技术规则' (Import Technical Rules), '导出技术规则' (Export Technical Rules), and '启动任务执行' (Start Task Execution). The '启动任务执行' button is highlighted with a red box. A table below lists rules, with two specific rows highlighted by blue boxes: '表行数检测' (Row Count Detection) with Rule ID 13 and Rule ID 20.

1. 执行的技术

2. 任务执行

This screenshot shows the task execution interface with a modal dialog for selecting the execution user. The dialog has a dropdown menu set to 'hdfs'. The background table lists rules, with the first two rows ('表行数检测' and '表行数检测2') selected, indicated by blue boxes around their checkboxes.

3. 执行用户

4. 定

#####

#

#### Step 4 : 任务查询

| 任务编号                              | 项目名称           | 技术规则数目 | 数据表                  | 任务提交时间              | 任务结束时间              | 任务状态    | 子任务:通过校验/失败/未通过校验 | 调度方式 |
|-----------------------------------|----------------|--------|----------------------|---------------------|---------------------|---------|-------------------|------|
| QUALITIS202008051151736420_111953 | test_0728_1    | 1      | default.dim.lain.bss | 2020-08-01 15:17:36 |                     | 任务初始化成功 | 0/0/0             | 接口调度 |
| QUALITIS2020080511515639_973847   | Flow_DataCheck | 1      | default.pdt_sales    | 2020-08-01 17:11:54 | 2020-08-01 17:12:41 | 未通过校验   | 0/0/1             | 接口调度 |
| QUALITIS2020080515252280_823179   | Flow_DataCheck | 1      | default.pdt_sales    | 2020-08-01 15:52:52 | 2020-08-01 15:52:58 | 未通过校验   | 0/0/1             | 接口调度 |
| QUALITIS20200805153544372_671308  | Flow_DataCheck | 1      | default.pdt_sales    | 2020-08-01 15:35:44 | 2020-08-01 15:35:57 | 未通过校验   | 0/0/1             | 接口调度 |
| QUALITIS20200805152453551_862309  | Flow_DataCheck | 1      | default.pdt_sales    | 2020-08-01 15:24:53 | 2020-08-01 15:24:57 | 失败      | 0/1/0             | 接口调度 |
| QUALITIS20200805151711439_083282  | Flow_DataCheck | 1      | default.pdt_sales    | 2020-08-01 15:17:11 | 2020-08-01 15:17:56 | 失败      | 0/1/0             | 接口调度 |
| QUALITIS2020080510016521_448597   | ch202008031606 | 1      | default.student      | 2020-08-01 11:00:16 | 2020-08-01 11:01:16 | 通过校验    | 1/0/0             | 接口调度 |
| QUALITIS20200801215316032_881128  | 测试1            | 1      | default.liyan2       | 2020-08-01 21:53:16 | 2020-08-01 21:54:10 | 通过校验    | 1/0/0             | 接口调度 |
| QUALITIS20200801175800756_865279  | 测试1            | 1      | default.liyan2       | 2020-08-01 17:58:00 | 2020-08-01 17:58:10 | 通过校验    | 1/0/0             | 接口调度 |
| QUALITIS20200801173325867_195633  | 测试1            | 1      | default.liyan2       | 2020-08-01 17:56:31 | 2020-08-01 17:56:40 | 未通过校验   | 0/0/1             | 接口调度 |
| QUALITIS20200801173325867_195638  | 测试1            | 1      | default.liyan2       | 2020-08-01 17:53:25 | 2020-08-01 17:53:30 | 未通过校验   | 0/0/1             | 接口调度 |

1. 任务查询
2. 到 执行的技术

```

2020-08-01 16:34:12.034 INFO Program is substituting variables for you
2020-08-01 16:34:12.034 INFO Variables substitution ended successfully
2020-08-01 16:34:12.034 INFO You have submitted a new job, script code (after variable substitution) is
-----SCRIPT CODE-----
var prop = new java.util.Properties();
prop.setProperty("user", "*****");
prop.setProperty("password", "*****");
var tmp1 = spark.sql("select count(*) from default.dim_lain_bss where prov_name != null");
tmp1.show();
-----LOG-----
2020-08-01 16:34:12.035 INFO You have submitted a new job, script code (after variable substitution) is
2020-08-01 16:34:12.035 INFO Your job is scheduled. Please wait for its execution.
2020-08-01 16:34:12.035 INFO Your job is running now. Please wait for its completion.
2020-08-01 16:34:12.035 INFO [sparkEngineEngineThread-2] com.webank.wechatphere.llm.engine.execute.CommonEngineJob[42] info sparkEngineEngine[1] change state Scheduled to Running
2020-08-01 16:34:12.035 INFO [sparkEngineEngineThread-2] com.webank.wechatphere.llm.engine.execute.CommonEngineJob[42] info sparkEngineEngine[1] executors[sparkEngineEngine[1].executors[0].llmEngineExecutor[0].sparkTask[0].task[0]] change state Running to Idle

```

3. 数据，查 任务执行的
4. 多，可以查 具 的

#### 数据治理 (Atlas)

#### 功 明

Atlas 是 apache 下的大数据的元数据 理和数据治理平台。 为 Hadoop 集群提供了包 数据分类、集中 管理在内的元数据治理 能力，支持 hive、storm、kafka、hbase、sqoop 等进行元数据 理以及以 式展示数据的 系。

## 使用流

### Step 1：配置 信

The screenshot shows the Apache Atlas search interface. On the left, there is a search bar with 'hive\_table (43)' selected. Below it are filters for 'Search By Classification' and 'Search By Term'. A search term 'dim\_latn\_bss' is entered in the 'Search By Text' field. On the right, the search results table displays one record: 'dim\_latn\_bss' (Owner: hdfs, Type: hive\_table). The 'Properties' column for this record is highlighted with a red box.

