UniCloud 集成平台

用户手册

紫光云技术有限公司
www.unicloud.com

资料版本：5W101-20230417
产品版本：UniCloud iPaaS (E6103)

前 言

本手册主要介绍了 UniCloud 集成平台概述、功能介绍、典型配置案例、常见问题解答等内容。

前言部分包含如下内容：

读者对象

本手册主要适用于如下工程师：

- 网络规划人员
- 现场技术支持与维护人员
- 负责网络配置和维护的网络管理员

本书约定

1. 图形界面格式约定

格　式	意　义
< >	带尖括号“< >”表示按钮名，如“单击<确定>按钮”。
[ ]	带方括号“[ ]”表示窗口名、菜单名和数据表，如“弹出[新建用户]窗口”。
/	多级菜单用“/”隔开。如[文件/新建/文件夹]多级菜单表示[文件]菜单下的[新建]子菜单下的[文件夹]菜单项。

2. 各类标志

本书还采用各种醒目标志来表示在操作过程中应该特别注意的地方，这些标志的意义如下：

 警告	该标志后的注释需给予格外关注，不当的操作可能会对人身造成伤害。
 注意	提醒操作中应注意的事项，不当的操作可能会导致数据丢失或者设备损坏。
 提示	为确保设备配置成功或者正常工作而需要特别关注的操作或信息。
 说明	对操作内容的描述进行必要的补充和说明。
 窍门	配置、操作、或使用设备的技巧、小窍门。

3. 端口编号示例约定

本手册中出现的端口编号仅作示例，并不代表设备上实际具有此编号的端口，实际使用中请以设备上存在的端口编号为准。

资料意见反馈

如果您在使用过程中发现产品资料的任何问题，可以通过以下方式反馈：

E-mail：unicloud-ts@unicloud.com

感谢您的反馈，让我们做得更好！

目　录

1 概述

1.1 简介

UniCloud 集成平台是一个全栈式的集成平台，旨在打通应用和数据孤岛，实现异构数据/API/消息/设备集成，提供异构数据集成、应用间通信集成能力、API 接口集成能力及物理设备设备集成能力，助力打造标准统一、融会贯通、资产化、服务化、闭环自优化的智能数据体系、以驱动应用创建，适用于多种常见的企业系统集成场景。

1.2 产品架构

图1-1 产品架构

UniCloud 集成平台平台定义为提供消息集成、服务集成、数据集成的统一集成平台，各组件既能够单独运行也能组合成套件，各集成共享相同的技术底座，用户可根据需要自由配置部署。

- 数据集成

  数据集成是一个以调度、监控和管理 ETL 过程为核心功能的应用系统。该系统通过图形化工具，快速灵活地设计与部署，实现数据抽取、转换及加载，并能在设计中设置统一的清洗规则，从而提升数据的质量，能为企业和组织提供一套完备的数据集成解决方案。

- 消息集成

消息集成旨在为集成平台提供可靠、无状态、满足各应用间信息最终一致性的消息集成服务。支持原生的 Kafka 特性，具备原生 Kafka 所有消息处理特性；支持安全的消息传输，通过 sasl 认证、消息存储加密等措施加强网络访问控制；支持消息数据高可靠，支持消息持久化、多副本存储机制。

- 服务集成

  服务集成主要用于数据库表字段的开放、第三方接口的代理转发、文件资源的开放等。支持 API 注册、API 测试、API 部署、API 授权、API 编辑、API 删除、API 版本管理等全生命周期管理。可以对 API 的访问进行统计分析，记录访问日志，实现对 API 访问的审计功能。

2 访问数字平台的集成服务

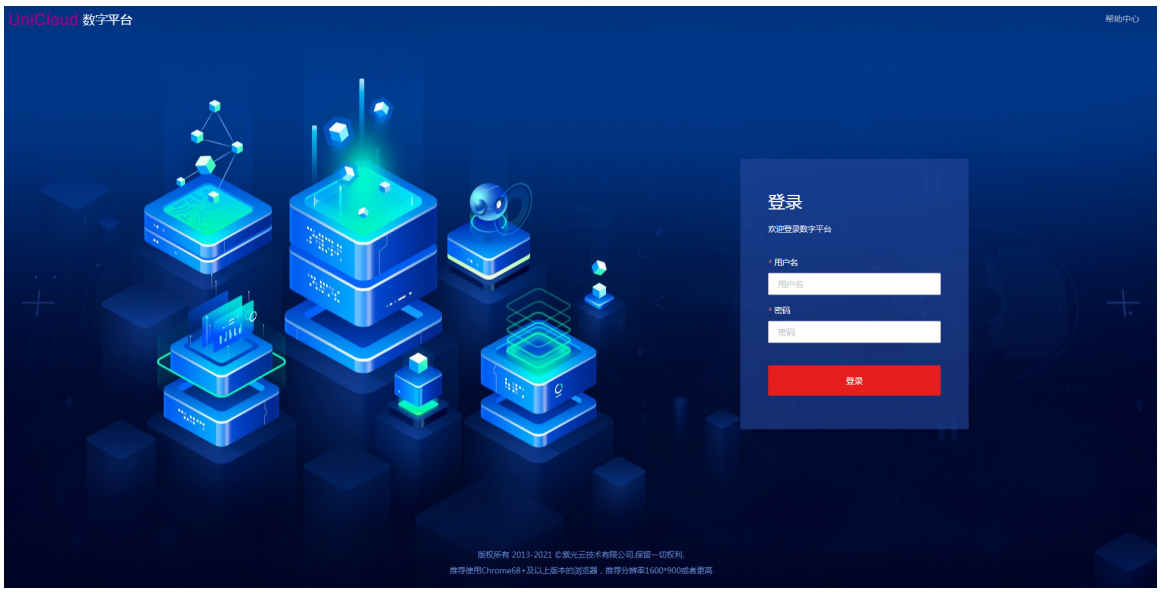
注意

为保证系统安全，登录系统后请及时修改登录密码。

数字平台的系统服务安装成功以后，数字平台的 URL 地址会在安装脚本成功执行完成后显示，格式为：https://VIP:32015。

在浏览器中输入地址：https://VIP:32015，进入登录页面，如图 2-1 所示。输入正确的用户名和密码，单击<登录>按钮即可登录数字平台。如果用户名或密码不正确，系统会弹出相应的错误提示。

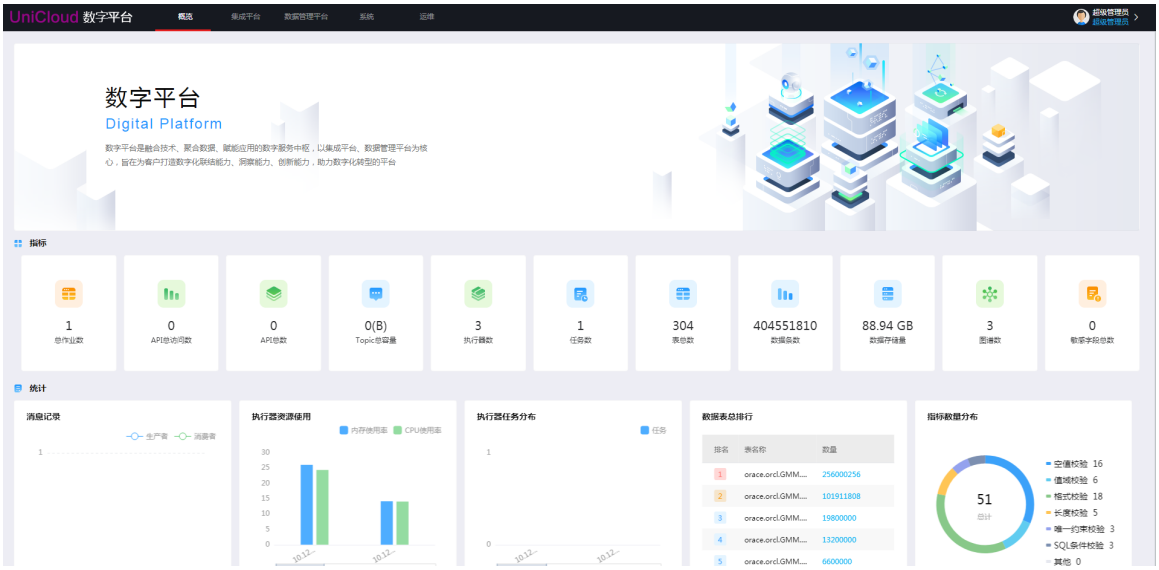
图2-1 登录页面



2.1 首页

首页展示了数字平台的统计信息，首页如图 2-2 所示。

图2-2 首页



2.2 退出登录

登录成功后，在右上角登录的当前用户下拉菜单中选择[退出]菜单项，即可退出系统。

3 功能介绍

说明

- 集成平台中包含联机帮助，单击页面左上角的帮助按钮，弹出窗口中提供各功能详细的配置说明、操作指导以及注意事项等，可帮助用户更好的使用集成平台。
- 本章节仅对各功能进行概括说明，以便用户快速了解数据集成平台提供的主要功能，关于各功能的详细说明请参见集成平台的联机帮助。

3.1 工作空间管理

工作空间是为了让用户更好的将项目相关资源进行统一管理，如数据源、数据集成作业、服务集成 API、消息集成 Topic 等。比如用户可以新建工作空间，然后将同一项目的相关资源创建在一个工作空间下，方便后续查看及操作。组织内的用户创建的工作空间属于该组织。该组织下的用户可以查看该组织下所有的工作空间并进行操作。工作空间提供导入导出，以及查看导入导出记录等功能。

3.2 标签管理

标签管理模块是对全局的标签进行管理，数据源、数据集成、服务集成、消息集成、数据运营等全局可以使用，用户可通过标签对相关资源进行分类。标签管理提供标签的新增、编辑、删除、查询、导入导出，以及查看导入导出记录等功能。

3.3 数据源管理

不同使用场景下需要连接不同的数据源，在数据源管理页面可新增数据源并管理系统中所有已添加的数据源。集成平台支持丰富的数据源类型，包括：DB2、达梦、Greenplum、HBase、MPP、MySQL、Oracle、PostgreSQL 等。

3.4 函数管理

函数管理页面是对组织下用户在当前工作空间下所有内置函数和自定义函数进行管理。函数管理页面管理的函数可用于数据集成各类组件、服务集成的 API 发布、消息集成的消息解析等。目前内置函数包含常用的字符串、日期、加解密、数学运算函数。支持在线编写 Java 代码开发自定义函数。

3.5 数据集成

数据集成是一个以设计、部署、调度、监控和管理 ETL 过程为核心功能的应用系统。

操作者借助该平台可以通过流程图式的图形化工具快速、灵活地设计 ETL 过程，并能方便的进行部署、调度及监控等管理活动，真正地提供一体化数据集成开发环境。

功能特性包括：

- 数据集成作业全生命周期管理，对作业进行多维度调度、监控，实现作业自动化处理。
- 根据已存在作业生成作业模板，通过作业模板快速生成作业，并自动部署作业，使大量重复的工作完成自动化。
- 通过标签标记作业，在有大量作业的场景中可根据标签快速定位作业。

3.5.1 主要功能

如表 3-1 所示，数据集成具备的主要功能包括：概览页、资源配置、作业管理、作业监控、整库迁移、定期清理等。

表3-1 数据集成主要功能说明

功能		说明
概览		概览页用于对作业和任务的执行状态进行监控统计，向用户展示系统中任务和作业的整体执行情况。包括作业/任务数量、状态及任务类型统计等
资源配置	执行器管理	执行器是执行任务的实际容器，用于执行服务端发送的作业中的任务
	调度器监控	调度器用于接收服务端下发的作业，并将作业中的环节调度到具体的执行器中运行。调度器根据执行器打分自动分配最优执行器运行作业。调度器列表显示了当前已注册的调度器
	资源收藏	展示同组织内用户收藏的任务组件资源。用户在新建数据集成任务时，可将常用组件添加到收藏夹中，方便组织内的用户使用
作业管理	作业定义	作业定义页面是对系统中的作业进行统一的管理，包括作业集新建/编辑、删除，作业列表展示，作业运行，作业上线，复制作业，删除作业、共享/批量共享、定时配置、版本管理、作业导入导出等。作业由一个或多个任务环节组成，作业分为作业和实时作业。 作业目前支持的任务类型包括： • 普通 ETL 任务可对源数据进行抽取和转换，然后将结果加载至目标数据源。数据集成支持丰富的数据源类型和多种数据转换操作，包含的组件详情请参见 3.5.2 ETL 任务组件简介 • Sqoop 任务支持丰富的数据源类型和 5 种任务类型（数据库到 HDFS、数据库到 HBase、数据库到 Hive、HDFS 到数据库、Hive 到数据库） • Shell 任务支持以 Shell 脚本形式直接运行任务 • SQL 任务支持 SQL 脚本形式直接运行任务 • Flume 任务支持在指定主机上运行 flume-ng agent 命令执行 flume 任务 • KETTLE TRANS 任务支持连接 Kettle carte 客户端，直接运用 Kettle 里任务保存后得到的.ktr 文件 • KETTLE JOB 任务连接 Kettle carte 客户端，直接运用 Kettle 里作业保存后得到的.kjb 文件 • CDC 任务基于数据库日志解析，完成数据库数据实时同步，包含的组件详

功能		说明
		情请参见表 3-6 • DataX 任务支持连接 DataX 客户端 • 子作业支持将已配置好的作业作为一个子作业插入到当前作业中 • New ETL 是对原 ETL 中的一些组件进行整合，在保证功能完全覆盖的基础上、提供更易用、更方便、更简洁的操作，并对于整合的组件进行功能的增强。包含的组件详情请参见表 3-7 实时作业采用流式处理方式，用于数据库中数据的迁移，推荐应用于需要捕获及同步实时变化数据的场景。系统中的作业支持CDC作业类型和MQ作业类型，包含的组件详情请参见表3-8 • CDC：CDC 类型实时作业基于数据库日志解析，完成数据库数据实时同步 • MQ：MQ 类型实时作业支持实时获取消息队列中的数据，可配置转换、过滤组件对数据进行加工处理，然后将处理后的数据写入到目标端
	作业模板	通过作业模板可直接快速需要的作业。作业模板模块可新增作业模板并管理系统中所有已创建的作业模板；作业模板暂不支持KETTLE TRANS、KETTLE JOB、JOB三类任务配置成模板 该功能适用于有大量重复作业，且这些作业的差异集中在部分参数设置值不同的场景中
作业监控	作业实例	作业实例页面是对系统中作业实例的运行情况进行展示，并可对作业实例执行重跑、失败恢复、停止等操作
	任务实例	任务实例页面是对系统中任务实例的运行情况进行展示，并可查看任务实例的详细信息及运行日志
整库迁移		整库迁移模块可直接创建作业，将源数据库中选定的某些表批量复制到目标数据库，节省批量创建任务的时间
定期清理		配置数据定期清理策略，可防止系统数据库因存在过多的冗余数据而导致系统变慢

3.5.2 ETL 任务组件简介

针对每个普通 ETL 组件，拖拽至任务设计面板上双击打开弹窗，单击页面左上角的帮助按钮，可查看各组件的操作步骤、配置说明及使用示例，帮助用户更好的使用组件。

表3-2 普通 ETL 任务数据抽取组件

数据抽取组件	描述
表抽取	表抽取组件用来利用已配置好的数据库连接和SQL语句，从数据库中读取数据

数据抽取组件	描述
文件抽取	通过文件抽取组件，可以读取单个或多个文本文件、指定读取的文件列表或者用正则表达式表示的目录列表。该组件可支持抽取本地文件或目录、FTP文件或目录、SFTP文件或目录、HTTP文件
生成记录	生成记录组件可根据配置输出指定数量的记录行，缺省为空。可选包括一定数量的静态字段
Kafka抽取	Kafka抽取组件可以根据配置从Kafka消息系统中抽取数据
Kafka流抽取	Kafka流抽取组件从Kafka抽取流数据，并运行子转换，该转换根据消息批量大小或持续时间执行，可近乎实时地处理连续的数据流
HDFS抽取	HDFS抽取组件可对HDFS文件系统中结构化的数据进行抽取
HBase抽取	HBase抽取组件可以将存储在HBase表中的数据抽取到其他类型数据库中
REST抽取	REST抽取组件用来从REST服务中抽取数据
Excel抽取	Excel抽取可以将存储在Excel表中的数据抽取到其他类型数据库中
WebService抽取	WebService组件用来从WebService服务中抽取数据
MongoDB抽取	MongoDB抽取组件可以将存储在MongoDB中的数据抽取到其他类型数据库中
Redis抽取	Redis抽取组件可以将数据从Redis数据库抽取出来，目前支持的数据类型包括String、Set、List、Hash、Hashall及ZSet等。该组件不能作为转换任务的开始步骤，需要在其前面有其他输入组件进行值传递
JSON抽取	JSON抽取组件用于从JSON输入源中抽取数据
XML抽取	XML抽取组件使用XPath规范将XML文件的数据解析到流中，支持本地/FTP的XML文件（*.xml）和以其他方式读取的XML格式数据的解析
XML输入流	XML输入流组件使用StAX解析器从XML源中读取数据，与XML抽取组件使用的DOM解析器相比，该组件适用于XML文档结构复杂和数据量大的场景
JMS抽取	JMS抽取组件可以从Apache ActiveMQ JMS服务器或IBM MQ中间件消费流数据
MQTTConsumer	MQTTConsumer组件可以从MQTT代理或客户端抽取流数据。MQTTConsumer步骤运行子转换，该转换根据消息批量大小或持续时间执行，可近乎实时地处理连续的数据流
CSV文件抽取	CSV文件抽取组件提供从定界文件中读取数据的功能
HttpServer	支持根据用户自定义的配置实现http服务器的实时部署，通过请求发布的http服务器接口可以实现用户数据主动推送到数据集成侧，接收到的数据可以通过其它组件进行数据处理
ORC文件抽取	ORC文件格式是Hive库的一种文件存储格式，利用ORC文件抽取组件可以对存储在服务器本地和HDFS文件系统中的ORC格式的结构化数据进行抽取

表3-3 普通 ETL 任务数据转换组件

数据转换组件	描述
字符串操作	字符串操作组件可以对输入字段类型为String的字段进行补位、改变大小写、转义等操作
字符串切割	该组件用于切割字符串的一部分。如果指定字段的起始位置超出范围，则返回空白
字段选择	字段选择组件用来选择字段、重命名字段、指定字段的长度或精度
正则表达式	正则表达式组件可以将输入字段的字符串值与正则表达式定义的文本模式匹配，将匹配结果存入新的字段中可以为后续连接的组件使用
对称加密	该组件可以对流中数据进行加密和解密，支持的对称加密算法有：AES/DES/DESede三种。建议配合生成密钥组件一起使用
JS代码	JS代码组件允许通过JavaScript语言对数据做复杂的运算。右侧JavaScript函数列表包含了该组件支持的JavaScript的函数
去重记录	去重记录组件从输入流中移除重复的记录，可以输入字段名称直接去重
排序记录	排序记录组件可以利用指定的字段对行按照升序或降序进行排序。当行数超过5000行的时候，将使用临时文件来排序行
增加常量	增加常量组件用于添加常量到流中，用字符串形式指定名称、类型和值。利用选择的数据类型指定转换格式
字符串替换	字符串替换组件允许指定的字符串替换输入流中的原指定字符串，并生成新的输出字段
计算器	计算器组件提供一个功能列表，可以在字段值上运行，操作更加便捷，且运算效率比JavaScript脚本更高
设置字段值	设置字段值组件用于将流中某一字段的值赋值给另一字段
增加序列	增加序列组件可以在流中新增一列，这一列为在某个起始值和增量的基础上的序列。可以生成两种序列，一种为自定义的序列，可以自己设置起始值，增长值及最大值，但每次转换运行的时候序列的值又会重新循环一次；另一种为使用数据库的序列
公式	公式组件可以实现复杂的逻辑判断，或者使用该组件自带的函数进行数据的转换清洗
过滤记录	过滤记录组件允许根据条件和比较符来过滤记录。这个步骤一旦连接到先前的步骤中，就可以通过简单的单击“<field>”、“=”和“<value>”区域来构建条件。该步骤支持2层条件嵌套，如果有复杂判断条件，需要配合JS代码组件一起使用
列拆分为多行	列拆分为多行组件可将输入数据行集中的某个列按照条件拆分为多行，这种条件可以是简单的一个分隔符，也可以指定正则表达式
拆分字段	该组件可根据分隔符信息拆分字段，拆分后的字段将形成新的列
行转列	行转列组件用于将数据某一列中的唯一值转换为输出中的多个列来旋转该数据，即将列值转换为列名，并可以在输出的时候对最后输出中的任何其余列值进行聚合

数据转换组件	描述
列转行	该组件可以将输入中的列转换为行
数据检验	数据检验组件通常用于确保传入数据的质量。通过定义一些简单的检验规则来确保传入的数据符合指定的数据规范，例如：数据的范围、长度和类型等，该组件可以同时定义多种检验规则
行比较	行比较组件用于比较两个不同来源的数据，这两个来源的数据分别为旧数据源和新数据源，该步骤将旧数据和新数据按照指定的关键字匹配、比较后进行合并，该组件主要用于数据表的同步更新
分组	分组组件允许通过定义字段实现分组后再执行计算，即可以按照某一个或某几个字段进行分组，同时将其余字段按照某种规则进行计算。例如：计算产品的平均销售额，获取某商品库存的数量等
数值范围	该组件比较一个字段的数字范围，在这个范围的话就输出一个新的列，并且打印出来你定好的结果。如果不在指定的范围则打印出来unknown，或者打印自定义的结果
流连接	流连接组件可以将多个前面步骤的数据以INNER/FULL OUTER的方式合并数据
Java代码	Java代码组件允许使用Java语言编写一个step插件
流查询	流查询组件允许使用来自其他组件的信息查找数据。来自查询组件的数据首先被读取到内存中，然后根据与查询关键字所匹配的字段从查询组件中查找数据
记录集连接	记录集连接组件用于将2个输入步骤的数据执行合并数据的操作
值映射	该组件用于将字符串值从一个值映射到另一个值
数据库查询	该组件可以使用设置的关键字在目标表中查询，并从查询结果中返回指定的字段
添加XML	该组件可以生成XML格式数据
SwitchCase	作为流程处理的转换组件，该组件可以通过设置case值对上一步传过来的数据流进行分流

表3-4 普通 ETL 任务数据加载组件

数据加载组件	描述
加载至表	加载至表组件可以将数据加载到数据库表中
加载至Excel表	加载至Excel表组件可以将数据加载到Excel表中
加载至文件	加载至文本文件组件用于将数据加载到文本文件中，并可以通过配置路径信息加载至远程文件中
丢弃数据	丢弃数据组件，表示不对数据做任何操作，常用于在调试ETL任务中测试和丢弃数据
加载至ES	加载至ES组件可以将数据写入到Elasticsearch中

数据加载组件	描述
加载至HDFS	加载至HDFS组件可以将数据加载至HDFS集群上的文本文件中
加载至HBase	加载至HBase组件可以将数据加载至HBase数据库中
加载至Kafka	加载至Kafka组件可以将数据发布至Kafka消息系统中
Kafka流加载	Kafka流加载组件允许以近乎实时的方式将消息发布到Kafka服务器
数据删除	数据删除组件可以利用查询关键字在表中搜索行。如果能被找到，行就会被删除
数据库更新	数据库更新组件可以利用查询关键字在表中搜索行。如果字段能被找到，并且没有任何改变，字段不更新；如果字段有改变，字段就会被更新
加载至Redis	加载至Redis组件可以将数据加载到Redis数据库中，目前支持的数据类型包括String、Set、List、Hash及Sorted Set等。该组件常和“字段选择”组件一起使用，即可以在“字段选择”组件中将需要的流字段按顺序先选择出来，然后再给该组件使用
GPLoad	GPLoad组件可以使用Greenplum的外部表并行加载功能进行大规模并行加载数据
加载至XML	加载至XML组件用于加载数据到XML文件中，可以根据配置路径加载至远程或本地文件中
加载至JSON	加载至JSON组件可以将数据生成JSON块，并选择输出到文件，或者生成包含JSON块的数据流
加载至JMS	加载至JMS（Java Messaging Service）组件几乎可以实时地将消息发布到Apache ActiveMQ JMS服务器或IBM MQ中间件
MQTTProducer	MQTTProducer组件允许以近乎实时的方式将消息发布到MQTT代理
插入更新	插入更新组件可以利用查询关键字在表中搜索行。如果找不到，则插入该行。如果可以找到，并且要更新的字段相同，则不执行任何操作；如果不完全相同，则更新表中的行
Oracle批量加载	Oracle批量加载组件可以将源数据以适当的加载格式写入，然后调用Oracle SQL*Loader将其传输到指定的表
MySQL批量加载	该组件可以将数据批量加载到MySQL数据库中，批量插入的原理是利用了MySQL一个高效导入方法load data infile将文本文件中的数据导入表中，文本文件是由系统后台根据前序步骤的数据流生成的FIFO文件，因为用户数据的不可控性，故后台自动生成的FIFO文件的相关配置需要用户自行配置
Vertica批量加载	Vertica批量加载组件通过JDBC使用STDIN语句将数据复制到Vertica数据库中
加载至MongoDB	加载至MongoDB组件可以将其它地方的数据，加载到MongoDB数据库中
PostgreSQL批量加载	PostgreSQL批量加载组件可以批量加载数据到PostgreSQL数据库中
数据同步	该组件可以与“行比较”组件结合使用。“行比较”组件使用标志字段来保存比较结果，该组件使用此标志字段对数据库表进行更新、插入及删除

数据加载组件	描述
加载至ORC文件	ORC文件格式是Hive库的一种文件存储格式，利用加载至ORC文件组件可以将需要存储到Hive库的数据进行ORC格式结构化保存

表3-5 普通 ETL 任务其他组件

其他组件	描述
存储过程	存储过程组件可以运行一个数据库存储过程，并获取返回结果
执行SQL	执行SQL组件用于对目标数据库执行SQL脚本，可以在转换初始化的时候执行。执行方式分为执行每一行、作为一个语句执行以及变量替换三种
发送邮件	发送邮件组件用来给指定邮箱发送E-Mail
检查文件存在	检查文件存在组件可以在本地系统或FTP服务器中检验是否存在某个文件，并将布尔标志字段添加到输出字段
FTP下载	FTP下载组件可以从FTP服务器上获取一个或者多个文件
HTTP下载	HTTP下载组件可以通过HTTP协议从Web服务器上获取一个文件
文件解压	文件解压组件可以解压文件，但只能解压zip格式的本地文件
表结构复制	表结构复制组件可以获取源数据库中的表结构，并在目标库中创建相同结构的表
从流中获取记录	从流中获取记录组件用于返回另一个任务生成的记录，您可以输入该任务所期望的字段元数据。该组件需作为任务的开始步骤，所在任务可以作为JMS抽取、MQTTConsumer抽取以及Kafka流抽取组件的子转换使用
生成密钥	该组件用于生成一定数量的密钥，生成的密钥可以在对称加密组件中使用
设置变量	设置变量组件可以在作业的维度内设置一个变量值，作业内的其他组件如果使用了“变量替换”功能，那么该组件内的所有变量都会在运行时被替换为变量组件中设置的同名变量值

表3-6 CDC 任务组件

CDC 组件	描述
MySQL CDC	捕获MySQL数据变更的Binlog信息，解析出相应数据并发送到Kafka Topic
Oracle CDC	使用LogMiner解析Oracle归档日志，获取数据库的数据变化信息，解析出相应数据并发送到Kafka Topic
JDBC Producer	从Kafka中获取CDC日志，解析后在目标数据库中进行相应的增删改操作

表3-7 NEW ETL 任务组件

New ETL 组件	描述
表抽取	表抽取组件用来利用已配置好的数据库连接和SQL语句，从数据库中读取数据
文件抽取	可以读取单个或多个文本文件、支持正则表达式过滤目录下的文件，支持（ftp、hdfs、sftp、http等）多种文件服务器和数据格式（cvs、json、execl、orc等）
函数	通过集成函数管理功能的函数，提供字段处理和转换的能力，更加灵活和可扩展
Route	可以通过函数表达式制定分发规则
数据库查询	该组件可以使用设置的关键字在目标表中查询，并从查询结果中返回指定的字段
文件加载	支持（ftp、hdfs、sftp、http等）多种文件服务器和数据格式（cvs、json、execl、orc等）
加载至Hbase	HBase抽取组件可以将存储在HBase表中的数据抽取到其他类型数据库中
表加载	加载至表组件可以将数据加载到数据库表中，可以设置模式为插入、更新、删除、批量加载等
Rest抽取	用来从Rest服务中抽取数据
数据解析	数据解析组件支持解析delimited、json、xml等三种格式的数据，用于从源数据中提取目标字段
消息加载	消息加载组件，用于将数据推送至相关MQ（Message Queue）服务的消息队列中
加载至ES	加载至ES组件可以将数据写入到ElasticSearch中
序列化	序列化是将无序数据转化为有序数据，例如将表字段数据转换为JSON数据

表3-8 实时作业组件

CDC 组件	描述
源CDC	源CDC组件支持Oracle、MySQL、MySQL8、SQL Server（navite）、PostgreSQL、Generic JDBC类型数据源全量无缝转增量的数据抽取，您还可以指定偏移量，指定时间进行增量抽取
JDBC加载	JDBC加载组件支持将源CDC抽取出来的数据加载至Postgre SQL、Vertica、MySQL、Oracle、ClickHouse、DB2、HANA、达梦、GBase、Kingbase8、MPP、Teradata、Greenplum、PostGIS、SeaSQL MPP、DRDS数仓，还可以进行打标签，指定并行度等操作
转换组件	转换组件可对源CDC组件传入的数据做函数处理（例如concat、split、replace等）、数据标签、字段过滤等，然后传入加载组件
写入Kafka	写入Kafka是将CDC源抽取的数据，按照配置的表和Topic的映射关系写入Kafka中
HUDI加载	HUDI加载是将CDC源抽取的数据按配置的表名映射关系，将数据按照HUDI格式加载至

CDC 组件	描述
	HDFS文件系统中
消息抽取	消息抽取组件，用于抽取相关消息中间件服务的消息队列的数据

3.6 服务集成

服务集成主要用于数据库表字段的开放、第三方接口的代理转发、文件资源的开放等。支持 API 注册、API 测试、API 部署、API 授权、API 编辑、API 删除、API 版本管理等全生命周期管理。可以对 API 的访问进行统计分析，记录访问日志，实现对 API 访问的审计功能。

表3-9 服务集成功能说明

功能		说明
概览		概览页面可以实时展示当前用户所属组织中开放 API 的今日访问次数、今日访问成功率、部署 API 的数量、授权 API 的数量以及激活工作空间的个数、组织内 API 开放统计、工作空间内 API 开放统计、API 类型开放统计以及 API 来源开放统计
API工厂	API 管理	API管理功能主要用于API注册、继承注册、API导入、API编辑、API测试、API部署、API撤销部署、API上线、API下线、API修改记录查询以及API删除等 • 数据 API 设计：数据 API 通过向导式的设计将数据库表里字段开放出去，以标准 resetful 接口的形式对外提供 • 通用 API 设计：通用 API 设计用来对已经存在的第三方接口进行代理 • 函数 API 设计：函数 API 可以在系统内通过编写 JavaScript 脚本，设计并生成标准 resetful 接口 • 继承注册：继承已经创建好的 API，快速生成一个新的 API • 模板注册：通过模板快速导入通用 API 和函数 API • API 测试：API 注册后可以通过在线测试查看接口是否可用 • API 部署：API 测试通过后可以部署到网关节点，进行授权访问 • API 上线：API 部署后可以申请上线，管理员审批后可以展示在资产市场供他人订阅 • API 版本管理：用来查看 API 的修改记录，并可以通过载入功能查看对应版本 API 的详情信息，并支持在该历史版本的基础上进行修改 • API 导入/导出：用于导出所选 API 信息，支持通过导出功能导出的 API 信息导入系统 • API 健康检查：API 注册时选择是否开启健康检查，开启后，当 API 部署到 API 网关后，可以在网关实时看到 API 健康状态
	服务编排	服务编排页面主要是对现有一些业务接口按照特定的业务执行流进行组织和编排，生成一个统一的对外访问接口，该接口实现了多个业务接口的功能

功能		说明
	认证模板	通过在认证模板页面添加认证模板，注册通用 API 时关联认证模板，解决带认证接口的访问权限问题，自动帮用户维护接口访问所需认证授权信息，使用户更能专注于自己的业务
	环境配置	环境配置功能主要用于函数 API 设计时工作空间秘钥和环境变量的获取
API网关	API列表	API 列表下展示了指定工作空间下已部署的的 API 信息。并可以根据实际需要将 API 授权/取消授权给指定的工作空间，并可对 API 的访问配置访问规则和熔断规则
	授权的API	授权的 API 页面下展示了当前工作空间下被授权的 API 信息。并可以根据实际需要对 API 进行查看详情、测试及取消授权等操作
	监控统计	监控统计页面对应用访问 API 的日志信息进行多维度分析展示，让用户对 API 的使用情况一目了然，包括 API 调用趋势分析和根据客户端 IP、响应状态、响应时间进行分类统计等
	日志搜索	日志搜索页面可以根据搜索条件来查看用户所在组织下的指定的 API 访问详情日志信息
	访问网络	访问网络页面定义访问网络分类，用于对转发节点进行分类标记。访问网络页面可维护平台部署的网络信息，支持多个网络的维护
	节点管理	节点管理页面用于配置可用的跨网转发节点，作为网关来代理发布接口。维护部署的转发节点信息，支持不同网络的转发节点维护，可通过不同网络的转发节点实现接口的跨网发布
	规则管理	规则管理分为 IP 规则和超限规则，IP 规则可以限制能访问 API 的 IP 地址或 IP 网段，超限规则可以限制每秒、每分钟、每小时或者每天的访问次数，支持 IP 黑白名单
	熔断管理	熔断管理分为信号量隔离模式和线程隔离模式，可以设置熔断开启的条件，在网关访问第三方接口发生错误时，出发熔断，避免系统雪崩
文件管理	文件发布	文件发布页面展示系统中共享的文件列表，并可对文件进行共享、编辑、删除及授权操作。文件可以申请上线到资产市场，管理审批通过后可以在资产市场展示供他人订阅。文件发布目前支持将本地文件、HDFS 文件数据源、ONEStor 文件数据源的文件发布共享
	文件资源	文件资源页面以列表形式展示当前用户所属的文件资源
	存储配额	存储配额管理页面用于对组织内用户的文件空间存储大小进行配置管理
订阅管理		订阅管理页面展示了当前组织下发布的服务列表以及订阅信息，可以进行规则配置和熔断配置，可以终止订阅

3.7 消息集成

消息集成定义为集成平台中各集成服务之间提供可靠的、可持久化的、高吞吐量的准实时消息管道系统。消息集成使用统一的消息接入机制，标准化的消息通道，具有如下几方面的优势：

- 支持原生的 Kafka 特性：具备原生 Kafka 所有消息处理特性。
- 支持安全的消息传输：通过 sasl 认证，消息存储加密等措施加强网络访问控制。
- 支持消息数据高可靠：支持消息持久化，多副本存储机制。支持节点级扩容与 Topic 重分配。

表3-10 消息集成功能说明

功能		说明
概览		概览页面主要展示了Topic总容量、历史记录，消费者，生产者等情况的统计信息
Topic管理	Topic列表	Topic列表页面展示Topic的相关信息，用户可以对Topic进行新建、编辑、删除、授权等操作
	Topic配置	Topic配置页面主要提供对Topic属性配置进行新增、删除、查看配置操作。当用户对已创建的Topic有属性配置的诉求时，可以使用该模块进行配置
	重分配	Topic重分配功能用于Topic修改分区所在broker节点位置
消息查询		消息查询主要提供毫秒级、可视化的Kafka集群中的消息查询能力，支持按照分区和生产时间进行过滤
消息转发		主要用于将一个Topic中的数据经过一定规则过滤后写入另外一个Topic。用户可在消息转发页面下配置新的消息转发规则，并对这些消息转发规则进行管理
消费进度		显示组消费进度和活跃Topic的情况
监控	Broker监控	显示整个系统中Brokers性能度量情况
	Kafka监控	显示Kafka运行情况，可根据自己需要选择指标进行展示
	Zookeeper监控	显示Zookeeper监控信息，包括发包数量、收包数量、活跃连接数、排队请求数在各时间节点的数据分布情况
集群信息		集群信息页面，展示了如下内容： • 基本信息：展示了 Kafka 集群用于与代理通信的协议、用户客户端连接的 SASL 机制、SASL 连接的 JAAS 登录配置以及用于建立与 Kafka 集群的初始连接的主机/端口对的列表 • Kafka 集群信息：Kafka 集群节点的具体信息，包括主机 IP、端口、内存、CPU 使用信息等 • Zookeeper 信息：各 Zookeeper 节点的主机 IP、端口、状态等信息

3.8 资产市场

表3-11 资产市场主要特性列表

功能		说明
资产中心		资产中心从不同维度对资产进行了展示，包括数据目录、热门来源、热门专区、推荐资产及最新发布资产。用户可根据需求从相应模块查看自己需要的资产，也可在资产中心搜索框中输入关键字进行资产检索
资产配置	目录管理	目录管理页面左侧以树结构展示系统中的数据目录，页面右侧展示选定数据目录下的子级数据目录。目录管理提供了目录的展示、查询、新增、修改和删除操作
	订阅管理	订阅管理页面展示了系统中所有的订阅信息，系统管理员可以在该页面对用户的订阅进行查看和取消订阅。取消订阅后，相关用户不能再使用对应资源
	标签管理	标签是资产的标识，为资产添加标签，可以方便用户识别和管理拥有的资产。标签管理支持用户根据需要自定义新增标签内容，并可对标签进行编辑、删除等操作 标签管理包括标签组和实体标签两种：标签组是具有某种共同特征的标签分类（比如资产类型），子标签是一种特定属性标签（API类型、MESSAGE类型）
	资产来源	资产来源功能用于对系统内的所有资产来源进行管理，可对资产来源信息进行新建、编辑或删除等操作，也可查看来源下的资产信息。
	资产管理	当资产由组织管理员通过来源进行上架后，会展示在资产管理页面，系统管理员可以在资产管理页面对资产进行统一管理，包括对资产的查询、下架、修改和恢复操作
	专区管理	专区是以应用场景为维度对资产进行展示，比如视频、经济、民生等专区。专区管理是对专区的增删改查及展示顺序进行管理，也可以对专区内的资产进行移入移除操作
	流控规则	流控规则页面管理规则模板，供API订阅审批的时候进行选择。规则的作用是限制API的调用方对API的调用。可以设置访问的IP白名单、黑名单以及超限规则（指定时间内的访问次数）
	评价管理	系统管理者在评价管理页面可以查看所有用户对所有资产的评价。可对评价进行搜索、查看和删除操作

3.9 系统

表3-12 系统管理主要特性列表

功能		说明
组织管理		用于对系统内的所有组织进行管理，如新建、编辑或删除组织等。系统管理员可根据用户需求为根组织划分子组织，便于对用户进行分类管理
用户管理		用户即可以登录本系统并使用系统中功能和资源的人。用户管理提供了统一管理系统中用户的功能。主要包括系统未对接认证通和对接认证通两种情况下的配置
角色管理		用于管理本系统中用户的角色。不同角色具有不同的系统资源使用权力。本系统通过角色实现分组式的权限管理，每种角色都有自己的权限集，被赋予该角色的用户即具有权限集所包含的权利。系统内置了基本角色，也支持创建自定义角色
操作日志		是以用户为行为主体，从而产生的行为日志，包含时间、操作用户的名称、产生操作行为的IP地址、操作对象、级别、操作结果等主要信息，方便后续的审计
系统配置	菜单管理	用于配置数字平台中各功能的布局及是否显示等，方便超级管理员对界面功能展示进行配置
	流程配置	系统管理员通过配置流程模板可以有效规范普通用户对资源的申请和使用，即系统提供了审批流程自定义功能，用户可以根据自身企业的情况自定义审批流程，通过在流程中添加多个审批人员实现多级审批的需求
	安全设置	对系统中的安全策略进行管理，包括密码策略、登录认证策略、SSO认证等
	基础设置	基础设置提供了系统消息的配置功能和关联BDP 大数据平台的配置、显示配置、操作日志失效配置功能
	大数据集群配置	大数据集群配置用于配置数字平台所使用的的BDP 大数据集群相关信息
	大数据集群资源	大数据集群资源用于管理系统中各组织的集群资源，包括指定Kerberos用户，可使用的YARN队列和Spark Oozie Launcher队列
	全景监控	全景监控提供了系统中数据的全方位展示，包括整体统计展示，分功能模块展示等
	多级部署配置	多级部署用于在分级部署多套系统的场景（如系统分县、市、省多个层级分别部署）中，将多个系统（含本系统）的数据信息汇集至一个系统中通过全景监控进行展示（如省级系统可以汇总展示下属市级、县级系统的数据）
软件授权		系统完成安装部署后，默认未完成License注册的系统为试用版本，用户可以使用试用授权来对资源进行申请。试用期到期后，用户需申请正式授权才能继续使用本系统

功能	说明
大数据环境管理	大数据环境管理功能为系统提供了统一的大数据集群管理功能，方便管理员管理对接的大数据集群并向各组织或工作空间分配大数据集群资源

3.10 运维

运维提供了数字平台的运行维护功能，包括服务的管理、资源监控及系统巡检等功能。

表3-13 系统功能特性

功能	说明
概览	运维模块默认展示概览页面，该页面展示了系统运维相关的监控统计信息，主要包括系统的节点信息、系统基础组件信息、以及系统中各项服务的信息
安装部署	安装部署功能用于可视化安装、卸载、扩容、升级、回退数字平台各产品服务，如集成平台或数据管理平台中的各服务
服务管理	服务管理用于可视化部署数字平台各服务，如集成平台或数据管理平台中的各服务功能等，并对各服务进行统一管理
告警管理	告警管理是对服务告警信息的统一管理，包括告警列表、告警阈值管理、告警配置及通知管理 • 告警列表页面展示了当前告警和历史告警的信息 • 告警阈值管理页面可对系统中各指标项的告警进行开启及关闭，并能配置各指标项的告警阈值。当指标项的告警开启后，并且使用率超过阈值，系统会发出告警，告警信息会展示在告警列表中 • 告警配置页面可配置告警联系人，当达到告警级别后，系统会发送告警消息给相关联系人 • 通知管理页面展示了针对联系人（用户）的通知配置。对于需要接收系统通知的用户，管理员可以在该处配置其联系方式
资源监控	资源监控提供了管理部署服务的主机和监控主机节点等资源功能，通过图和表直观地呈现了资源状况，方便用户对资源进行监控和管理 • 监控详情可以对部署服务的主机节点进行监控，包括主机节点信息、MySQL 信息和服务信息，为用户提供主机当前的运行状态，帮助用户更合理地使用主机资源 • 主机管理用于对部署服务的主机进行监控管理，用户可根据需要进行新增
防火墙管理	防火墙管理页面展示了系统的防火墙状态，可执行开启或关闭防火墙的操作，也可以添加或移除白名单，开放或者关闭相关服务端口，同时支持检查防火墙状态及一键同步等功能
系统巡检	系统维护支持一键巡检、定时巡检、查看报告、下载报告的功能
系统备份	系统提供了对系统数据进行备份，并管理备份的功能，支持一键备份和定时备份

3.11 个人中心

表3-14 个人中心主要特性列表

功能	说明
个人信息	提供了当前登录用户修改个人信息和密码的功能
我的申请	展示了当前用户提交的申请流程，用户可查看申请流程详情、审批进度，并可根据实际需要撤销流程
待办审批	为用户提供了审批流程管理功能。我的审批页面展示了当前登录用户待办的审批流程，用户可对这些流程进行审批（同意/驳回）、更改责任人、删除等操作
所有申请	所有申请页面展示了当前登录用户相关的所有的申请流程，包括审批中和已完成的申请流程
资产订阅	资产订阅页面展示了当前用户所在组织下的应用信息及资产订阅信息，包含应用、资产列表和资产使用总览三部分。 • 应用管理功能用于对当前用户所在组织下的所有应用进行管理，如新建、编辑、查看秘钥和删除应用信息等 • 订阅资产列表下展示了当前登录用户所在组织下申请订阅的资产信息 • 资产使用总览展示了当前登录用户所在组织下的资产订阅使用情况统计总览
我的收藏	我的收藏中展示了当前登录用户收藏的资产信息，可进行查看和取消收藏操作
我的资产	我的资产下展示了当前登录用户负责的资产列表，并可对相应资产的纠错和客户评价进行处理。 • 资产列表：资产列表页面提供了对当前用户负责的资产进行统一管理的功能，包括资产的上架、下架、恢复、修改等。资产列表页面中包括已上架、待上架、已下架及不可用页签，分类展示了不同状态的资产。 • 纠错回复：纠错回复页面提供了当前用户负责的所有资产的纠错信息，用户可对纠错信息执行拒绝、接收、删除、查看详情等操作。 • 客户评价：客户评价页面提供了当前用户负责的所有资产的评价信息，用户可对评价信息进行查看
我的评价	页面中展示了当前用户的已评价订阅的列表
我的纠错	我的纠错页面展示用户提交的资产纠错信息

4 数据集成典型配置案例

4.1 增量抽取场景-时间戳

目前很多企业都会构建自己的数据仓库，通过整合所有业务系统数据形成企业的数据资产，并对其进行深度分析挖掘数据背后的价值，以支持企业做出一些重要的决策。一般情况下，企业会采用增量数据抽取方式完成业务系统数据向数据仓库中的同步。为了实现数据的增量抽取，在设计数据表时新增一个时间戳字段，数据在入库时，会为时间戳字段赋值为入库时间。

1. 场景描述

A 交通运输管理公司的业务系统数据库中，维护一张车辆通过卡口信息记录表 vv_1000。现公司需要将该表中每天新增的数据定时向数据仓库中的 vv_1000000 表做数据同步。公司业务系统使用 PostgreSQL 数据库，数据仓库使用 SeaSQL MPP。