1. 在 , 写 查询的 的 信

2.

3. 主页面上 示 查询的 的 信

The screenshot shows the Apache Atlas Properties page for the 'dim\_latn\_bss' table. The left sidebar has 'hive\_table (43)' selected. The main area shows tabs for 'Properties', 'Lineage', 'Relationships', 'Classifications', 'Audits', and 'Schema'. The 'Properties' tab is active and displays a table of key-value pairs. The 'columns (7)' section lists 'prov\_id', 'city\_id', 'prov\_name', 'city\_name', 'area\_name', and 'area\_node'. Other properties include 'createTime' (Fri Jul 24 2020 10:58:00 GMT+0800 (中国标准时间)), 'db' (default), 'lastAccessTime' (Fri Jul 24 2020 10:58:00 GMT+0800 (中国标准时间)), and 'name' (dim\_latn\_bss).

4. 进 页面

5. Properties 中 示的是 的元数据的 信

6. Lineage 表示的是数据的血缘

#

## 数据权限(Ranger)

### 功能

数据权限(Ranger)是一个支持、管理和治理全面数据安全的功能。

#

### 功能

#### 配置 Hive 权限

- 在页面中添加hive文件中，在页面中写service name- Active Status- Select Tag  
Service-Username-Password-jdbc.driverClassName-jdbc.url-save

Service Details :

|                    |                                                                         |   |
|--------------------|-------------------------------------------------------------------------|---|
| Service Name *     | <input type="text" value="hive_yt"/>                                    | ① |
| Description        | <input type="text" value="hive_ytal"/>                                  | ② |
| Active Status      | <input checked="" type="radio"/> Enabled <input type="radio"/> Disabled | ③ |
| Select Tag Service | <input type="text" value="hive_tag"/>                                   | ④ |

Config Properties :

|                             |                                                               |   |
|-----------------------------|---------------------------------------------------------------|---|
| Username *                  | <input type="text" value="hive"/>                             | ⑤ |
| Password *                  | <input type="password" value="....."/>                        | ⑥ |
| jdbc.driverClassName *      | <input type="text" value="org.apache.hive.jdbc"/>             | ⑦ |
| jdbc.url *                  | <input type="text" value="jdbc:hive2://192.168.1.128:10000"/> | ⑧ |
| Common Name for Certificate | <input type="text"/>                                          |   |

Add New Configurations

|                        |       |
|------------------------|-------|
| Name                   | Value |
| tag.download.auth      | hive  |
| policy.download.all    | hive  |
| policy.grantrevoke.all | hive  |

2. service 进页面，页面 add new policy 加权 Permissions。

Ranger Service Manager

Security Zone : Select Zone Name

|                                |                                        |                                                                 |
|--------------------------------|----------------------------------------|-----------------------------------------------------------------|
| <input type="checkbox"/> HDFS  | <input type="checkbox"/> HBASE         | <input type="checkbox"/> HIVE <span style="color:red;">①</span> |
| <input type="checkbox"/> YARN  | <input type="checkbox"/> KNOX          | <input type="checkbox"/> STORM                                  |
| <input type="checkbox"/> SOLR  | <input type="checkbox"/> KAFKA         | <input type="checkbox"/> NIFI                                   |
| <input type="checkbox"/> KYLIN | <input type="checkbox"/> NIFI-REGISTRY | <input type="checkbox"/> SQUIOP                                 |
| <input type="checkbox"/> ATLAS | <input type="checkbox"/> ELASTICSEARCH | <input type="checkbox"/> PRESTO                                 |
| <input type="checkbox"/> OZONE |                                        |                                                                 |

Licensed under the Apache License, Version 2.0

Ranger Service Manager > hive\_yt Policies > Create Policy

Create Policy

Policy Details :

|               |                                           |                                                    |
|---------------|-------------------------------------------|----------------------------------------------------|
| Policy Type   | <input checked="" type="radio"/> Access   | <input type="button" value="Add Validity Period"/> |
| Policy Name * | <input type="text" value="test001"/>      | ①                                                  |
| Policy Label  | <input type="text" value="Policy Label"/> | ②                                                  |
| hiveserv. *   | <input type="text" value="hive_yt"/>      | ③                                                  |
| Description   | <input type="text"/>                      |                                                    |
| Audit Logging | <input checked="" type="radio"/> YES      |                                                    |

Allow Conditions :

| Select Role                               | Select Group                               | Select User                            | Permissions                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | Delegate Admin           |
|-------------------------------------------|--------------------------------------------|----------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| <input type="text" value="Select Roles"/> | <input type="text" value="Select Groups"/> | <input type="text" value="x cheng"/> ④ | <input type="checkbox"/> All<br><input type="checkbox"/> Alter<br><input type="checkbox"/> Create<br><input type="checkbox"/> Drop<br><input type="checkbox"/> Index<br><input type="checkbox"/> Lock<br><input type="checkbox"/> Read<br><input type="checkbox"/> Refresh<br><input type="checkbox"/> ReplAdmin<br><input type="checkbox"/> select<br><input type="checkbox"/> Service Admin<br><input type="checkbox"/> Temporary UDF Admin<br><input type="checkbox"/> Update<br><input type="checkbox"/> Write | <input type="checkbox"/> |

The screenshot shows the Ranger Access Manager interface. At the top, there are tabs for Service Manager, hive\_yt Policies, Create Policy, Security Zone, and Settings. A user 'admin' is logged in. The main area is titled 'Create Policy' under 'hive\_yt Policies'. It has sections for 'Deny All Other' (set to False) and 'Deny Conditions'. Each 'Deny Conditions' section has tabs for 'Select Role', 'Select Group', 'Select User', 'Permissions', and 'Delegate Admin'. There are two such sections, each with an 'Add' button. Below these is a 'Deny Conditions' table with columns: Policy ID, Policy Name, Policy Labels, Status, Audit Logging, Roles, Groups, Users, and Action. The table contains several rows, including 'all - database, table, column', 'all - database, udf', 'all - url', 'user\_info\_access\_hdfs', and 'Hive Global UDF Allow'. The 'Action' column for the last row shows icons for edit, delete, and copy. On the right side of the table, there are buttons for 'Add New Policy' and 'Edit'.

## 案 1：屏（配置 hdfs 用户 phone 屏 .）

The screenshot shows the Ranger Access Manager interface. At the top, there are tabs for Service Manager, hive\_yt Policies, Masking, and Row Level Filter. A user 'admin' is logged in. The main area is titled 'hive\_yt Policies' and shows a table of policies. The 'Masking' tab is selected. The table has columns: Policy ID, Policy Name, Policy Labels, Status, Audit Logging, Roles, Groups, Users, and Action. One row is visible, labeled 'phone\_masking'. The 'Action' column for this row shows icons for edit, delete, and copy. A blue arrow points from the number 1 to the search bar at the top of the table. Another blue arrow points from the number 2 to the 'Add New Policy' button on the right side of the table. A success message 'Policy deleted successfully' is displayed above the table.

**Service Manager > hive\_yt Policies > Create Policy**

**Policy Details:**

- Policy Type: Masking (1)
- Policy Name: test003 (2)
- Policy Label: (3)
- Hive Database: default (4)
- Hive Table: userinfo (5)
- Hive Column: phone (6)
- Description: (7)
- Audit Logging: YES (8)

**Mask Conditions:**

| Select Role  | Select Group  | Select User | Access Types | Select Masking Option     |
|--------------|---------------|-------------|--------------|---------------------------|
| Select Roles | Select Groups | (x cheng)   | select       | Partial mask: show last 4 |

**Ranger Access Manager > hive\_yt Policies > Create Policy**

**Mask Conditions:**

| Select Role  | Select Group  | Select User | Access Types | Select Masking Option     |
|--------------|---------------|-------------|--------------|---------------------------|
| Select Roles | Select Groups | (x cheng)   | select       | Partial mask: show last 4 |

Licensed under the Apache License, Version 2.0.

**Ranger Access Manager > hive\_yt Policies**

**List of Policies: hive\_yt**

| Policy ID | Policy Name   | Policy Labels | Status  | Audit Logging | Roles | Groups | Users | Action |  |
|-----------|---------------|---------------|---------|---------------|-------|--------|-------|--------|--|
| 6         | phone_masking | --            | Enabled | Enabled       | --    | --     | hdfs  |        |  |
| 17        | test003       | --            | Enabled | Enabled       | --    | --     | cheng |        |  |

Licensed under the Apache License, Version 2.0.

案 2：行 (配置 用户 id=29 的数据.)

Service Manager > hive\_yt Policies

Access Masking Row Level Filter

List of Policies : hive\_yt

Policy ID Policy Name Policy Labels Status Audit Logging Roles Groups Users Action

|   |              |    |         |         |    |    |      |  |
|---|--------------|----|---------|---------|----|----|------|--|
| 7 | score_filter | -- | Enabled | Enabled | -- | -- | hive |  |
|---|--------------|----|---------|---------|----|----|------|--|

Add New

Policy Details :

Policy Type: Row Level Filter 1

Policy Name \*: test006 2 enabled normal

Policy Label: Policy Label

Hive Database \*: x\_default 2

Hive Table \*: x\_student 3

Description:

Audit Logging: YES

Row Filter Conditions :

| Select Role                                 | Select Group                                 | Select User                                                           | Access Types                                                                                                                     | Row Level Filter                 |
|---------------------------------------------|----------------------------------------------|-----------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|----------------------------------|
| <input type="button" value="Select Roles"/> | <input type="button" value="Select Groups"/> | <input type="text" value="x_cheng"/> <span style="color:red">4</span> | <input type="checkbox"/> select <input type="checkbox"/> update <input type="checkbox"/> delete <span style="color:red">5</span> | <span style="color:red">6</span> |

Hive Table \*: x\_student

Description:

Audit Logging: YES

Row Filter Conditions :

| Select Role                                                                                               | Select Group                                 | Select User                                                           | Access Types                                                                                                                     | Row Level Filter                 |
|-----------------------------------------------------------------------------------------------------------|----------------------------------------------|-----------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|----------------------------------|
| <input type="button" value="Select Roles"/>                                                               | <input type="button" value="Select Groups"/> | <input type="text" value="x_cheng"/> <span style="color:red">4</span> | <input type="checkbox"/> select <input type="checkbox"/> update <input type="checkbox"/> delete <span style="color:red">5</span> | <span style="color:red">6</span> |
| <span style="color:red">1</span> <input type="button" value="Add"/> <input type="button" value="Cancel"/> |                                              |                                                                       |                                                                                                                                  |                                  |

请文件 测试 首页 控制台

admin Ranger Access Manager Audits Security Zone Settings

Service Manager > hive\_yt Policies

Access Masking Row Level Filter

List of Policies : hive\_yt

Search for your policy... Add New Policy

| Policy ID | Policy Name  | Policy Labels | Status  | Audit Logging | Roles | Groups | Users | Action |  |  |
|-----------|--------------|---------------|---------|---------------|-------|--------|-------|--------|--|--|
| 7         | score_filter | --            | Enabled | Enabled       | --    | --     | hive  |        |  |  |
| 18        | test006      | --            | Enabled | Enabled       | --    | --     | cheng |        |  |  |

#

## 附录1：新闻舆情分析案例代码

### 1. hive\_ddl\_createtable 代码

```
create table if not exists ODS_T_NEWS (
`ID` STRING comment "ID",
`TITLE` STRING comment '新闻标题',
`DATE` STRING comment '新闻日期',
`COMPNAME` STRING comment '公司名字',
`NEWSCODE` STRING comment '新闻编号码',
`SENTIMENT` STRING comment '新闻评分，正面为 1，中性为 0，负面为-1')
partitioned BY (dt STRING)
stored as ORC
location 'hdfs:///tmp/linkis/hdfs/NewAnalysis/hive/ods_news';

create table if not exists DWS_T_NEWS (
`STATISTICDATE` STRING comment '统计日期',
`COMPNAME` STRING comment '公司名字',
`POSCOUNT` int comment '新闻评分正面为 1 条数',
`NEUTRALCOUNT` int comment '新闻评分中性为 0 条数',
`NEGCOUNT` int comment '新闻评分负面为-1 条数')
stored as ORC
location 'hdfs:///tmp/linkis/hdfs/NewAnalysis/hive/dws_news';

create table if not exists DWA_T_NEW_NEGATIVE (
`STATISTICDATE` STRING comment '统计日期',
`COMPNAME` STRING comment '公司名字',
`WEEK1COUNT` int comment '新闻评分负面为-1 条数' ,
`WEEK2COUNT` int comment '新闻评分负面为-1 条数' ,
`WEEK3COUNT` int comment '新闻评分负面为-1 条数' ,
`WEEK4COUNT` int comment '新闻评分负面为-1 条数' ,
`WEEK5COUNT` int comment '新闻评分负面为-1 条数' ,
`WEEK6COUNT` int comment '新闻评分负面为-1 条数' ,
`WEEK7COUNT` int comment '新闻评分负面为-1 条数' ,
`WEEK8COUNT` int comment '新闻评分负面为-1 条数')
stored as ORC
location 'hdfs:///tmp/linkis/hdfs/NewAnalysis/hive/dwa_t_news_neg';

create table if not exists DWA_T_NEW_POSTIVE (
`STATISTICDATE` STRING comment '统计日期',
`COMPNAME` STRING comment '公司名字',
`WEEK1COUNT` int comment '新闻评分正面为 1 条数' ,
```

```

`WEEK2COUNT` int comment '新闻评分正面为1条数' ,
`WEEK3COUNT` int comment '新闻评分正面为1条数' ,
`WEEK4COUNT` int comment '新闻评分正面为1条数' ,
`WEEK5COUNT` int comment '新闻评分正面为1条数' ,
`WEEK6COUNT` int comment '新闻评分正面为1条数' ,
`WEEK7COUNT` int comment '新闻评分正面为1条数' ,
`WEEK8COUNT` int comment '新闻评分正面为1条数')
stored as ORC
location 'hdfs:///tmp/linkis/hdfs/NewAnalysis/hive/dwa_t_news_pos';
#

```

## 2. genData\_1、genData\_2、genData\_3 代码(3者代码相同)

```

import java.text.SimpleDateFormat
import java.util
import java.util.{Calendar, Random}

import org.apache.spark.broadcast.Broadcast
import org.apache.spark.sql.{SQLContext, SaveMode, SparkSession}

val sqlCtx: SQLContext = spark.sqlContext
sqlCtx.setConf("hive.exec.dynamic.partition","true")
sqlCtx.setConf("hive.exec.dynamic.partition.mode","nonstrict")

val current_date = "2020-06-06"
//数据复制倍数
val copy_times = 1

val dtFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")
val date_arr = new util.ArrayList[String]()
for(i <- 1 to 68){
 val calendar: Calendar = Calendar.getInstance()
 calendar.setTime(new SimpleDateFormat("yyyy-MM-dd").parse(current_date))
 calendar.add(Calendar.DATE,-i)
 date_arr.add(dtFormat.format(calendar.getTimeInMillis))
}

val schemaString = "ID,TITLE,DATE,DT,COMPNAME,NEWSCODE,SENTIMENT"

val dtBroad: Broadcast[util.ArrayList[String]] = spark.sparkContext.broadcast(date_arr)
val fileDF = spark.read.format("csv").option("header", true).load("hdfs:///tmp/linkis/hdfs/NewAn
alysis/sample2020.csv") //需替换成自己上传的文件路径