表4-1 源表（vv_1000）结构

KKWZBM	HPHM	HPZL	TGSJ	HBSJ	SSSD	RYLX	RKSJ
3116250005	莆P3Z907	2	2020/3/10 15:30:58	26	68	01	2020/3/10 20:10:47

2. 场景分析

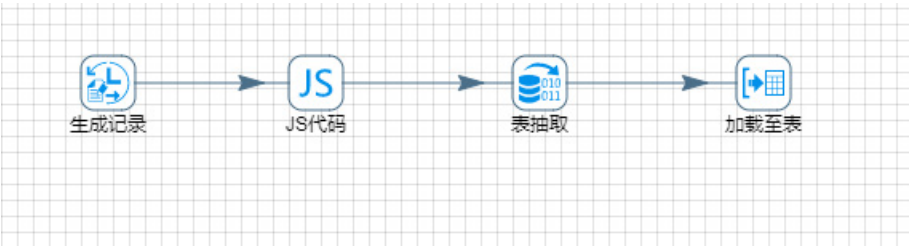
企业设计增量抽取的表比较多，可以先创建一个增量抽取的作业模板，然后通过作业模板批量部署 ETL 作业，最后配置作业定时调度实现增量的抽取。

3. ETL 设计方案

数据流向：生成记录 –> JS 代码 –> 表抽取 -> 加载至表

ETL 方案：数据增量是基于当前调度时间，要求每天凌晨 2 点将前一天的数据同步到目标表。比如当前调度时间为 2020/3/10 02:00:00，则增量抽取的数据是 2020/3/9 00:00:00 到 2020/3/9 23:59:59 这段时间入库的数据，这个时间范围可以使用 JS 代码组件依据当前调度时间计算得出，然后将起始时间作为参数传入表抽取步骤，将源表中时间戳（RKSJ 字段值）在指定时间范围内的数据传给“加载至表”步骤。

图4-1 ETL 任务设计图示



4. 示例前置条件

PostgreSQL 数据库中 vv_1000 表已创建完成。

SeaSQL MPP 数据库中 vv_1000000 表已创建完成。

5. 示例详细步骤

创建作业模板

作业模板功能适用于有大量重复作业，且这些作业的差异集中在部分参数设置值不同的场景中。用户可通过填写构成作业模板的参数自动生成对应的作业，并且可以通过批量部署功能批量部署作业。

(1) 新建作业模板

进入[作业管理/作业模板]页面，单击<新建>按钮，弹出新建作业模板窗口。配置完作业名称及描述后，单击<确定>按钮，进入作业模板设计页面。

图4-2 新建作业模板

(2) 配置作业模板参数

- 作业模板设计画布左侧为模板替换参数区域，参数类型分为普通参数、数据源连接、HBase 连接、HDFS 连接、Redis 连接、Kafka 连接和 ES 连接。单击<增加参数>按钮可新增参数，参数名称请根据实际参数作用自定义取名，参数替换值不可更改（请直接将替换值复制至任务信息对应的参数处）。
- 作业模板设计画布右侧为作业设计页面，用户可将各类型任务添加到作业设计画布中进行作业设计。

表4-2 作业模板设计

(3) 作业设计时可将任务中的配置项替换为模板参数，后续模板在部署时，填写模板参数即可完成任务中参数的配置。比如页面左侧增加了替换参数“{{para1}}”，将{{para1}}直接配置在任务节点配置参数中，后续模板部署时，用户配置的{{para1}}的值会直接替换到作业中。

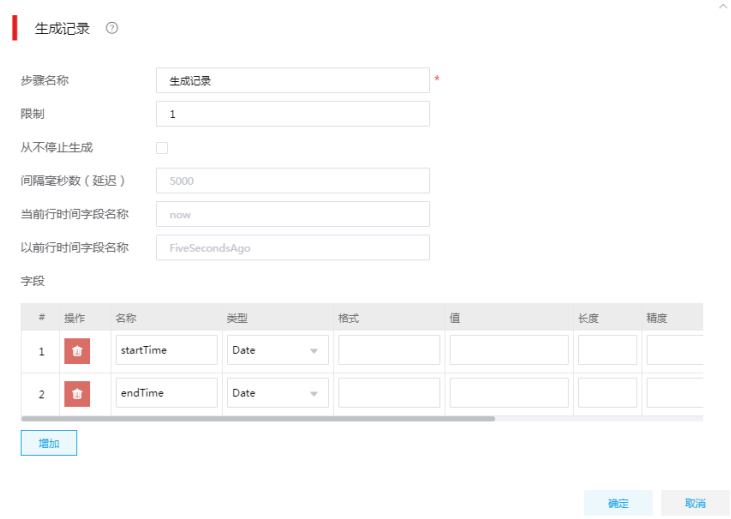
(4) 双击 ETL 任务组件，进入 ETL 任务设计画布。在左侧填写参数区域增加参数，并分别双击画布上组件将组件内容替换为参数的替换值，参数及替换内如图 4-3 所示。

图4-3 作业模板编辑

a. 生成记录

该步骤为了配合 JS 代码组件，生成基于当前调度时间的抽取时间范围，无需替换参数。

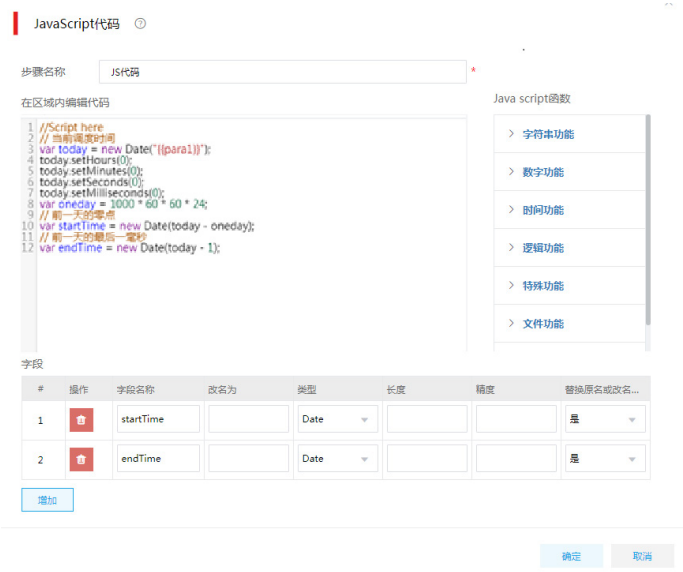
图4-4 生成记录具体配置图示



b. JS 代码

JS 代码步骤根据当前调度时间，取得一个入库的时间段，可以将“调度时间”配置为参数“{{para1}}”。

图4-5 JS 代码替换参数



c. 表抽取

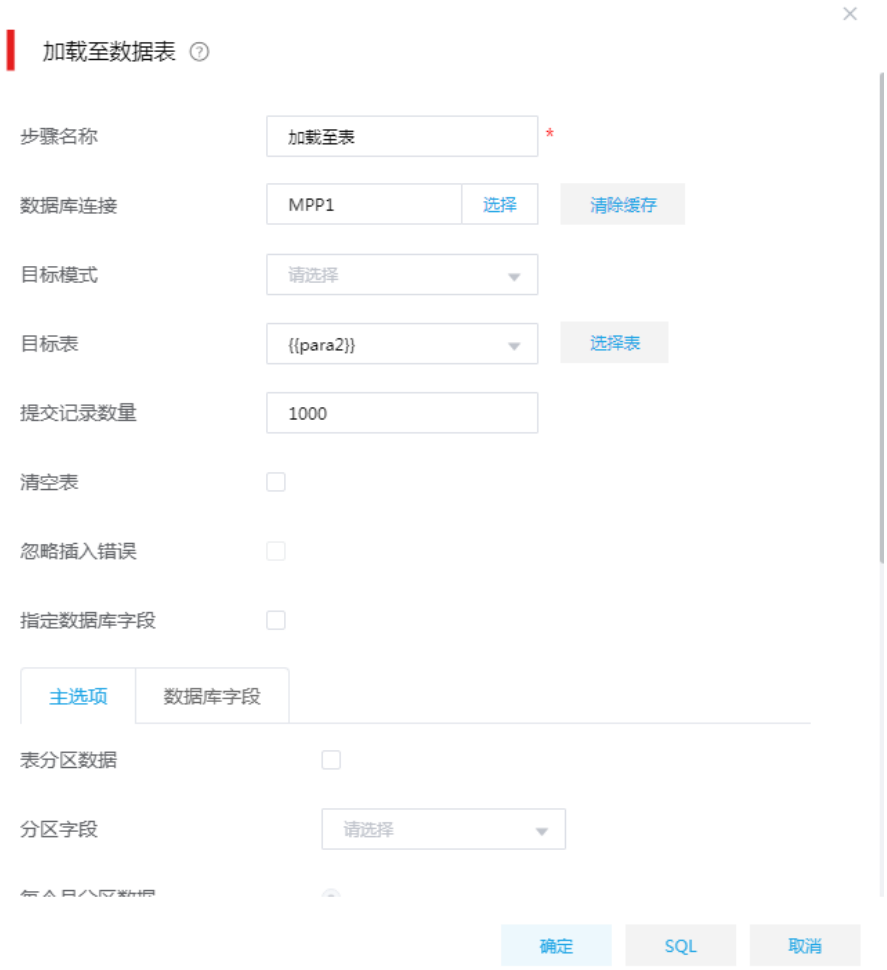
通过数据表抽取步骤，源表为 PostgreSQL 数据库中名为 vv_1000 的表。在“从步骤插入数据”字段选择步骤二 JS 代码，SQL 语句如图 4-6 所示，SQL 中可以自动将数据表抽取步骤获取的变量按照顺序替换给 SQL 脚本中的“?”。可以将、“源表名”、“源表时间戳”字段配置为参数“{{para3}}”、“{{para4}}”。

图4-6 表抽取替换参数

d. 加载至数据库表

将最终的增量数据加载至 SeaSQL MPP 库中 vv_1000000 表中。可以将“目的表”配置为参数“{{para2}}”。

图4-7 加载至表替换参数



(5) 配置完成后，单击右上角<保存>按钮即可保存作业模板。保存作业模板后退出到作业模板页面，可以在作业模板页面查看到新创建的作业。

通过作业模板部署作业

(1) 模板部署作业

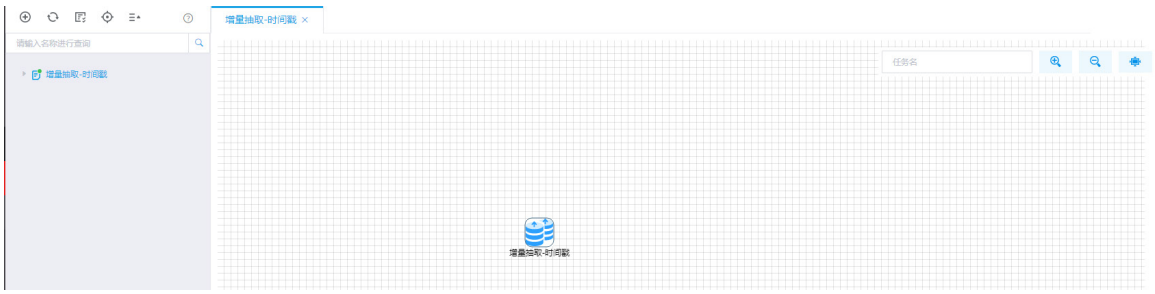
作业模板页面，单击模板列表里模板右侧的<部署>按钮，弹出部署作业弹出框，配置如图 4-8 所示。

图4-8 作业模板创建作业参数

(2) 在部署作业页面配置各作业模板参数，模板参数配置完成后，会在作业部署时替换进作业中。

(3) 部署作业配置完成后，单击<确定>按钮，可以在作业定义页面查看到以该作业模板部署的作业。

图4-9 通过作业模板创建的作业



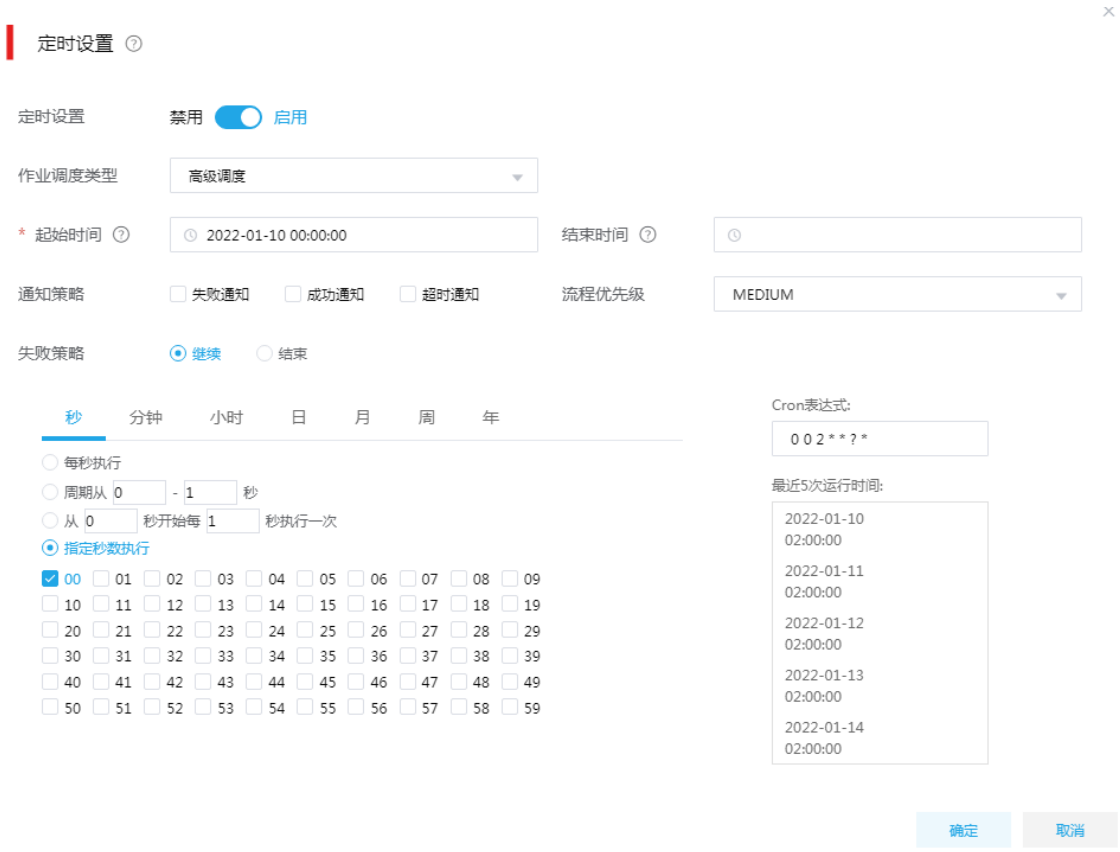
配置作业定时运行策略

[作业定义]页面，鼠标右键单击作业目录中作业名称，在弹出列表中选择设置定时，配置作业的定时运行策略，步骤如下：

(1) 如图 4-10 所示，将作业配置成高级调度，实现每天的凌晨两点执行一次作业。

调度配置方法：对于 Cron 表达式实现定义时间规则为每天凌晨两点执行的具体配置，即指定每一分钟的第 0 秒，每小时的第 0 分钟，指定小时执行（勾选 02 前复选框）。日、月、周、年都按默认配置，即可将 Cron 表达式定义为每天凌晨两点执行。

图4-10 配置作业定时调度策略



(2) 配置完成后，单击<确定>按钮进行保存即可。

4.2 增量抽取场景-自增ID

为了实现数据的增量抽取，除了增加时间戳字段外，还可以为数据表设置自增 ID，当一条数据入库时，无需传入 ID 字段，数据库自动获取当前表中最大 ID，然后将其加 1 作为当前记录的 ID。

1. 场景描述

A 交通运输管理公司的业务系统数据库中，维护一张车辆通过卡口信息记录表 kk_1000。现公司需要将该表中新增的数据向数据仓库中的 kk_1000000 表做数据同步。公司业务系统使用 PostgreSQL 数据库，数据仓库使用 SeaSQL MPP。

表4-3 源表（kk_1000）结构

ID	KKWZBM	HPHM	HPZL	TGSJ	HBSJ	SSSD	RYLX	CRSJ
1	3116250005	莆P3Z907	2	2014/4/8 23:36:58	26	68	01	2016/6/6 10:10:47

2. 场景分析

企业设计增量抽取的表比较多，可以先创建一个增量抽取的作业模板，然后通过作业模板批量部署 ETL 作业，最后配置作业定时调度实现增量的抽取。

3. ETL 设计方案

数据流向：表抽取–> 表抽取 -> 加载至表

ETL 方案：表 4-3 中有一列列名为 ID 的自增序列，判断需要被加载的表 ID 字段的最大数值，作为参数传入第二个数据抽取步骤，与数据库表抽取 2 中被抽取的表 ID 进行对比，数据库表抽取 2 中 ID 比传入参数大的数据传给“加载至表”步骤。

图4-11 ETL 任务设计图示

4. 示例前置条件

PostgreSQL 数据库中 kk_1000 表已创建完成，创建序列 seq_id，设置 ID 字段的默认值为：next('seq_id')

SeaSQL MPP 数据库中 kk_1000000 表已创建完成。

5. 示例详细步骤

创建作业模板

作业模板功能适用于有大量重复作业，且这些作业的差异集中在部分参数设置值不同的场景中。用户可通过填写构成作业模板的参数自动生成对应的作业，并且可以通过批量部署功能批量部署作业。

(1) 新建作业模板

进入[作业管理/作业模板]页面，单击<新建>按钮，弹出新建作业模板窗口。配置完作业名称及描述后，单击<确定>按钮，进入作业模板设计页面。

图4-12 新建作业模板

(2) 配置作业模板参数

- 作业模板设计画布左侧为模板替换参数区域，参数类型分为普通参数、数据源连接、HBase 连接、HDFS 连接、Redis 连接、Kafka 连接和 ES 连接。单击<增加参数>按钮可新增参数，参数名称请根据实际参数作用自定义取名，参数替换值不可更改（请直接将替换值复制至任务信息对应的参数处）。
- 作业模板设计画布右侧为作业设计页面，用户可将各类型任务拖拽到作业设计画布中进行作业设计。

表4-4 作业模板设计

(3) 作业设计时可将任务中的配置项替换为模板参数，后续模板在部署时，填写模板参数即可完成任务中参数的配置。比如页面左侧增加了替换参数“{{para1}}”，将{{para1}}直接配置在任务节点配置参数中，后续模板部署时，用户配置的{{para1}}的值会直接替换到作业中。

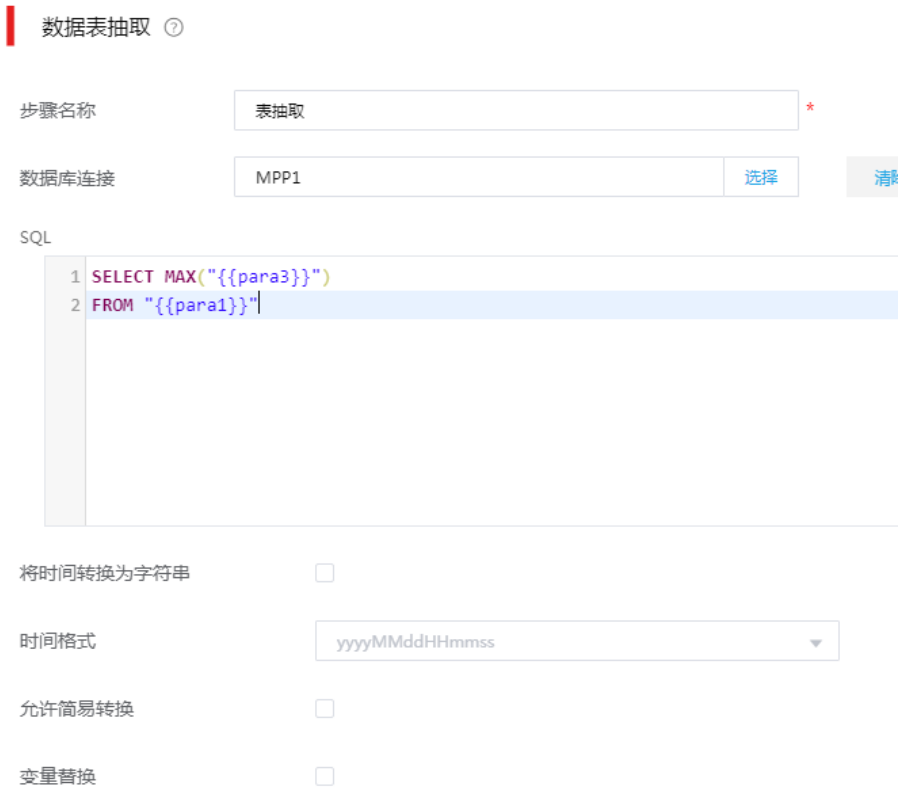
(4) 双击 ETL 任务组件，进入 ETL 任务设计画布。在左侧填写参数区域增加参数，并分别双击画布上组件将组件内容替换为参数的替换值，参数及替换内如图 4-13 所示。

图4-13 作业模板配置

a. 数据表抽取

最终被插入数据的目标表为 kk_1000000。在这一步骤中通过 SQL 语句查询出 kk_1000000 表中的最大 id 作为参数传输至下一环节。其中“目标表”和“目的增量字段”可以替换为模板参数“{{para1}}”和“{{para3}}”。

图4-14 表抽取替换参数



b. 数据表抽取 2

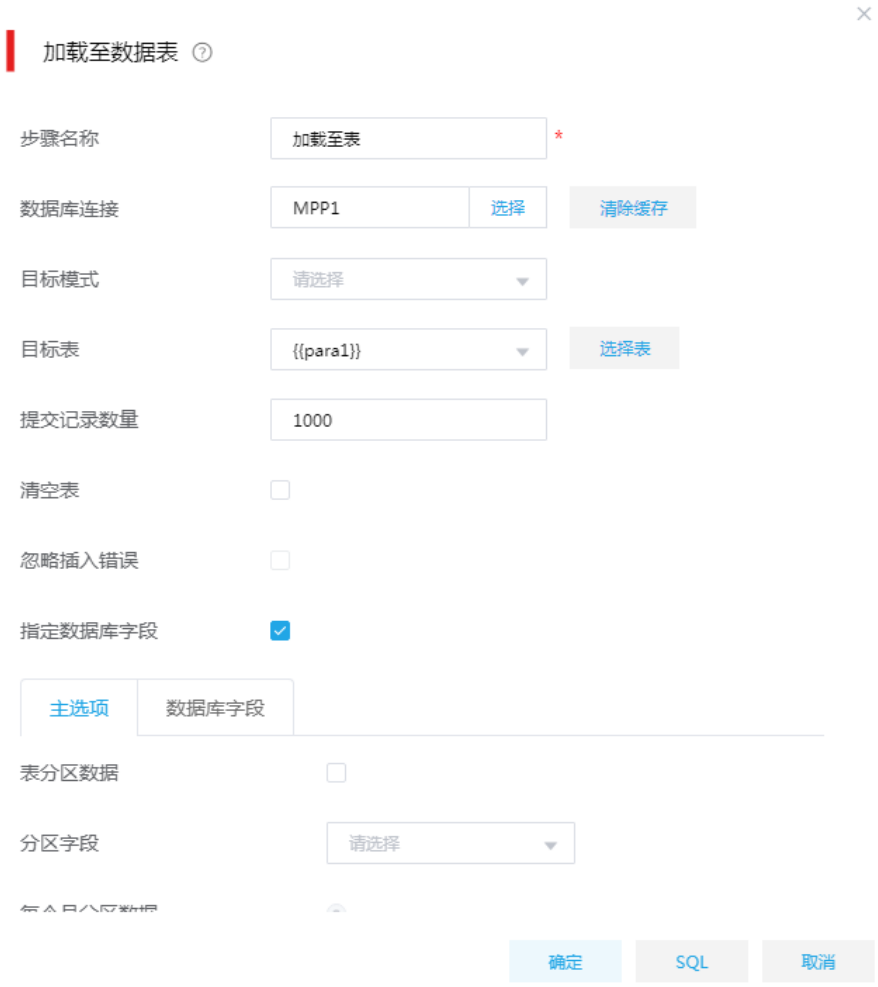
通过数据表抽取步骤，源表为 PostgreSQL 数据库中名为 kk_1000 的表。在“从步骤插入数据”字段选择步骤一数据表抽取，SQL 语句如图 4-15 所示，SQL 中可以自动将数据表抽取步骤获取的变量按照顺序替换给 SQL 脚本中的“?”。其中“源表”和“源表增量字段”可以替换为模板参数“{{para2}}”和“{{para4}}”。

图4-15 表抽取 2 替换参数

c. 加载至数据库表

将最终的增量数据加载至 SeaSQL MPP 库中 vv_1000000 表中。可以将“目的表名”配置为参数。其中“目的表名”可以替换为模板参数“{{para1}}”。

图4-16 加载至表替换参数



(5) 配置完成后，单击右上角<保存>按钮即可保存作业模板。保存作业模板后退出到作业模板页面，可以在作业模板页面查看到新创建的作业。

通过作业模板部署作业

(1) 模板部署作业

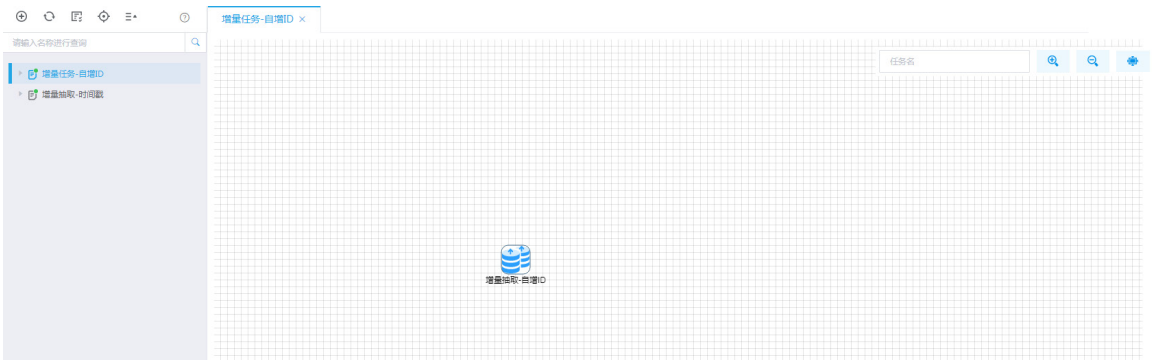
作业模板页面，单击模板列表里模板右侧的<部署>按钮，弹出部署作业弹出框，配置如图 4-17 所示。

图4-17 作业模板创建作业参数

(2) 在部署作业页面配置各作业模板参数，模板参数配置完成后，会在作业部署时替换进作业中。

(3) 部署作业配置完成后，单击<确定>按钮，可以在作业定义页面查看到以该作业模板部署的作业。

图4-18 通过作业模板创建的作业



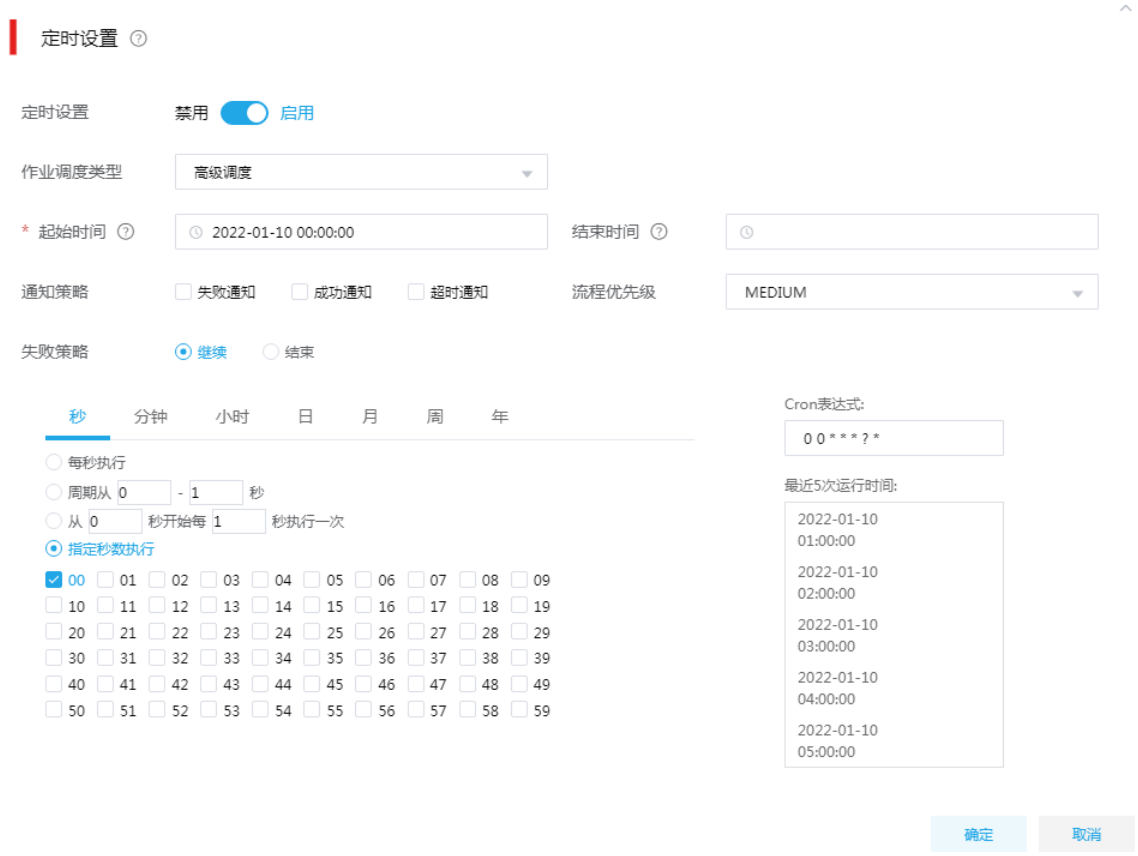
配置作业定时运行策略

[作业定义]页面，鼠标右键单击作业目录中作业名称，在弹出列表中选择设置定时，配置作业的定时运行策略，步骤如下：

(1) 如图 4-19 所示，将作业配置成高级调度，实现每个小时执行一次作业。

调度配置方法：对于 Cron 表达式实现定义时间规则为每个小时执行的具体配置，即指定每一分钟的第 0 秒，每小时的第 0 分钟，每小时执行。日，月，周，年都按默认配置，即可将 Cron 表达式定义为每个小时执行。

图4-19 配置作业定时运行策略



(2) 配置完成后，单击<确定>按钮进行保存即可。

4.3 数据转换场景

半结构与结构化之间的数据转换，支持半结构化（xml/JSON）数据转换为结构化数据，也支持结构化数据转换为半结构化数据，比如：JSON 数据采集并解析写入关系型数据库，或者关系型数据库抽取转为 JSON 格式文件等。

1. 场景描述

A 公司对接客户 REST 接口，数据交换格式为 JSON，A 公司需将对接的数据解析并写入关系型数据库。

- 接口示例：
  - REST：http://10.121.56.134:6688/getinfo/1
  - 请求方式：GET
- 采集 JSON 数据示例：

```
[
  {"sid": "100001", "sname": "fox"},
  {"sid": "100002", "sname": "tom"},
  {"sid": "100003", "sname": "fix"},
```

```
    {"sid": "100004", "sname": "bug"},
    {"sid": "100005", "sname": "dog"}
]
```

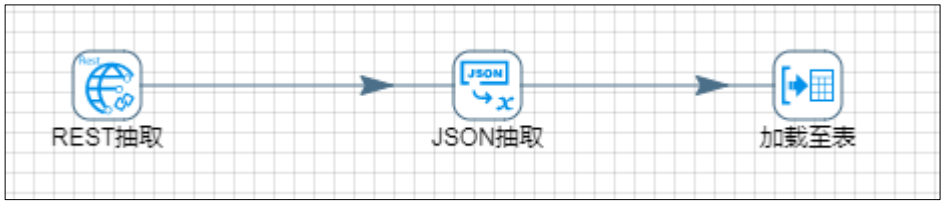
2. 场景分析

数据集成已支持 REST 接口数据采集，可采用 JSON 抽取组件实现数据转换（将半结构化数据转换为结构化数据），并通过加载至表将数据写入数据库。

3. ETL 设计方案

- 数据流向：REST 抽取组件—>JSON 抽取组件—>加载至表组件。ETL 图示如图 4-20。

图4-20 ETL 图示



- ETL 方案：REST 抽取采集 REST 接口数据并输出，JSON 抽取将输入的 JSON 格式数据解析为字段然后输出，加载至表将数据加载至数据库。

4. 示例前置条件

(1) 模拟一个 REST 接口，接口返回数据参考 JSON 数据示例。

(2) MySQL 数据源，名称：mysql-gzh，目标表：test_stu，SQL 脚本：

```
CREATE TABLE `test_stu` (
  `sid` varchar(18) DEFAULT NULL,
  `sname` varchar(50) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

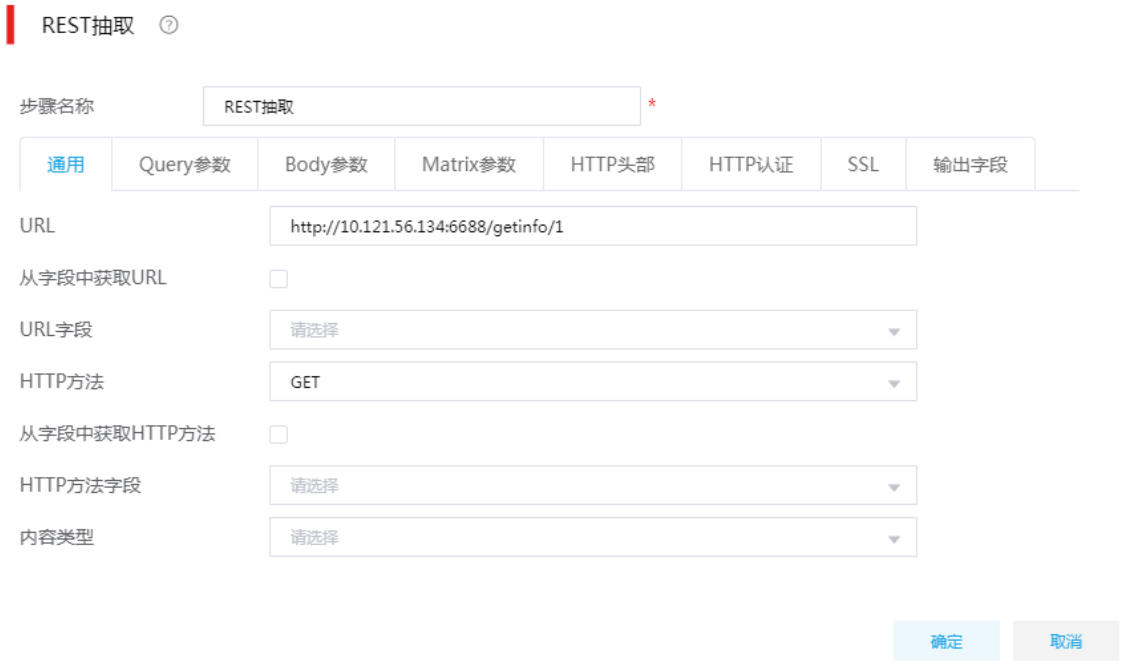
5. 示例详细步骤

(1) 第一步：REST 抽取组件配置

“通用”页配置如图 4-21。

- URL：http://10.121.56.134:6688/getinfo/1
- HTTP 方法：下拉选中 GET

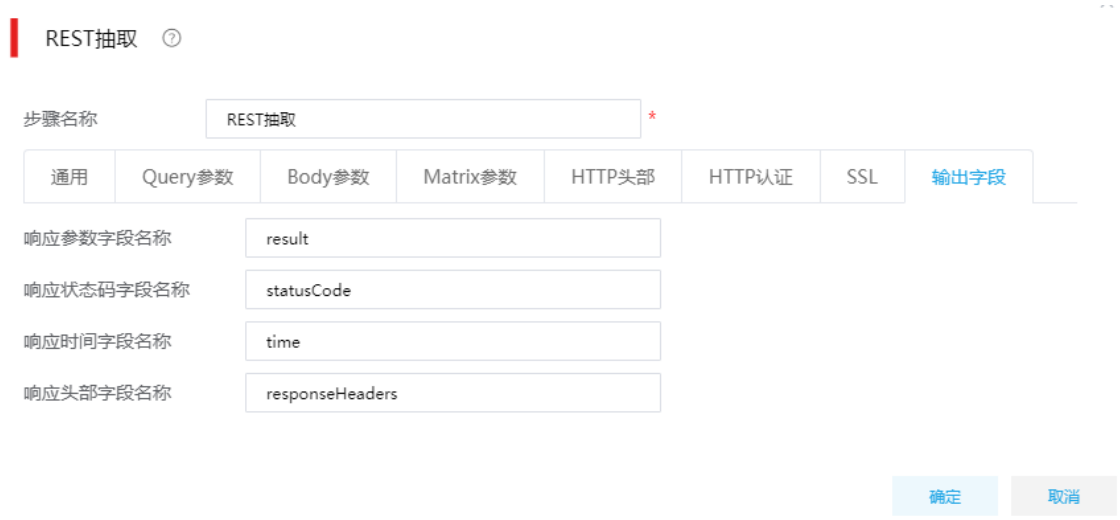
图4-21 “通用页”图示



“输出字段”配置如图 4-22。

- 响应参数字段名称：此字段接收接口返回数据。

图4-22 “输出字段”页图示

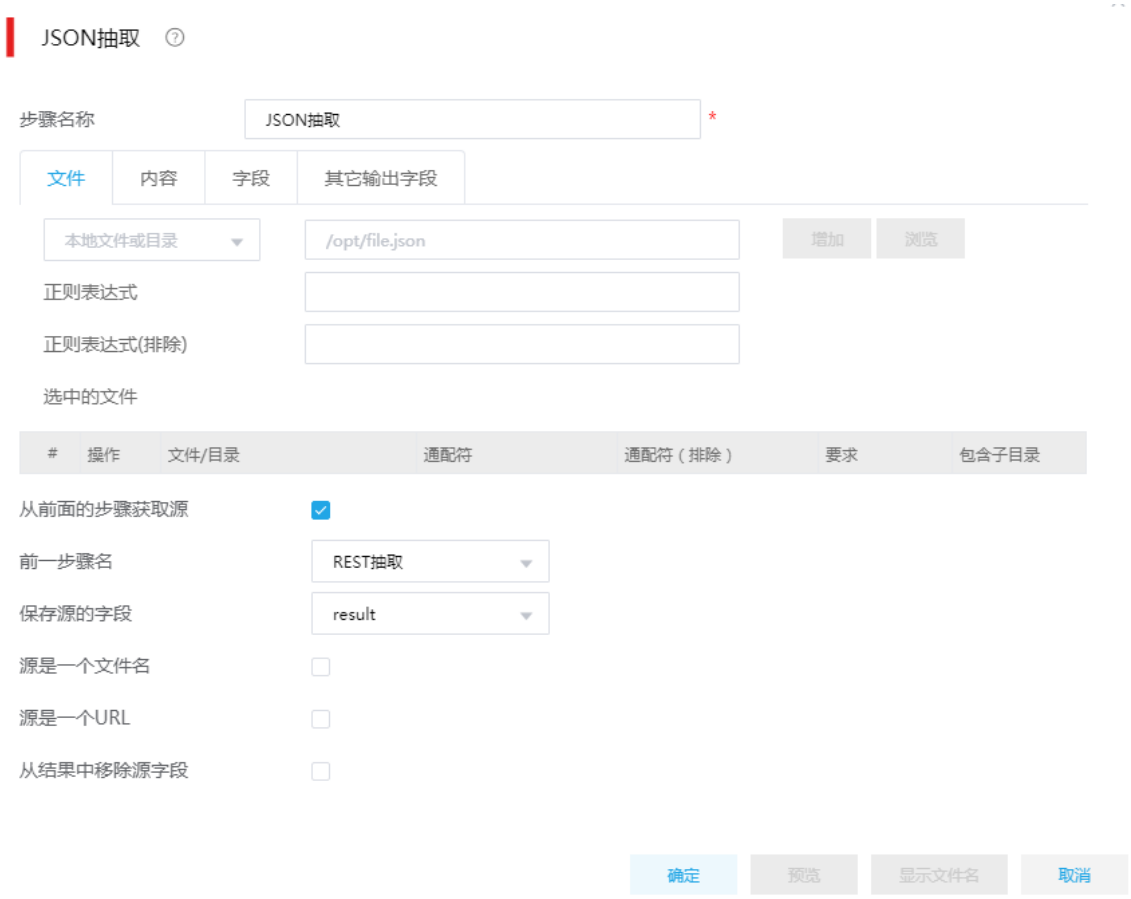


(2) 第二步：JSON 抽取组件配置

“文件”页配置如图 4-23。

- 从前面的步骤获取源：勾选。
- 前一步骤名：下拉选中“REST 抽取”。
- 保存源的字段：下拉选择“result”，注意需要与上一步骤“REST 抽取”输出字段的“响应参数字段名”保持一致。

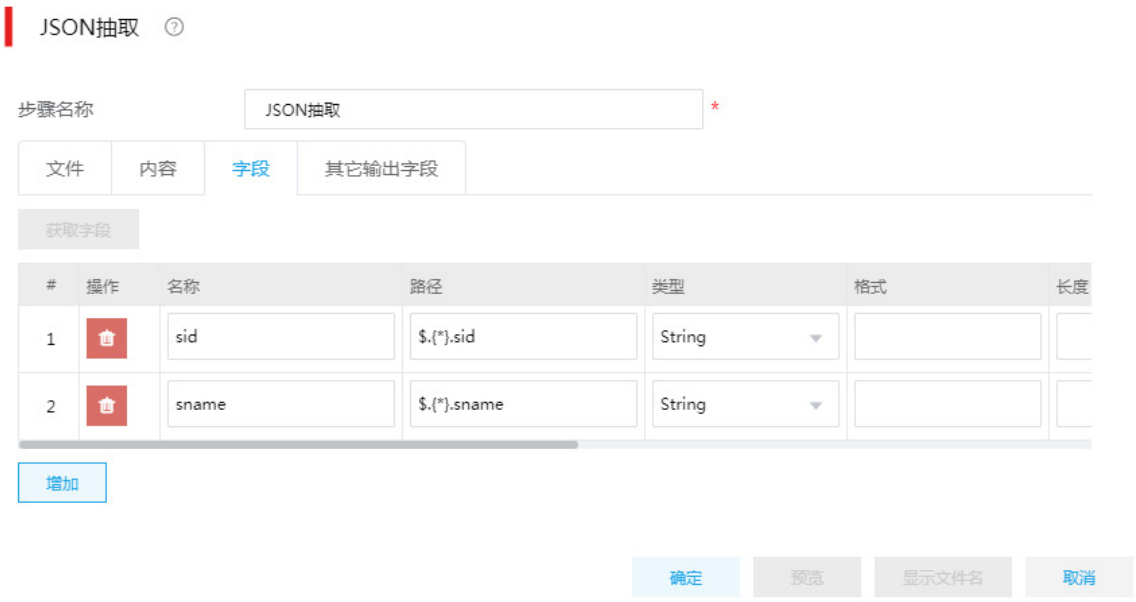
图4-23 “文件”页图示



“字段”页配置如图 4-24。

- 字段名称：配置示例 JSON 的字段名称。
- 路径：参考 JSONPath 填写，填写对应要读取的 JSONPath。
- 类型：下拉选中 String。

图4-24 “字段”页图示



(3) 第三步：加载至表组件配置

- 数据库连接：单击“选择”，然后选中“mysql-gzh”（选择要写入的数据库）。
- 目标表：单击“选择”，然后选中“test_stu”（选择要写入的表）
- 指定数据库字段：勾选。

图4-25 “主选项”图示

“数据库字段”图示：

- 通过获取字段，配置“表字段”和“流字段”的映射关系。“表字段”是 test_stu 表的字段名称，“流字段”是数据流中的字段。

图4-26 数据库字段图示

4.4 实时流场景

与消息中间件通信，支持向 Kafka、ActiveMQ、IBM MQ 等消息队列发送数据，也支持从这些消息队列中抽取数据，比如：订阅 ActiveMQ 中某个主题，实时拉取主题中发布的数据，并经过解析处理后写入关系型数据库。

1. 场景描述

A 物联网平台接入很多传感器设备，这些设备使用 MQTT 协议与物联网平台通信，传递数据为 JSON 文档，主要内容是各设备上报的自身状态信息和各类传感器采集到的环境数据，平台使用 ActiveMQ 作为 MQTT 服务端来接收这些数据。现要求实时抽取消息队列中的数据，经过简单转换处理之后写入关系型数据库。

传感器上报 JSON 数据示例：

```
{
    "deviceid": "100001",
    "devicetype": "2",
    "status": "ok",
    "data": "25,42",
    "data": "2020/03/11 16:00:00"
}
```

2. 场景分析

数据集成目前已经支持从消息队列中实时采集数据，可使用 MQTTConsumer 组件实时抽取数据，通过设置组件的批量参数，实现定时/定量触发子转换。子转换中对 MQTTConsumer 抽取过来的一批数据使用 JSON 抽取组件进行解析，并通过加载至表将数据写入数据库。

3. ETL 设计方案

- 数据流向：MQTTConsumer 组件—>从流中获取记录—>JSON 抽取组件—>加载至表组件。

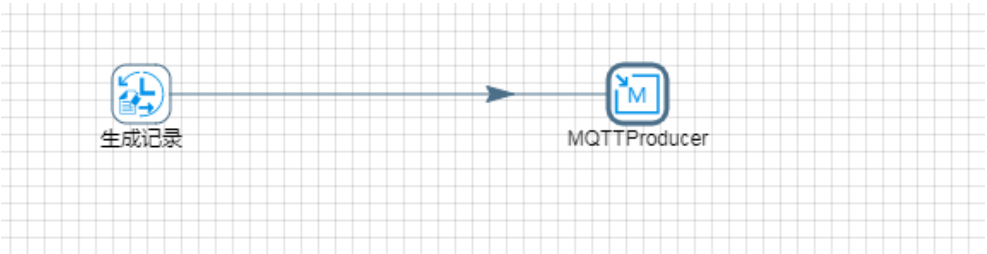
- ETL 方案：一个作业中需要设计两个 ETL 任务，我们分别称之为主任务、子任务。主任务中主要使用 MQTTConsumer 组件，实时采集 ActiveMQ 中某个主题的数据；子任务中负责解析入库。原则是：先创建主任务，然后在主任务的 MQTTConsumer 中配置子任务。
- 运行逻辑：运行主任务时，根据设置的批量参数，周期性触发配置的子任务，同时将一批数据传进子任务中，然后 JSON 抽取组件将输入的 JSON 格式数据解析为字段，最后加载至表将数据加载至数据库。

  注意事项：子任务必须以“从流中获取记录”组件为开始节点，该组件专门用于接收从主任务中 MQTTConsumer 组件传递进来的数据。

4. 示例前置条件

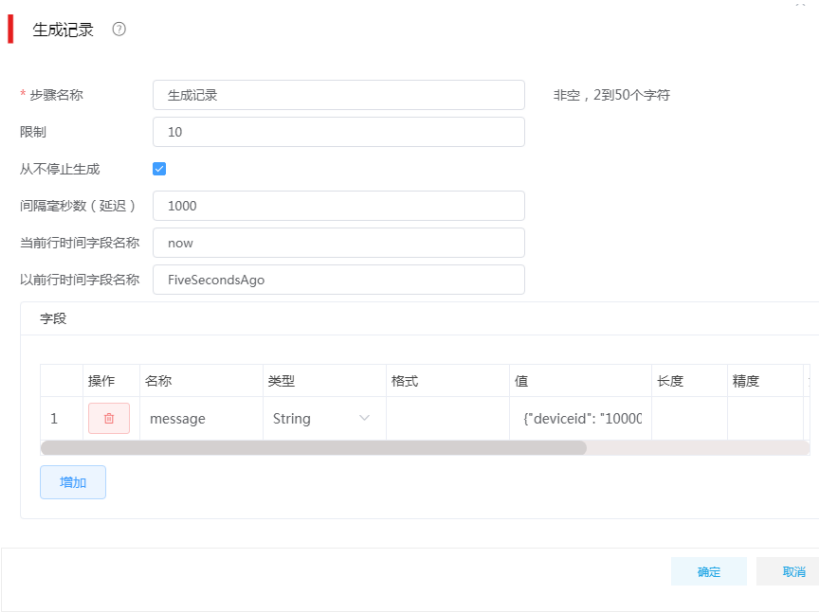
为了便于实时采集任务设计调试，需要模拟传感器上报数据到 ActiveMQ 的过程。

图4-27 “模拟上报”任务设计图示



使用生成记录组件，不停地生成一个 JSON 字符串，JSON 字符串具体内容参见场景描述中的数据示例。

图4-28 生成记录具体配置图示



MQTTProducer 组件使用 MQTT 协议将生成的 JSON 串不断发送到 ActiveMQ 的“device_info”主题中。

图4-29 MQTTProducer 基本设置

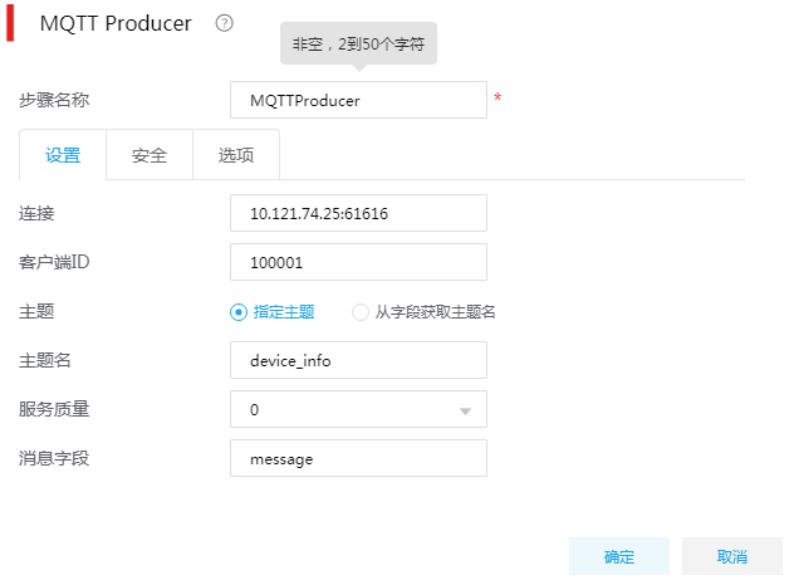
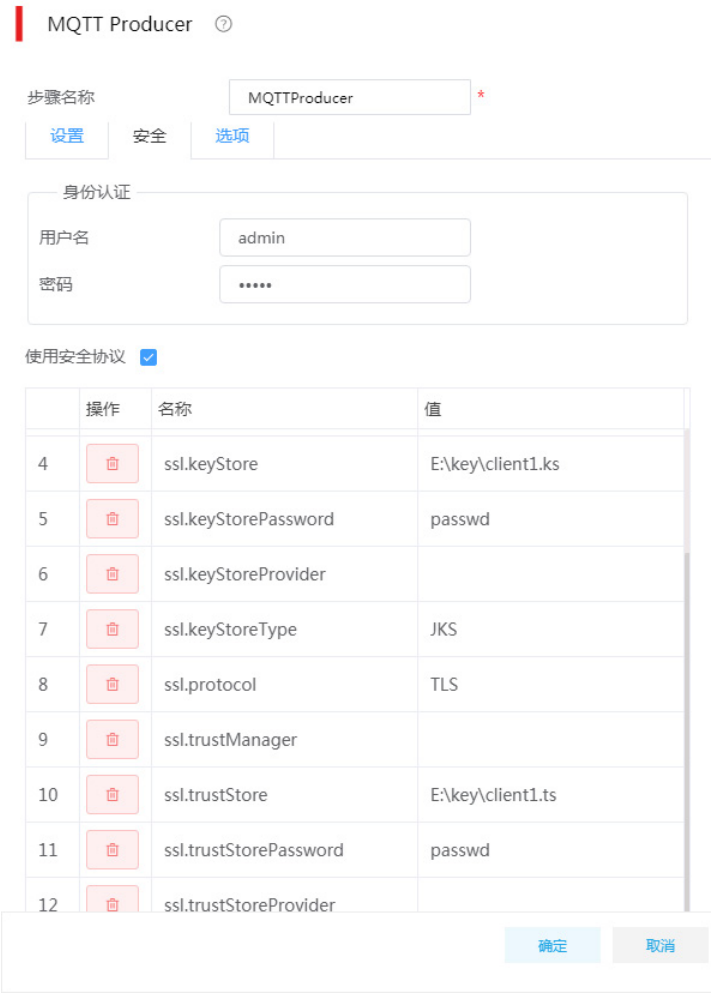


图4-30 MQTTProducer 安全设置



MQTTProducer 的安全配置包含两部分内容：身份认证、SSL 认证，各配置项的值取决于 ActiveMQ 中 broker 的设置。具体而言：

- 若 broker 设置为不允许匿名访问，则需要在身份认证部分填写创建 broker 时设置的用户名、密码。
- 若 broker 开启了 SSL 认证，这里需要勾选“使用安全协议”，然后如实填写图 4-30 所示的几个参数。

完成上述配置工作后，保存并运行“模拟上报”任务。此时，不断发送数据到 device_info 主题中。

5. 示例详细步骤

准备工作完成后。现在按照设计的实时采集方案在作业中来创建任务。

图4-31 主任务 ETL 图示

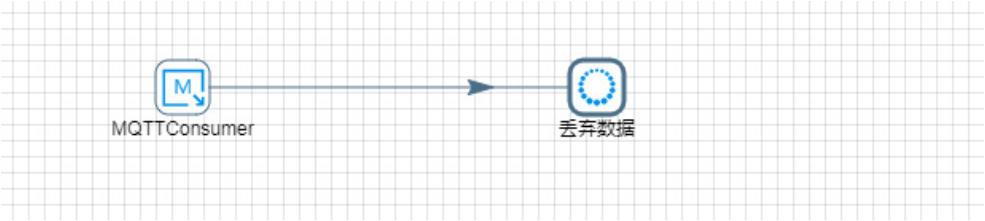
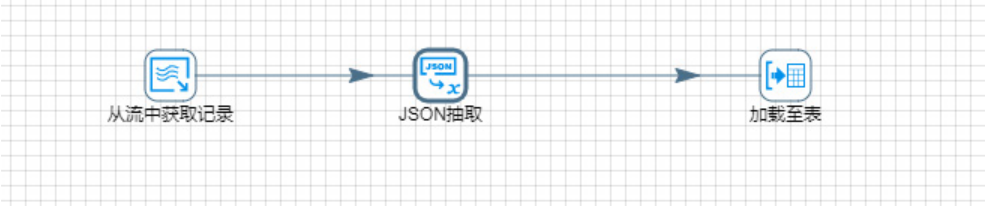


图4-32 子任务 ETL 图示

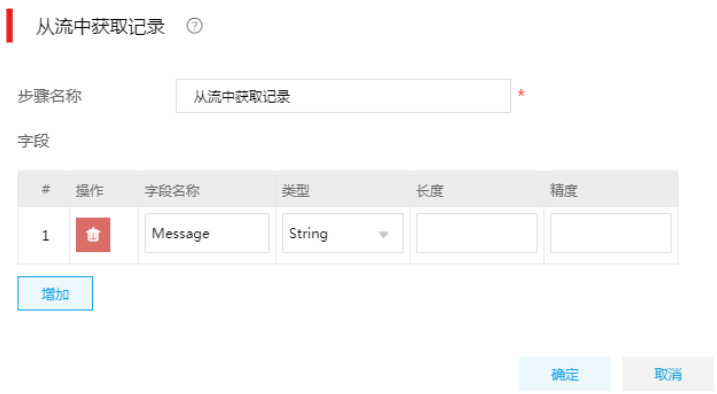


子转换任务中各组件配置说明：

图4-33 MQTTConsumer 基本设置

配置连接地址，添加一个主题。这里的连接地址和主题使用与模拟上报数据任务一样的配置。单击<打开子装换画布>按钮，打开子转换设计画布，设计子转换任务。

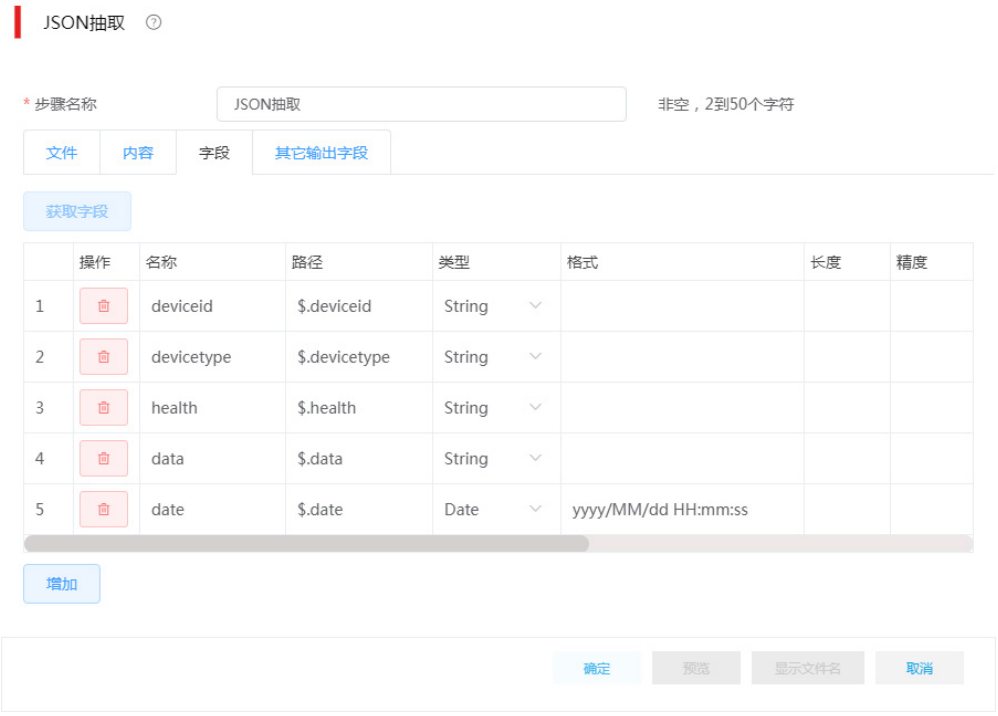
图4-34 从流中获取记录具体配置图示



这里只需要增加一个 Message 字段，该字段名称对应主任务中 MQTTConsumer 组件的输出字段。如果后面步骤需要用到消息所属主题，这里还可以再增加一个 Topic 字段。

图4-35 JSON 抽取配置图示-文件页签

图4-36 JSON 抽取配置图示-字段页签



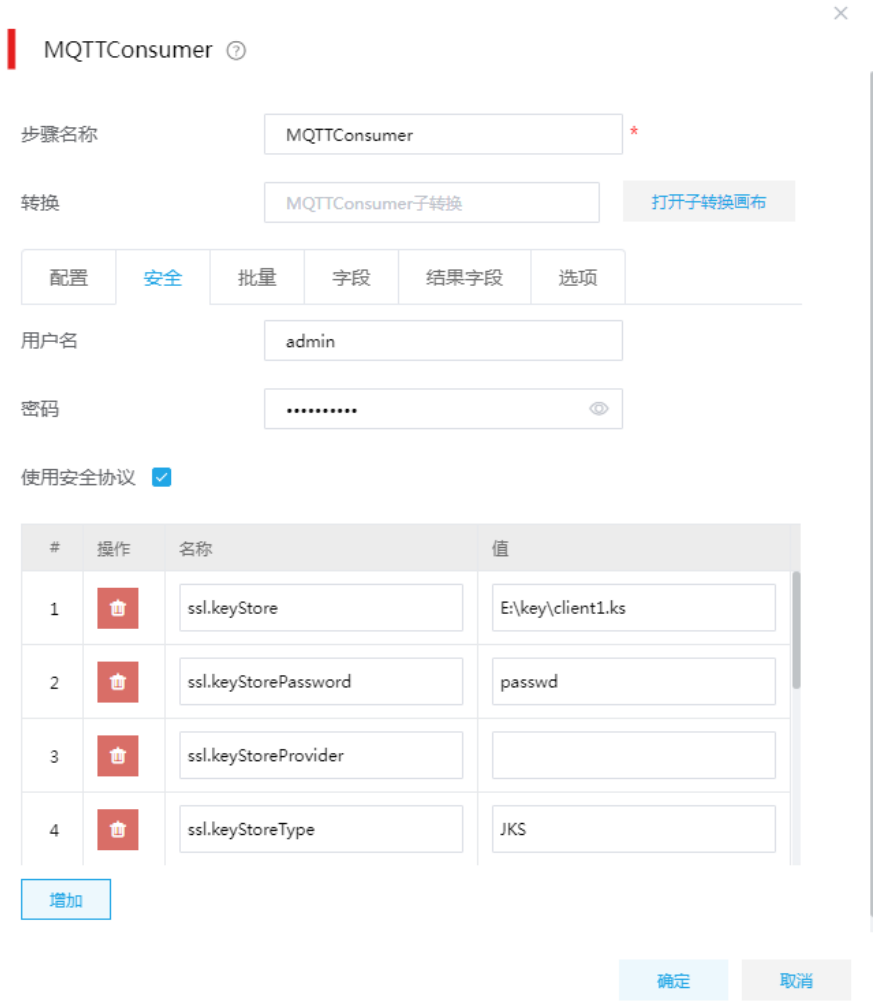
JSON 抽取组件，选择从前面步骤获取数据，这里的前一步骤也就是“从流中获取记录”组件。在字段页签下需要手动添加字段，这就需要预先了解采集的 JSON 数据的结构，重点关注 date 字段的配置，因为 date 字段在 JSON 文档中是一个字符串，比如：“2020/03/11 00:00:00”，所以这里选择使用 Date 类型去解析这个 JSON 属性情况下，需要指定与其匹配的格式。

图4-37 加载至表具体配置图示

完成上述配置工作后，保存子任务并退出子任务设计器。

主任务中组件配置说明：

图4-38 MQTTConsumer 安全设置



安全页签下的配置与 MQTTProducer 保持一致，填写说明参见模拟上报数据任务。

图4-39 MQTTConsumer 批量设置

批量页签下的两个配置项：持续时间、记录数量是触发子任务的两个条件。

- 持续时间：每隔多长时间触发一次子任务，单位毫秒。
- 记录数量：每采集到多少条数据触发一次子任务。

两个条件任意一个满足就会触发子任务，每次触发子任务，就会重新计时、计数。

至此，实时流数据采集任务设计全部结束。同时运行“模拟上报”任务、主任务，观察数据库中目标表数据量变化，发现有数据被周期性地写入。

4.5 大数据组件场景

对于 HDFS、Hive、HBase 等大数据组件，数据集成也做了适配。通过简单的拖拉拽，既可实现大数据组件中数据与关系数据库或者文本数据的相互转换。

1. 场景描述

A 公司存在多个孤立的业务系统，数据分散在各个业务数据库中。这样就导致了存在重复数据，并且数据核心价值没有很好被利用。现公司要求，将分散的数据统一集中，作为数据仓库。该公司数据主要分散在文本文件、关系型数据库（例如 MySQL）以及部分消息系统中，数据仓库建设以 Hive 为主，分层存储；ODS 与源数据保持同步，采用 Hive 外部表存储；DW 存储经过沉淀积累下来的数据，使用 Hive 内部表实现。

数据流向：各类数据源--> ODS 层—> DW 层。

2. 场景分析

分析业务需求，针对 A 公司的数据量，基本分为两种采集方式，全量采集和增量采集。以 100W 表中存量数据为分界，大于 100W 数据采用增量采集方式，小于则采用全量采集方式。

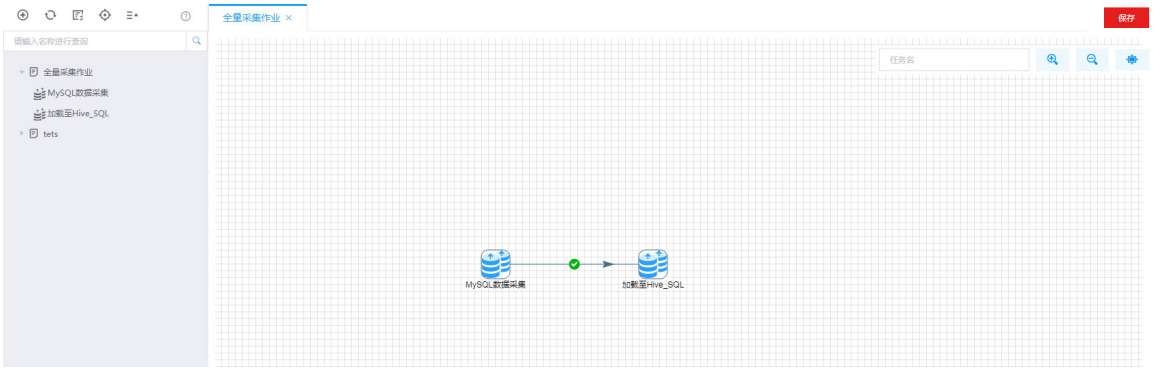
现仅以 dig(数据库名称 dig，是 MySQL)数据库为例，全量采集 behiver 表中。

注：增量采集思路，可参考其他增量采集思想，此处不做重复介绍。

3. ETL 设计方案

全量采集作业全流程：采集 behiver 数据到 ODS 层—>同步 ODS 数据到 DW。

图4-40 作业设计方案图示



图中，“MySQL 数据采集”是一个任务，将 MySQL 中数据抽取到 ODS 层指定的 hdfs 路径下，“加载至 Hive_SQL”也是一个任务，通过 insert overwrite 将数据同步到 DW 内部表中。

4. 示例前置条件

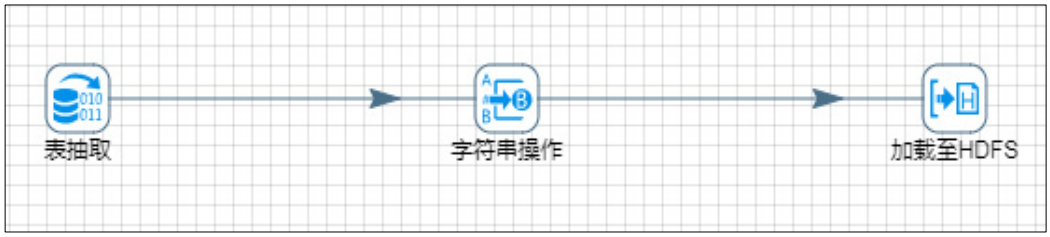
- ODS 层 ods_behavior 表已创建完成，location 路径“/usr/hdfs/gzh/behavior/”。
- DW 层 dw_behavior 表已创建完成。
- MySQL 数据源 dig，采集表 behiver，正常有数据。

5. 示例详细步骤

任务一：MySQL 数据采集

ETL 流程：MySQL 表数据抽取—>去除空格和换行字符—->数据加载至 HDFS 中。

图4-41 MySQL 数据采集图示

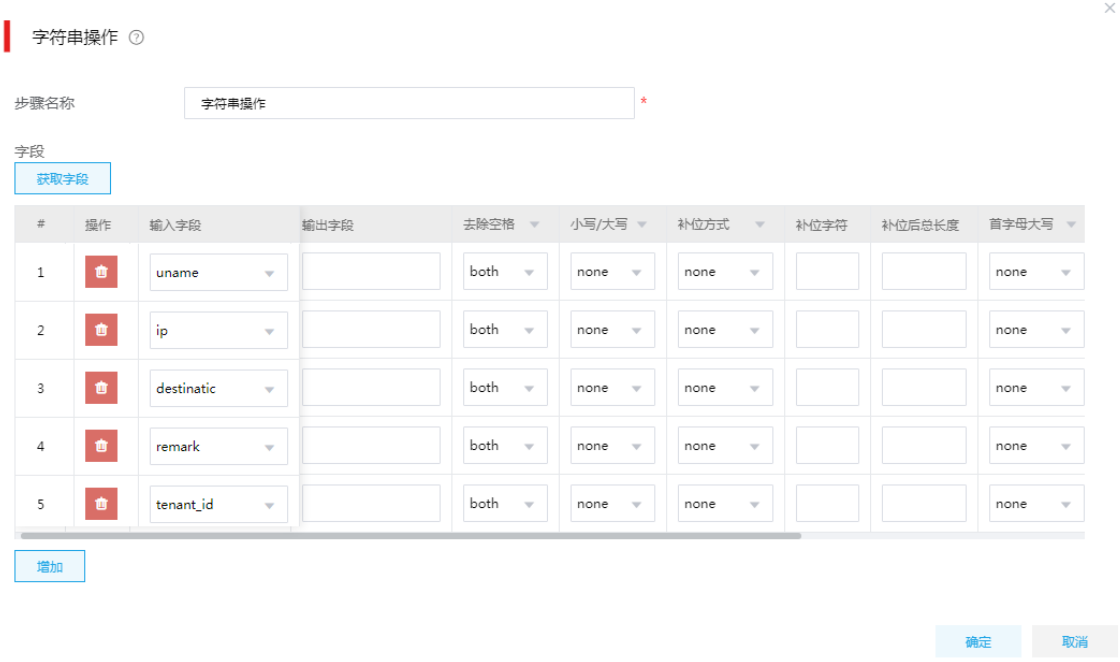


(1) 第一步：表抽取配置，数据源选择 dig，查询 SQL 如图 4-42。

图4-42 数据表抽取图示

(2) 第二步：字符串操作，获取字段并批量设置去除空格，清洗数据两端多余空格。“去除空格”批量设置 both，清洗数据两端空格。

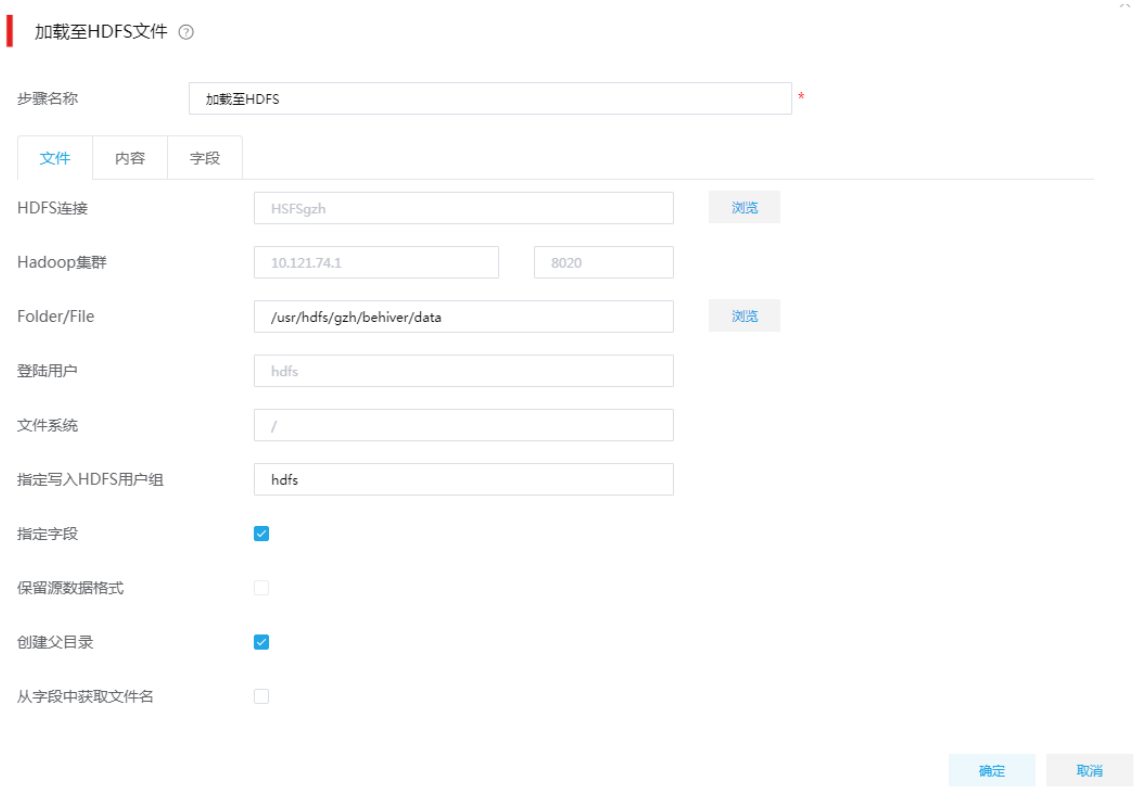
图4-43 字符串操作图示



(3) 第三步：加载至 HDFS 文件，浏览配置 HDFS 数据源及 HDFS 路径。

- Folder/File：/usr/hdfs/gzh/behiver/data。
- 指定字段：勾选。
- 创建父目录：勾选。

图4-44 加载至 HDFS 文件图示



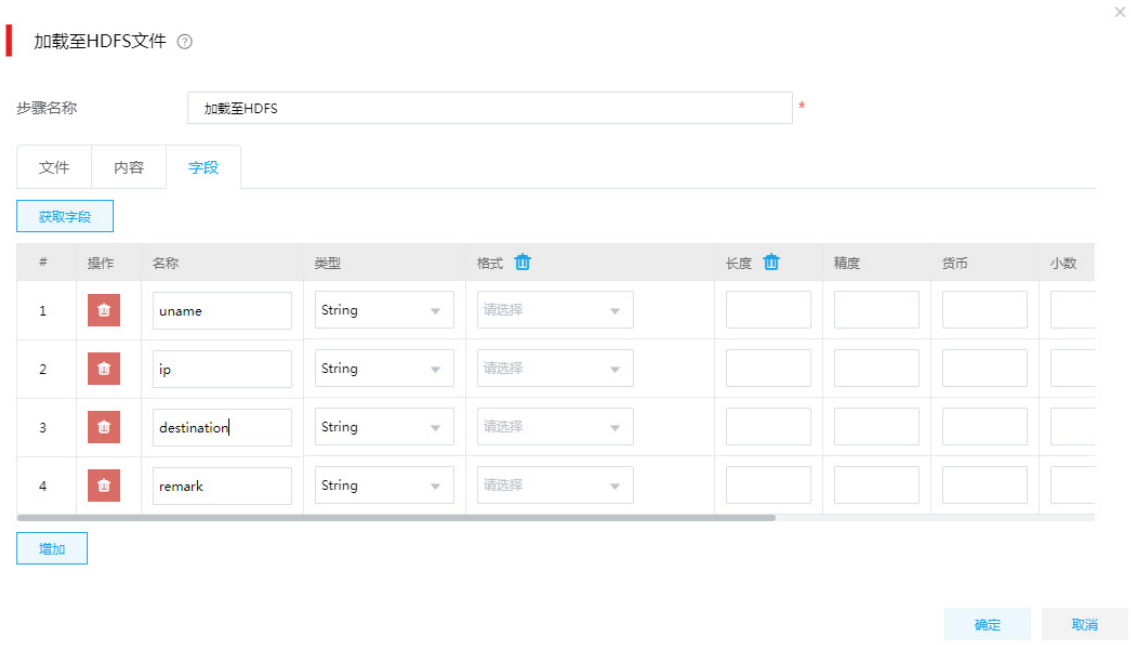
(4) 加载至 HDFS 文件“内容”页签配置：

- 分隔符：使用竖线“|”，如图。
- 格式：下拉选择 Unix。
- 编码方式：下拉选择 UTF-8。

图4-45 HDFS 文件“内容”页签配置图示

(5) “字段”页签配置：获取字段，并去除不需要的字段即可。

图4-46 “字段”页签配置图示



任务二：ODS 数据同步到 DW

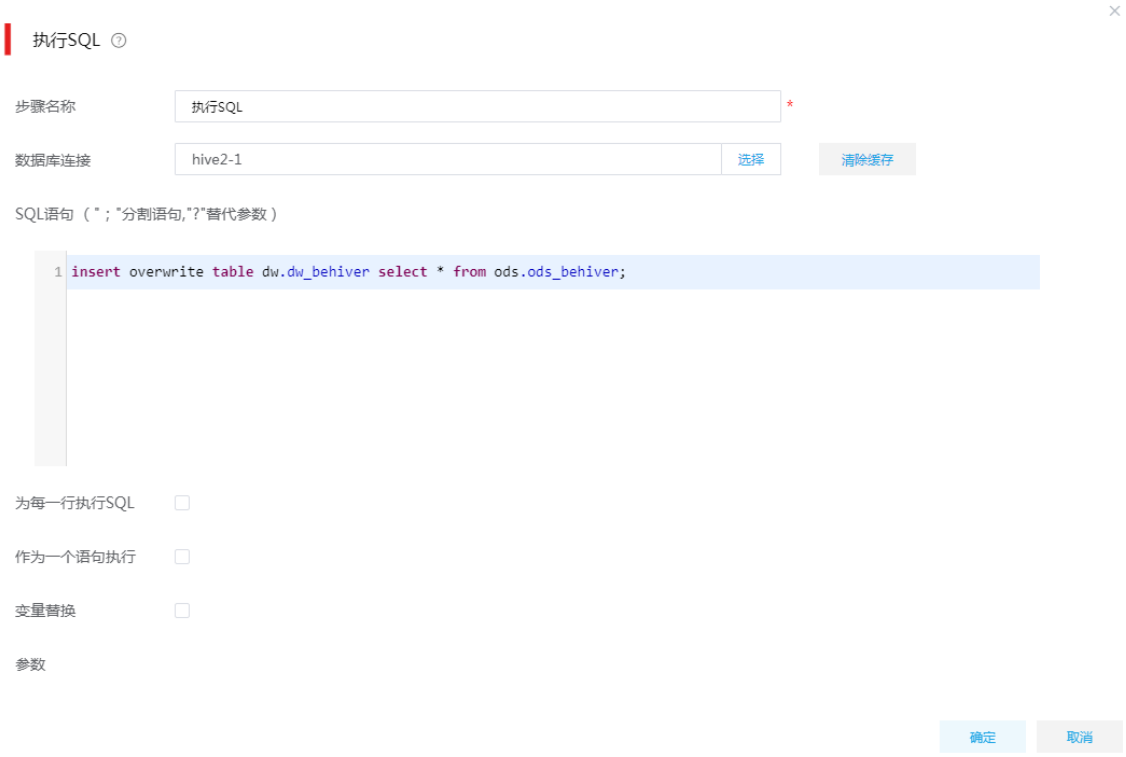
执行 SQL，将数据从 Hive 外部表加载至内部表中。

图4-47 执行 SQL 图示

“执行 SQL”配置：

- 数据库连接：选择 Hive2-1。
- SQL 语句：采用 insert overwrite 方式同步数据，如图 4-48 所示。

图4-48 执行 SQL 配置图示



4.6 数据清洗场景

1. 场景描述

数据集成支持文本、数据库、大数据组件等多种数据源，在数据集成过程中，经常会需要将数据中不符合规则的数据进行清洗或转换，格式归一后存储到目标仓库中。

2. 场景分析

A 公司需要将数据库中一张表数据，按照规定的格式加载至远程文件中。该表是一个车辆信息表，要求去除车牌号不符合格式的数据，另外将人员类型（RYLX）为空的默认设置为 2。

图4-49 示例数据源（test 表）

	id	kkwzbm	hphm	hpzl	rylx	tgsj
1	1	311,001	豫A11111	1	2	1990-01-01 12:12:12
2	2	311,002	xyz	1	2	2010-01-01 12:12:12
3	3	311,001	豫A11111	1	[NULL]	1990-01-01 12:12:12

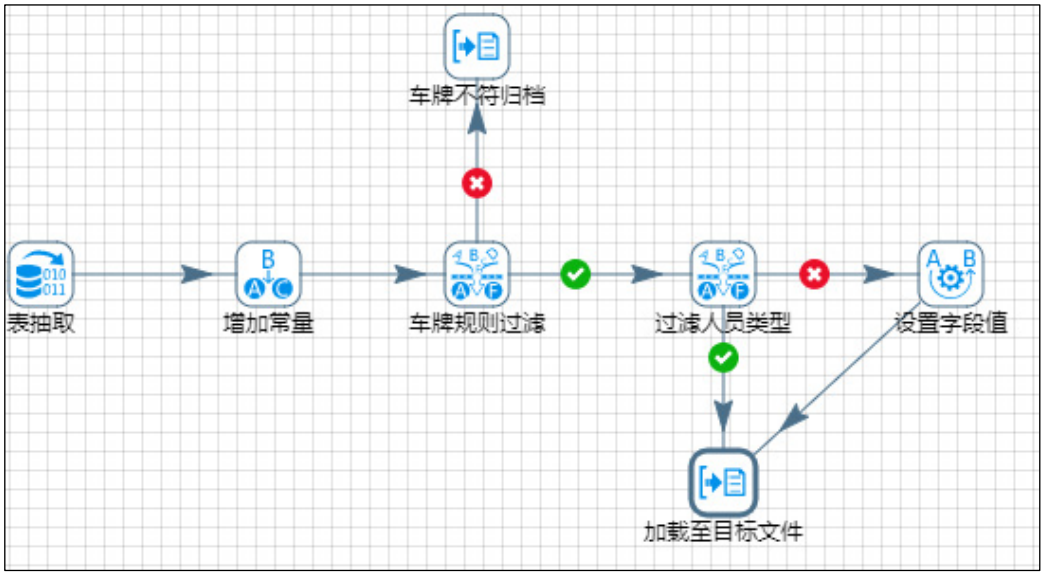
测试数据源说明：

- hphm：是车牌号字段，其中可能出现未知或不符合车牌号码的车牌号。要求将此字段数据进行清洗，去除不符合规则的纪录。
- rylx：是人员类型字段，可能为空，需要对此字段进行处理，将空值转换为 2。

3. ETL 设计方案

使用表抽取组件进行数据抽取，通过过滤记录组件中正则表达式将 hphm 中不符合要求的数据过滤掉，加载至文本文件归档。正确的数据再次过滤 rylx 为空的数据，为空的数据传递设置字段值，将空值赋值为 2，最后将据全部加载至步骤加载至目标文件。

图4-50 任务图示



4. 示例前置条件

无

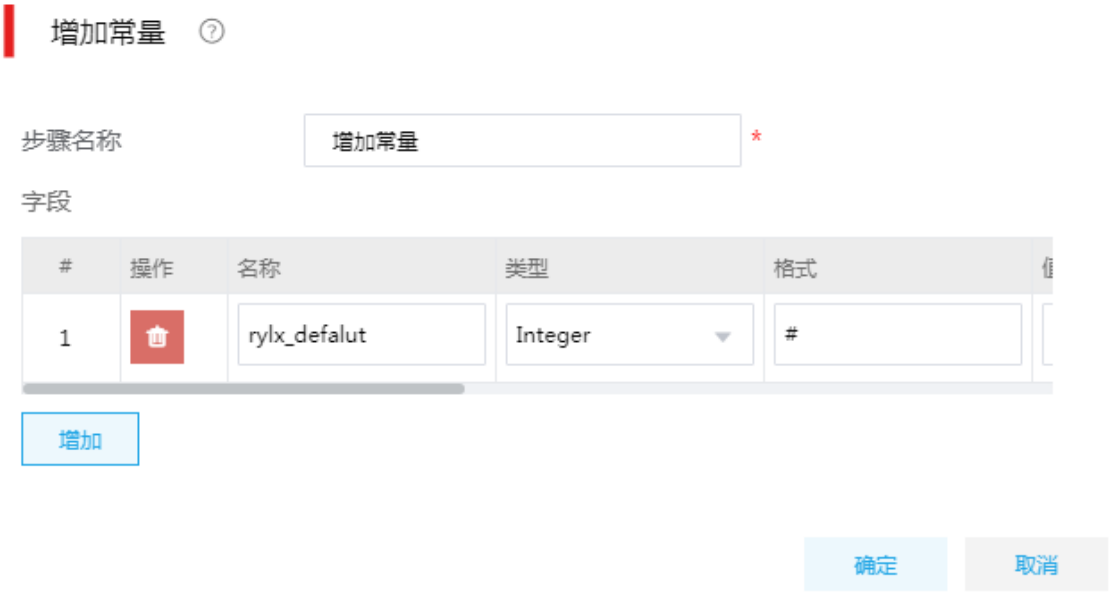
5. 示例详细步骤

(1) 第一步：表抽取配置，配置要采集的数据源及 SQL 语句。

图4-51 表抽取配置图示

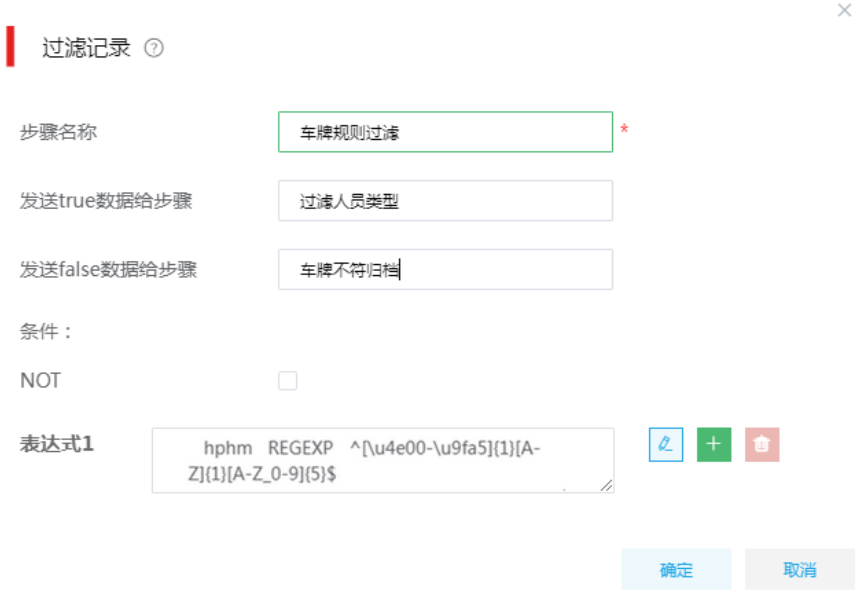
(2) 第二步：增加常量配置，配置 rylx_default 的默认值为 2。

图4-52 增加常量配置图示



(3) 第三步：车牌规则过滤，使用过滤记录组件，配置过滤车牌的规则。用于测试用的车牌的正则表达式为：^[\u4e00-\u9fa5]{1}[A-Z]{1}[A-Z_0-9]{5}$。

图4-53 车牌规则过滤图示

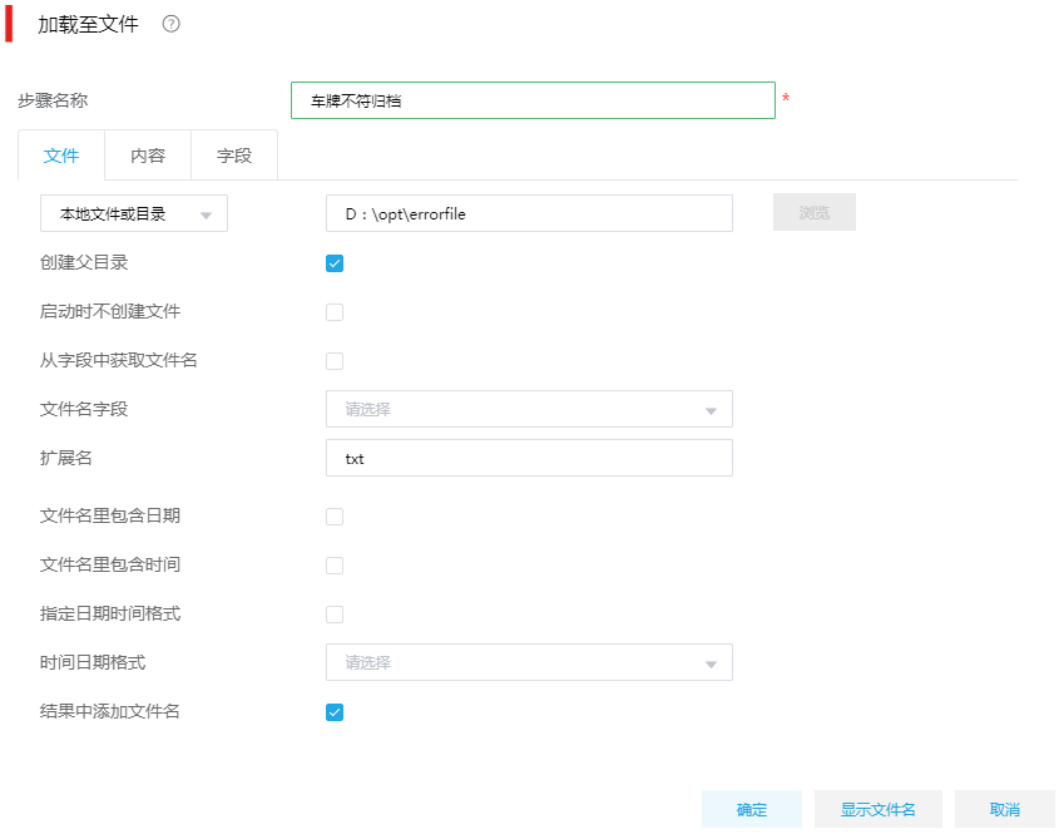


(4) 第四步：车牌不符归档配置，使用加载至文件。

(5) “文件”页签配置：

- 本地文件或目录：指定归档文件，“D:\opt\errorfile”。

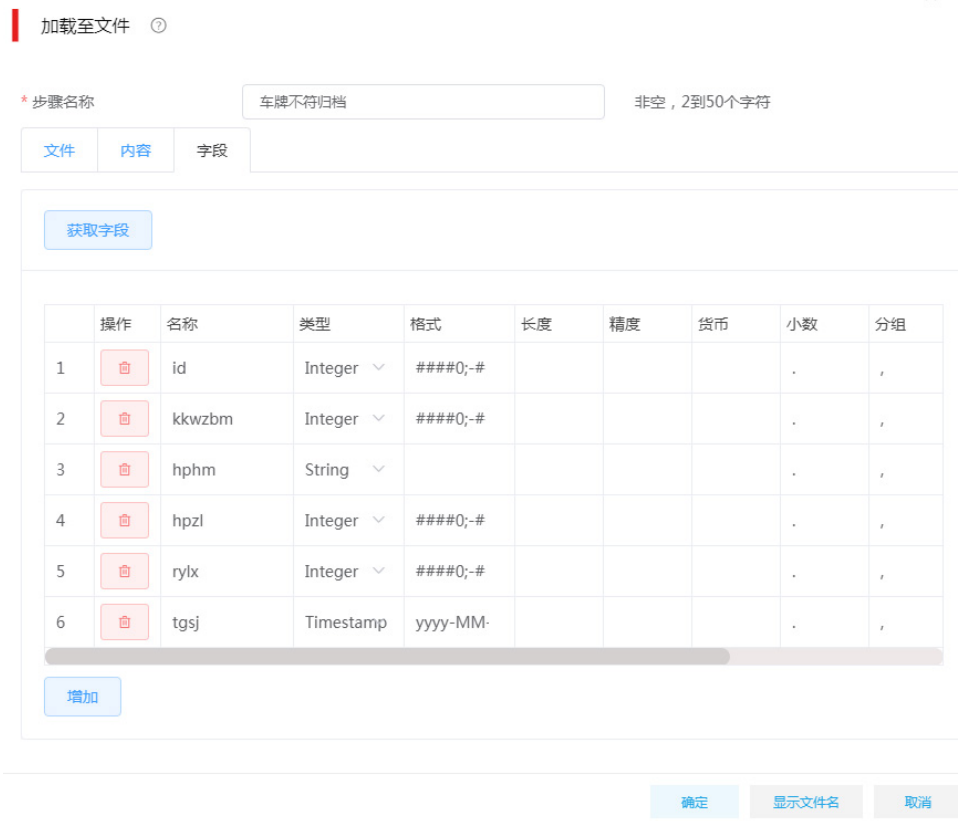
图4-54 “文件”页签配置图示



(6) “字段”页签配置：

- 注意：需移除 rylx_default 字段。

图4-55 “字段”页签配置图示



(7) 第五步：过滤人员类型，使用过滤记录，过滤出 rylx 为空的记录。

- 发送 true 数据给步骤：指定校验通过的数据流向，此处为“加载至目标文件”。
- 发送 false 数据给步骤：指定校验未通过的数据流向，此处为“设置字段值”。
- 条件：配置表达式 1 的内容，如图 4-56 所示。

图4-56 过滤记录图示

(8) 第六步：设置字段值，将 rylx 的值设置默认值（流中 rylx_default 的值）。

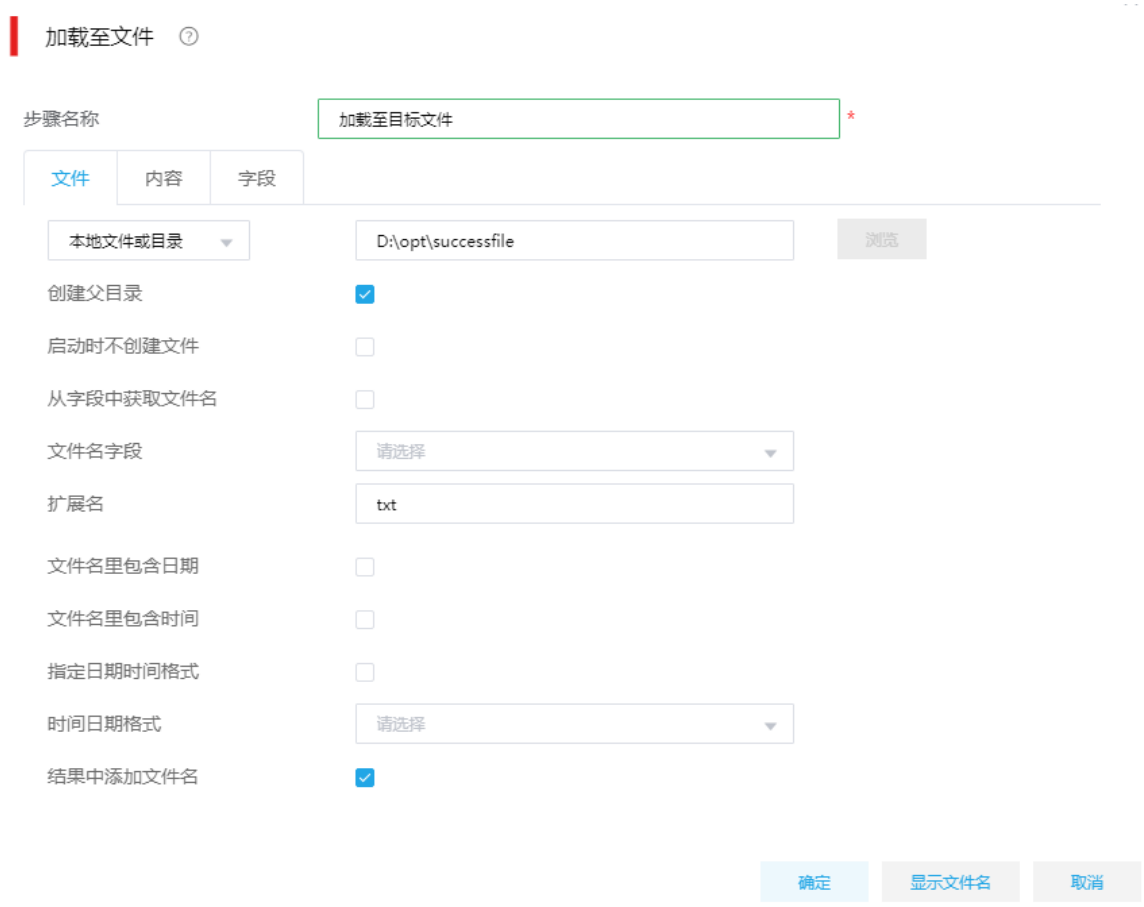
- 输入字段名称：rylx
- 替换字段名称：rylx_default

图4-57 设置字段值图示

(9) 第七步：加载至目标文件，使用加载至文件组件。

- 本地文件或目录：指定目标文件路径。

图4-58 加载至文件图示



(10) “内容”页签配置：

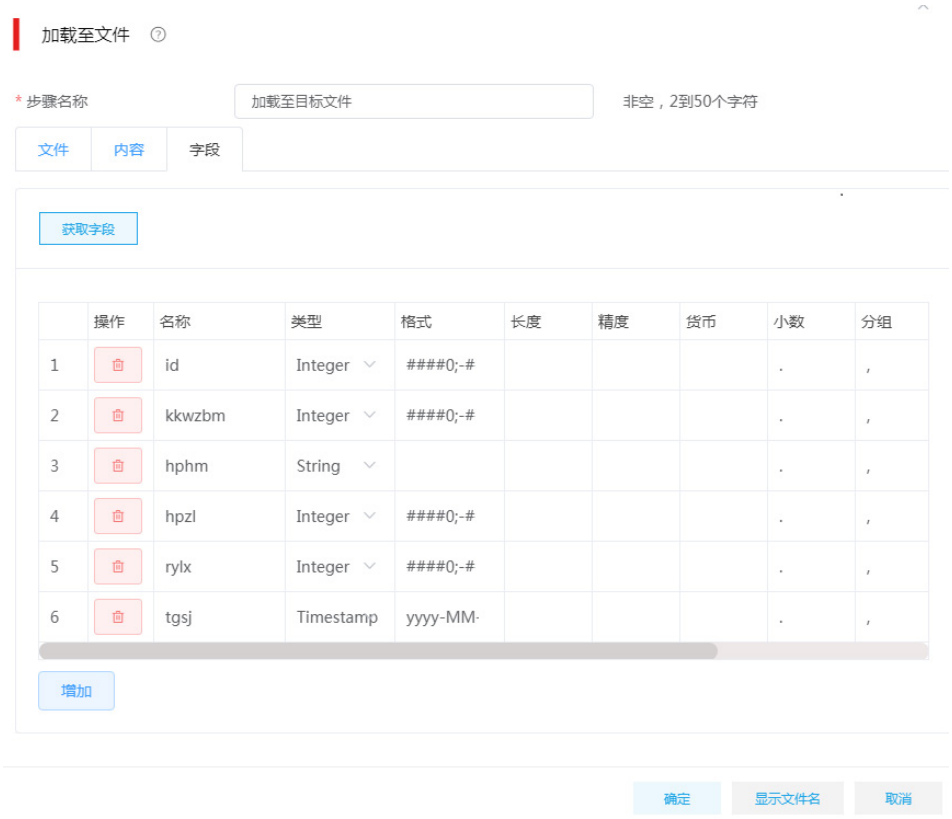
- 格式：下拉选择 Unix 格式。
- 编码：下拉选择 UTF-8 格式。

图4-59 “内容”页签配置

(11) “字段”页签配置：

注意：删除字段 rylx_default 字段。

图4-60 “字段”页签配置



4.7 整库迁移

在做数据迁移时，会遇到数据表较多的情况，虽然使用数据集成模板的批量部署可完成数据迁移。但最优的选择是“整库迁移”，它提供了自动建表、自动采集的能力，可通过一次简单的配置，即可完成多个表的迁移工作。

1. 场景描述

A 公司 MySQL 数据库中存有历史数据，大约 20+张表，需要将数据迁移到 PostgreSQL 数据库中。

2. 场景分析

根据 A 公司的需求，我们可通过数据集成的全量抽取方式完成数据迁移，但需要重复创建 20+的任务。而使用整库迁移，可通过一次创建即可完成多表的全量抽取。

3. ETL 设计方案

使用数据集成的整库迁移功能完成。

4. 示例前置条件

源库：MySQL 数据库。

目标库：PostgreSQL 数据库。

5. 示例详细步骤

(1)　第一步：选择[整库迁移]，进入整库迁移页面，单击“创建作业”，弹出配置框。

图4-61 创建整库迁移作业

(2) 第二步：配置要迁移的表及目标库。

- 作业名称：MySQL-2-PG。
- 是否立即运行：勾选“否”。
- 是否清空表：勾选“是”（建议勾选）。
- 每个任务复制表数：配置为 6。此处示例中有 27 张表，且数据量不大。
- 要复制的表：通过单击“选择”，勾选要迁移的源表。
- 目标模式：选择 mysql_bak。
- 目标数据库：选择 mysql_bak_pg。

图4-62 配置图示

(3) 第三步：单击创建作业，完成作业创建。在作业上线运行即可。

4.8 GPLoad加载

GPLoad 加载组件适用于 GreenPlum、SeaSQL MPP 及 Generic JDBC 数据库，在大数据量、字段较多场景下，相较于加载至表组件，提供了更好的加载性能。

1. 场景描述

A 集团业务系统数据库使用 PostgreSQL，其中有一张物料信息记录表 mm_1000，字段数近 100 个，现有数据 100 万条，每天物料信息的变化会导致表中记录的修改和增加。该集团现需要定期将业务系统数据库表 mm_1000 中的数据，同步到数据仓库 SeaSQL MPP 中的 mm_1000000。

2. 场景分析

考虑到数据量较大、表字段数较多，这里可以使用数据集成的 GPLoad 批量加载组件。GPLoad 组件运行时先将源表数据写入到落地文件，然后经由 greenplum-loaders 插件提供的文件分发功能，将落地文件发送给 SeaSQL MPP 数据库的各个计算节点，然后在数据库端通过落地文件建立外部表，最后 SeaSQL MPP 各个计算节点在主节点调度下，并行地从外部表查询数据插入到目标表中。

3. ETL 设计方案

数据流向：执行 SQL—>表抽取—>加载至表。

ETL 方案：考虑每次同步数据时，源表数据部分发生了修改，同时又有新增数据，首先通过执行 SQL 组件，清空目标表 mm_1000000，然后使用表抽取组件对源表 mm_1000 进行全量抽取，最后使用 GPLoad 加载至目标表 mm_1000000。

图4-63 ETL 任务设计图示

4. 示例前置条件

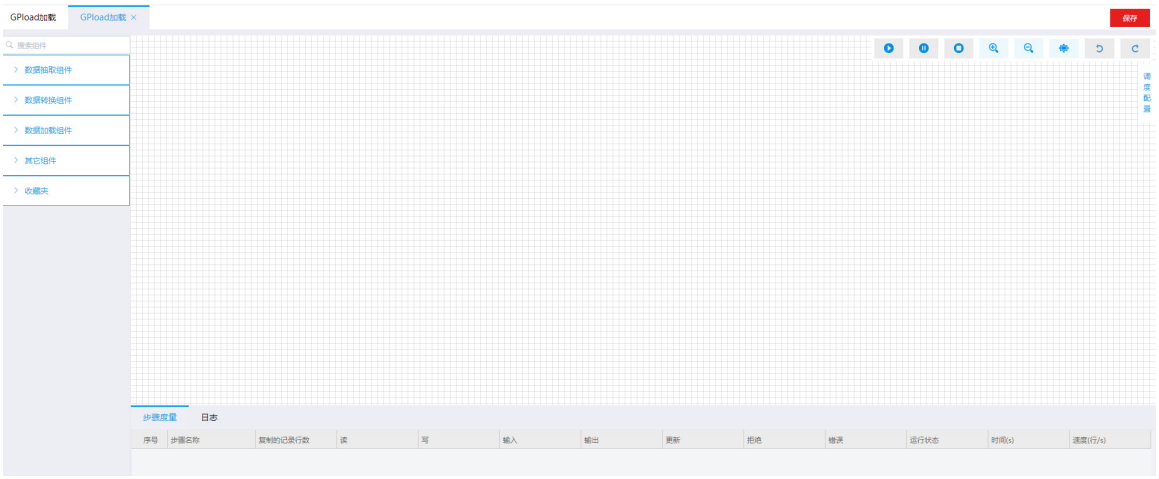
PostgreSQL 数据库中 mm_1000 表已创建完成，数据量不少于 10 万条。

SeaSQL MPP 数据库中 mm_1000000 表已创建完成。

5. 示例详细步骤

进入[作业管理/作业定义]页面，单击<新建>按钮，新建作业。将 ETL 任务添加到作业设计画布中，双击任务名称后跳转至 ETL 任务设计页面，如图 4-64 所示。

图4-64 新增作业图示

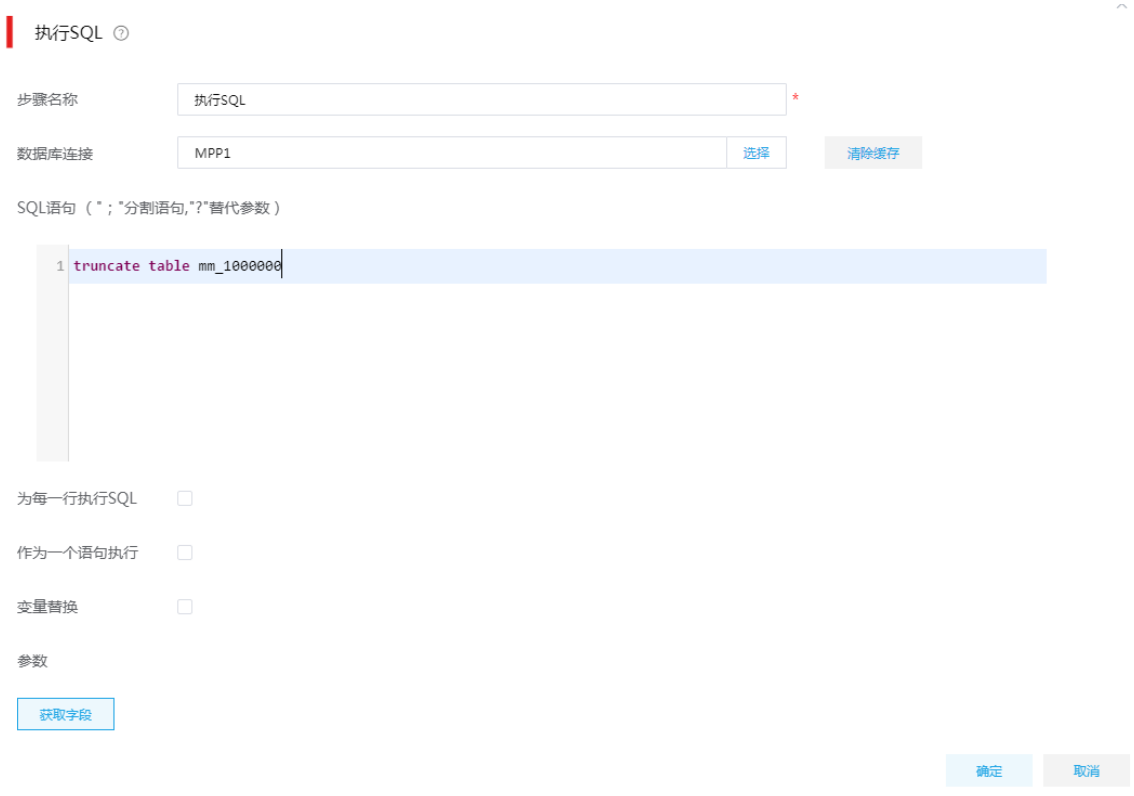


按照设计方案，拖拽组件步骤、建立连接。各步骤详细配置：

(1) 执行 SQL

最终被插入数据的目标表为 mm_1000000。在这一步骤中通过 SQL 语句清空 mm_1000000 表中所有数据。

图4-65 执行 SQL 具体配置图示

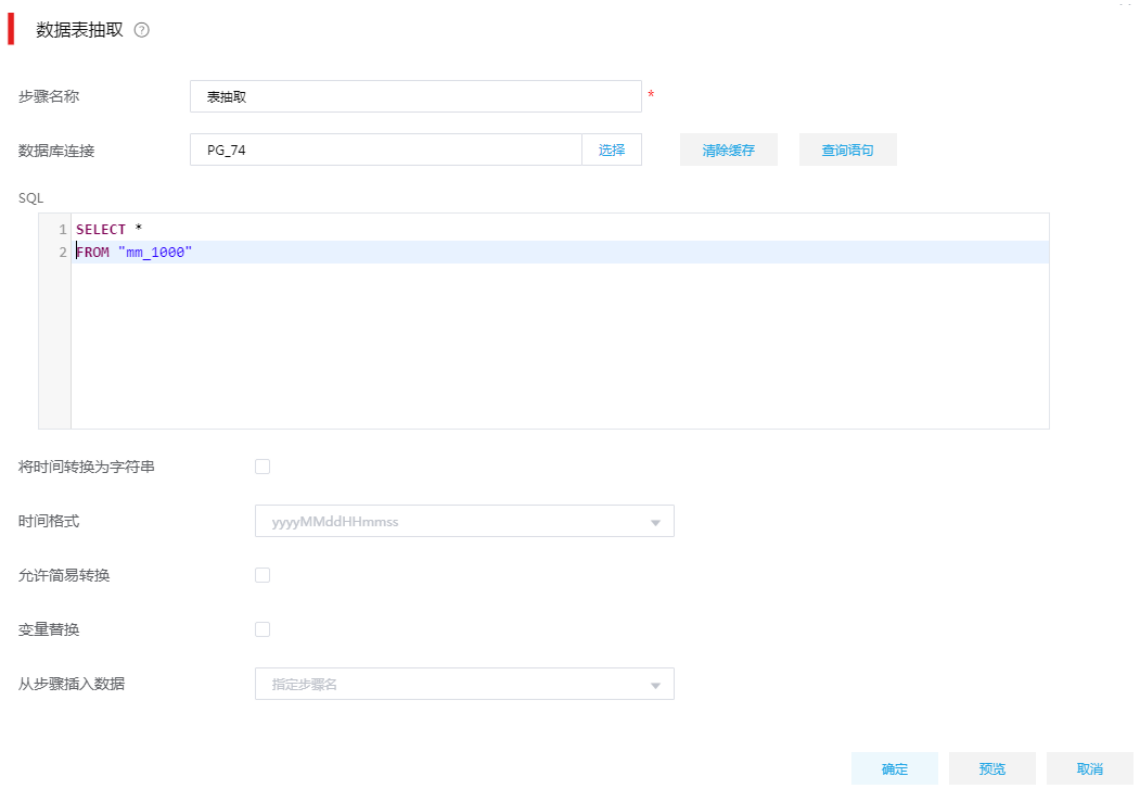


(2) 数据表抽取

通过数据表抽取步骤，抽取 PostgreSQL 数据库中表 mm_1000 的所有数据。

【注意】：目标数据库 SeaSQL MPP 是基于 PostgreSQL 实现的分布式数据库，因为这里的业务库是 PostgreSQL，所以简单的使用如图 4-66 所示的查询语句，若业务库使用的是其他类型数据库，比如：mysql、oracle 等，则需要考虑源表中数据可能包含 PostgreSQL 无法处理的字符，此时查询语句需要适当调整。

图4-66 数据表抽取具体配置图示



(3) GPLoad 加载

将从源表中抽取出的数据，使用 GPLoad 组件批量加载至 SeaSQL MPP 库的 mm_1000000 表中。

图4-67 GPLoad 组件-基本配置

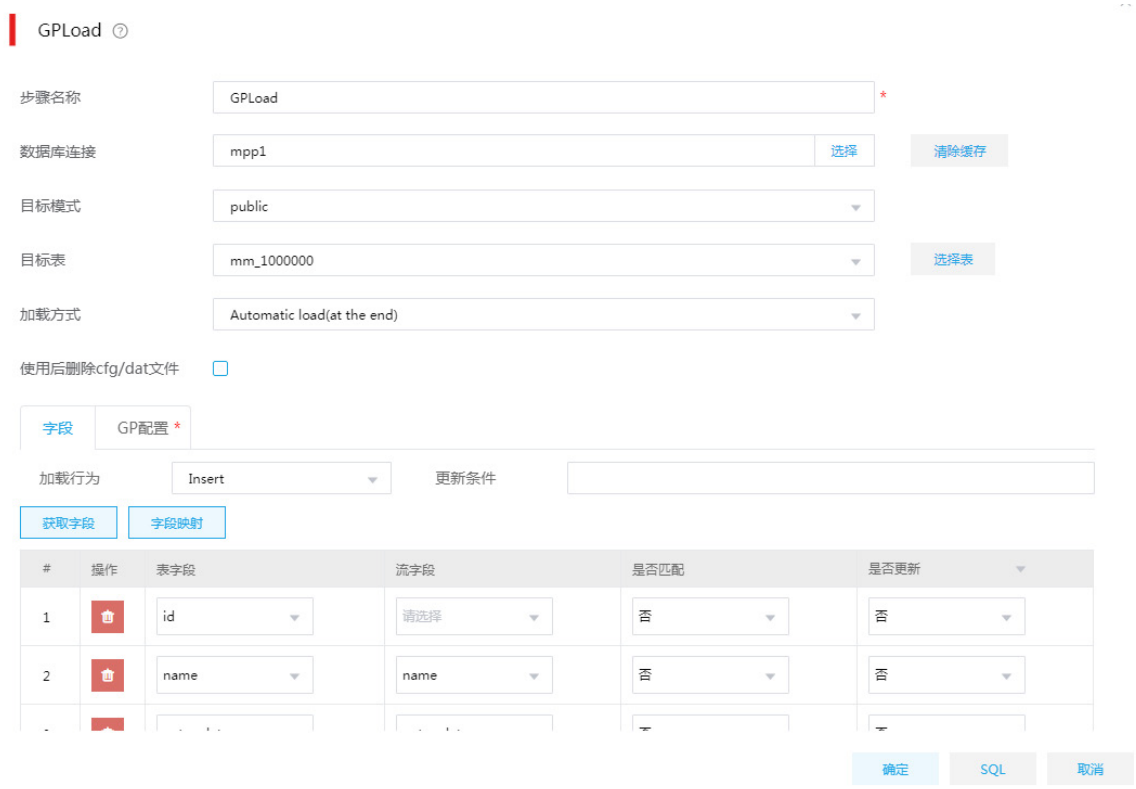
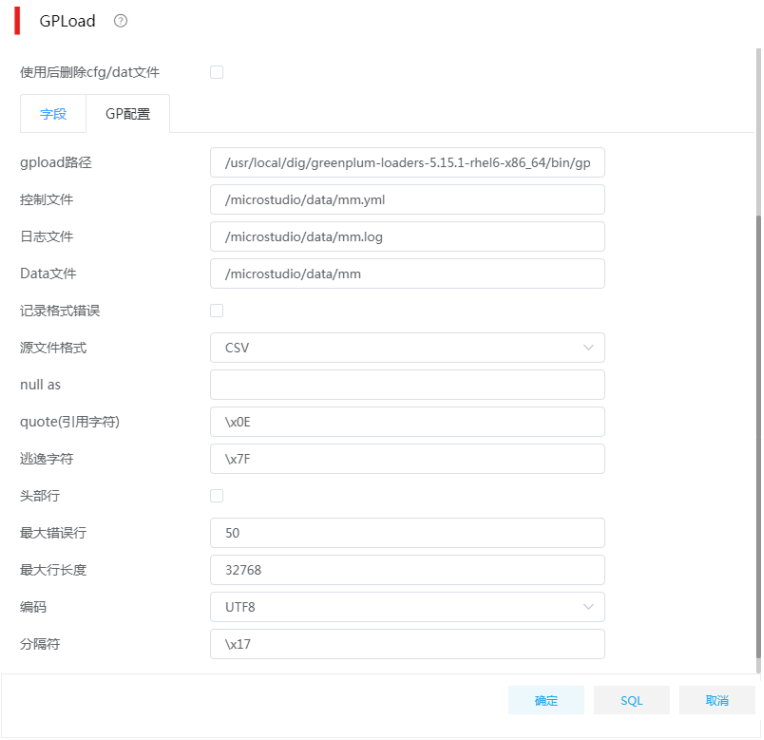


图4-68 GPLoad 组件-GP 配置



(4) GPLoad 组件配置大概分两部分：

- 配置目标表相关信息，包括数据库、目标表、字段等。
- GP 配置，该部分信息用于指导生成落地文件，为 greenplum-loaders 插件运行做准备。主要包括以下几项配置：
  - 控制文件：生成的控制文件全路径，要求文件的父目录必须存在。
  - 日志文件：生成的插件运行日志文件全路径，要求文件的父目录必须存在。
  - Data 文件：生成的数据文件的全路径，要求文件的父目录必须存在，Data 文件中会被写入源表抽取出的全部数据。
  - 源文件格式：将源表数据写入 Data 文件时使用的文件格式，可选 TEXT/CSV。
  - quote(引用字符)、逃逸字符、分隔符参数的含义详见组件帮助信息，要求分割符不能与 quote(引用字符)、逃逸字符相同。建议从下拉菜单中选择“非可打印字符”，避免与源表数据冲突。
  - 最大错误行：设置最大允许出错行数，仅适用于格式问题导致的某行数据加载错误，若出错行数不超过该参数值，这些错误数据会被忽略，其他数据正常入库，若错误数超过该参数，所有数据加载失败。
  - 最大行长度：Data 文件中一行数据的最大字节数，根据源数据一行记录的实际大小设置。

(5) 任务设计完成后，单击<运行>，等待数据同步完成。

4.9 REST抽取

越来越多的数据服务提供商，通过 REST API 为客户提供数据服务，比如天气、位置、公交、火车、交通违章、快递等数据定制查询服务等。API 接口常用的数据交换格式有 XML、JSON 等。

1. 场景描述

A 公司对接客户 REST 接口，该接口提供了健康人员注册信息查询服务，数据交换格式为 JSON，A 公司需将对接的数据解析并写入关系型数据库。

接口基本信息：

- 接口地址：http://10.121.57.38:8089/person
- 请求方式：GET
- 请求参数：
  - appKey
    - 参数描述：header，授权码;
    - 参数类型：varchar
    - 参数位置：header
    - 是否必填：是
  - pageNumber.cludove
    - 参数描述：请求页数，页数从 0 开始;
    - 参数类型：varchar
    - 参数位置：query
    - 是否必填：是

- recordNumber.cludove
  - 参数描述：请求每页显示条数，最大 1000 条，不传此参数默认为每页请求 200 条信息
  - 参数类型：varchar
  - 参数位置：query
  - 是否必填：是
- LAST_MODIFY_TIME
  - 参数描述：时间段查询参数，以天为单位步进查询，必填，例如要查询 2020 年 3 月 19 日的数据，则参数值为 2020-03-19 00:00:00,2020-03-19 23:59:59 ,以小值在前,大值在后,中间以英文逗号分隔的形式传参，因为参数值中有空格等特殊字符，需要将这个参数的值进行两次 url 编码后，再用来查询
  - 参数类型：varchar
  - 参数位置：query
  - 是否必填：是

- 请求示例：

  例如：查询时间段 2020 年 3 月 19 日这天的数据（从第 0 页开始，每页 1000 条记录），LAST_MODIFY_TIME 的参数值实际为 2020-03-19 00:00:00,2020-03-19 23:59:59 ，下面的值是经过了两次 URL 编码的：

  pageNumber.cludove=0&recordNumber.cludove=1000&LAST_MODIFY_TIME=2020-03-19%2b00%253a00%253a00%252c2020-03-19%2b23%253a59%253a59

- 采集 JSON 数据示例：