```

```
import spark.implicits._

val ods_t_newsDF = fileDF.na.fill("unknow")
.map(x => {
 val buffer = new StringBuffer()
 for(i <- 1 to copy_times){
 val x1 = x(1).toString + new Random().nextInt()
 val index = Math.abs(new Random().nextInt(Integer.MAX_VALUE)) % 66
 val x2 = dtBroad.value.get(index)
 var x5 = x(5)
 if(x5.toString.trim.isEmpty) {
 x5 = -1
 }
 }
})
```

### 3. NewByDay 代码

```
import org.apache.spark.sql.{SQLContext, SaveMode, SparkSession}

val sqlCtx: SQLContext = spark.sqlContext
sqlCtx.setConf("hive.exec.dynamic.partition","true")
sqlCtx.setConf("hive.exec.dynamic.partition.mode","nonstrict")
val ods_t_news_negDF = spark.sql("select DT ,COMPNAME from ODS_T_NEWS where SENTIMENT = -1")
var schema_neg = "STATISTICDATE,COMPNAME,NEGCOUNT"
val dwt_t_news_negDF = ods_t_news_negDF.groupBy("DT", "COMPNAME").count().toDF(schema_neg.split(","))
val ods_t_news_neaDF = spark.sql("select DT ,COMPNAME from ODS_T_NEWS where SENTIMENT = 0")
var schema_nea = "STATISTICDATE,COMPNAME,NEUTRALCOUNT"
val dwt_t_news_neaDF = ods_t_news_neaDF.groupBy("DT", "COMPNAME").count().toDF(schema_nea.split(","))
val ods_t_news_posDF = spark.sql("select DT ,COMPNAME from ODS_T_NEWS where SENTIMENT = 1")
var schema_pos = "STATISTICDATE,COMPNAME,POSCOUNT"
val dwt_t_news_posDF = ods_t_news_posDF.groupBy("DT", "COMPNAME").count().toDF(schema_pos.split(","))
val dwt_t_newsDF = dwt_t_news_negDF.join(dwt_t_news_neaDF, "STATISTICDATE,COMPNAME".split(","))
.djoin(dwt_t_news_posDF, "STATISTICDATE,COMPNAME".split(","))
dwt_t_newsDF.show()
dwt_t_newsDF.write.format("hive").mode(SaveMode.Append).saveAsTable("DWS_T_NEWS")
```

### 4. NegNewByWeek 代码

```
import java.text.SimpleDateFormat
import java.util.Calendar

import org.apache.spark.sql.{SQLContext, SaveMode, SparkSession}

def pre_n_week(date:String,n:Int) = {
 val dateFormat = new SimpleDateFormat("yyyy-MM-dd")
 val calendar = Calendar.getInstance()
 calendar.setTime(dateFormat.parse(date))
 calendar.set(Calendar.DAY_OF_WEEK,Calendar.MONDAY)

 val monday = new SimpleDateFormat("yyyy-MM-dd").format(calendar.getTime)
 calendar.setTime(dateFormat.parse(monday))
 calendar.set(Calendar.DATE, -7* (n-1))
```

```

 val last_sun = new SimpleDateFormat("yyyy-MM-dd").format(calendar.getTime)
 calendar.setTime(dateFormat.parse(monday))
 calendar.set(Calendar.DATE, - 7*n + 1)
 val last_mon = new SimpleDateFormat("yyyy-MM-dd").format(calendar.getTime)
 (last_mon,last_sun)
}

val date_str = "2020-06-06"

var week_1 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_news where STATISTICDATE >= '${pre_n_week(date_str,1)._1}' and STATISTICDATE <= '${pre_n_week(date_str,1)._2}' group by COMPNAME"
"
val week_1DF = spark.sql(week_1).toDF("COMPNAME","WEEK1COUNT")

var week_2 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_news where STATISTICDATE >= '${pre_n_week(date_str,2)._1}' and STATISTICDATE <= '${pre_n_week(date_str,2)._2}' group by COMPNAME"
"
val week_2DF = spark.sql(week_2).toDF("COMPNAME","WEEK2COUNT")

var week_3 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_NEWS where STATISTICDATE >= '${pre_n_week(date_str,3)._1}' and STATISTICDATE <= '${pre_n_week(date_str,3)._2}' group by COMPNAME"
"
val week_3DF = spark.sql(week_3).toDF("COMPNAME","WEEK3COUNT")

var week_4 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_NEWS where STATISTICDATE >= '${pre_n_week(date_str,4)._1}' and STATISTICDATE <= '${pre_n_week(date_str,4)._2}' group by COMPNAME"
"
val week_4DF = spark.sql(week_4).toDF("COMPNAME","WEEK4COUNT")

var week_5 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_NEWS where STATISTICDATE >= '${pre_n_week(date_str,5)._1}' and STATISTICDATE <= '${pre_n_week(date_str,5)._2}' group by COMPNAME"
"
val week_5DF = spark.sql(week_5).toDF("COMPNAME","WEEK5COUNT")

var week_6 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_NEWS where STATISTICDATE >= '${pre_n_week(date_str,6)._1}' and STATISTICDATE <= '${pre_n_week(date_str,6)._2}' group by COMPNAME"
"
val week_6DF = spark.sql(week_6).toDF("COMPNAME","WEEK6COUNT")

var week_7 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_NEWS where STATISTICDATE >= '${pre_n_week(date_str,7)._1}' and STATISTICDATE <= '${pre_n_week(date_str,7)._2}' group by COMPNAME"
"
val week_7DF = spark.sql(week_7).toDF("COMPNAME","WEEK7COUNT")