```
{
  "records" : [
        {
            "ID_CARD": "411678199905092345"     ,
            "NAME": "张三",
            "POSITION": "郑州市高新区金梭社区云都会小区南门",
            "TEMPERATURE": "36.6",
            "LAST_UPDATE_TIME": "2020-03-23 17:35:26"
        },{…},…,{…}
  ]
}
```

2. 场景分析

数据集成已支持 REST 接口数据采集，可采用数据集成的 JSON 抽取组件实现数据转换（将半结构化数据转换为结构化数据），并通过加载至表将数据写入数据库。

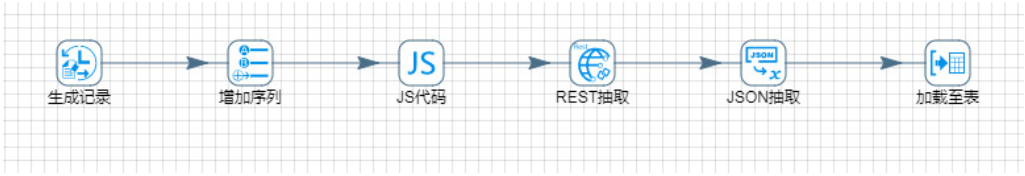
3. ETL 方案设计

数据流向：生成记录 –> 增加序列 -> JS 代码 –> REST 抽取 -> JSON 抽取 ->加载至表

ETL 方案：要求每次运行作业时，将当前运行时间前一天更新的数据同步到目标表。比如当前调度时间为 2020/3/20 09:00:00，则 REST 抽取组件的 Query 参数中的 LAST_MODIFY_TIME 应该设置为 2020-03-19 00:00:00,2020-03-19 23:59:59 ，这个字符串格式的“时间范围”可以使用 JS

代码组件依据当前调度时间计算得出；另外 REST 抽取还需要传入 pageNumber.cludove（请求页数）、recordNumber.cludove（每页记录数）、appKey（API 调用授权码），其中请求页数应该是递增的，可以使用增加序列组件产生，每页记录数和授权码是固定的，使用生成记录组件生成即可。

图4-69 ETL 任务设计图示



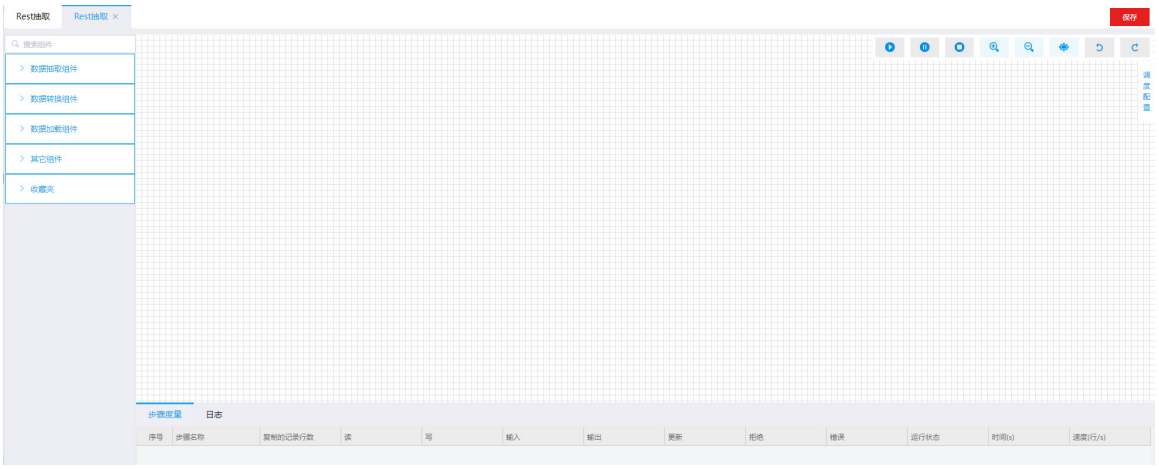
4. 示例前置条件

PostgreSQL 数据库中 hh_1000 表已创建完成。

5. 示例详细步骤

进入[作业管理/作业定义]页面，单击<新建>按钮，新建作业。将 ETL 任务添加到作业设计画布中，双击任务名称后跳转至 ETL 任务设计页面，如图 4-70 所示。

图4-70 新增作业图示

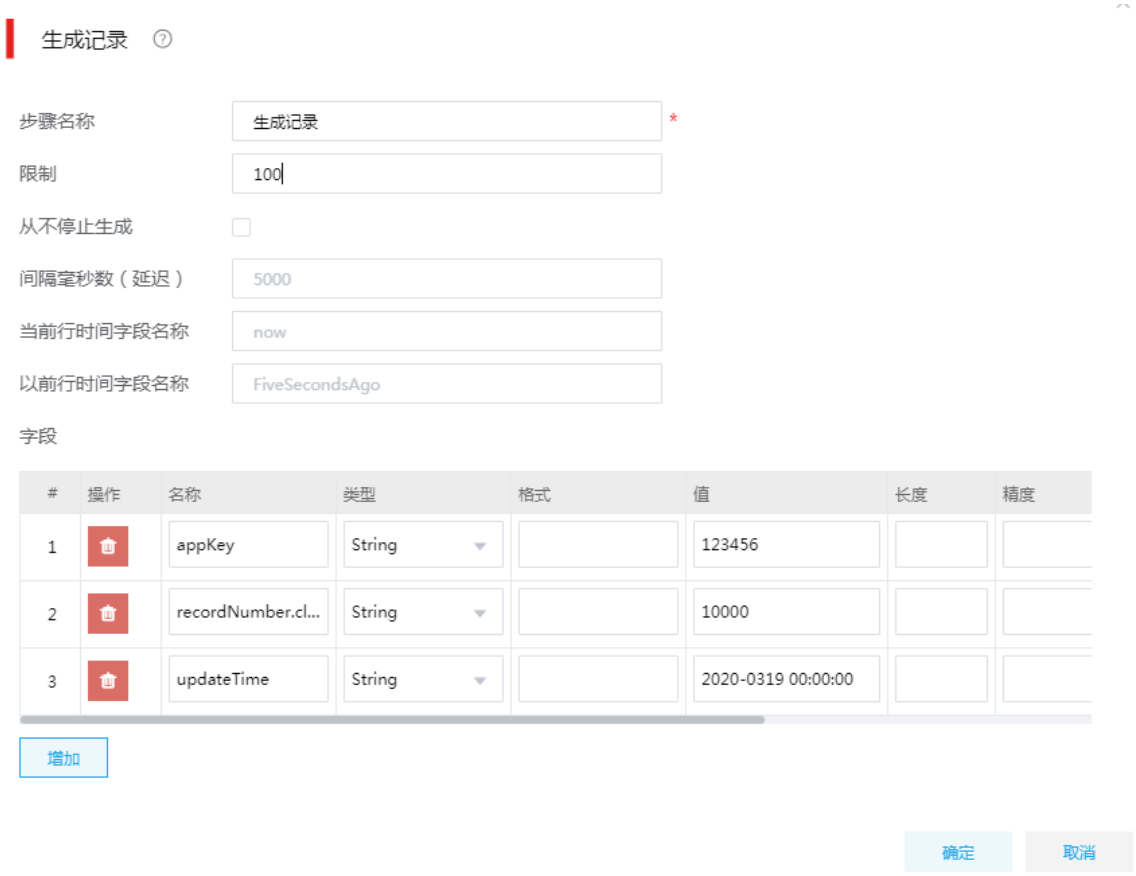


按照设计方案，拖拽组件步骤、建立连接。各步骤详细配置：

(1) 生成记录

该步骤为了配合 JS 代码组件，生成基于当前调度时间的 LAST_MODIFY_TIME 的参数值，同时为了生成 REST 抽取组件需要用到的 recordNumber.cludove（每页记录数）、appKey（API 调用授权码）两个参数的值。

图4-71 生成记录具体配置图示



(2) 增加序列

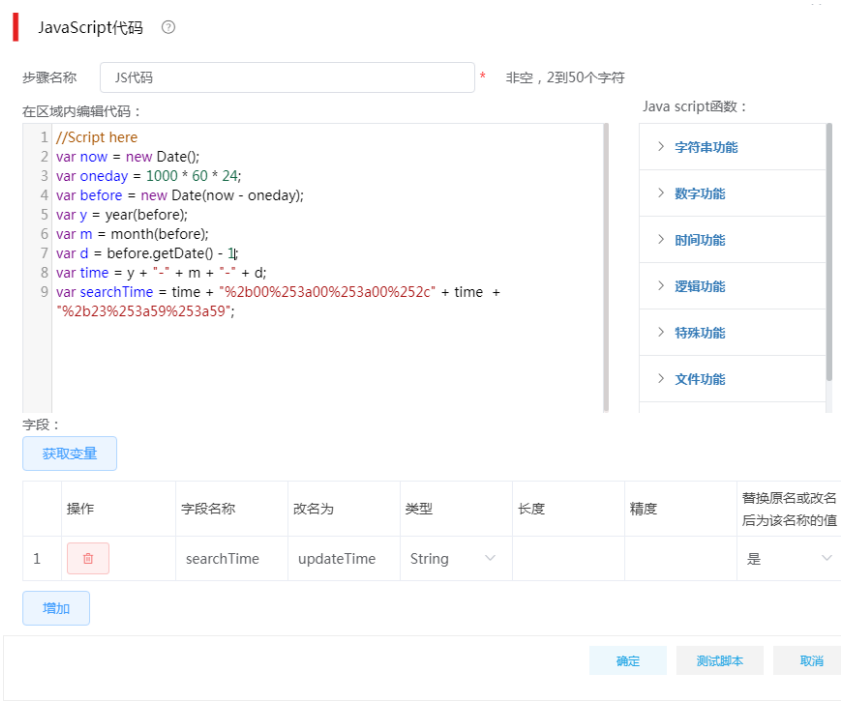
增加序列步骤主要是生成递增的 pageNumber.cludove（请求页数）参数。

图4-72 增加序列具体配置图示

(3) JS 代码

JS 代码步骤根据当前调度时间，计算拼接出 LAST_MODIFY_TIME 参数的值。

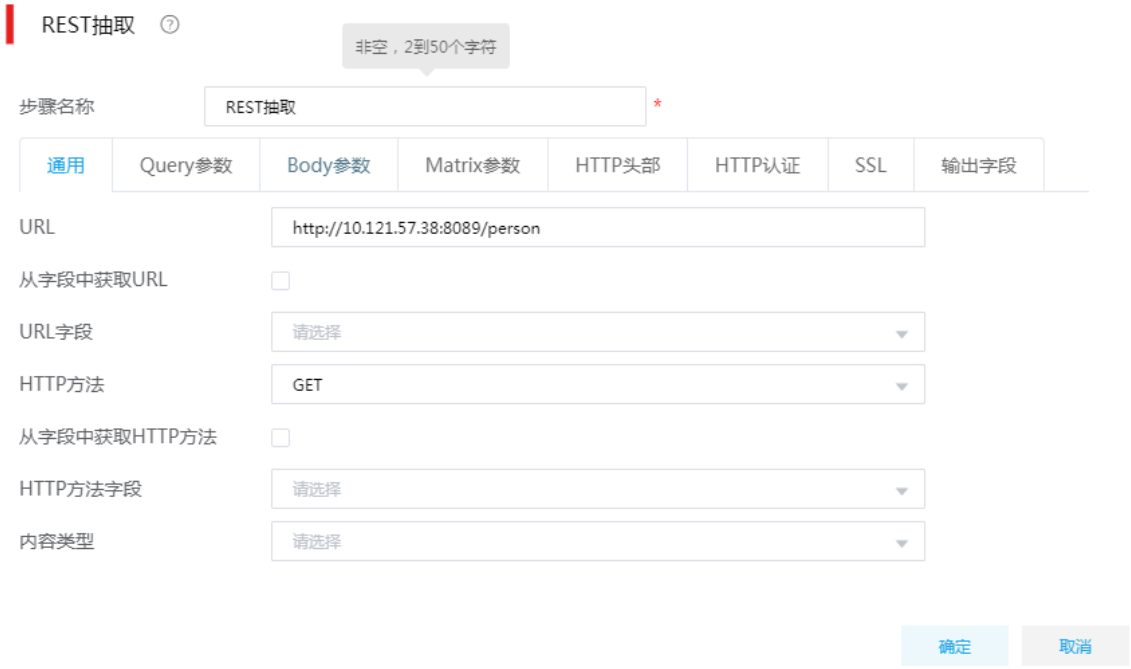
图4-73 JS 代码具体配置图示



(4) REST 抽取

通过 REST 抽取步骤，抽取接口数据。

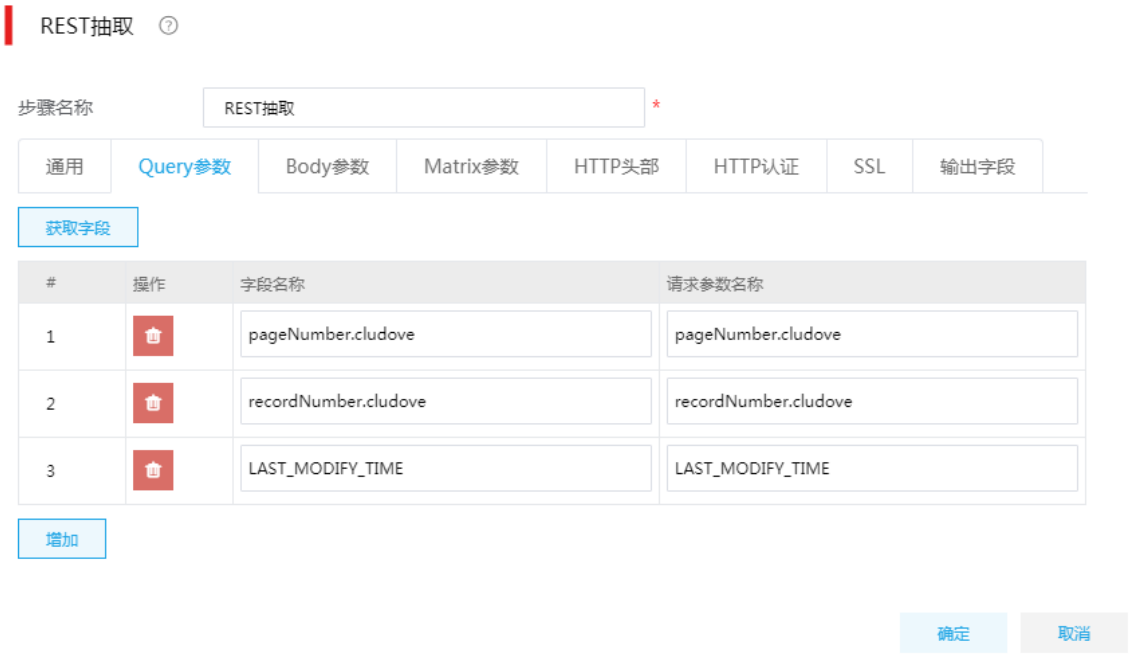
图4-74 REST 抽取-通用



按照待抽取的 REST 接口定义，在组件的通用页签下，需要配置 URL、HTTP 方法

- URL: http://10.121.57.38:8089/person
- HTTP 方法：GET

图4-75 REST 抽取-Query 参数



按照待抽取的 REST 接口定义，在组件的 Query 参数下，需要添加三个参数：pageNumber.cludove、recordNumber.cludove、LAST_MODIFY_TIME

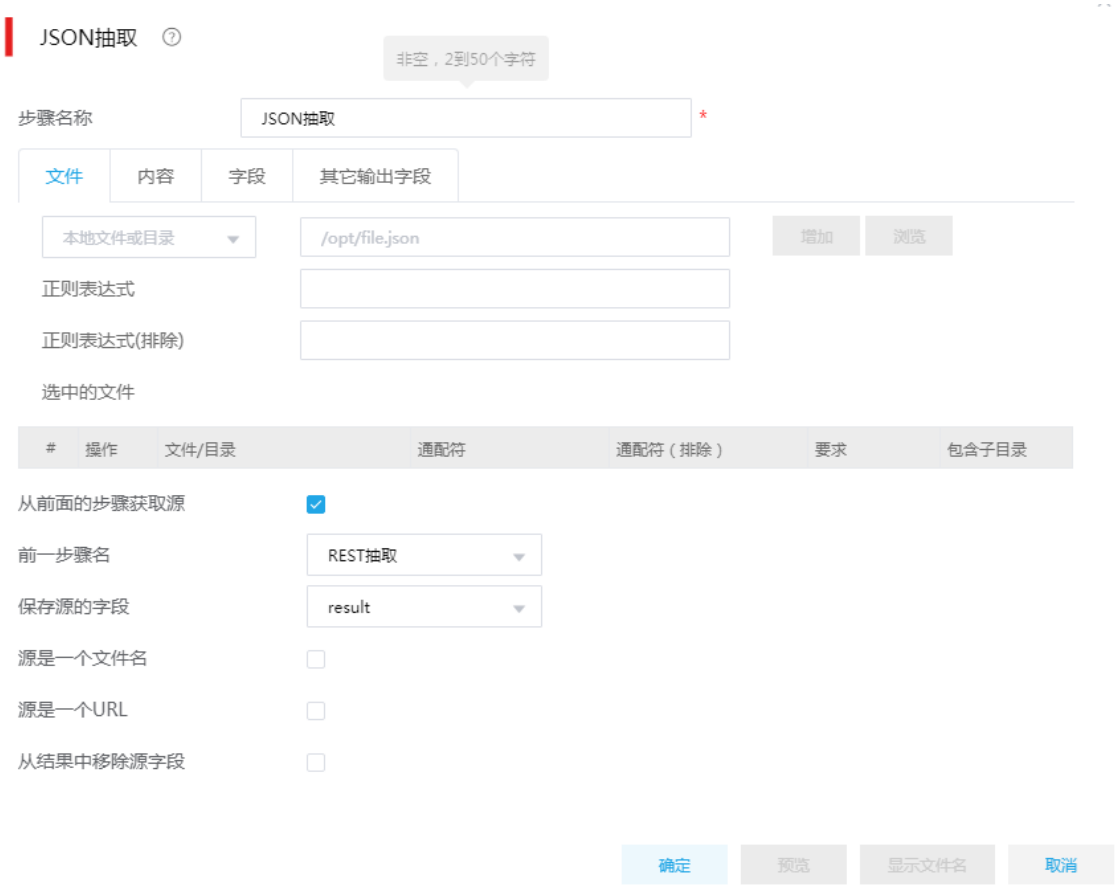
图4-76 REST 抽取-HTTP 头部

按照待抽取的 REST 接口定义，在组件的 HTTP 头部页签中，需要添加 appKey。

(5) JSON 抽取

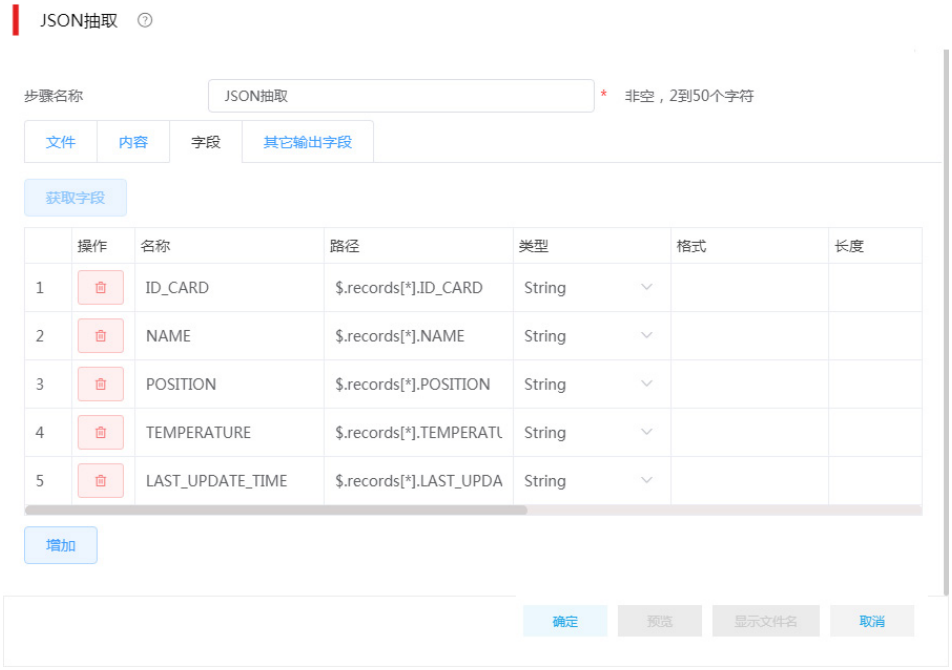
选择从 REST 抽取步骤中接收数据（JSON 字符串），然后解析出各个字段的值，并将解析结果发送给加载至表组件。

图4-77 JSON 抽取-文件页签



在文件页签下，勾选从前面的步骤获取源，选择前一步骤为“REST 抽取”，选择保存源的字段为“result”，该字段名称在 REST 抽取组件的“输出字段”页签下配置，可以手动修改。

图4-78 JSON 抽取-字段页签



在字段页签下，添加 REST 请求返回结果中包含的各个字段，根据接口返回的 JSON 格式数据示例，我们这里添加 5 个待解析字段，每个字段需要填写字段名称、路径、类型。其中字段名称与示例数据保持一致，与最终加载到数据库表的列名对应，路径则是表名该字段在 JSON 字符串中的层级定位，遵循标准的 JsonPath 定义，类型则统一使用 String，后面若需要用到其他类型，也可以结合字段选择等转换组件做处理。

(6) 加载至数据库表

将 JSON 解析出来的数据加载至 PostgreSQL 数据库中 hh_1000 表中。

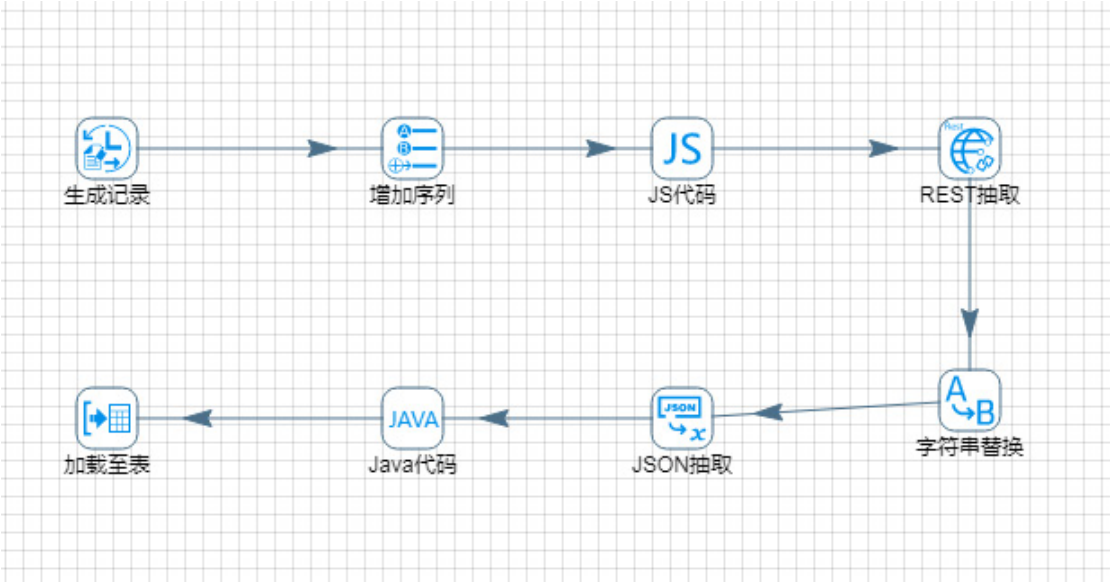
图4-79 加载至数据库表具体配置图示

(7) 设计好任务后，单击<运行>按钮，根据我们之前的设置会发送 100 次 REST 请求，每次请求抽取 1000 条数据。如果前一天更新的数据总数少于 10 万条，则当前设置即可满足抽取前一天修改的全部数据，如果大于 10 万条，我们还需调整生成记录组件的生成数量、以及增加序列中的最大值。

6. 补充场景

根据现场接口变化，REST 接口返会的 JSON 数据中每个“”“ 前都会有“\\”字符，需要替换掉才能进行 JSON 解析；同时，某些隐私字段会进行加密处理，比如：证件号码、住址以及电话号码等。因此，根据以上场景需求，需要进行设计补充，如图 4-80 所示。

图4-80 补充场景设计图



(1) “字符串替换”组件

该组件用于替换 REST 接口返回的 JSON 数据中多余的符号“\\”，使返回数据符合 JSON 解析格式。图 4-81 是该组件具体的配置信息

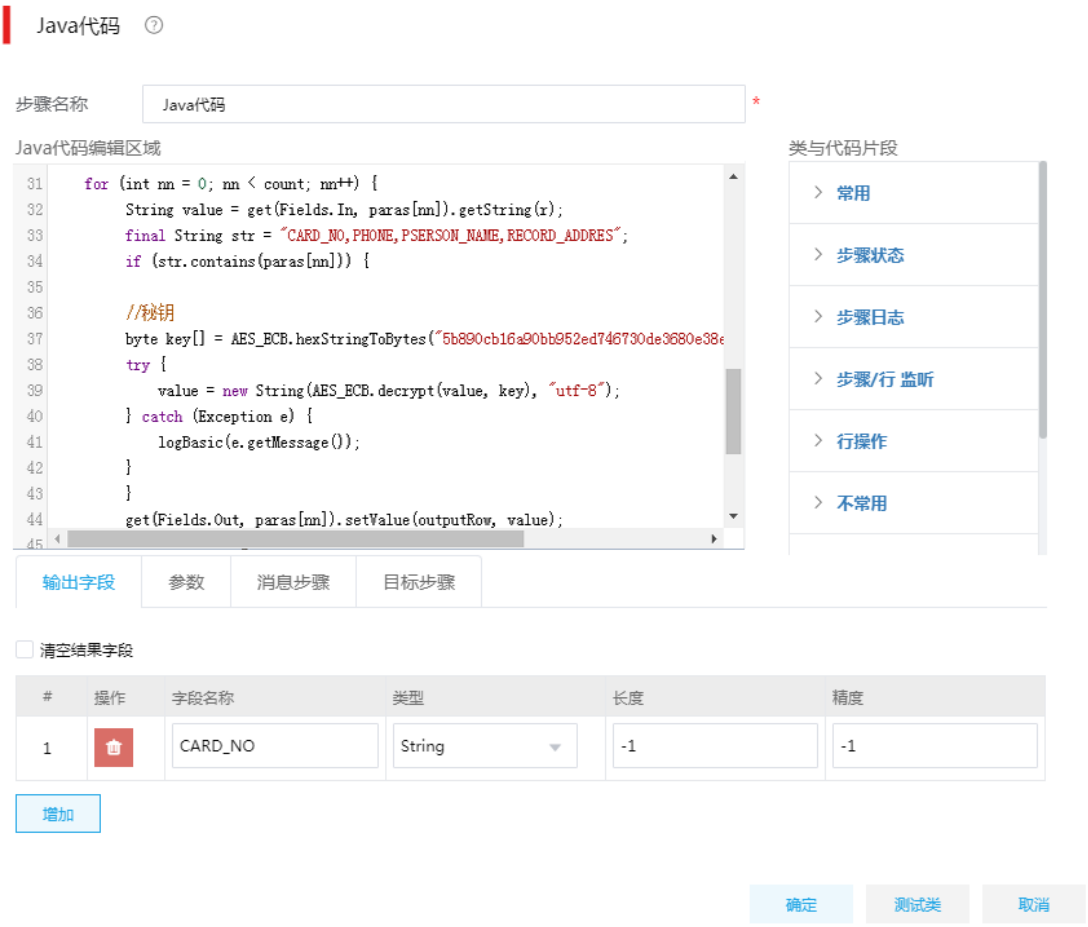
图4-81 字符串替换组件配置

(2) “Java 代码”组件

Java 代码组件主要是对“JSON 抽取”组件输出的字段中，需要进行解密的字段进行解密操作；实现方式可以有多种方式，可以通过引入 REST 接口方提供的加解密工具包，也可以自己通过写代码实现加解密工具包（一般都是需要第三方工具包导入 DI 包路径中，重启服务即可直接使用）。

图 4-82 是“java 代码”组件的具体配置页面，我们只需要按照组件的要求规范在“Java 代码编辑区域”根据实际需求编写 java 代码实现对字段的处理即可。

图4-82 Java 代码组件配置页面



4.10 外部调用触发数据集成作业下发

1. 场景描述

A 公司通过数据集成设计了 ETL 任务，希望通过外部 API 调用的方式触发作业的下发，这里可以通过服务集成函数 API 调用作业下发的接口来触发，函数 API 设计完成后可以部署到 API 网关对外提供使用。

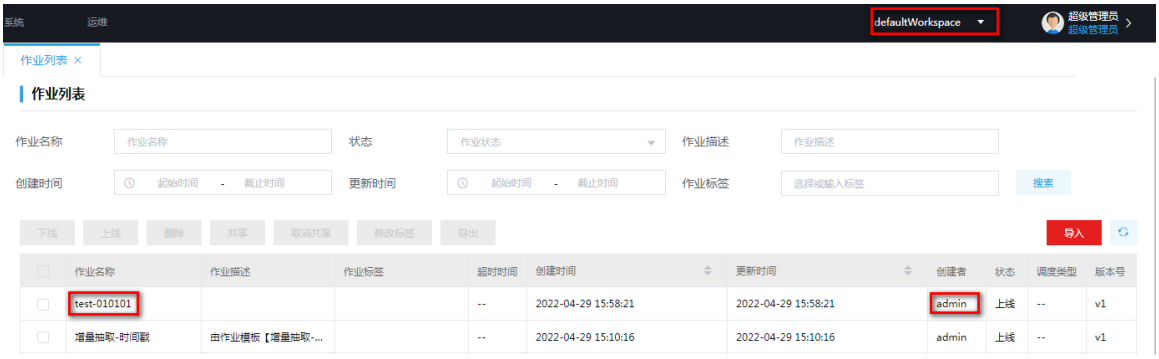
2. 场景分析

数据集成可以设计 ETL 任务，设计完成后通过服务集成函数 API 调用作业下发接口。

3. 示例前置条件

数据集成中已设计完成一个可正常运行的作业。记录作业名称、作业所属的工作空间及创建人，创建 API 编写 JS 脚本时需要指定工作空间名称，当 API 创建完成后，用户可通过作业名称、创建人及工作空间名称对作业进行调度。

图4-83 记录作业名称、创建者及工作空间名称



在[服务集成/API 工厂/环境配置/环境变量]页面新增环境变量。其中 key 配置为“openIp”,Value 配置为数字平台的虚 IP。后续编写 JS 脚本时，需要使用该变量获取接口 IP。

图4-84 创建环境变量

4. 示例详细步骤

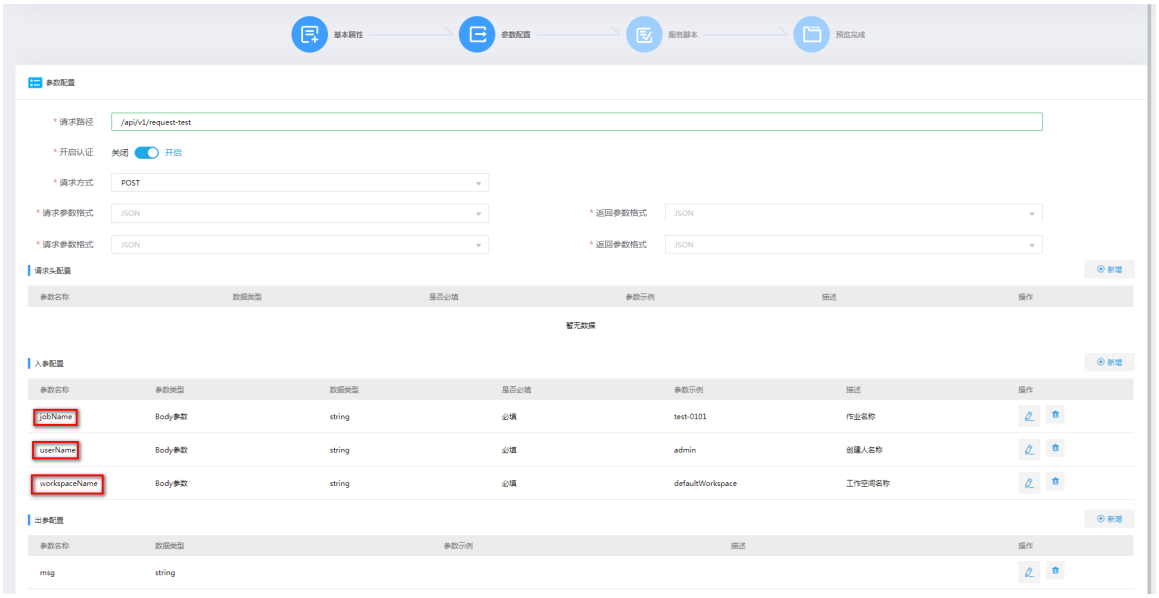
(1) [服务集成/API 工厂/API 管理]页面，单击<API 注册>按钮，选择注册类型为“函数 API”，进入函数 API 设计页面，配置函数 API 基本属性。

图4-85 配置函数 API 基本属性

(2) 配置完基本属性后，单击<下一步>按钮，进入函数 API 的参数配置页面。部分参数配置说明如下：

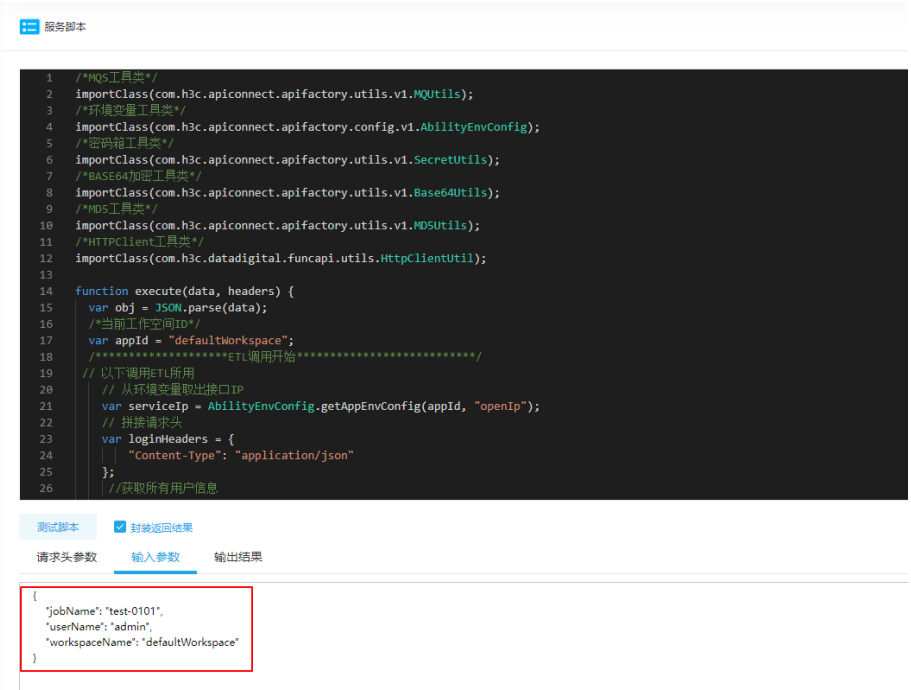
- 请求路径：用户可根据实际需要自定义该请求路径。以“/”开头的自定义访问路径，且只能包含字母、数字、“-”、“_”“.”、“/”，如“/api/v1.0/api-test”。
- 开启认证：默认是开启认证的，开启认证后，用户在授权时需选择静态认证或动态认证；如果不开启认证，则在授权时无需选择认证方式，用户在实际使用该 API 时也无需进行认证，用户可根据实际需要进行配置。
- 请求方式：选择接口的请求方式，目前支持 POST 和 GET，用户可根据实际需要进行配置。
- 入参配置：入参配置需配置作业名称、创建人名称及工作空间名称，如图 4-86 所示。
  - jobName：作业名称。用户后续调用该 API 时需通过传入作业名称、作业创建人名称及工作空间名称调用指定的作业。
  - userName：作业创建人名称。用户后续调用该 API 时需通过传入作业名称、作业创建人名称及工作空间名称调用指定的作业。
  - workspaceName：工作空间名称。用户后续调用该 API 时需通过传入作业名称、作业创建人名称及工作空间名称调用指定的作业。
- 出参配置：配置出参的字段名和数据类型等信息，需根据所设计的 API 返回值进行填写。

图4-86 参数配置



(3) 单击<下一步>，编写 JS 脚本，调用数据集成作业下发接口。

图4-87 编写 JS 脚本。



(4) JS 脚本编写示例如下。

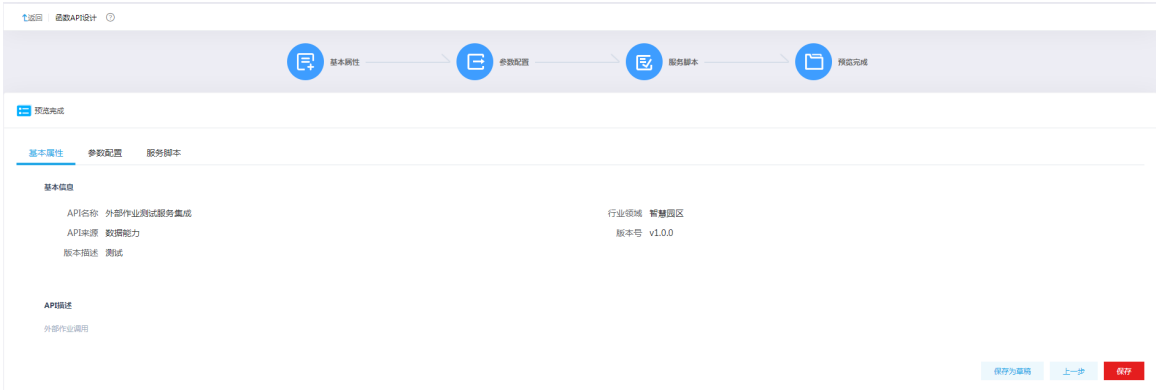
```
function execute(data) {
    var obj = JSON.parse(data);
  /*当前工作空间ID*/
  var appId = "defaultWorkspace";
  /*******************ETL调用开始*************************/
 // 以下调用ETL所用
    // 从环境变量取出接口IP
    var serviceIp = "127.0.0.1"
     // 拼接请求头
    var loginHeaders = {
        "Content-Type": "application/json"
    };
     //获取所有用户信息
    var get_all_users = "http://" + serviceIp + ":32007/users";
     // 发送get请求，传入url,入参，请求头
    var result = HttpClientUtil.sendGetRequestAddHeader(get_all_users, "UTF-8", null, loginHeaders);
    var users = com.alibaba.fastjson.JSON.parse(result).data;
```

```
//遍历所有用户找到传入的用户名
var userName = data.userName;
var user = {};
//循环获取匹配运行用户,使用已经存在的用户名
for (var num = 0; num < users.length; num++) {
   var tempUser = users[num];
   if (userName == tempUser.name) {
      user = tempUser
      break;
   }
}
 // 解析返回结果
var user = [{
      "userId": user.id,
      "roleName": user.roleName,
      "userName": user.name,
      "roleId": user.role_id,
      "projectName": user.projectName,
      "workspacename":data.workspaceName,
      "projectId": user.defaultProjectId
}];
 // 拼接根据作业名获取作业id接口url
 var getJobIdUrl="http://" + serviceIp + ":20005/openapi/v1/process/list-paging?pageNo=1&pageSize=10";
 // 封装请求头
 var getJobIdHeaders = {
      "userinfo": JSON.stringify(user),
      "Content-Type": "application/json;charset=UTF-8"
};
 // 封装请求参数
 var search = {
"name": data.jobName
};
 // 发送post请求
 var
```

```
getJobIdResult=HttpClientUtil.sendPostRequestAddHeader(getJobIdUrl,JSON.stringify(search),getJobIdHeaders);
    // 解析返回结果,由于返回data是数组，需要用alibaba的fastjson来解析
    var JobIdResult = com.alibaba.fastjson.JSON.parse(getJobIdResult);
    // 拿到作业code
    var jobCode=JobIdResult.data.records[0].code;
   // 拼接下发作业url
   var sendjobUrl = "http://" + serviceIp + ":20005/openapi/v1/process/start-process-instance";
    // 拼接下发作业请求头
   var jobHeaders = {
        "userinfo": JSON.stringify(user),
        "Content-Type": "application/json"
   };
   //封装作业下发command
    var command = {
        "commandType": "START_PROCESS",
        "taskDependType": "TASK_POST",
        "failureStrategy": "CONTINUE",
        "processInstancePriority": "MEDIUM",
        "notifyType": "EMAIL",
        "processDefinitionCode": jobCode
   };
    // 发生请求
   var sendJob_result = HttpClientUtil.sendPostRequestAddHeader(sendjobUrl, JSON.stringify(command), jobHeaders);
    // 返回结果
   return (JSON.parse(sendJob_result));
}
```

(5) 单击<下一步>，查看配置的函数 API 的整体信息。

图4-88 查看 API 整体信息



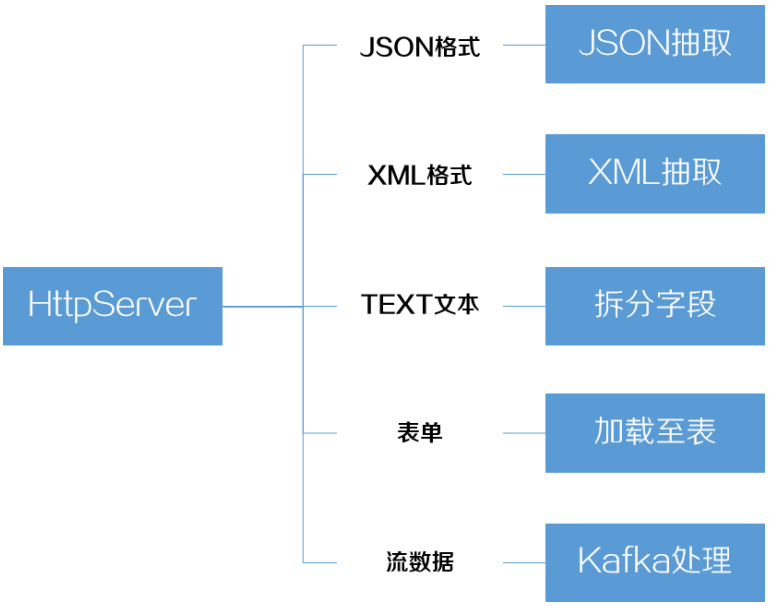
(6) 确认配置无误后，单击<保存>按钮，返回 API 管理页面。在 API 管理页面可以看到新增的 API，接下来按照 API 的通用操作进行测试、部署、授权等操作，可参考 5.1　3. (8)中数据 API 的操作步骤。

(7) API 部署并授权成功之后，用户通过调用该 API，传入作业名称、作业创建者名称及工作空间名称即可对指定的数据集成作业进行调用。注意：外部作业调用的时候一定是已上线的作业。

4.11 HttpServer使用场景介绍

HttpServer 目前可支持 Text、Json、Xml、form-data、流数据类型的数据处理，整体的处理方案如下。下面只是示例性展示常用的几种场景，实际可根据自己需求设计对应的任务流程。

图4-89 处理方案



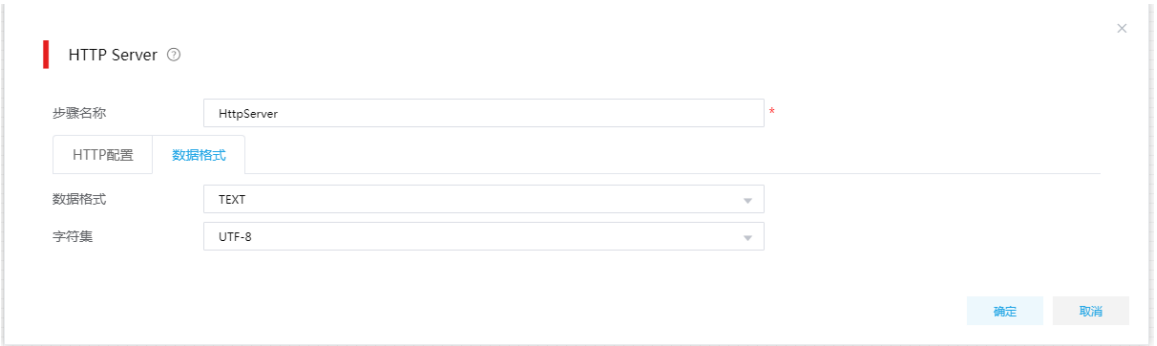
1. 配置 HttpServer

新建一个作业，将 ETL 任务拖拽到作业设计画布中，然后打开 ETL 任务设计器画布，在“数据抽取组件”中选择“HTTP Server”组件并拖拽到画布中。

根据需求配置对应参数，包括“HTTP 配置”和“数据格式”两项配置。“HTTP 配置”配置各项 HTTP 服务端参数；“数据格式”指定该接口接收的数据格式类型和字符集。

图4-90 配置各项 HTTP 服务端参数

图4-91 指定接口接收的数据格式类型和字符集

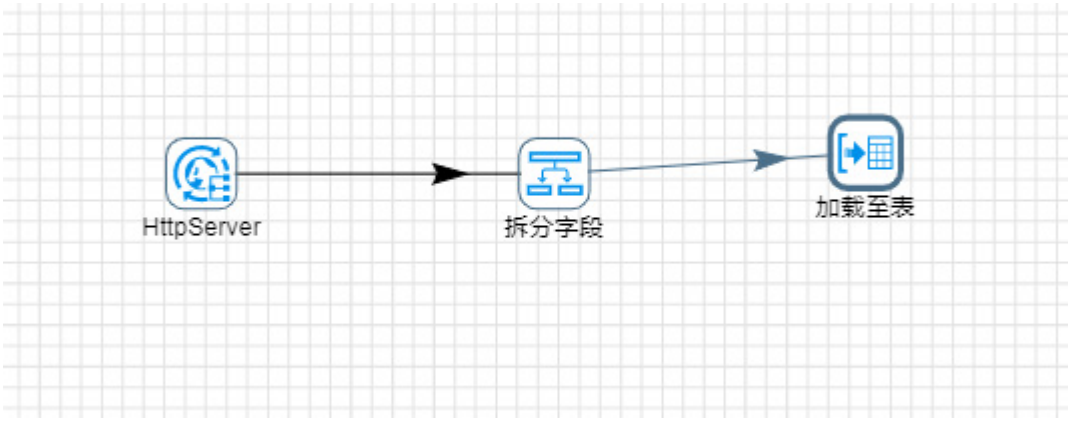


配置完 HTTP Server 后，需要根据其配置的数据格式选择后续相应的组件进行数据处理。

2. 文本类型

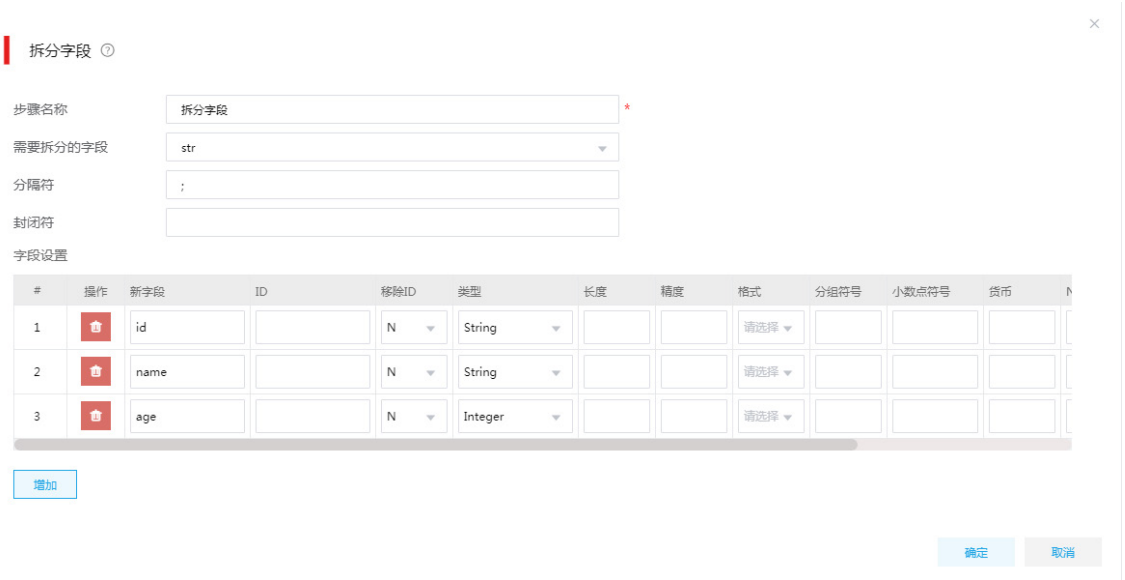
对于文本类型数据，目前只支持固定分隔符分割的数据，例如“111;222;333”对应三个字段，按照“;”符号进行分割；后面可连接“拆分字段”组件，按照分隔符配置对数据进行分割，并赋值给配置的字段。总体 ETL 任务流程设计如下：

图4-92 文本类型数据处理流程



对应的“拆分字段”组件配置如下：

图4-93 拆分字段组件配置

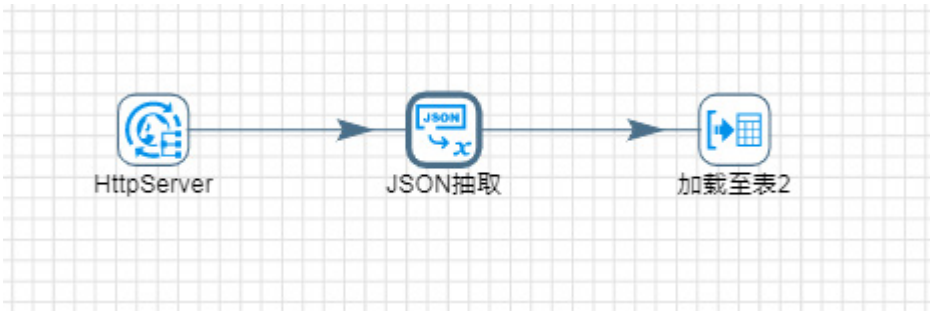


最终，数据可以使用“加载至表”或者“加载至文件”等对数据进行落地处理（根据实际需要选择对应的加载组件落地数据即可）。

3. JSON 类型

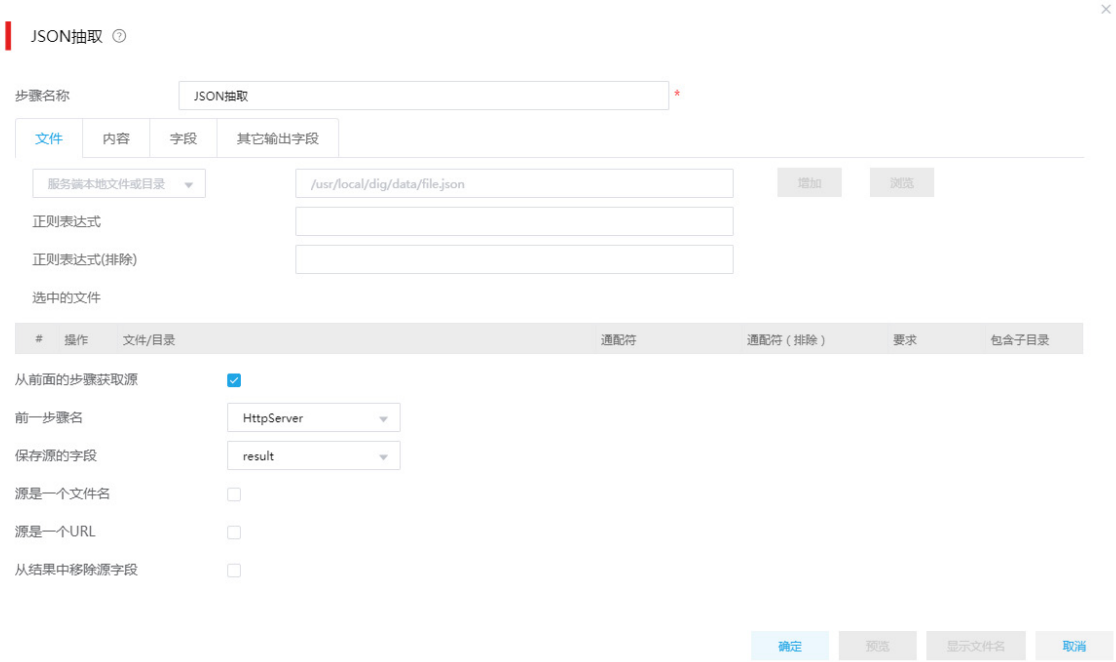
JSON 类型数据处理时，HTTP Server 组件只需要将“数据格式”中配置为“JSON”即可。后续流程需要连接“JSON 抽取”组件对接收数据进行处理，总体流程如下：

图4-94 JSON 类型数据处理流程



“JSON 抽取”组件配置如下（抽取来源需要配置为“从前面的步骤获取源”，不可以配置为从文件中抽取）：

图4-95 JSON 抽取组件配置

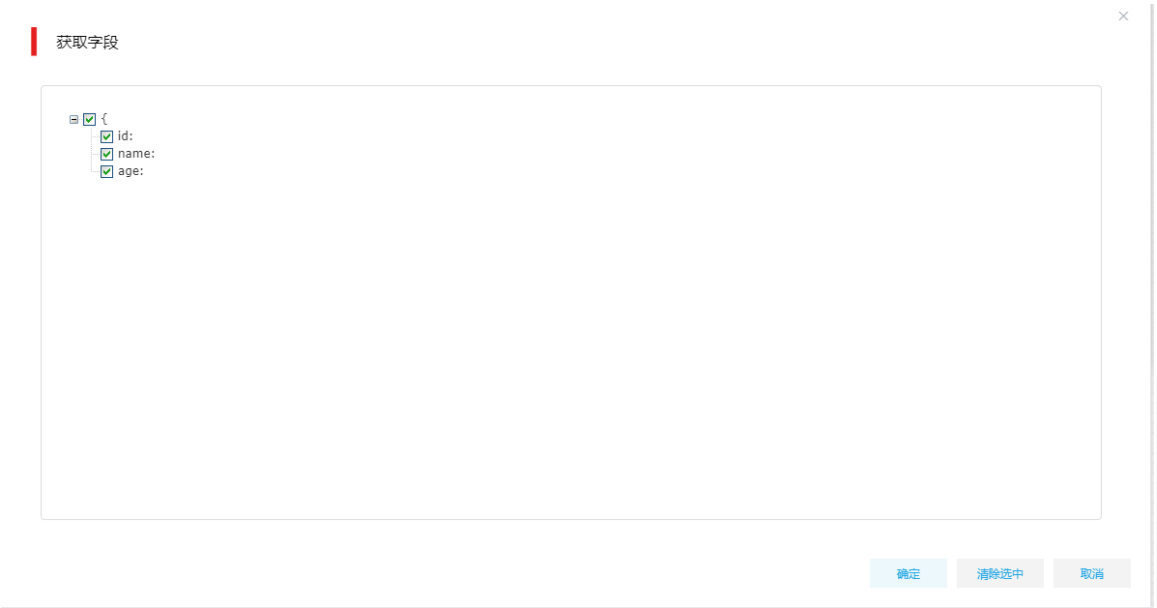


对于字段配置，可以通过数据样例自动解析字段，具体如下：

图4-96 字段配置

图4-97 获取字段具体配置

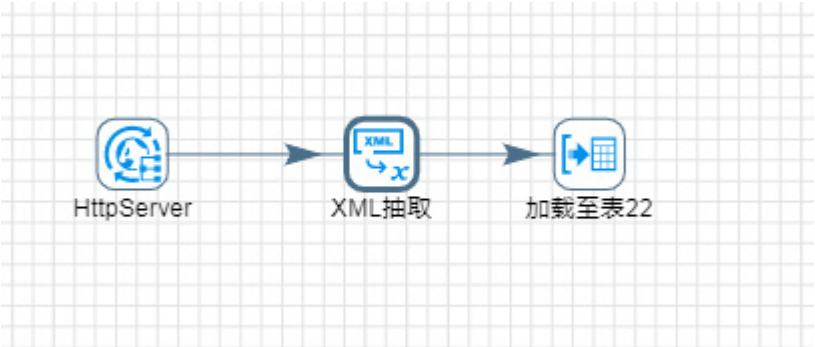
图4-98 JSON 内容字段选择



4. XML 类型

HTTP Server 组件后连接“XML 抽取”组件，用于解析接收到的 XML 数据。总体流程如下：

图4-99 XML 类型数据处理流程



XML 抽取组件在“文件”中配置“XML 源定义在一个字段里”，然后，在“内容”中“获取 XML 文档的所有路径”，输入数据样例。

图4-100 输入 XML 示例

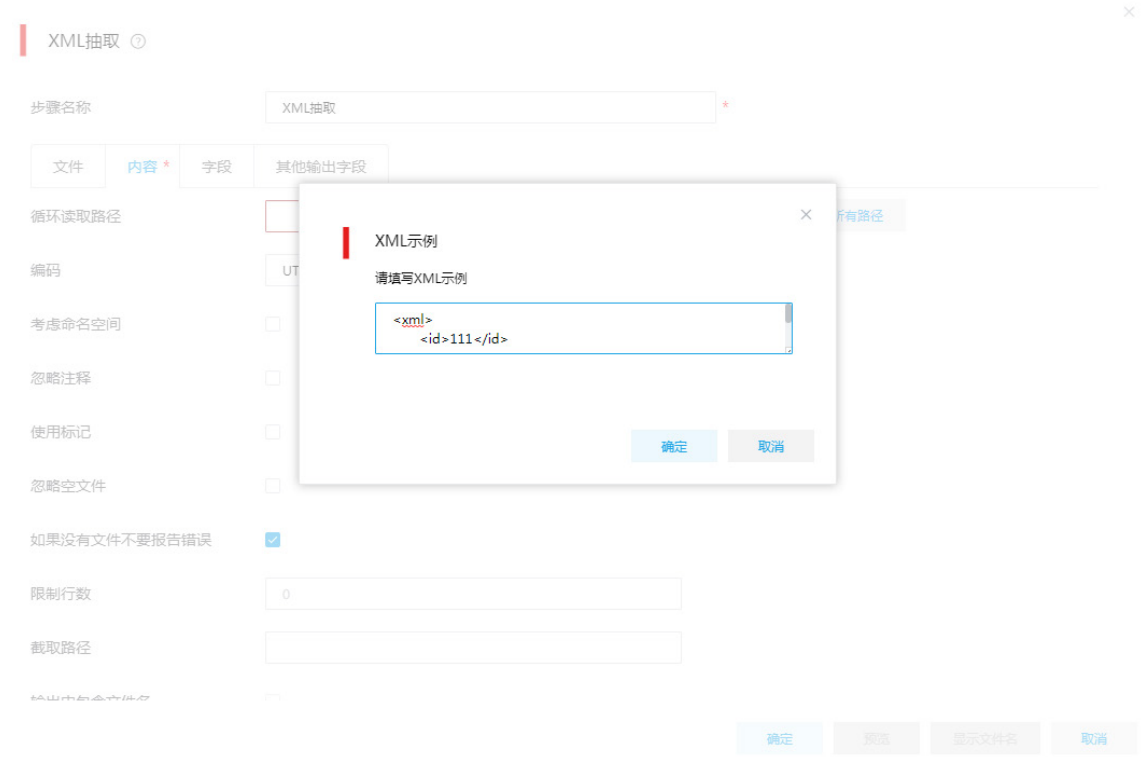
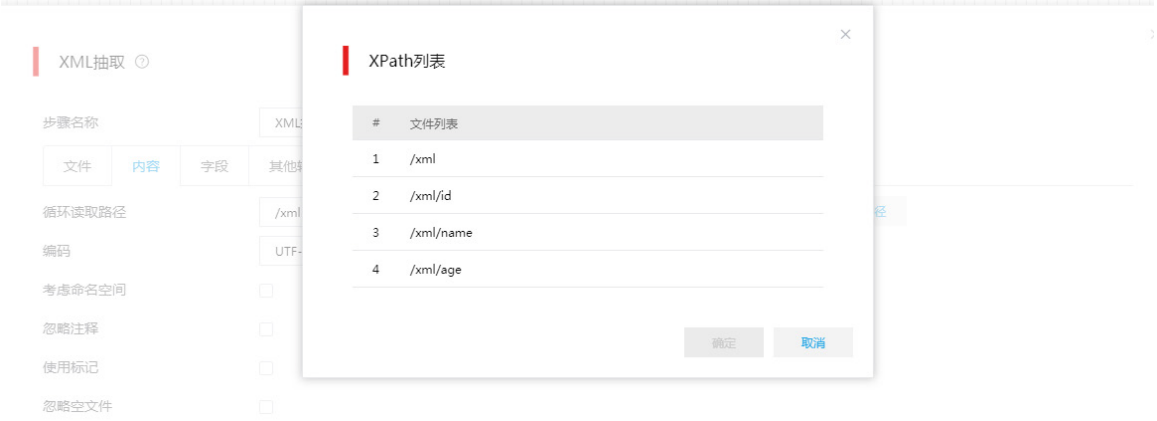


图4-101 XPath 展示



“字段”中单击“获取字段”，输入样例数据获取数据字段。

图4-102 获取字段

图4-103 获取字段 XML 示例

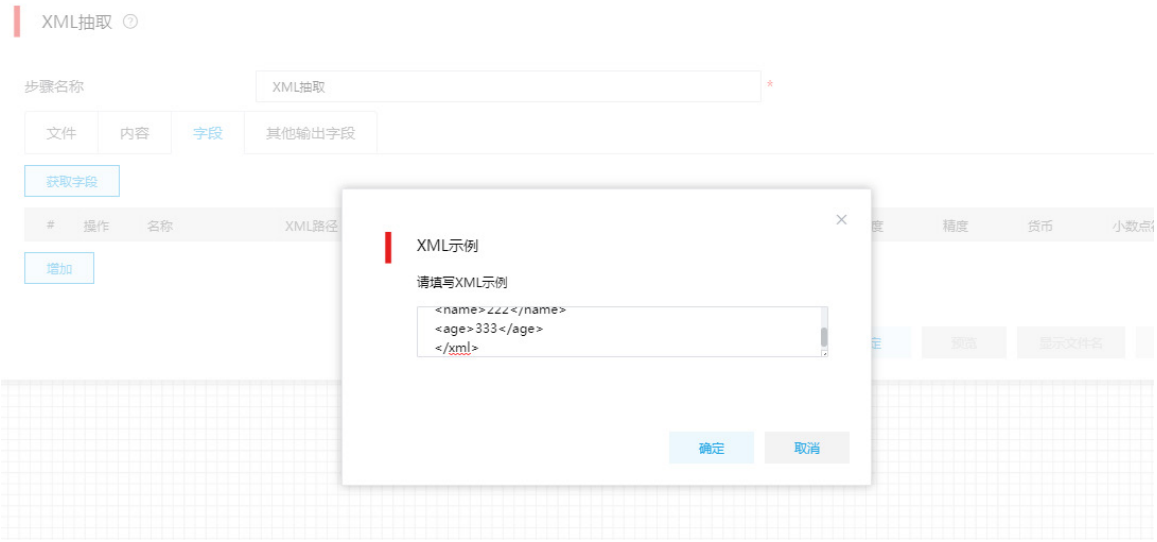
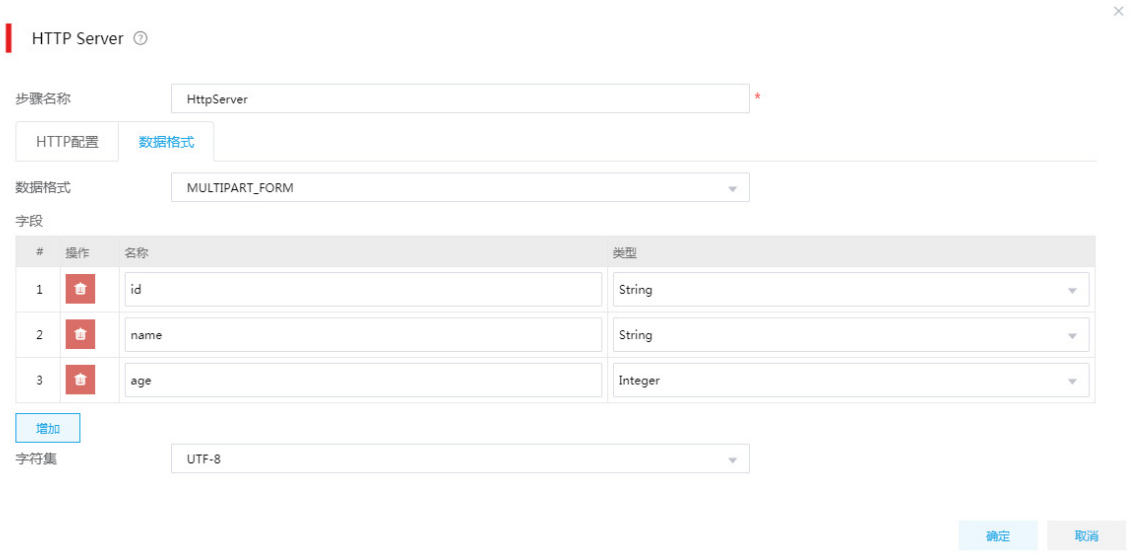


图4-104 XML 抽取获取字段列表

5. form-data（表单类型）

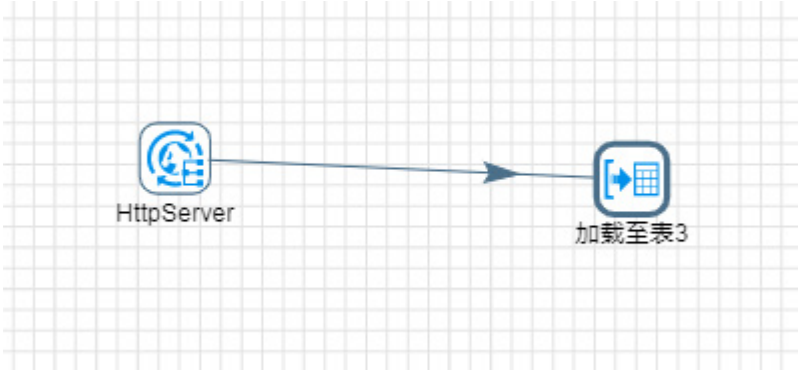
表单类型数据需要直接在 HTTP Server 中配置表单的参数列表即可。

图4-105 配置表单参数列表



配置好表单字段后，后续步骤可以根据实际需要选择对应的组件，无需连接类似于 JSON 抽取的组件进行数据解析（表单数据直接在 http 请求处理时解析完成）；总体流程如下：

图4-106 表单数据处理流程

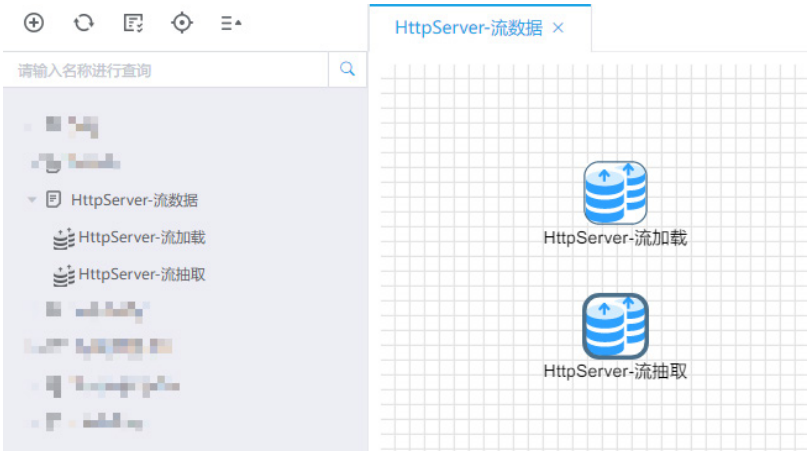


6. 流数据类型

HttpServer 获取的数据需要入库时，直接连接“表加载”在数据推送时间不确定时可能会导致“表加载连接丢失”，建议使用 Kafka 作为数据中转，先使用“Kafka 流加载”将数据推送到 Kafka 指定 Topic 中，再用“Kafka 流抽取”将数据抽取并加载至对应表中。

(1) HTTP Server 处理流数据由流加载、流抽取两个 ETL 任务组成。

图4-107 HTTP Server 处理流数据作业



(2) 流加载任务：HttpServer 参数配置完成后，直接连接“Kafka 流加载”组件。“Kafka 流加载”组件配置 Kafka 数据源、主题等参数。当 HttpServer 推送数据时，直接写入组件配置的主题中，等待消费。

图4-108 流加载任务

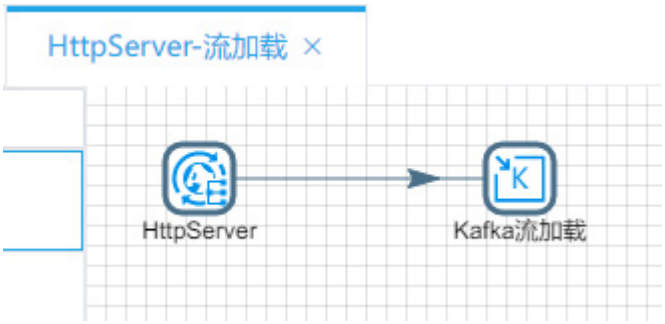
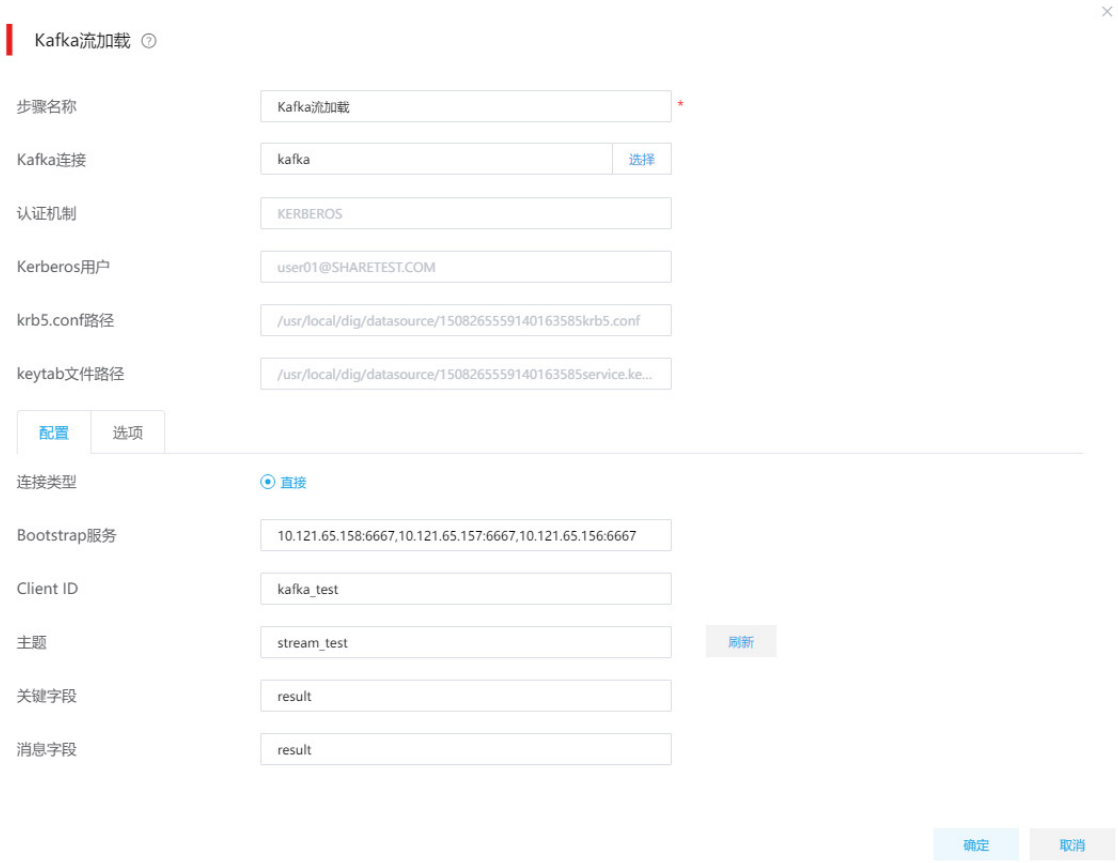


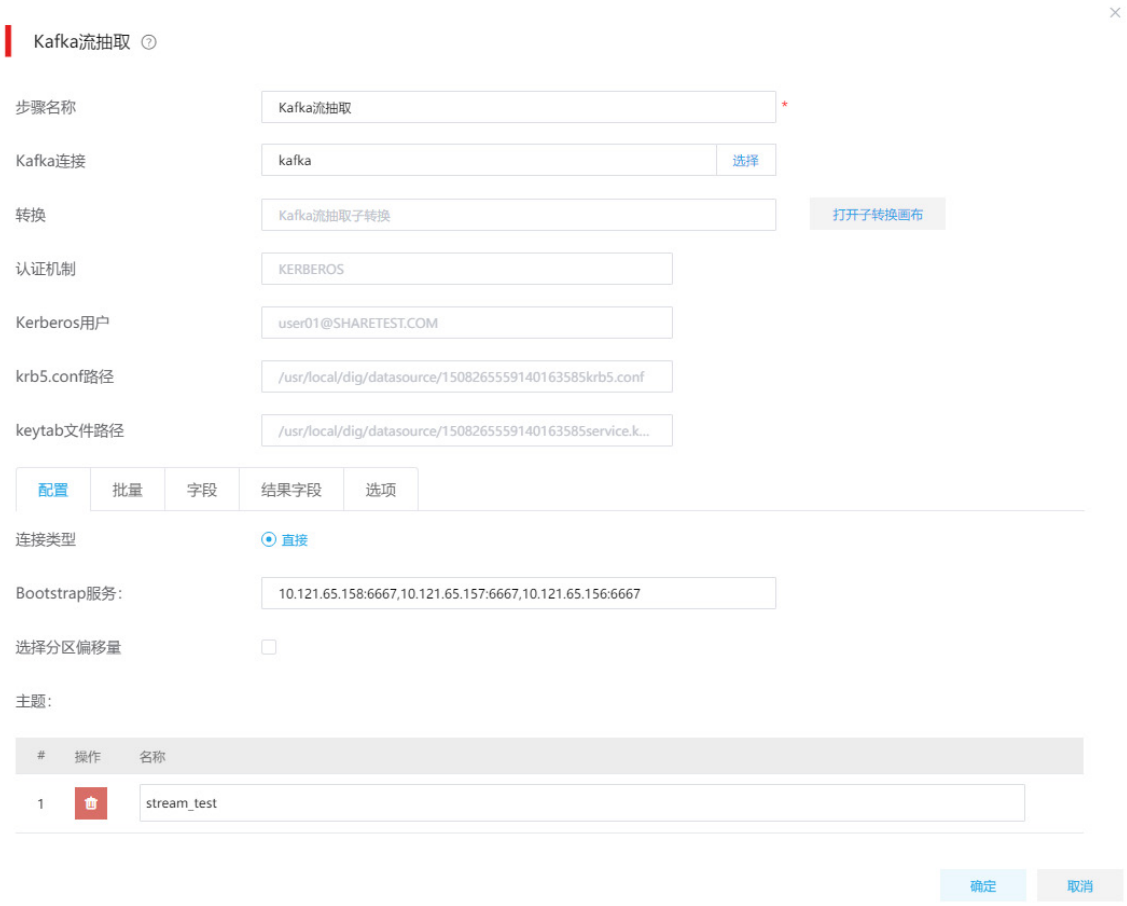
图4-109 Kafka 流加载配置



(3) 流抽取任务：仅由一个“Kafka 流抽取”组件组成。配置 Kafka 数据源、主题等参数，在“子转换画布”中配置“从流中获取记录”，选择对应字段，并连接相关的数据处理组件即可。

图4-110 流抽取任务

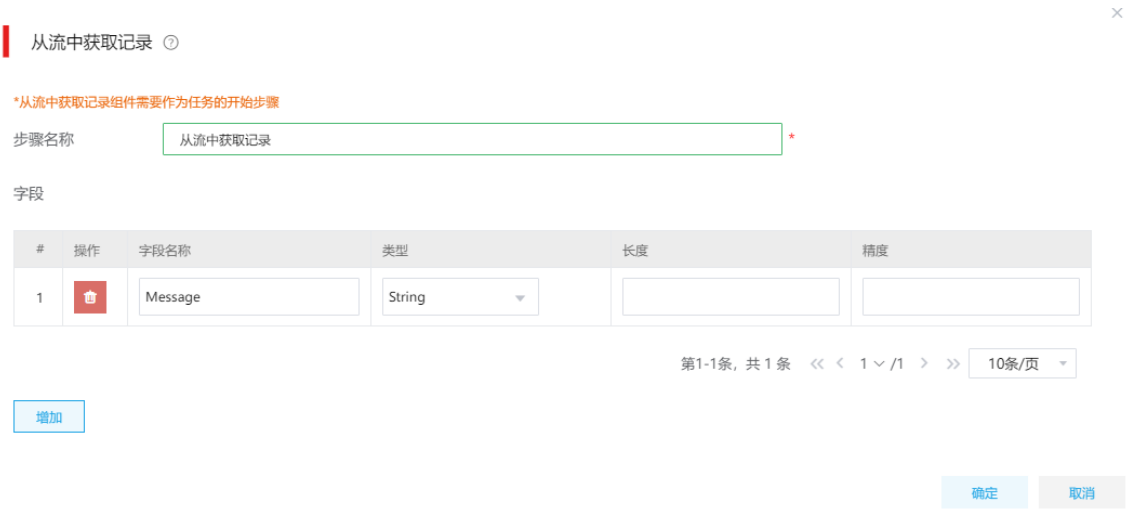
图4-111 流抽取配置



(4) 子转换画布任务设计如下所示，主要从 Topic 中消费数据并进行处理，最终加载至表。

图4-112 子转换画布任务设计

图4-113 从流中获取记录



4.12 插入更新

插入更新组件利用查询关键字在表中搜索行。如果未搜索到，则插入该行。如果可以搜索到，并且与要更新的字段相同，则不执行任何操作；如果不完全相同，则更新表中的行。

1. 场景描述

对于增量数据既要新增又要插入操作同一张表时，需要使用插入更新组件。

2. 场景分析

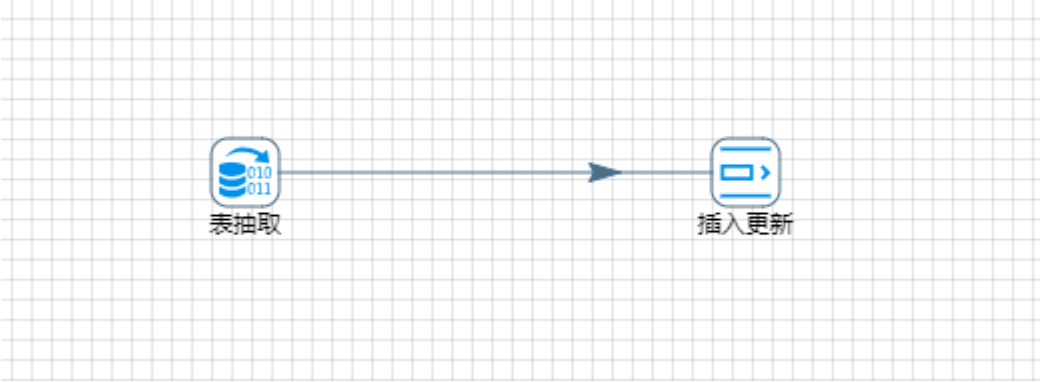
A 院校需要维护一张学生信息表，院校根据学生信息的变化定时更新这张表，新来的学生要把信息新增到这张表中，在校学生信息发生变化时，需要更新这张学生信息表对应的信息。

3. ETL 设计方案

数据流向：表抽取—>插入更新。

ETL 方案：用表抽取组件获取增量数据，用插入更新组件同步到学生信息表，根据学生学号进行对比，学生信息表中没有，则插入更新组件就会新增该条数据；学生信息表中已存在其他字段有变动，插入更新组件就会更新该条数据，没有变动插入更新不做任何操作。