```

```

var week_8 = s"select COMPNAME, count(1) WEEK1COUNT from DWS_T_NEWS where STATISTICDATE >= '${pre_n_week(date_str,8)._1}' and STATISTICDATE <= '${pre_n_week(date_str,8)._2}' group by COMPNAME"
"

val week_8DF = spark.sql(week_8).toDF("COMPNAME", "WEEK8COUNT")

val schema = "STATISTICDATE,COMPNAME,WEEK1COUNT,WEEK2COUNT,WEEK3COUNT,WEEK4COUNT,WEEK5COUNT,WEEK6COUNT,WEEK7COUNT,WEEK8COUNT"

val dt_broad = spark.sparkContext.broadcast(date_str)

import spark.implicits._

val joinCol = Seq("COMPNAME")

val dwa_t_news = week_1DF
 .join(week_2DF, joinCol, "inner")
 .join(week_3DF, joinCol, "inner")
 .join(week_4DF, joinCol, "inner")
 .join(week_5DF, joinCol, "inner")
 .join(week_6DF, joinCol, "inner")
 .join(week_7DF, joinCol, "inner")
 .join(week_8DF, joinCol, "inner")
 .map(x => {
 (dt_broad.value, x(0).toString, x(1).toString.toInt, x(2).toString.toInt, x(3).toString.toInt,
 x(4).toString.toInt, x(5).toString.toInt, x(6).toString.toInt, x(7).toString.toInt, x(8).toString.toInt)
 })
 .toDF(schema.split(","):_*)

dwa_t_news.show(10)

dwa_t_news.write.format("hive").mode(SaveMode.Append).saveAsTable("DWA_T_NEW_NEGATIVE")

```

## 5. hiveToMysql 代码：

```

import org.apache.spark.sql.{SaveMode, SparkSession}

val date_str = "2020-06-06"
val title = "STATISTICDATE,COMPNAME,WEEK1COUNT ,WEEK2COUNT, WEEK3COUNT, WEEK4COUNT ,WEEK5COUNT ,
WEEK6COUNT ,WEEK7COUNT ,WEEK8COUNT"

val loadDateFromHive = s"select STATISTICDATE,COMPNAME,WEEK1COUNT ,WEEK2COUNT, WEEK3COUNT, WEEK4COUNT ,WEEK5COUNT ,WEEK6COUNT ,WEEK7COUNT ,WEEK8COUNT from DWA_T_NEW_NEGATIVE "

```

```

import spark.implicits._

val hiveDF = spark.sql(loadDateFromHive)

hiveDF .toDF(title.split(","):_*).show(10)

val url = "jdbc:mysql://192.168.0.38:3306/news?useUnicode=true&characterEncoding=gbk&autoReconnect=true&failOverReadOnly=false"
val driver = "com.mysql.cj.jdbc.Driver"
val username = "root"
val password = "Bigdata2@20"
val tablename = "DWA_T_NEW"

hiveDF.write
 .format("jdbc")
 .option("driver",driver)
 .option("url",url)
 .option("user",username)
 .option("password",password)
 .option("dbtable",tablename)
 .option("numPartitions","50")
 .option("batchsize","500")
 .mode(SaveMode.Append)
 .save()

```

## 6. mysql\_ddl 代码:

```

create database if not exists news;
drop table if exists news.DWA_T_NEW;
create table if not exists news.DWA_T_NEW (
`id` int not null AUTO_INCREMENT comment '主键',
`STATISTICDATE` varchar(32) comment '统计日期',
`COMPNAME` Text not null comment '公司名字',
`WEEK1COUNT` int comment '新闻评分正面为-1 条数' ,
`WEEK2COUNT` int comment '新闻评分正面为-1 条数' ,
`WEEK3COUNT` int comment '新闻评分正面为-1 条数' ,
`WEEK4COUNT` int comment '新闻评分正面为-1 条数' ,
`WEEK5COUNT` int comment '新闻评分正面为-1 条数' ,
`WEEK6COUNT` int comment '新闻评分正面为-1 条数' ,
`WEEK7COUNT` int comment '新闻评分正面为-1 条数' ,
`WEEK8COUNT` int comment '新闻评分正面为-1 条数' ,
constraint pk_news primary key(id)
) ENGINE=InnoDB AUTO_INCREMENT=10000 DEFAULT CHARSET=utf8;