图4-114 ETL 任务设计图示



4. 示例前置条件

增量原表 student 和目标表 student_info 已存在，数据源已经在数字平台[数据源管理]页面创建好。

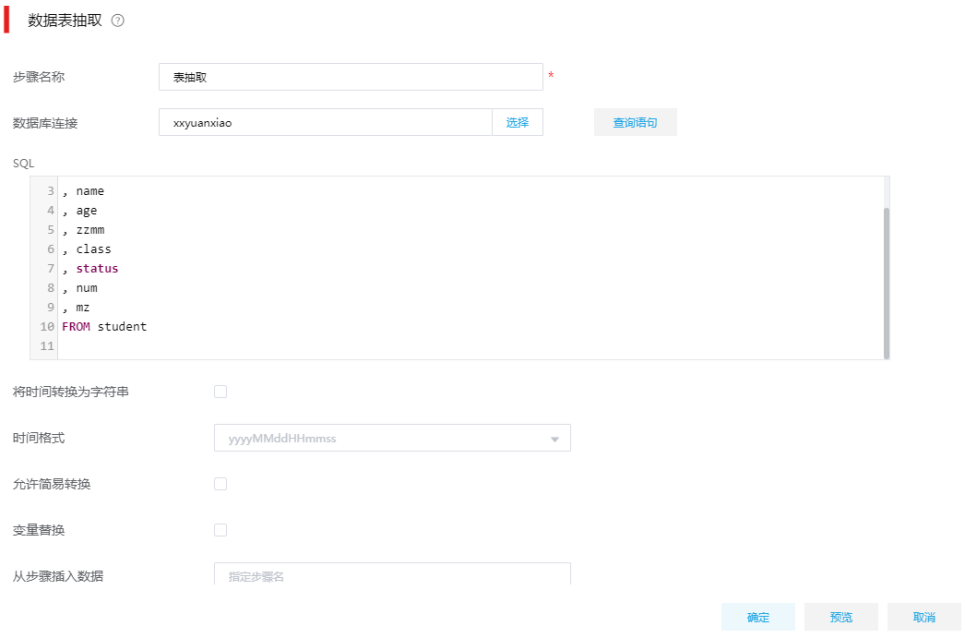
图4-115 示意数据源 student 表

图4-116 示意数据源 student_info 表

5. 示例详细步骤

(1) 第一步：表抽取组件配置，配置进行数据抽取的数据库，以及需要抽取的字段。

图4-117 表抽取组件配置



(2) 第二步：配置需要进行插入更新的数据源连接及表，用表抽取组件获取数据，用插入更新组件同步到学生信息表。

- “查询字段”页签：如图 4-118 所示，通过指定主键字段 sno 在目标表中进行查询，查询原表的 sno 数值在目标表中是否存在。
- “更新字段”页签：如图 4-119 所示，根据学生学号进行对比，学生信息表中没有，则插入更新组件就会新增该条数据；学生信息表中已存在，其他字段有变动，插入更新组件就会更新该条数据，没有变动插入更新不做任何操作。

图4-118 插入更新组件“查询字段”配置

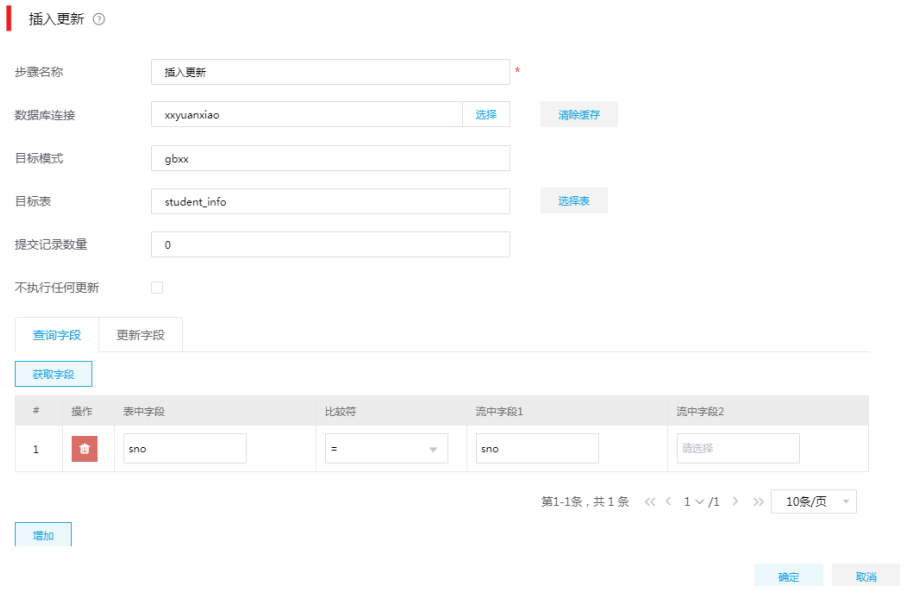
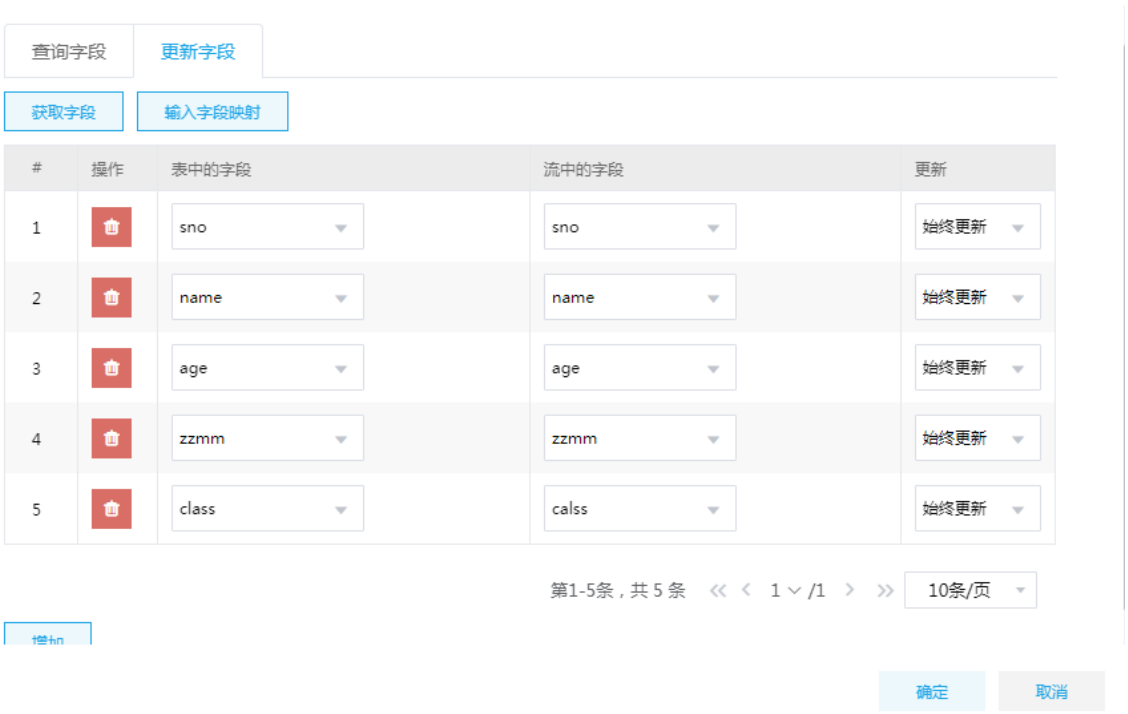


图4-119 插入更新字段“更新字段”配置



(3) 任务设计完成后，单击<运行>，等待数据同步完成，同步完成后 student_info 表新增一条 sno=“20225520279”学生信息，并且 sno=“20205520147”学生的政治面貌由 zzmm=“共青团员”更新成了 zzmm=“党员”。

图4-120 示意数据源 student_info 表

4.13 实时作业（Oracle到Vertica）

1. 场景描述

A 公司基于 Vertica 的数仓建设，需要将 Oracle 数据库中历史数据和增量数据实时同步至 Vertica 数仓用于数据分析。

2. 场景分析

通常数仓建设会将源库中的所有业务表同步至数仓，源 CDC 组件可以模糊匹配某个模式下的所有表，通过源 CDC 可以方便的配置监控 Oracle 库下的所有表，指定并行度后实现全量并行抽取历史数据并且无缝转增量抽取。

3. 实时作业设计方案

通过源 CDC 组件抽取 Oracle 数据库中历史数据和增量数据，然后通过 JDBC 加载组件将数据实时同步至 Vertica 数仓。

图4-121 实时作业图示

4. 示例前置条件

- Oracle 数据源已添加，测试连接通过并且用户有查看归档日志的权限。
- Oracle 库已开启日志归档并且启用 logminer。
- Vertica 数据源已添加并测试连接通过。

5. 示例详细步骤

(1) 配置源 CDC 相关参数

a. 基本配置页签：配置 Oracle 数据库连接，抽取方式选择“增量+全量”，根据抽取的数据量选择推荐的批量大小和并行度配置，如果抽取的表中包含 blob、clob、xmltype 等大字段时建议调小批量大小，批量大小建议不超过 100。

图4-122 基本配置

b. 抽取的表页签：选择需要抽取的模式和表，其中%代表该模式下的所有表。

图4-123 抽取的表

(2) 配置转换组件相关参数

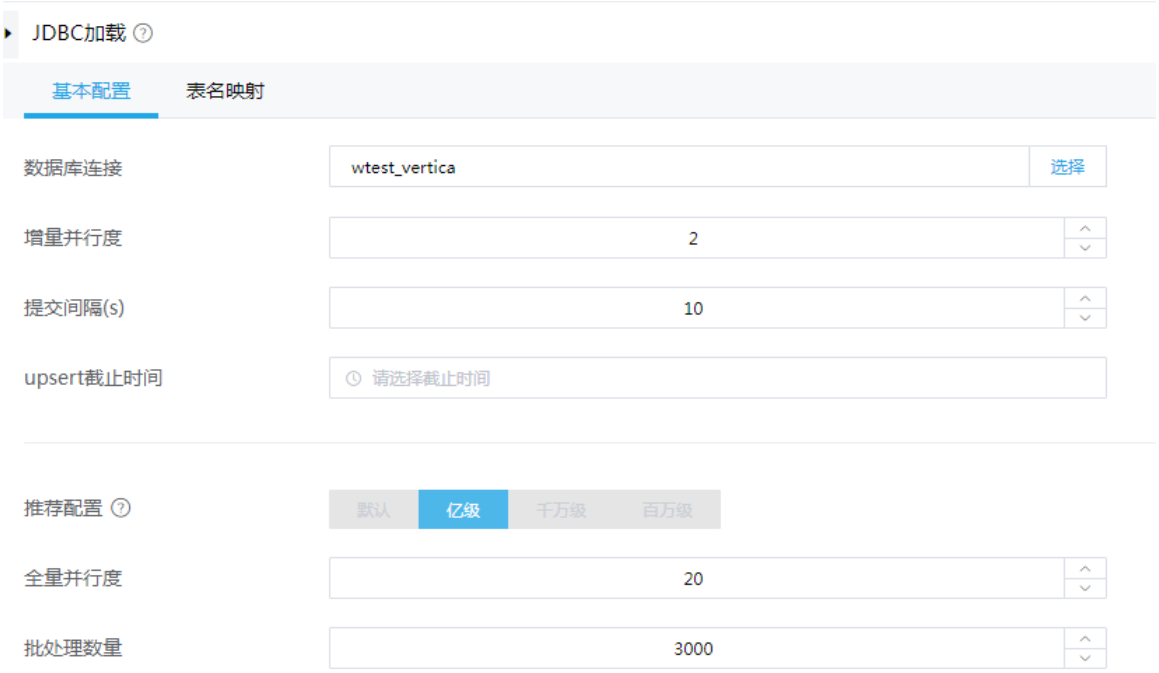
数据标签页签下对源库中抽取出来的数据增加标签字段。表达式 option 代表操作类型(INSERT、DELETE、UPDATE)；original time 代表源库数据变更的时间；now time 代表当前数据抽取出来的时间。

图4-124 数据标签配置

(3) 配置 JDBC 加载配置参数

a. 基本配置页签：配置数据库连接，增量并行度、提交间隔等参数。

图4-125 基本配置



b. 表名映射页签：当源库中模式名和表名与目标库中模式名和表名不一致时，需使用该配置项配置映射关系。

- 配合源模式、源表名与目标模式、目标表名的匹配关系。源表名和目标表名支持通过*进行匹配，*代表源表名和目标表名相同的部分。
- 自动建表如果选择是，则当目标表名匹配不到时会自动在目标库创建表。

图4-126 表名映射配置

(4) 配置完成后，保存作业，然后上线、运行即可开始抽取数据。

4.14 实时作业（SQL Server到Vertica）

1. 场景描述

A 公司基于 Vertica 的数仓建设，需要将 SQL Server 数据库中历史数据和增量数据实时同步至 Vertica 数仓用于数据分析。

2. 场景分析

通常数仓建设会将源库中的所有业务表同步至数仓，源 CDC 组件可以模糊匹配某个模式下的所有表，通过源 CDC 可以方便的配置监控 SQL Server 库下的所有表，实现全量+增量抽取历史数据。

3. 示例前置条件

- SQL Server 数据源已添加，对要抽取的库开启 CDC，同时对要抽取的表开启 CDC。
- Vertica 数据源已添加并测试连接通过。

4. 实时作业设计方案

通过源 CDC 组件抽取 SQL Server 数据库中历史数据和增量数据，然后通过 JDBC 加载组件将数据实时同步至 Vertica 数仓。

图4-127 实时作业图示

5. 示例详细步骤

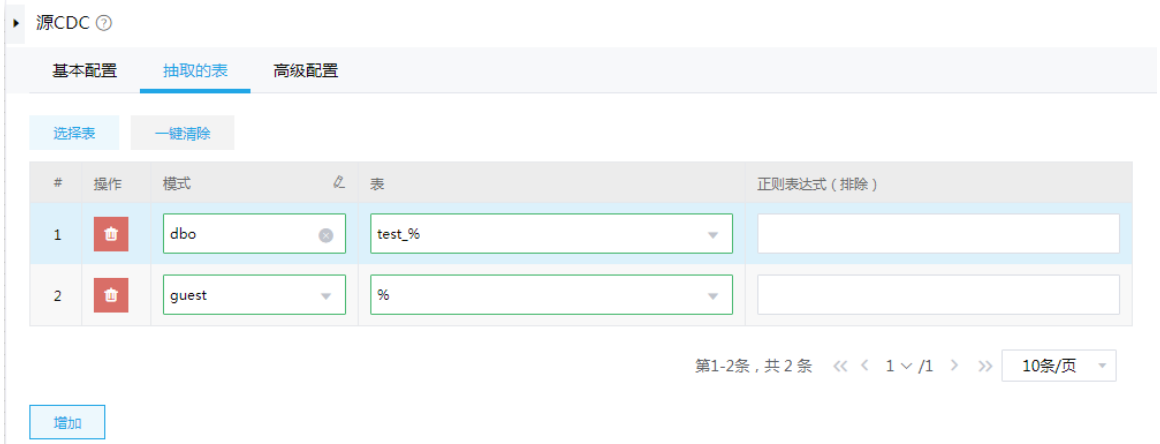
(1) 配置源 CDC

a. 基本配置页签：配置 SQL Server 数据库连接，抽取方式选择“增量+全量”，根据抽取的数据量选择推荐的批量大小和并行度配置，如果抽取的表中包含 varbinary，text、image 等大字段时建议调小批量大小，批量大小建议不超过 100。

图4-128 基本配置

b. 抽取的表页签：选择需要抽取模式和表，其中%代表该模式下的所有表。

图4-129 抽取的表



(2) 配置转换组件相关参数

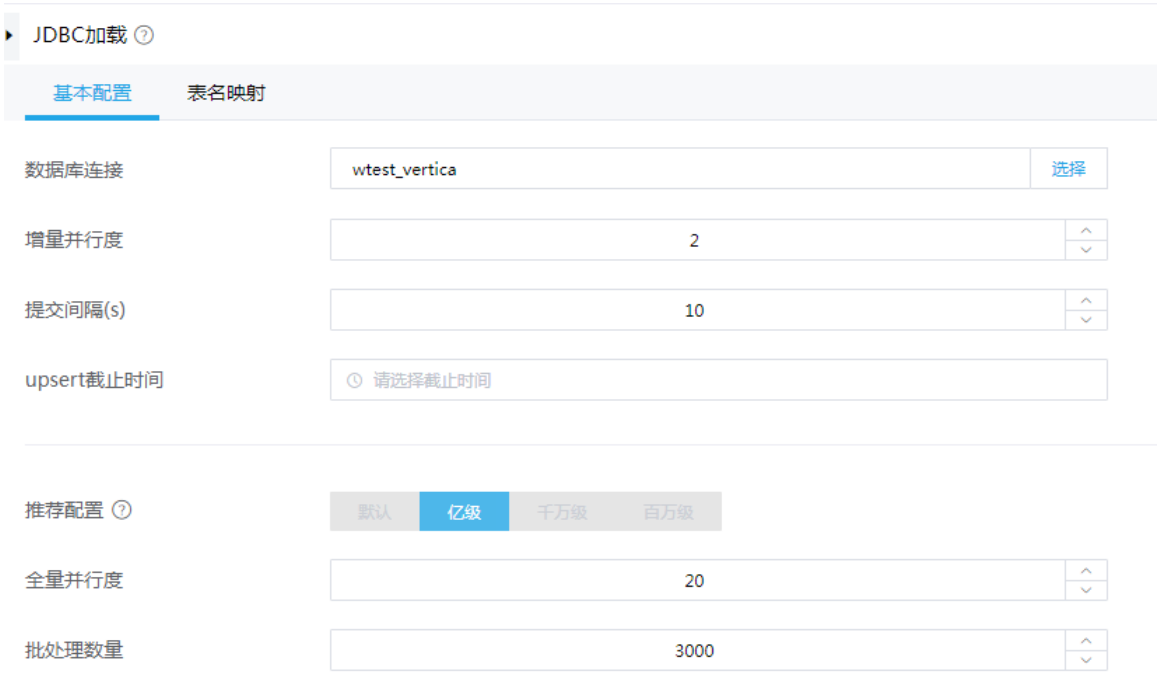
数据标签页签下对源库中抽取出来的数据增加标签字段。表达式 option 代表操作类型(INSERT、DELETE、UPDATE)；original time 代表源库数据变更的时间；now time 代表当前数据抽取出来的时间。

图4-130 数据标签配置

(3) 配置 JDBC 加载。

a. 基本配置页签：配置数据库连接，增量并行度、提交间隔等参数。

图4-131 基本配置



b. 表名映射：当源库中模式名和表名与目标库中模式名和表名不一致时，需使用该配置项配置映射关系。

   – 配合源模式、源表名与目标模式、目标表名的匹配关系。源表名和目标表名支持通过*进行模糊匹配，*代表源表名和目标表名相同的部分。

   – 自动建表如果选择是，则当目标表名匹配不到时会自动在目标库创建表。

图4-132 表名映射页签

(4) 配置完成后，保存作业，然后上线、运行即可开始抽取数据。

4.15 New ETL结构化数据加载至文件

1. 场景描述

A 公司需要将公司数据库数据持久化，需要将数据库数据输出至 FTP 的文件中。

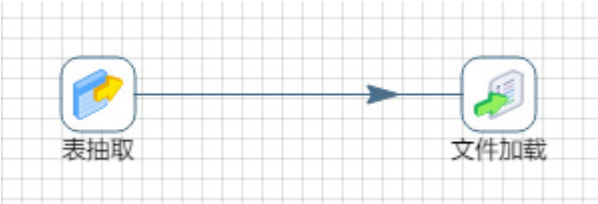
2. 场景分析

通过表抽取组件抽取数据库数据，连接文件加载组件配置 FTP 数据源，输出至指定 FTP 目录下。

3. New ETL 设计方案

表抽取抽取数据，通过文件加载生成 txt 文件加载至 FTP 服务器中，New ETL 图示如图 4-133。

图4-133 结构化数据加载至文件



4. 示例前置条件

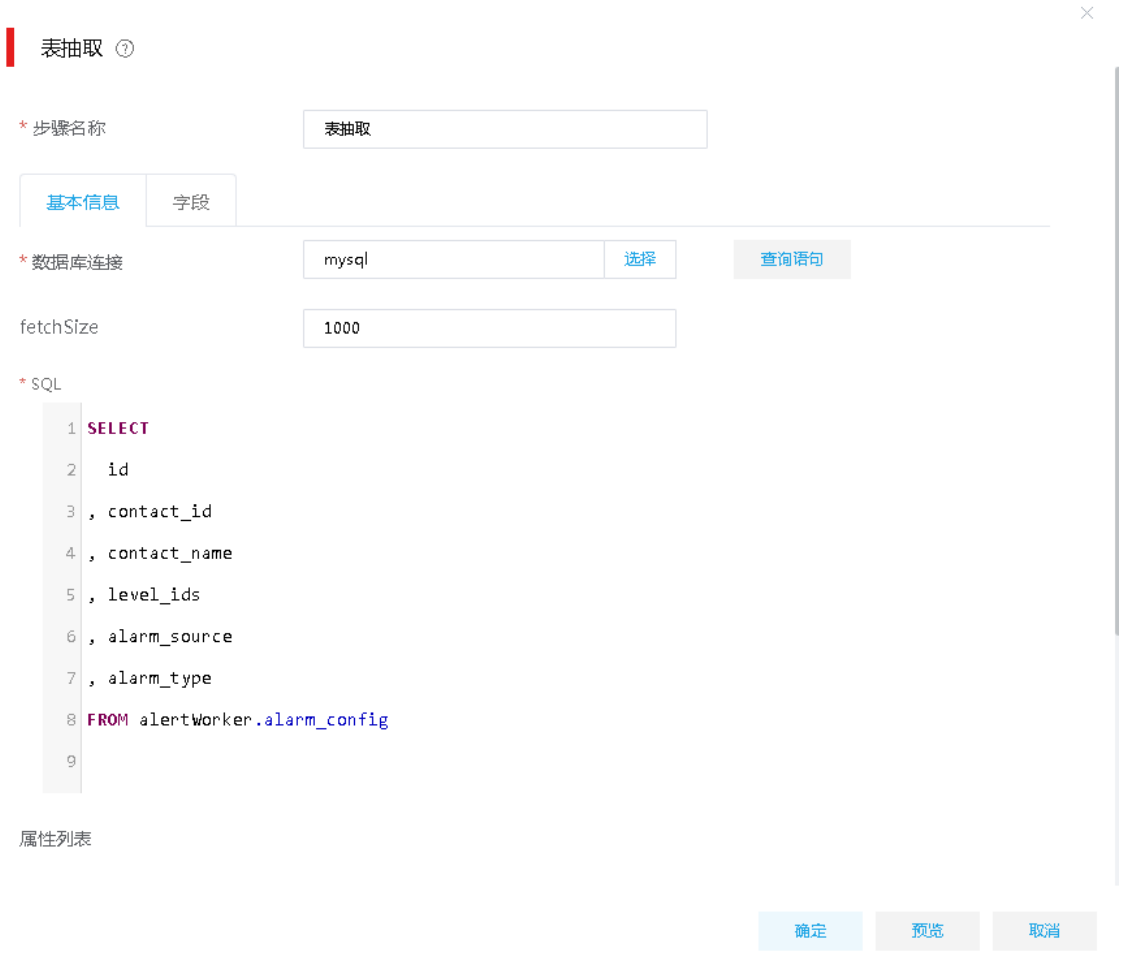
在数据源管理页面中已完成数据库数据源和 FTP 数据源的创建。

5. 示例详细步骤

(1) 表抽取组件配置

a. “基本信息”页配置：配置进行数据抽取的数据库，以及需要抽取的数据。

图4-134 表抽取“基本信息”配置



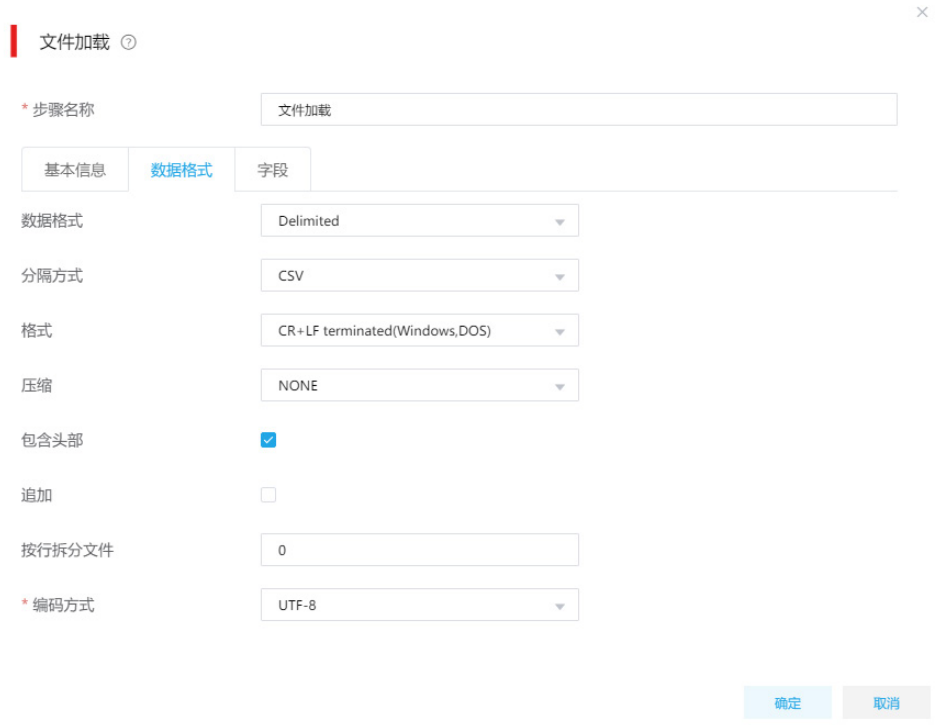
(2) 文件加载组件配置

a. “基本信息”页配置：确定待加载的文件类型及路径。

图4-135 文件加载“基本信息”配置

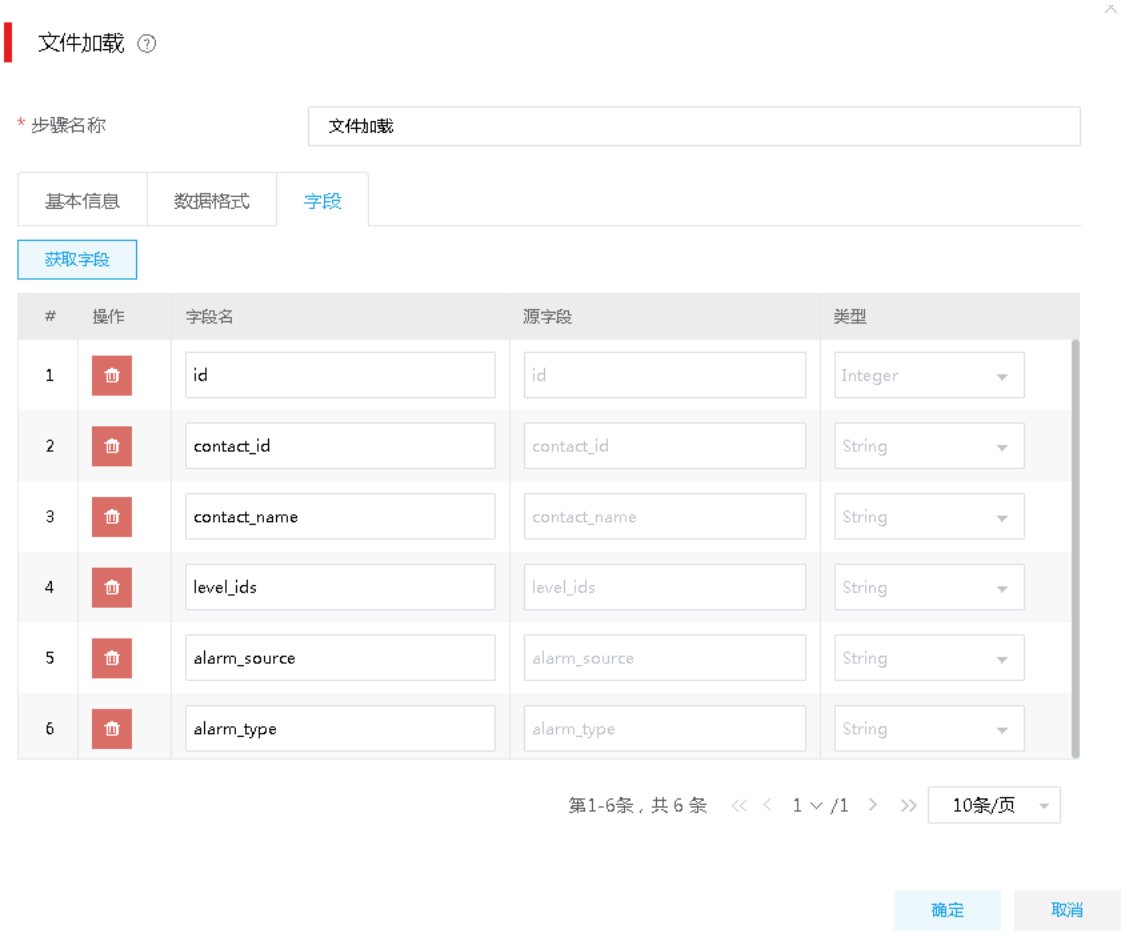
b. “数据格式”页配置：设置加载的文件格式与内容。

图4-136 文件加载“数据格式”配置



c. “字段”页配置：获取前置步骤传来过的数据，并可根据需要重新设置字段名

图4-137 “字段”页配置



(3) 配置完成后，保存作业，然后上线、运行即可开始抽取、加载数据。

4.16 New ETL非结构化数据加载至文件

1. 场景描述

A 公司需要将 FTP 服务器上的一组图片转移至 HDFS。

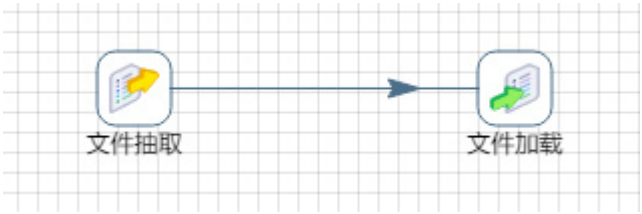
2. 场景分析

NewETL 支持整个文件的抽取与接收，使用文件抽取和文件加载，数据格式选择“WHOLE_FILE”即可。

3. NewETL 设计方案

文件抽取、文件加载数据类型均选择“WHOLE_FILE”。文件抽取从 FTP 服务器中抽取文件，通过文件加载加载至 HDFS 中，New ETL 图示如图 4-138。

图4-138 非结构话数据加载至文件



4. 示例前置条件

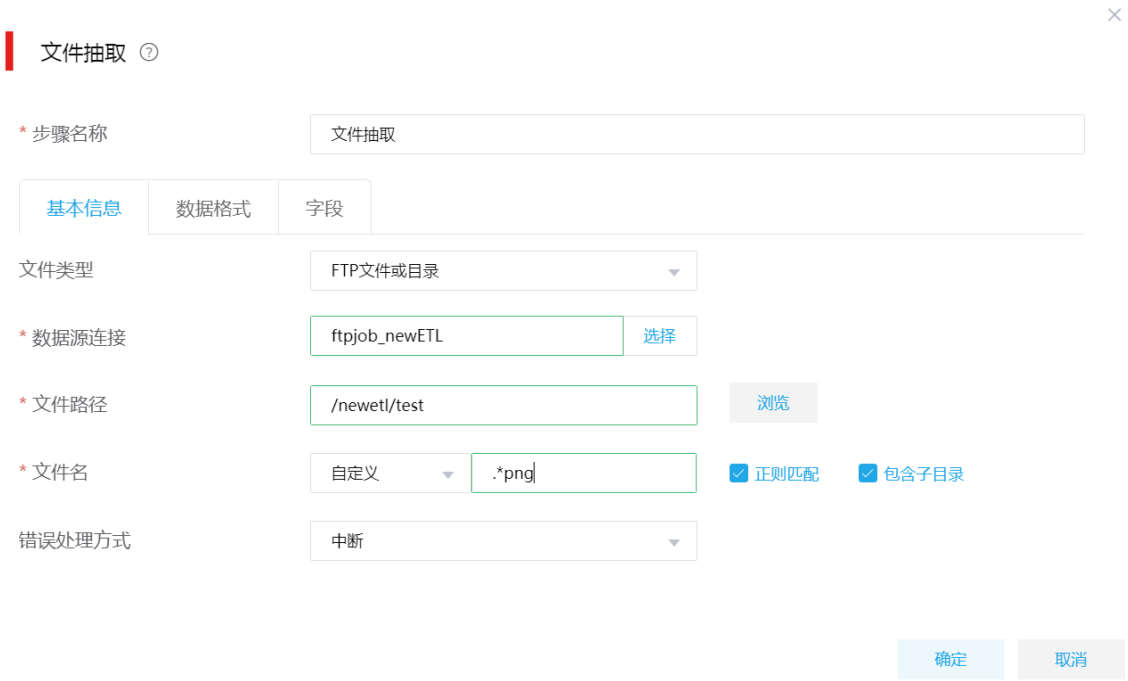
在数据源页面提前配置好 FTP 和 HDFS 数据源。

5. 示例详细步骤

(1) 第一步：文件抽取组件配置

a. “基本信息”页配置：配置待抽取的文件。

图4-139 文件抽取“基本信息”配置



b. “数据格式”页配置：配置需要抽取的格式，此处数据格式配置为 WHOLE_FILE。

图4-140 文件抽取“数据格式”配置

c. “字段”页配置：单击<获取字段>指定从文本文件中读取的字段名称或格式等信息，此处采用默认值即可。

图4-141 文件抽取“字段”配置

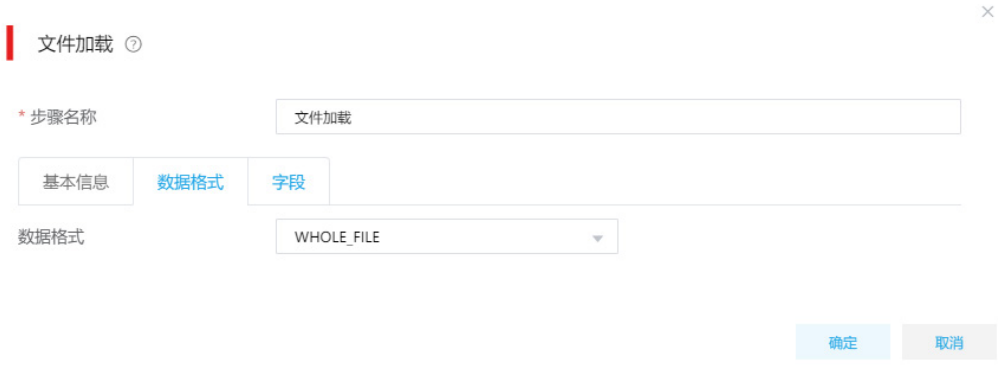
(2) 第二步：文件加载组件配置。

a. “基本信息”页配置：确定待加载的文件类型及路径。

图4-142 文件加载“基本信息”配置

b. “数据格式”页配置：设置加载的文件格式与内容，此处配置数据格式为 WHOLE_FILE。

图4-143 文件加载“数据格式”配置



c. “字段”页配置：获取前置步骤传来过的数据，并重新设置字段名。

图4-144 文件加载“字段”配置

(3) 配置完成后，保存作业，然后上线、运行即可开始抽取、加载数据。

4.17 New ETL数据转换场景

支持对字段加解密、日期转换、加减乘除等运算、字段串截取、字段串替换等处理。

1. 场景描述

A 公司从文件中读取数据，然后需要对部分数据加密、截取、运算后写入关系型数据库。

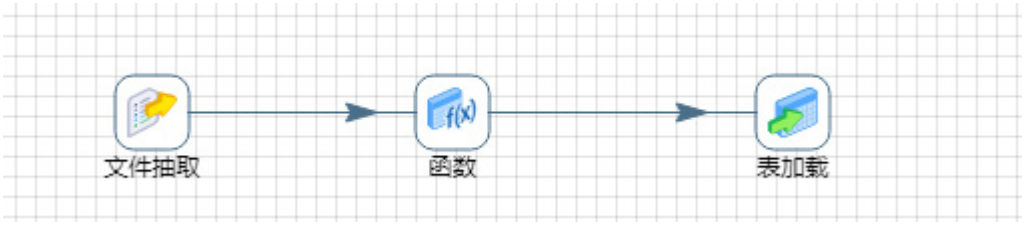
2. 场景分析

通过文件抽取组件从文件中抽取数据，再经过函数组件对字段进行处理后加载至 MySQL 数据库。

3. NEW ETL 设计方案

- 数据流向：文件抽取组件—>函数组件—>表加载组件。NEW ETL 图示如图 4-145。
- NEW ETL 方案：文件抽取采集文件数据并输出，函数组件对通过函数对字段进行处理然后输出，加载至表将数据加载至数据库。

图4-145 NEW ETL 图示



4. 示例前置条件

(1) 创建一个文件，文件内容参考文件数据示例。文件数据示例如下：

id,productname,price,num

10001,apple,2,10

10002,banana,3,20

10003,mongo,8,11

(2) 已创建 MySQL 数据源，名称：mysql_product，目标表：test_product，SQL 脚本：

```
CREATE TABLE `test_product` (
  `id` varchar(18) DEFAULT NULL,
  `productname` varchar(50) DEFAULT null,
  `price` long,
  `num` long,
  `today` date DEFAULT null,
  `sum` long
)ENGINE=InnoDB DEFAULT CHARSET=utf8
```

5. 示例详细步骤

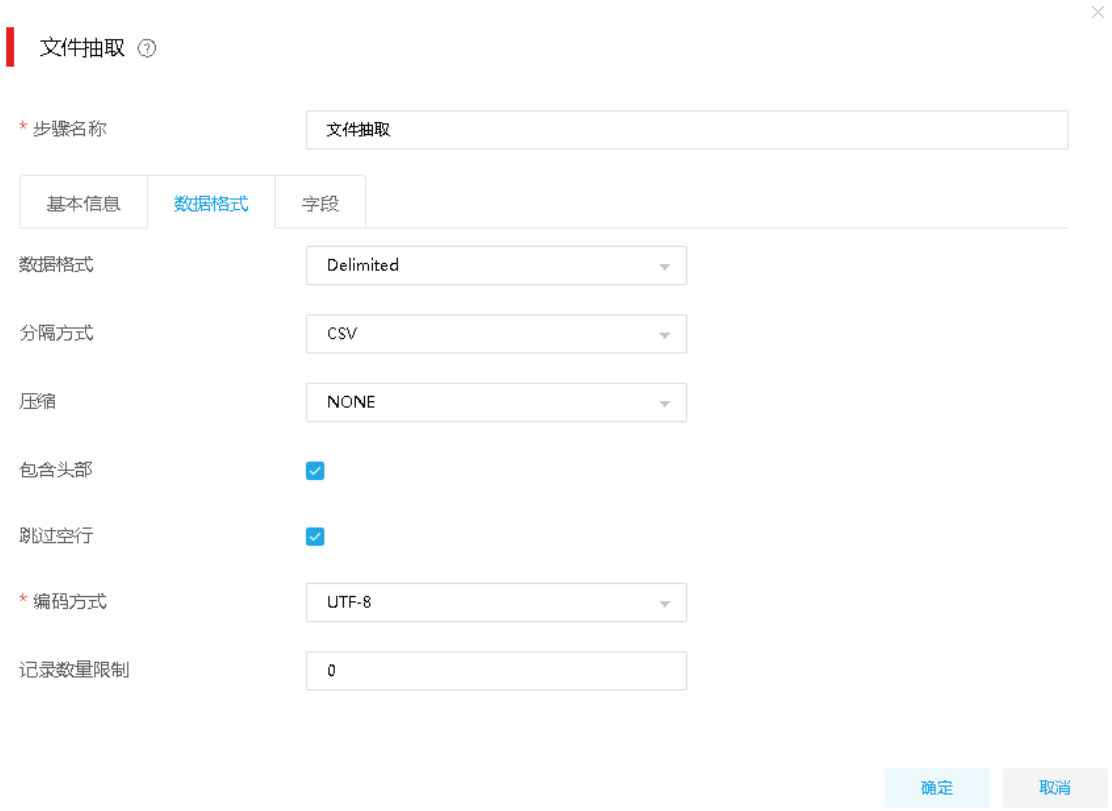
(1) 第一步：文件抽取组件配置

a. “基本信息”页配置：配置待抽取的文件信息。

图4-146 文件抽取“基本信息”

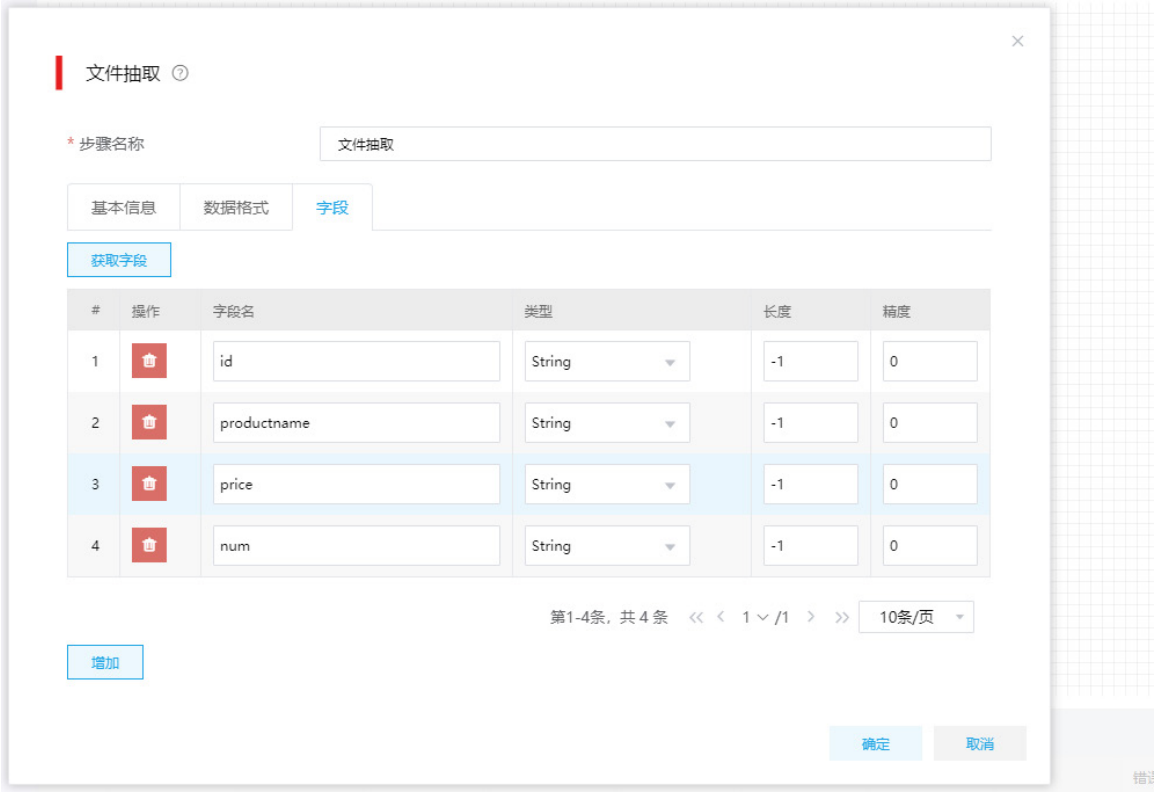
b. “数据格式”页配置：配置处理文件格式及内容，因源文件中的数据是以“,”进行分割，所以采用如下配置进行处理。用户可根据自身实际需要进行配置。

图4-147 “数据格式”页配置



c. “输出字段”配置：从要抽取的文件中获取字段。用于指定从文本文件中读取的字段名称或格式等信息。

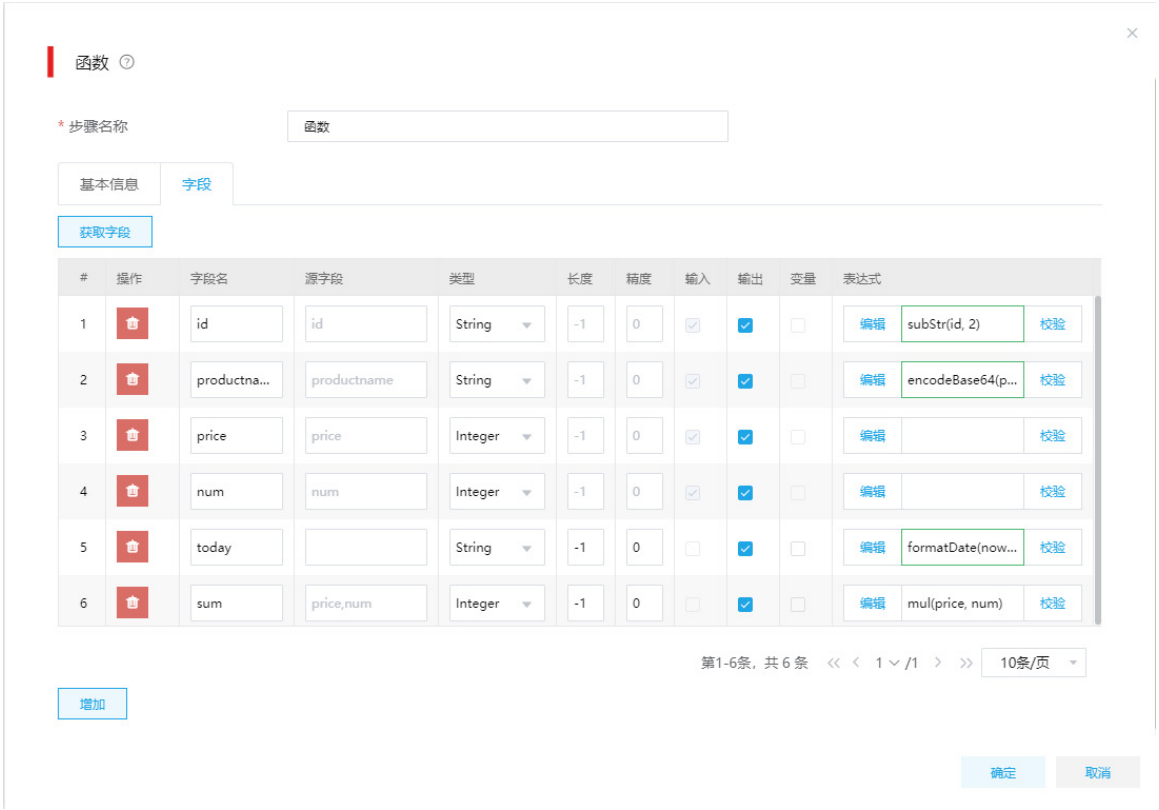
图4-148 文件抽取“输出字段”



(2) 第二步：函数组件配置。函数组件可以单击<编辑>按钮选择可用的内置函数及运算符，选择相应的函数对字段进行处理，编写完表达式后可以单击<校验>按钮对编写的表达式进行校验。

- 对 id 字段进行截取操作，从第三位截取到末尾，使用函数 subStr(id,2)
- 对 productname 进行 Base64 加密，使用函数 encodeBase64(productname)
- 新增字段 today 记录日期，使用函数 formatDate(now(),"yyyy-MM-dd")
- 新增字段 sum 计算总金额,使用乘法函数 mul(price,num)

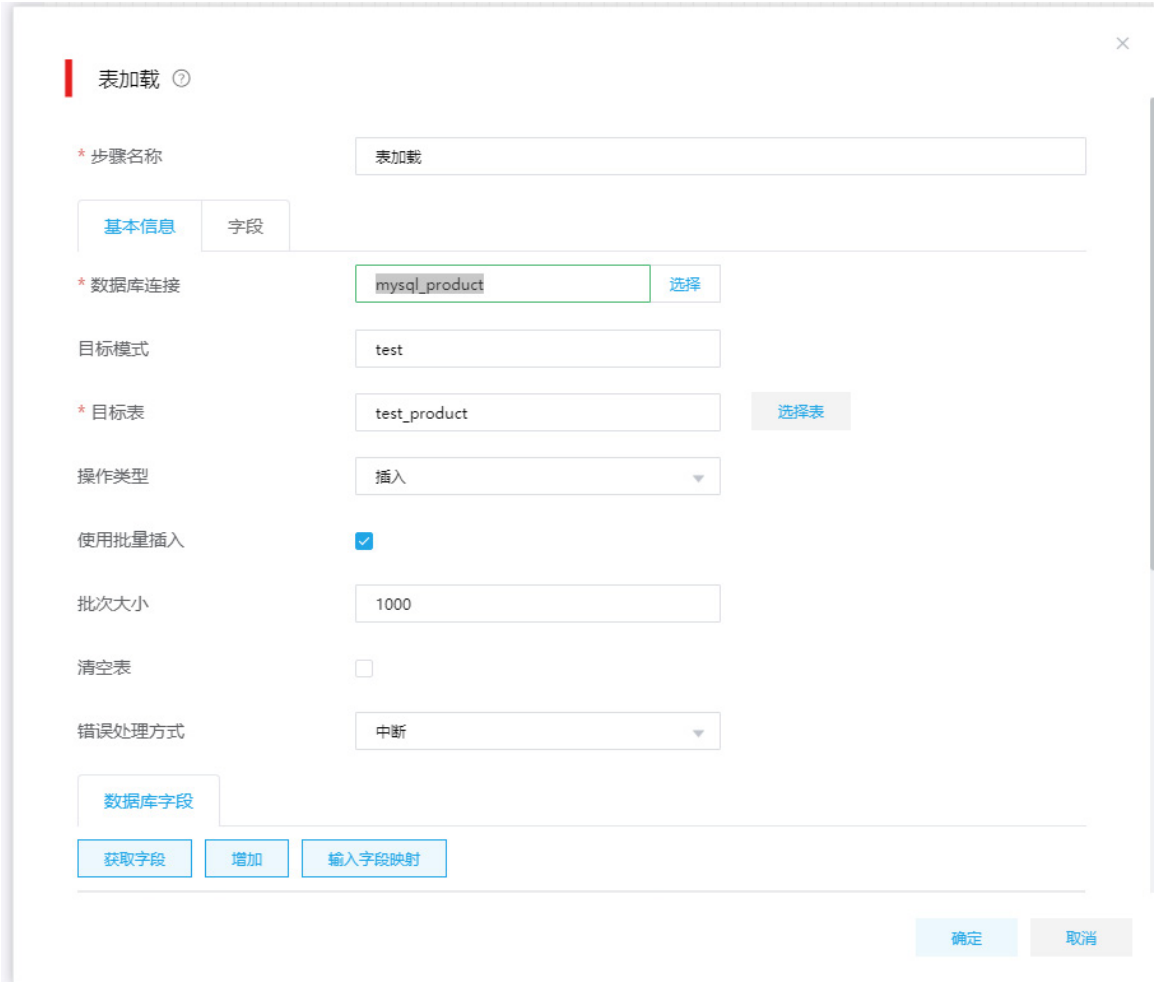
图4-149 函数组件“字段”配置



(3) 第三步：加载至表组件配置。

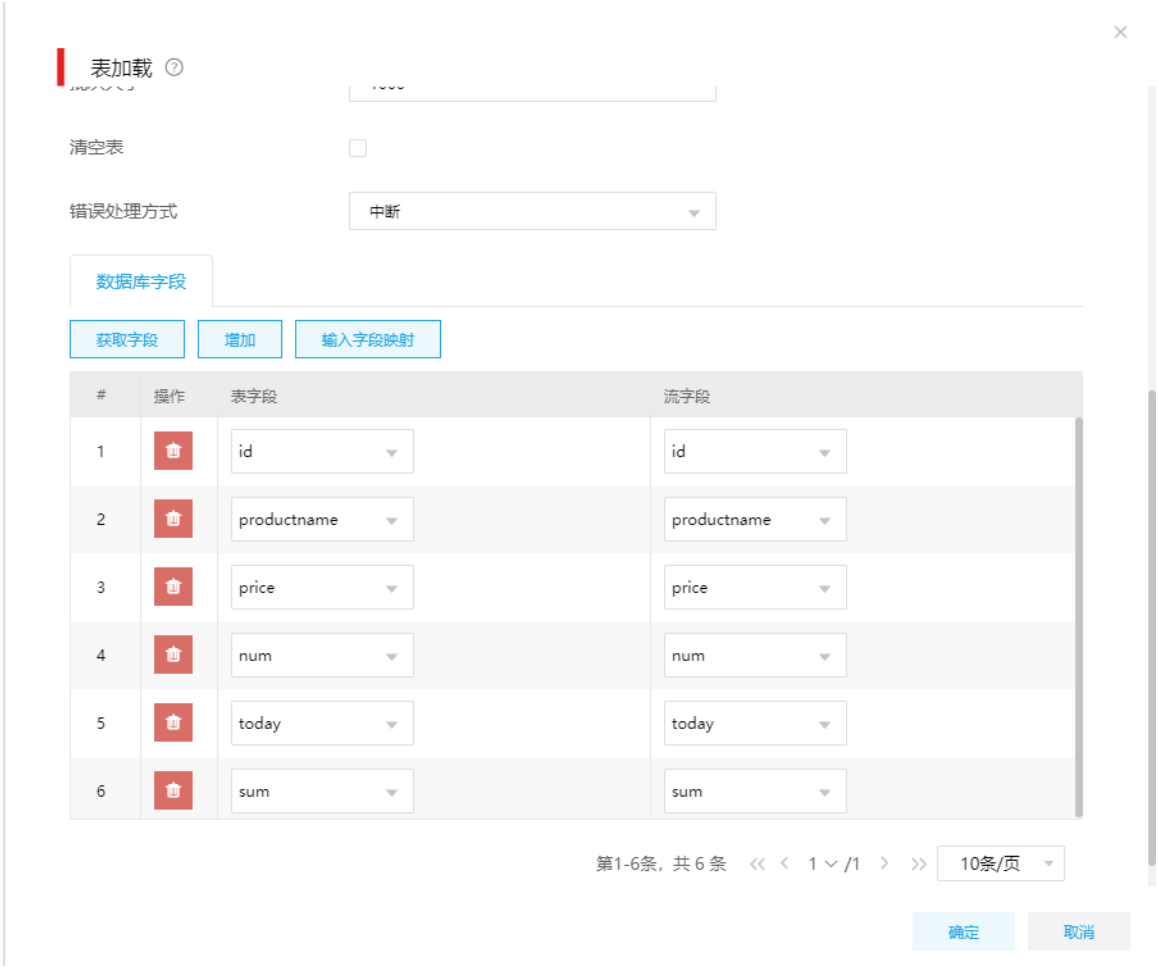
a. “基本信息”页配置：选择需要加载至的库表。

图4-150 表加载“基本信息”



b. “数据库字段”图示：通过获取字段，配置“表字段”和“流字段”的映射关系。“表字段”是 test_product 表(目标表)的字段名称，“流字段”是数据流中的字段。

图4-151 数据库字段图示



(4) 配置完成后，保存作业，然后上线、运行即可进行数据处理。

4.18 New ETL结构化数据加载至HBase

1. 场景描述

A 公司将公司结构化数据转存至大数据集群中，需要将数据库数据输出至 HBase 中。

2. 场景分析

通过表抽取组件抽取数据库数据，连接加载至 HBase 组件配置 HBase 数据源，指定不同列簇。

3. NewETL 设计方案

- 数据流向：表抽取—>加载至 HBase，New ETL 图示如图 4-152。
- ETL 方案：表抽取抽取数据，直接加载至 HBase 中。

图4-152 NewETL 设计方案

4. 示例前置条件

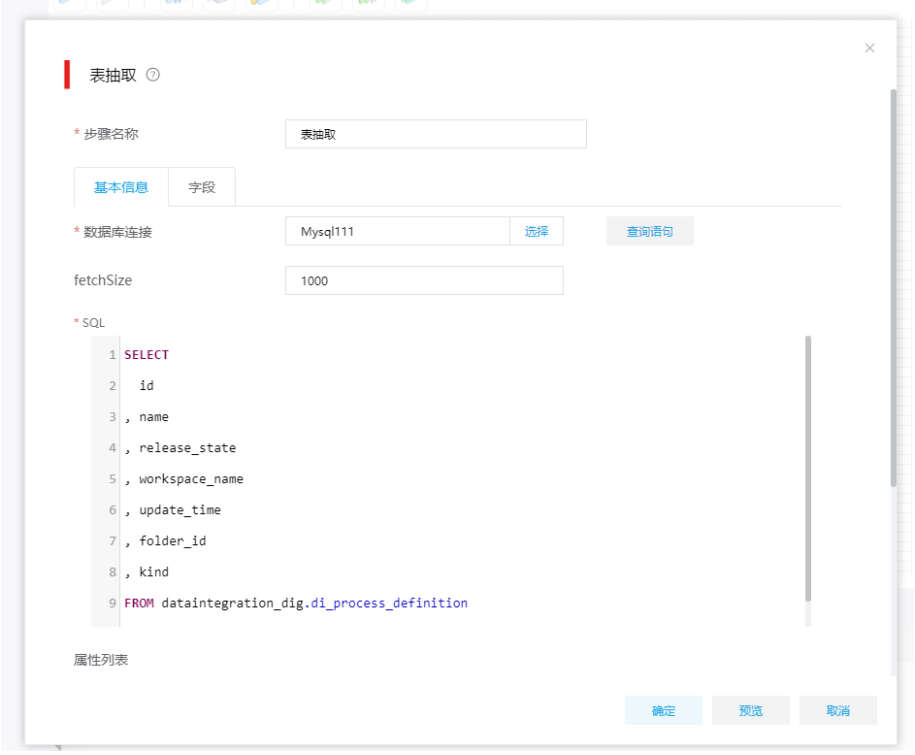
在数据源管理页面中创建好数据库数据源和 HBase 数据源。

5. 示例详细步骤

(1) 第一步：表抽取组件配置

a. “基本信息”页配置：配置进行数据抽取的数据库，以及需要抽取的数据。

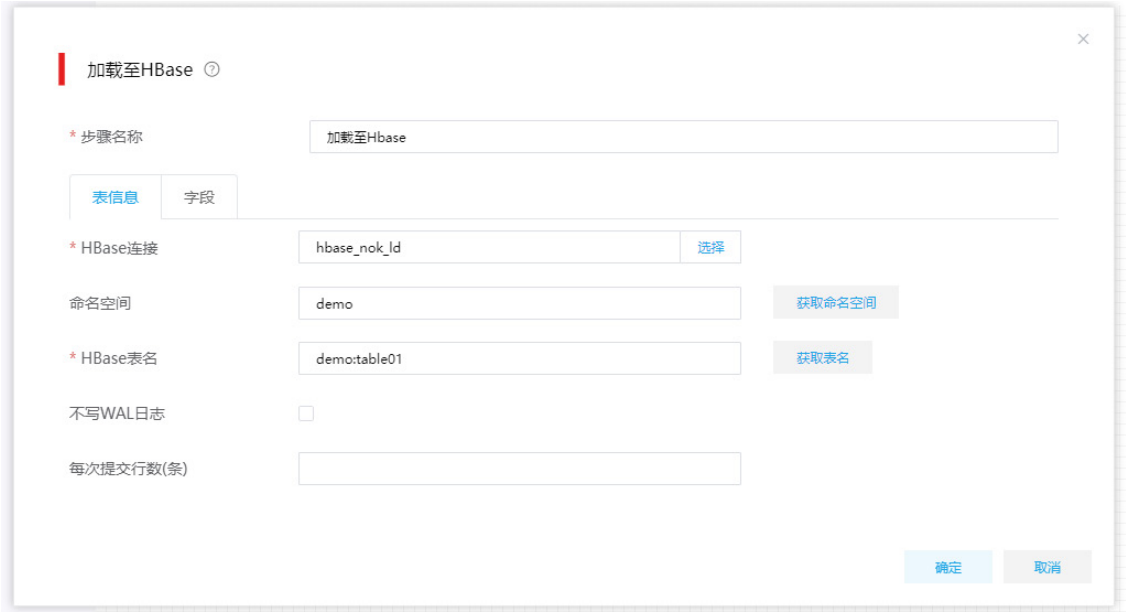
图4-153 表抽取“基本信息”



(2) 第二步：加载至 HBase 组件配置

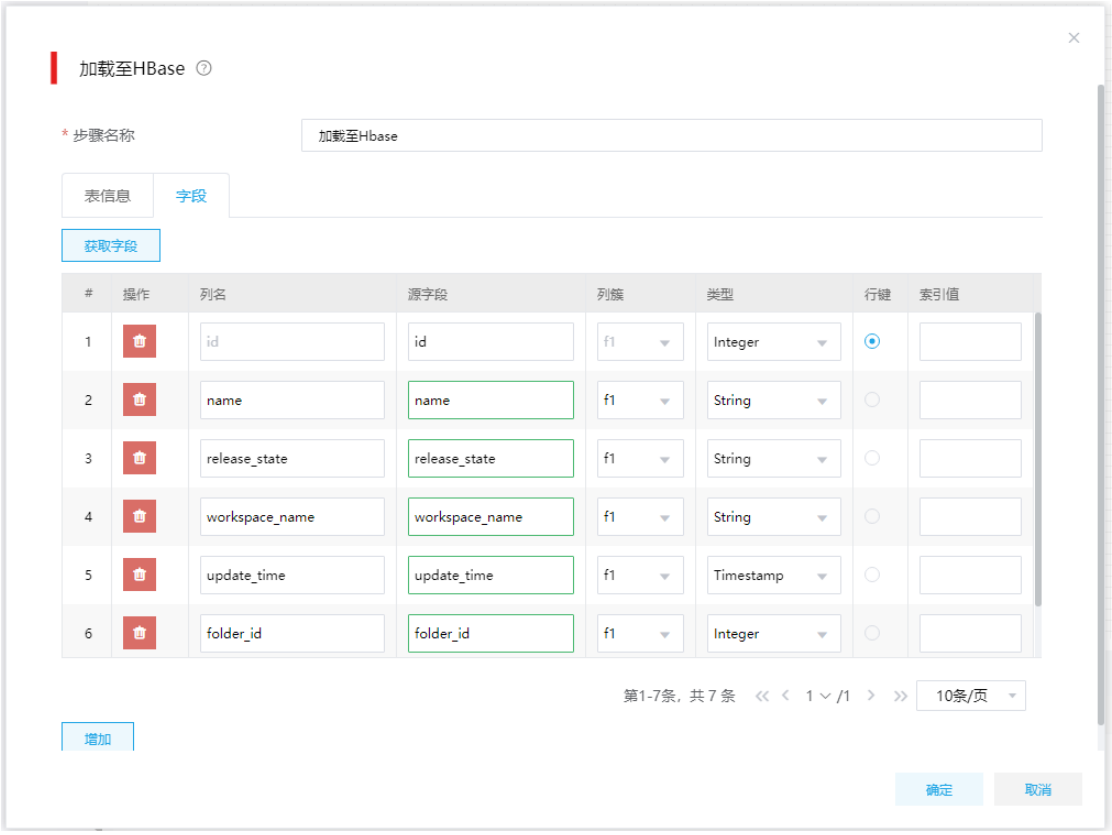
a. “表信息”页配置：选择需要加载数据的 HBase 数据源、命名空间及表名。

图4-154 “表信息”页配置



b. “字段”页配置：获取前一组件传递过来的字段作为输出字段，然后写入 HBase 中。

图4-155 “字段”页配置



4.19 New ETL非结构化数据（单一类型）加载至HBase

1. 场景描述

A 公司需要将 FTP 服务器上的一组图片转移至 HBase。

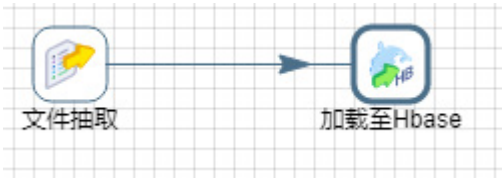
2. 场景分析

New ETL 支持整个文件的抽取与接收，使用文件抽取和文件加载，数据格式选择“WHOLE_FILE”即可。

3. NewETL 设计方案

- 数据流向：文件抽取—>加载至 HBase，ETL 图示如图 4-156。
- New ETL 方案：文件抽取数据类型选择“WHOLE_FILE”。文件抽取从 FTP 服务器中抽取文件，通过加载至 HBase 组件加载至 HBase 中。

图4-156 NewETL 设计方案



4. 示例前置条件

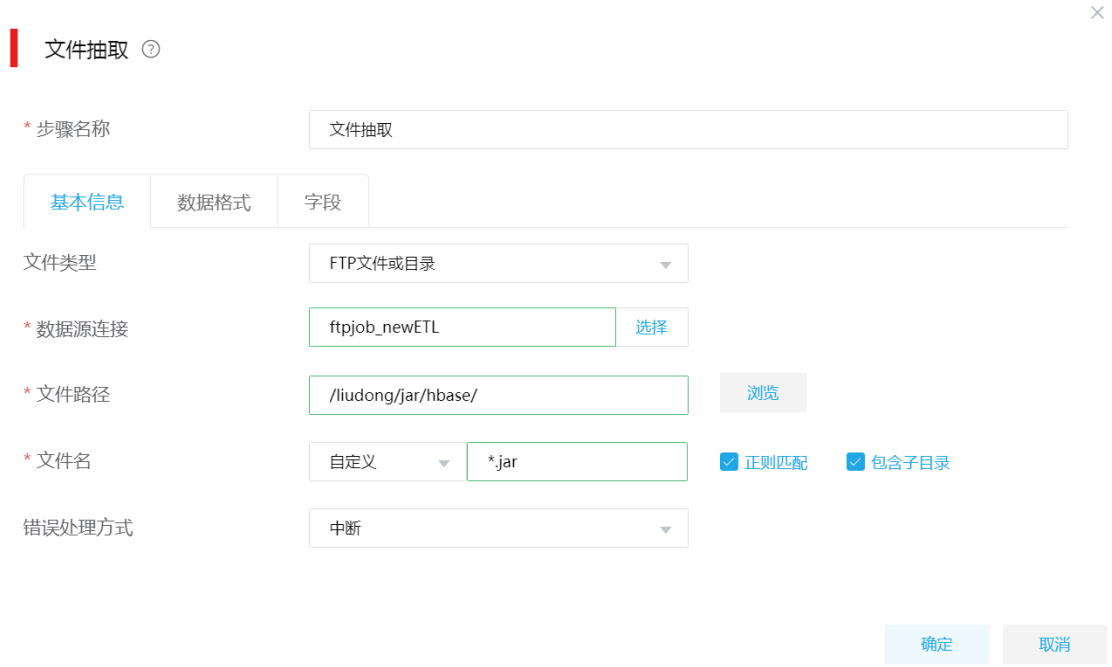
在数据源管理页面提前配置好 FTP 和 HBase 数据源

5. 示例详细步骤

(1) 第一步：文件抽取组件配置。

a. “基本信息”页配置：选择待抽取文件所在文件路径及文件名。

图4-157 文件抽取“基本信息”



b. “数据格式”页配置：设置待处理文件的格式与内容，此处设置为 WHOLE_FILE。当配置为 WHOLE_FILE 格式时，仅以二进制方式读取原文件内容，不做任何格式化解析。

图4-158 文件抽取“数据格式”

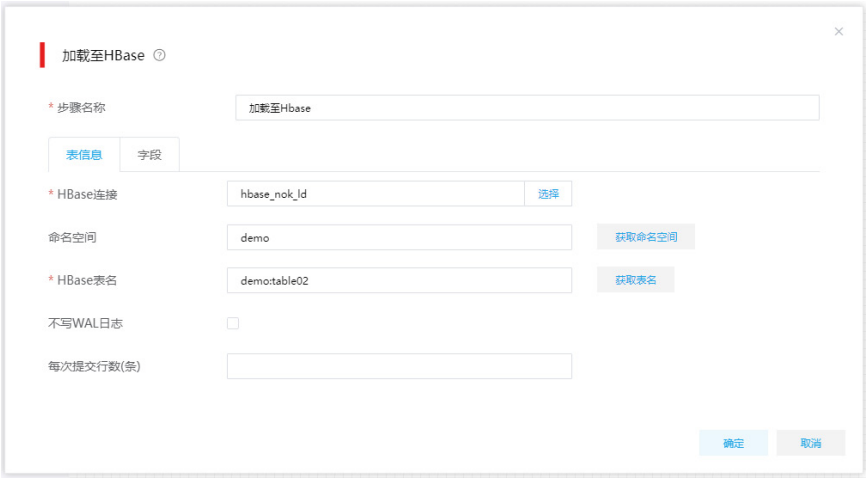
c. “字段”页配置：指定从文本文件中读取的字段名称或格式等信息。

图4-159 文件抽取“字段”

(2) 第二步：加载至 HBase 组件配置

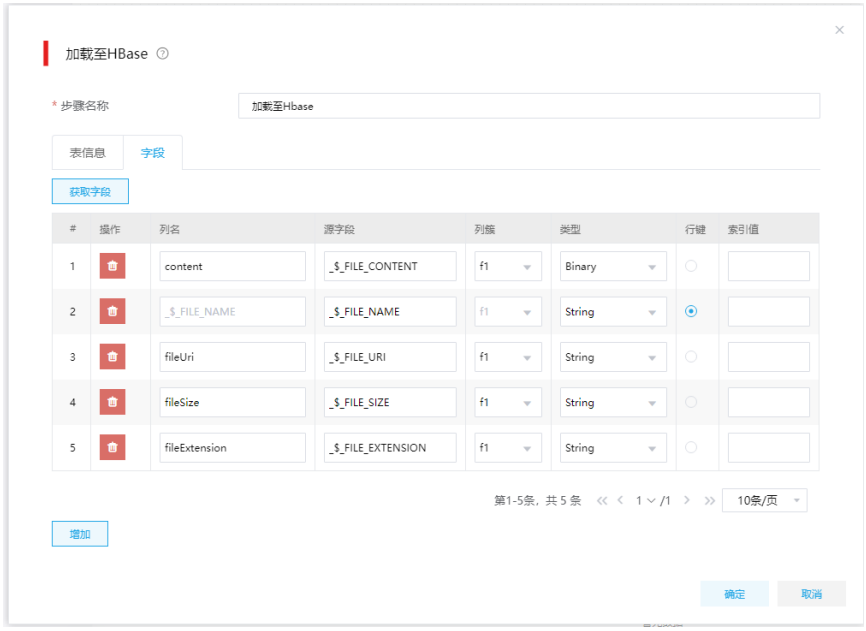
a. “表信息”页配置：选择需要加载数据的 HBase 数据源、命名空间及表名。

图4-160 “表信息”页配置



b. “字段”页配置：获取前一组件传递过来的字段作为输出字段，然后写入 HBase 中。

图4-161 加载至 HBase“字段”



4.20 New ETL非结构化数据（多种类型）加载至HBase

1. 场景描述

A 公司需要将 FTP 服务器上的一组包含 png 及 sha1 类型的文件转移至 HBase 中。

2. 场景分析

随着公司业务系统数据量的增加。一些小型的文件。例如 pdf 或一些图片在常规储存中出现诸多不便，故需要将一些小型的文件存入大数据集群中，而 HBase 基于列簇的数据仓库恰恰满足这一点。

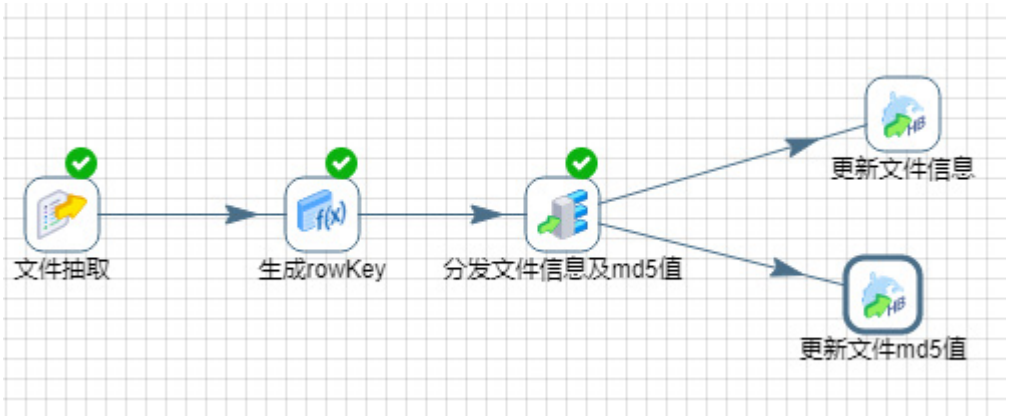
表4-5 Hbase 列簇信息

rowKey	fileInfo:name	fileInfo:size	fileInfo:uri	fileInfo:extension	fileInfo:content	fileInfo:md5
json-lib-2.2.2-jdk15	json-lib-2.2.2-jdk15.jar	143977	ftp://ls:ls@10.121.74.26/liudong/jar/hbase/json-lib-2.2.2-jdk15.jar	jar	\x00\x00\x12\x0B\x00\x001\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xA4\x818	564983fce666c9b3878b90ba753b62a4eca0791c

3. NewETL 设计方案

New ETL 方案：文件抽取数据类型选择“WHOLE_FILE”。文件抽取从 FTP 服务器中抽取文件，通过函数组件截取文件内容生成 rowKey，然后通过 route 组件将 sha1 文件与其他文件及文件信息进行区分，分别发送到 HBase 的不同列中。

图4-162 NewETL 设计方案



4. 示例前置条件

在数据源页面提前配置好 FTP 和 HBase 数据源。

5. 示例详细步骤

(1) 第一步：文件抽取组件配置

a. “基本信息”页配置：选择待抽取文件所在文件路径及文件名。

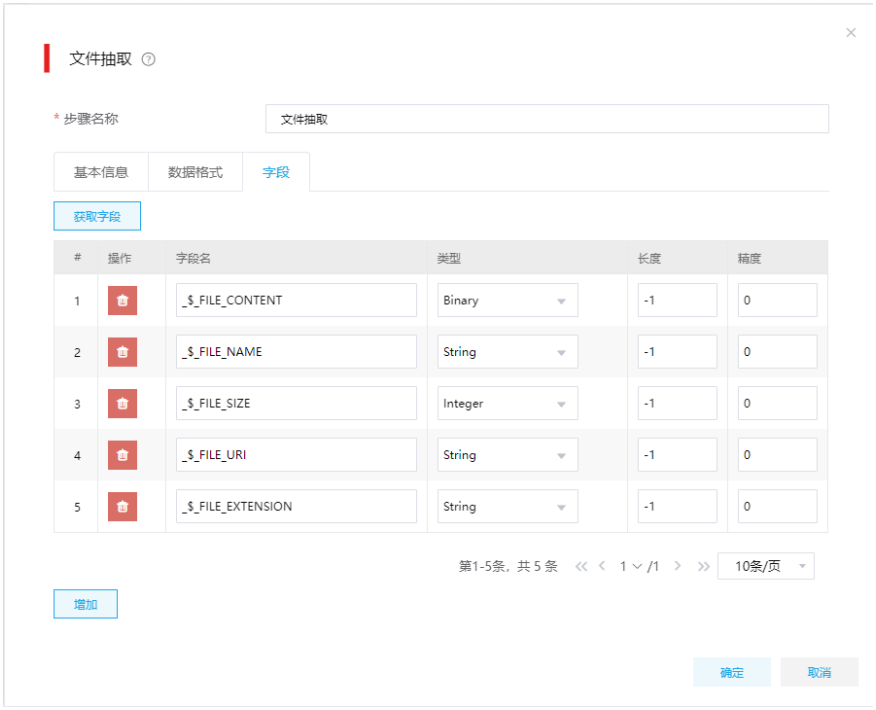
图4-163 “基本信息”页配置

b. “数据格式”页配置：设置待处理文件的格式与内容，此处设置为 WHOLE_FILE。当配置为 WHOLE_FILE 格式时，仅以二进制方式读取原文件内容，不做任何格式化解析。

图4-164 “数据格式”页配置

c. “字段”页配置：指定从文件中读取的字段名称或格式等信息。

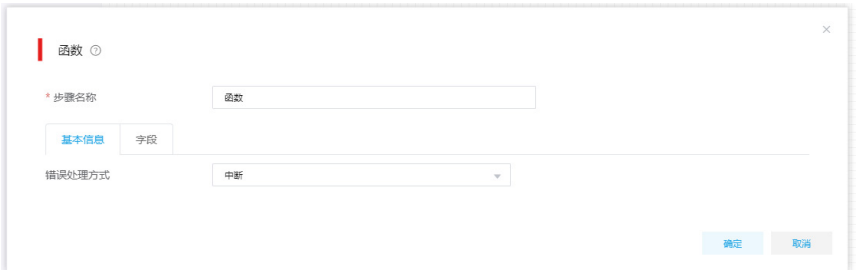
图4-165 “字段”页配置



(2) 第二步：通过 Route 组件生成 rowKey 配置。

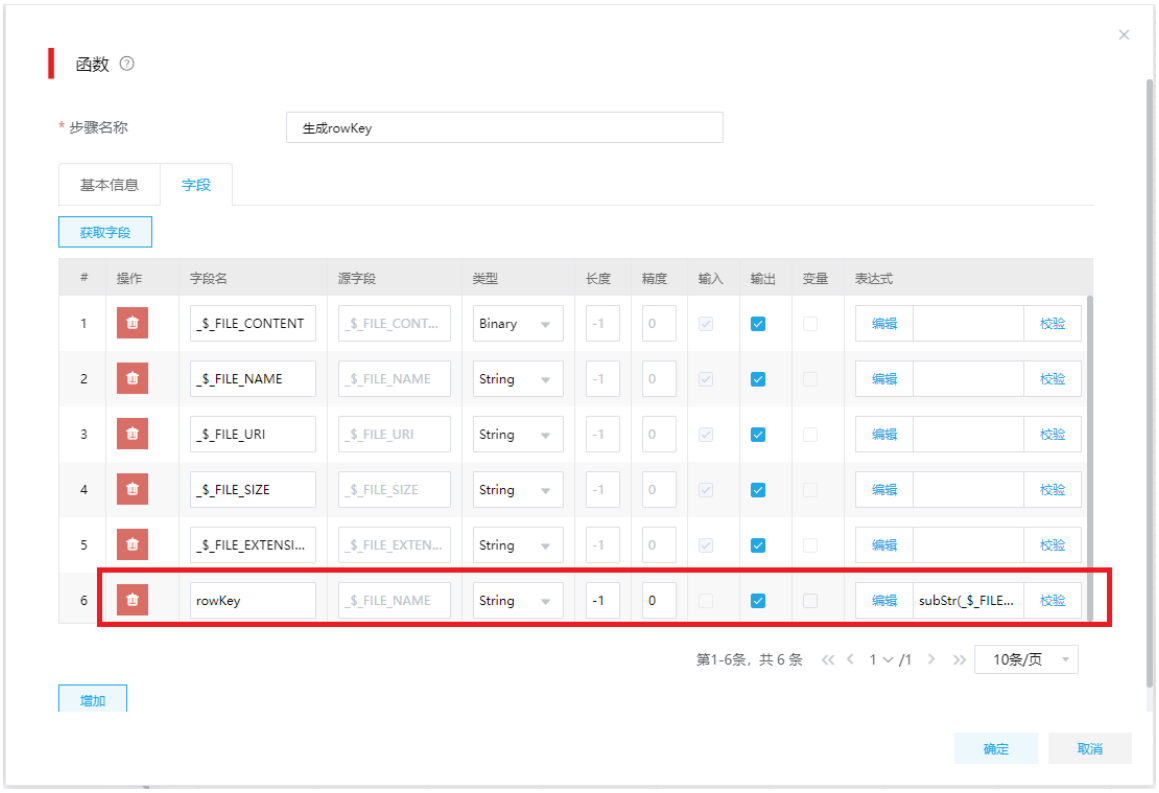
a. “基本信息”页配置：出现错误处理的方式，包括中断及忽略。中断指立即中断抽取，忽略指忽略一定数量的错误数后才中断，用户可根据实际需要进行选择。

图4-166 “基本信息”页配置



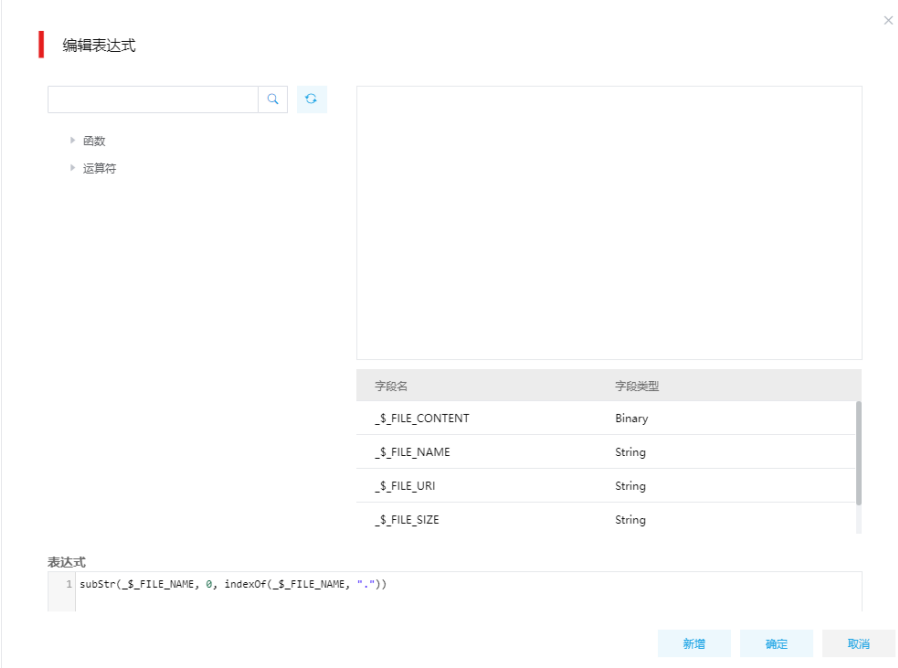
b. “字段”页配置：新增“rowKey”字段，通过对文件名的截取（取除去扩展名之前的文件名）生成 rowKey。

图4-167 “字段”页配置



c. 单击<编辑>按钮，可对源字段编辑表达式，对前序组件传递过来的源字段进行处理。

图4-168 编辑函数表达式



(3) 第三步：通过 Route 组件分发文件信息及 md5 值

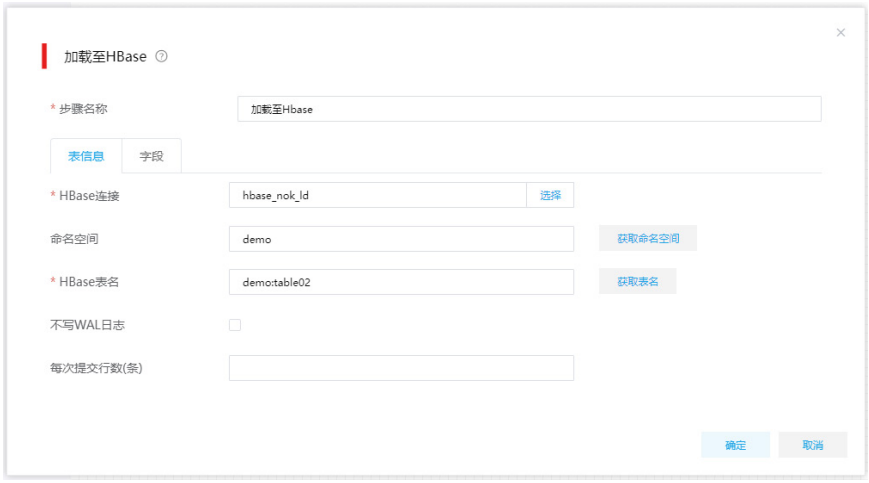
a. “基本信息”页配置：配置分发规则将 sha1 文件与其他文件及文件信息进行区分。Sha1 文件加载至 HBase2，其他文件加载至 HBase。

图4-169 “基本信息”页配置

(4) 第四步：通过“加载至 HBase”组件将数据加载至 HBase 的不同列中。

a. “表信息”页配置：选择需要加载数据的 HBase 数据源、命名空间及表名。

图4-170 “基本信息”页配置



b. 两个“加载至 HBase”组件均通过 rowkey 进行更新，如图 4-171、图 4-172。

图4-171 更新文件信息配置

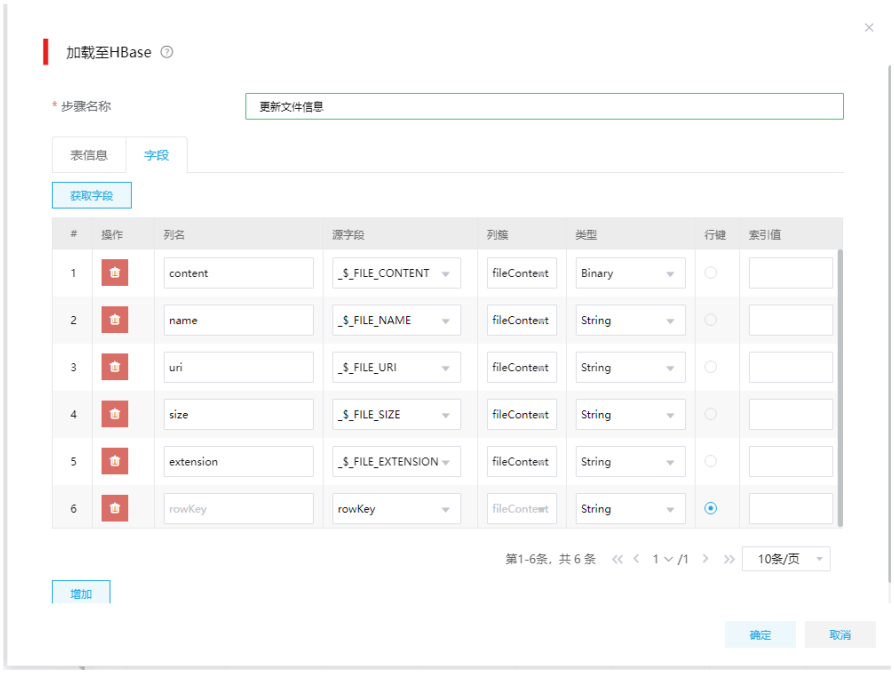
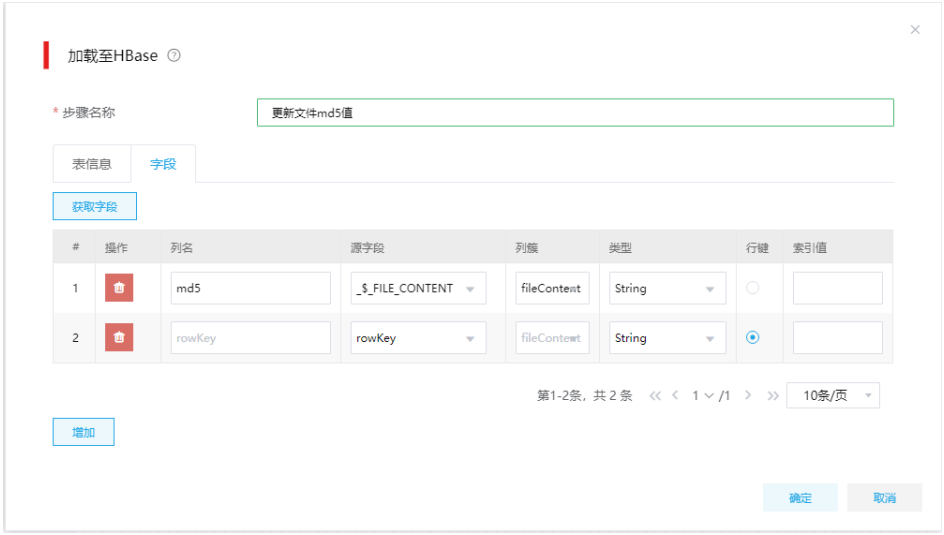


图4-172 sha1 文件更新文件 md5 值



4.21 New ETL数据库查询

1. 场景描述

A 公司需要将用户数据和消费数据关联为宽表，并加载至目标表中，以便后续进行分析。

数据示例如下：

表4-6 用户数据结构

ID	NICKNAME	PHONE	CTEATETIME
1	张三	13888888888	2014/4/8 23:36:58
2	李四	13999999999	2015/3/26 13:15:06

表4-7 消费数据结构

ID	USERID	CREATETIME	PRICE
1	1	2016/2/8 10:36:58	100
2	2	2016/3/8 11:36:58	80
3	1	2016/4/8 12:36:58	60
4	2	2016/5/8 13:36:58	90

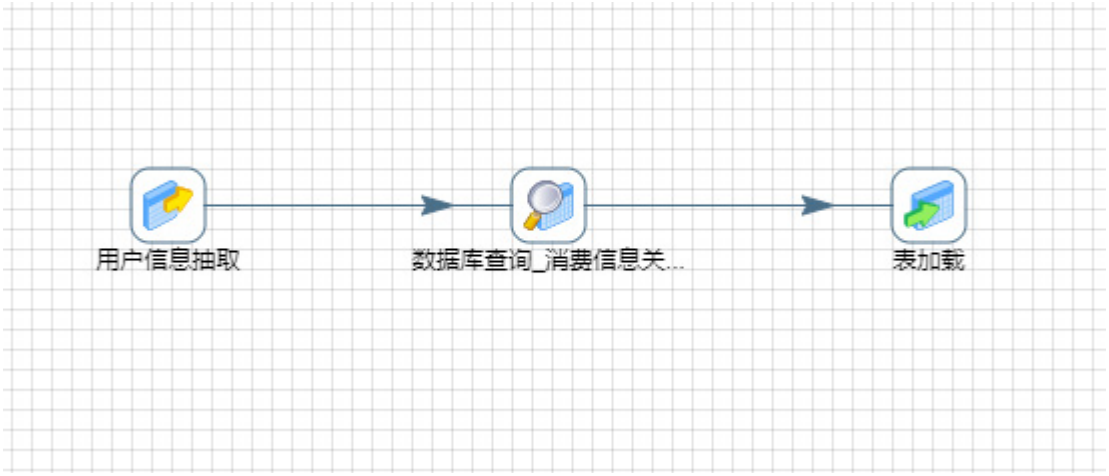
2. 场景分析

对于分布在不同数据库中的数据，可以通过数据库查询组件进行查询关联，再使用表加载组件将数据加载至目标表中。

3. New ETL 任务设计

- 数据流向：表抽取 → 数据库查询 → 表加载
- New ETL 方案：先使用表抽取组件抽取用户数据，然后使用数据库查询组件关联查询消费数据，关联条件为：用户数据.ID = 消费数据.USERID。最后将数据加载是目标表中，如图 4-173 所示。

图4-173 New ETL 任务设计图示

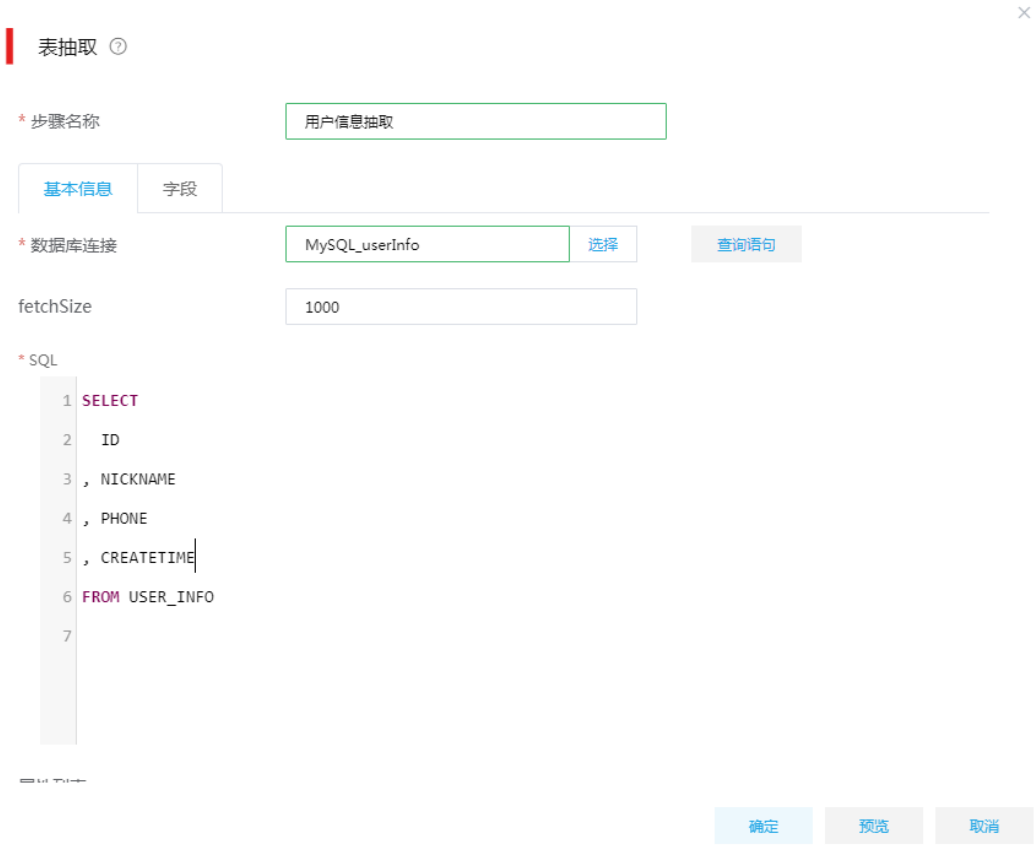


4. 示例详细步骤

(1) 第一步：配置表抽取组件

a. “基本信息”配置：配置进行数据抽取的数据库，以及需要抽取的数据。

图4-174 “基本信息”配置



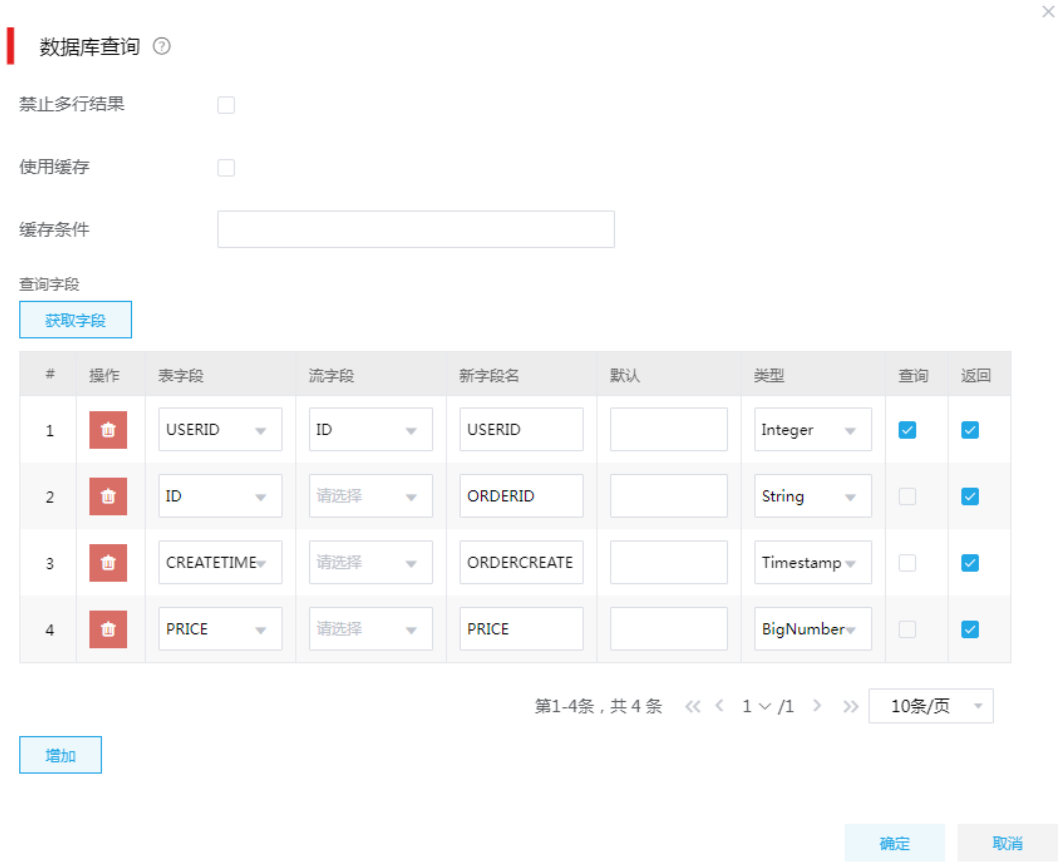
b. “字段”配置：“字段”页签下展示通过 SQL 语句查询出来的字段。如果对 SQL 进行了手动调整，需要进入字段页签重新获取字段，如图 4-175 所示。

图4-175 “字段”配置

c. 配置完成后，单击<确定>按钮保存组件配置。

(2) 第二步：配置数据库查询组件，该组件可以使用设置的关键字在目标表中查询，并从查询结果中返回指定的字段。配置用户数据.ID 为查询条件，查询用户数据.ID = 消费数据.USERID 的数据，如图 4-176。

图4-176 数据库查询



(3) 第三步：配置表加载组件。

选择数据库连接、目标表后，根据需要配置字段映射即可。

4.22 New ETL增量数据同步

1. 场景描述

A 公司需要将源数据库中某张表的增量数据同步到目标数据库的对应表中。

2. 场景分析

随着业务系统数据量增加，数据同步操作也从每次全量覆盖，转变为一次全量同步+定时增量同步。为了实现数据的增量同步，在设计数据表时新增一个时间戳字段。新增数据在入库时，会为时间戳字段赋值为入库时间，已有数据发生变更时，时间戳字段被赋值为数据的更新时间。

表4-8 源表（source）、目标表（target）结构

id	name	age	updt_time
1001	tom	18	2022-07-07 12:25:36

3. New ETL 设计方案

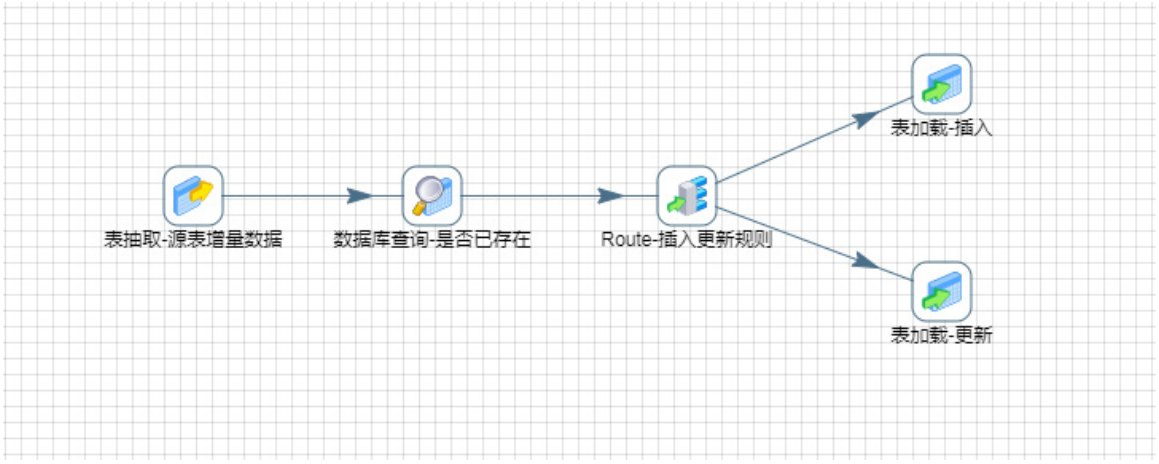
数据流向：表抽取—>数据库查询—>Route—>表加载。

增量数据同步基于当前调度时间，要求每天凌晨 2 点将前一天的增量数据同步到目标表。比如当前调度时间为 2022/07/07 02:00:00，则增量抽取的数据是 2022/07/06 00:00:00（包含）到 2022/07/07 00:00:00（不包含）这段时间入库或者更新的数据，在表抽取组件中可以通过数据库的内置时间函数来设置 updt_time 字段值的范围，如：源数据库是 MySQL 时，可以通过如下 SQL 语句抽取当前调度所对应的前一天的增量数据。