```

## 附录 2：航空公司的客户价值分析案例代码

### 1. JDBC1 代码

```
CREATE DATABASE IF NOT EXISTS `lubanbackup` ;
USE `lubanbackup` ;#
DROP TABLE IF EXISTS `cvtest`;#
CREATE TABLE `cvtest` (
 `MEMBER_NO` varchar(50) DEFAULT NULL,
 `FFP_DATE` varchar(50) DEFAULT NULL,
 `FIRST_FLIGHT_DATE` varchar(50) DEFAULT NULL,
 `GENDER` varchar(50) DEFAULT NULL,
 `FFP_TIER` varchar(50) DEFAULT NULL,
 `WORK_CITY` varchar(50) DEFAULT NULL,
 `WORK_PROVINCE` varchar(50) DEFAULT NULL,
 `WORK_COUNTRY` varchar(50) DEFAULT NULL,
 `age` varchar(50) DEFAULT NULL,
 `LOAD_TIME` varchar(50) DEFAULT NULL,
 `FLIGHT_COUNT` varchar(50) DEFAULT NULL,
 `BP_SUM` varchar(50) DEFAULT NULL,
 `EP_SUM_YR_1` varchar(60) DEFAULT NULL,
 `EP_SUM_YR_2` varchar(60) DEFAULT NULL,
 `SUM_YR_1` varchar(60) DEFAULT NULL,
 `SUM_YR_2` varchar(60) DEFAULT NULL,
 `SEG_KM_SUM` varchar(60) DEFAULT NULL,
 `WEIGHTED_SEG_KM` varchar(60) DEFAULT NULL,
 `LAST_FLIGHT_DATE` varchar(50) DEFAULT NULL,
 `AVG_FLIGHT_COUNT` varchar(60) DEFAULT NULL,
 `AVG_BP_SUM` varchar(60) DEFAULT NULL,
 `BEGIN_TO_FIRST` varchar(50) DEFAULT NULL,
 `LAST_TO_END` varchar(50) DEFAULT NULL,
 `AVG_INTERVAL` varchar(50) DEFAULT NULL,
 `MAX_INTERVAL` varchar(50) DEFAULT NULL,
 `ADD_POINTS_SUM_YR_1` varchar(50) DEFAULT NULL,
 `ADD_POINTS_SUM_YR_2` varchar(50) DEFAULT NULL,
 `EXCHANGE_COUNT` varchar(50) DEFAULT NULL,
 `avg_discount` varchar(60) DEFAULT NULL,
 `P1Y_Flight_Count` varchar(50) DEFAULT NULL,
 `L1Y_Flight_Count` varchar(50) DEFAULT NULL,
 `P1Y_BP_SUM` varchar(60) DEFAULT NULL,
 `L1Y_BP_SUM` varchar(60) DEFAULT NULL,
 `EP_SUM` varchar(60) DEFAULT NULL,
```

```

`ADD_Point_SUM` varchar(60) DEFAULT NULL,
`Eli_Add_Point_Sum` varchar(60) DEFAULT NULL,
`L1Y_ELI_Add_Points` varchar(60) DEFAULT NULL,
`Points_Sum` varchar(60) DEFAULT NULL,
`L1Y_Points_Sum` varchar(60) DEFAULT NULL,
`Ration_L1Y_Flight_Count` varchar(60) DEFAULT NULL,
`Ration_P1Y_Flight_Count` varchar(60) DEFAULT NULL,
`Ration_P1Y_BPS` varchar(60) DEFAULT NULL,
`Ration_L1Y_BPS` varchar(60) DEFAULT NULL,
`Point_NotFlight` varchar(60) DEFAULT NULL,
KEY `MEMBER_NO` (`MEMBER_NO`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

use lubanbackup ;#
DROP TABLE IF EXISTS `cvresult`;
CREATE TABLE `cvresult` (
`ZL` double DEFAULT NULL,
`ZR` double DEFAULT NULL,
`ZF` double DEFAULT NULL,
`ZM` double DEFAULT NULL,
`ZC` double DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
show tables;

```

## 2.SPARKML 代码

```

import java.sql.{Connection, DriverManager, PreparedStatement}
import java.text.{ParseException, SimpleDateFormat}
import java.util.{Calendar, Date}

import com.sun.xml.internal.ws.spi.db.BindingContextFactory.LOGGER
import org.apache.spark.SparkConf
import org.apache.spark.ml.clustering.KMeans
import org.apache.spark.ml.feature.{StandardScaler, VectorAssembler}
import org.apache.spark.sql.SparkSession
#
case class LRFMC(L:Int,R:Double,F:Double,M:Double,C:Double)
#
val url = "jdbc:mysql://192.168.0.38:3306/lubanbackup?serverTimezone=UTC&useUnicode=true&characterEncoding=gbk&autoReconnect=true&failOverReadOnly=false"
val driver = "com.mysql.cj.jdbc.Driver"
val username = "root"
val password = "Bigdata2@20"

```

```
val tablename = "cvtest"#
val myqlDf=spark.read
 .format("jdbc")
 .option("driver",driver)
 .option("url",url)
 .option("user",username)
 .option("password",password)
 .option("dbtable",tablename)
 .load()
#
import spark.implicits._
#
val seq = Seq("LOAD_TIME", "FFP_DATE", "LAST_TO_END", "FLIGHT_COUNT", "SEG_KM_SUM", "avg_discount")
val mysqlETLDF = myqlDf.filter((myqlDf("SUM_YR_1") != 0) || (myqlDf("SUM_YR_2") != 0) || (myqlDf("SEG_KM_SUM") == 0 && myqlDf("avg_discount") == 0)).select(seq.head,seq.tail:_*)
#
def monthDiff(bef: String, aft: String): Int = { //解析出日期对象
 val sdf = new SimpleDateFormat("yyyy/MM/dd")
 var date1:Date= null
 var date2:Date = null
 try {
 date1 = sdf.parse(bef)
 date2 = sdf.parse(aft)
 } catch {
 case e: ParseException =>
 //e.printStackTrace();
 LOGGER.info("error mess :" + e.getMessage + "," + e.toString + date1 + "," + date2)
 return -100
 }
}
//使用 Calendar 对象来操作日期对象
val c1 = Calendar.getInstance
val c2 = Calendar.getInstance
c1.setTime(date1)
c2.setTime(date2)
//获取日期的毫秒数，用于比较日期谁先谁后
val time1 = date1.getTime
val time2 = date2.getTime
//年数相减
val yearSubtract = c2.get(Calendar.YEAR) - c1.get(Calendar.YEAR)
//月数相减
var monthSubtract = 0
//根据日期的先后应用不同的公式
if (time1 < time2) monthSubtract = c2.get(Calendar.MONTH) - c1.get(Calendar.MONTH) + 1
```

```

 else monthSubtract = c2.get(Calendar.MONTH) - c1.get(Calendar.MONTH) - 1

 Math.abs(yearSubtract * 12 + monthSubtract).toInt

 }

#

 val schema = "L,R,F,M,C".split(",")

 val transDF = mysqlETLDF.map(line => {

 val L = monthDiff(line.get(0).toString, line.get(1).toString)

 if (L == -100) {

 null

 } else {

#
 LRFMC(L, line.get(2).toString.toDouble, line.get(3).toString.toDouble, line.get(4).toString.toDouble, line.get(5).toString.toDouble)

 }

 }).filter(line => line != null)

#

 val assembler = new VectorAssembler()

 .setInputCols(schema)

 .setOutputCol("feature_vec")

#

//regular

 val scaler = new StandardScaler()

 .setInputCol("feature_vec")

 .setOutputCol("features")

 .setWithStd(true)

 .setWithMean(true)

#

//df

 val df_vec = assembler.transform(transDF)

 val df_scaler = scaler.fit(df_vec).transform(df_vec)

 df_scaler.show()

#

 val kmeans = new KMeans().setFeaturesCol("features").setK(5).setMaxIter(20)

 val model = kmeans.fit(df_scaler)

 val df_Predict = model.transform(df_scaler)

 (df_Predict,model)

#

def conn():Connection = {

 var conn: Connection = null

 try {

 Class.forName(driver)

 conn = DriverManager.getConnection(url,username,password)

 }catch {

 case e: Exception => println("Mysql Exception" + e.getMessage + "," + e.getStackTrace)

 }

}

```

```

 }

 conn

}

#

def write(conn:Connection,arr:Array[Double]): Unit = {
 var ps: PreparedStatement = null
 val sql = "insert into cvresult(ZL,ZR,ZF,ZM,ZC) values (?,?,?,?,?,?)"
 try {
 ps = conn.prepareStatement(sql)
 ps.setDouble(1,arr(0))
 ps.setDouble(2,arr(1))
 ps.setDouble(3,arr(2))
 ps.setDouble(4,arr(3))
 ps.setDouble(5,arr(4))
 ps.executeUpdate()
 } catch {
 case e: Exception => println("Mysql Exception"+ e.getMessage + "," + e.getStackTrace)
 } finally {
 ps.close()
 }
}

#

val res = model.clusterCenters.foreach(row => {
 println("cluster center: " + row)
 write(conn(),row.toArray)
 // ZLRFCMC(row(0),row(1),row(2),row(3),row(4))
})
println("tmp res: " + res)
#

//show stat
df_Predict.groupBy("prediction").count().show()
#
}

```

### 3.JDBC2行

```

USE `lubanbackup`;

/*Table structure for table `cvresultn` */
#
DROP TABLE IF EXISTS `cvresultn`;
#
CREATE TABLE `cvresultn` (
`colname` varchar(20) DEFAULT NULL,

```

```

`customer1` double DEFAULT NULL,
`customer2` double DEFAULT NULL,
`customer3` double DEFAULT NULL,
`customer4` double DEFAULT NULL,
`customer5` double DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

#
/*Data for the table `cvresultn` */

#
insert into `cvresultn`(`colname`,`customer1`,`customer2`,`customer3`,`customer4`,`customer5`) values
('ZL',-0.397,-0.706,-0.271,0.481,1.187),
('ZR',-0.301,-0.353,1.771,-0.8,-0.375),
('ZF',-0.105,-0.229,-0.578,2.47,-0.081),
('ZM',-0.166,-0.198,-0.54,2.415,-0.087),
('ZC',1.17,-0.577,-0.064,0.324,-0.14);
#

```

#### 4 JDBC3 代

```

USE `lubanbackup`;

/*Table structure for table `cv_result_num` */

#
DROP TABLE IF EXISTS `cv_result_num`;
#

CREATE TABLE `cv_result_num` (
`重要保持客户` varchar(20) DEFAULT NULL,
`重要发展客户` varchar(20) DEFAULT NULL,
`重要挽留客户` varchar(20) DEFAULT NULL,
`一般客户` varchar(20) DEFAULT NULL,
`低价值客户` varchar(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

#
/*Data for the table `cv_result_num` */

#
insert into `cv_result_num`(`重要保持客户`, `重要发展客户`, `重要挽留客户`, `一般客户`, `低价值客户`) values
('19692','30876','22740','39940','10840');
select * from cv_result_num;

```