```
select
*
from
test.source
where
updt_time >= date_sub(current_date,  interval 1 day)
and updt_time < current_date
```

“表抽取”组件查询出的增量数据，流入“数据库查询”组件，通过指定主键字段在目标表中进行查询，并要求返回目标表中查询出的主键值，然后将查询结果再流入“Route”组件，在 Route 组件中配置分发规则，将输入数据分流至两个“表加载”组件，分别处理新增数据的插入、已有数据的更新。

图4-177 增量同步 NewETL 任务图示



4. 示例前置条件

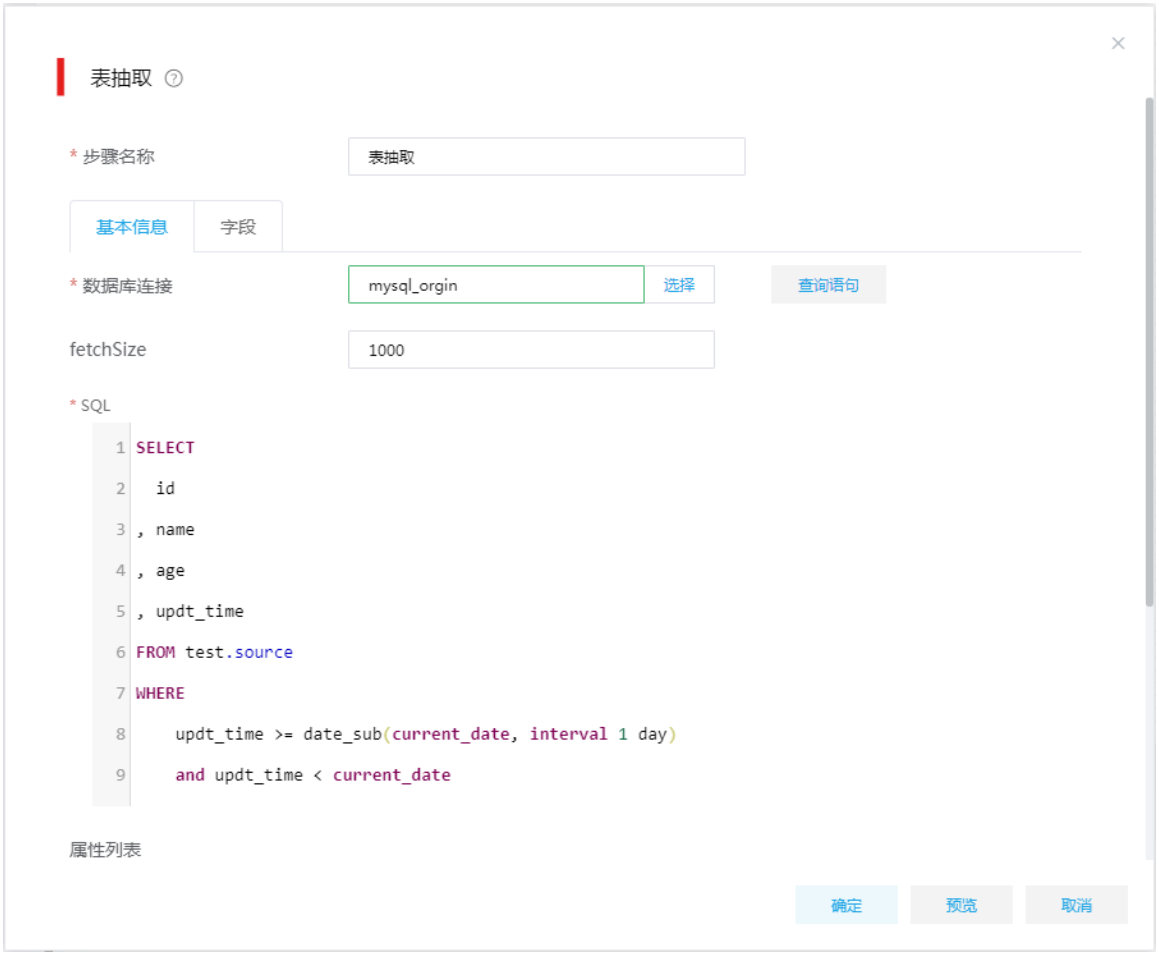
在数据源页面提前配置好源库、目标库的数据源，参照表 4-8 分别在源库、目标库中创建表，id 为主键字段。

5. 示例详细步骤

(1) 第一步：表抽取组件配置

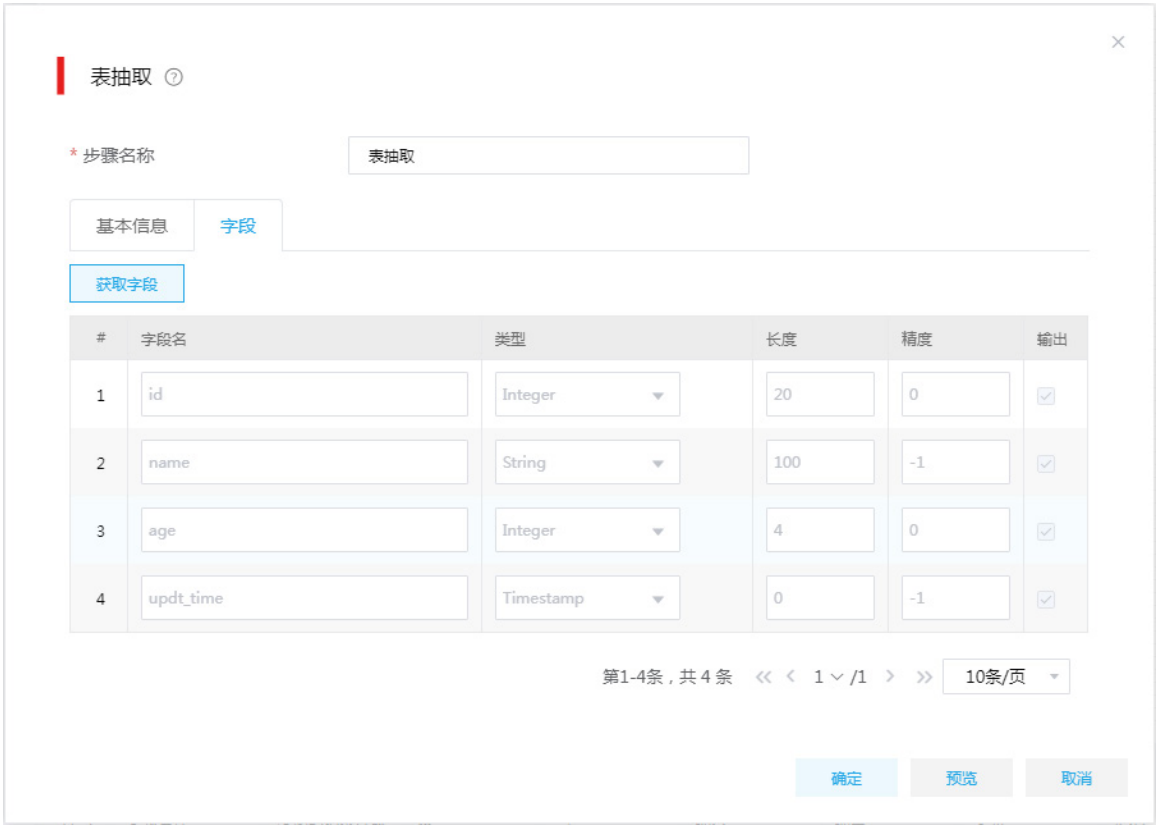
a. “基本信息”页配置：配置进行数据抽取的数据库，并通过 SQL 语句抽取当前调度所对应的前一天的增量数据。

图4-178 “基本信息”页配置



b. “字段”页配置：展示了通过 SQL 语句查询出来的字段。

图4-179 “字段”页配置



(2) 第二步：数据库查询组件配置

“查询”页配置：配置需要进行数据库查询的数据源连接及表。“表抽取”组件查询出的增量数据，流入“数据库查询”组件，通过指定主键字段在目标表中进行查询，并要求返回目标表中查询出的主键值。

图4-180 数据库查询

(3) 第三步：Route 组件配置

在 Route 组件中配置分发规则，根据 id 字段是否为空，将输入数据分流至两个“表加载”组件，分别处理新增数据的插入、已有数据的更新。

图4-181 Route 组件配置

(4) 第四步：表加载组件配置（表加载-插入）

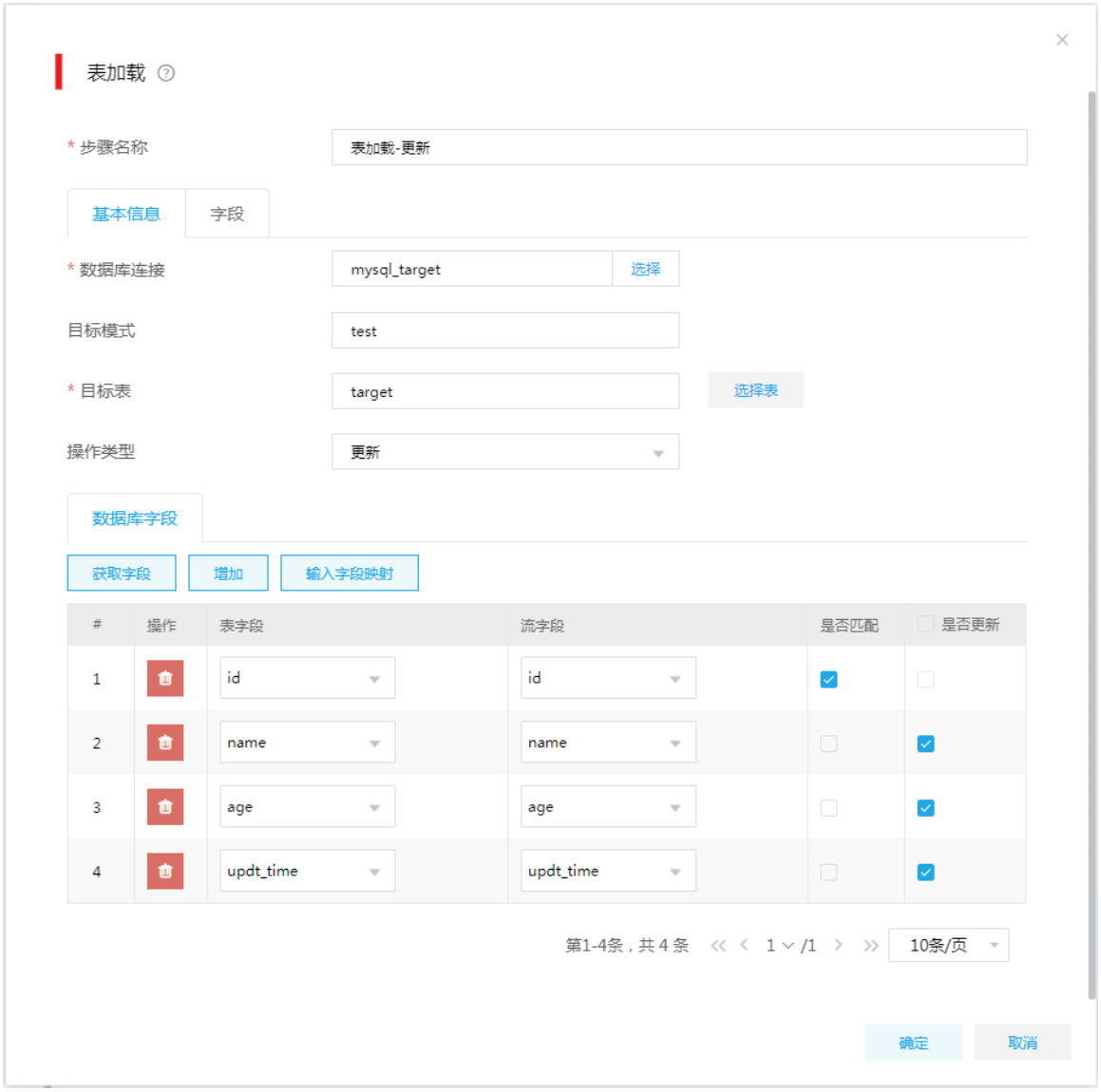
操作类型选择“插入”，用 insert 方式向目标表插入数据。

图4-182 表加载组件配置（表加载-插入）

(5) 第五步：表加载组件配置（表加载-更新）

操作类型选择“更新”，按照配置的字段映射规则，对目标表数据执行更新。

图4-183 表加载组件配置（表加载-更新）



5 服务集成典型配置案例

5.1 将数据库字段共享开放成接口场景-数据API

1. 场景描述

数据拥有者希望将数据库里的数据提供给使用者，但是为了安全考虑，又不想直接将数据库的用户名和密码提供给对方，这时可以通过调用接口的方式获取数据库里的数据。

2. 场景分析

集成平台的数据源管理可以添加数据源，服务集成可以选择需要开放的数据源的库表字段，以 restful 接口形式对外提供，返回数据格式为 JSON。

3. 示例详细步骤

(1) [数据源管理]页面，单击<新增>按钮，新增需要对外开放的数据源。

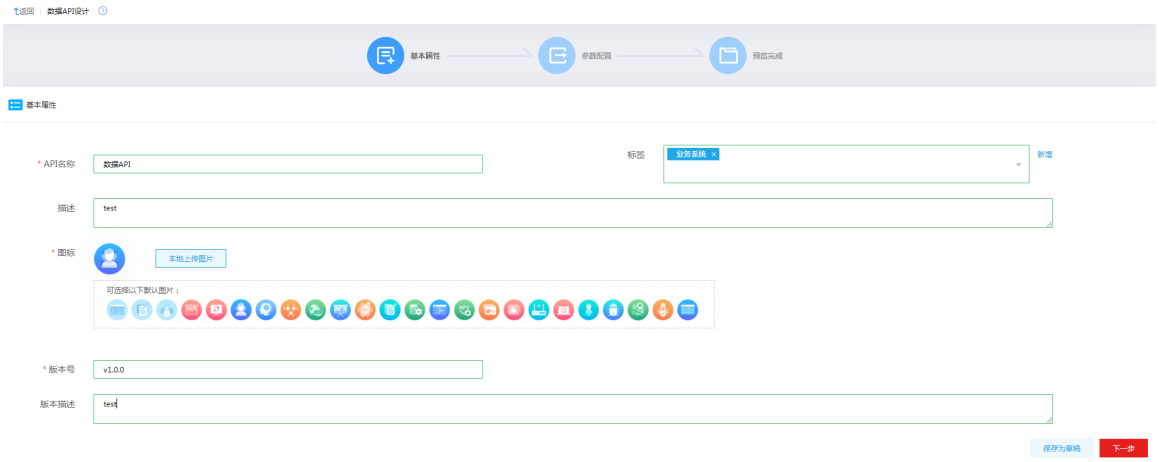
图5-1 新增数据源

(2) [服务集成/API 工厂/API 管理]页面，单击<API 注册>按钮，选择注册类型为“数据 API”。

图5-2 选择注册类型

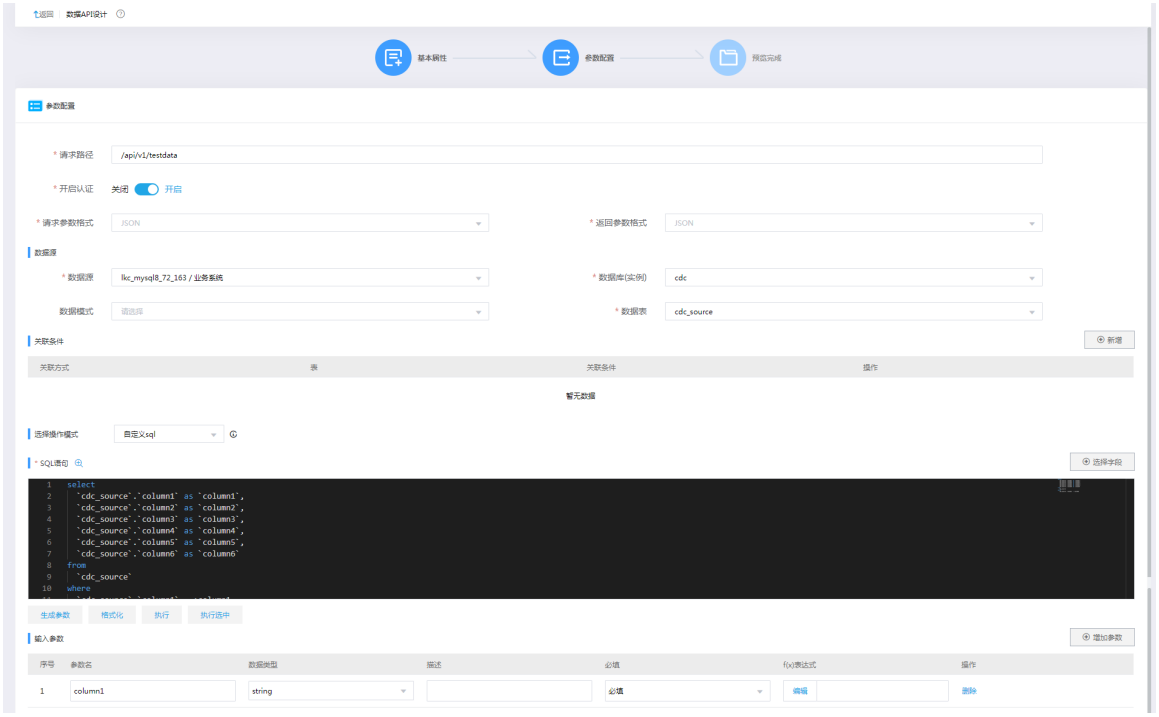
(3) 进入数据 API 设计页面，填写数据 API 相关信息。

图5-3 配置数据 API 基本信息



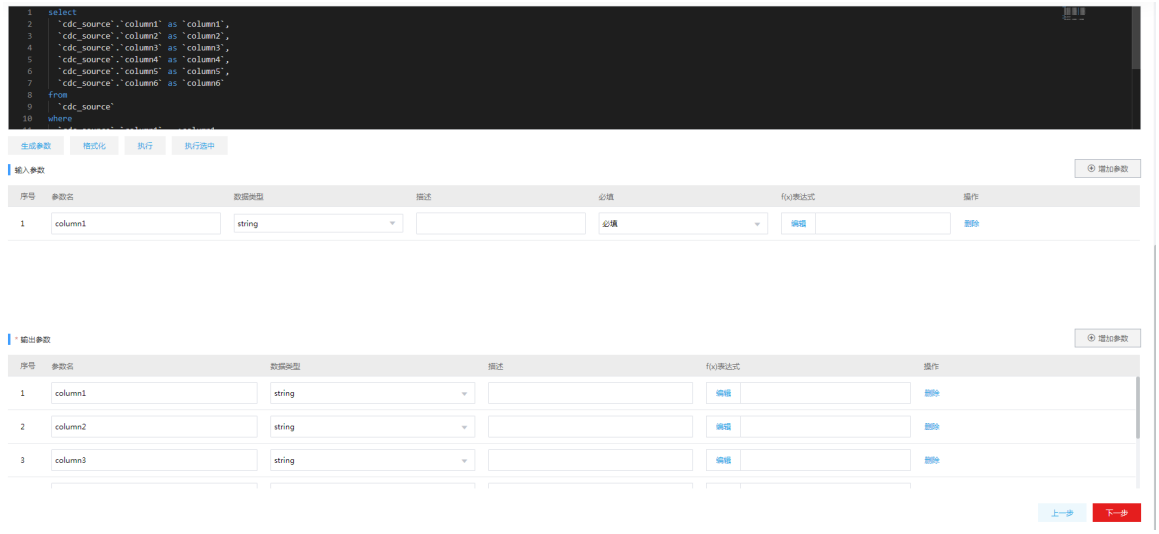
(4) 基本信息配置完成后，单击<下一步>，进入参数配置页面。配置接口的请求路径，选择刚才配置的数据源及数据源下的数据库、数据表和字段，生成 SQL 查询语句。

图5-4 参数配置



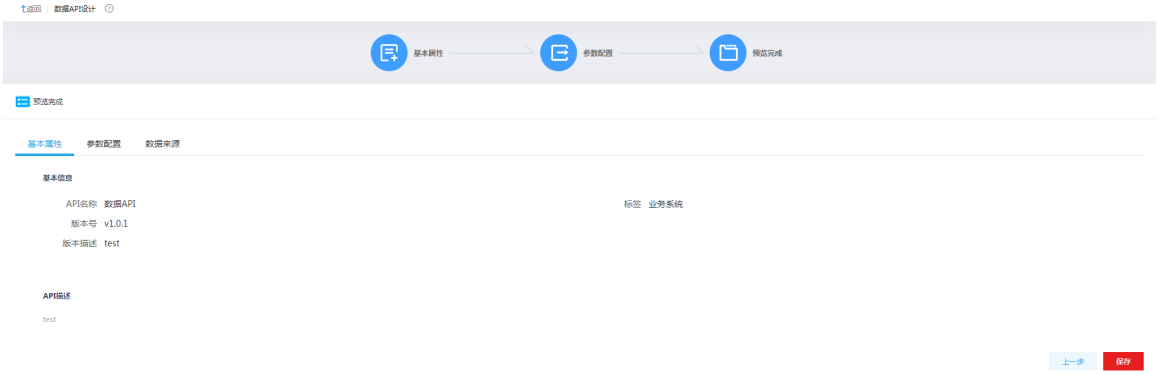
(5) 单击<生成参数>，生成接口的输入和输出参数。

图5-5 生成接口的输入和输出参数



(6) 单击<下一步>，查看配置的数据 API 的整体信息。

图5-6 查看预览信息

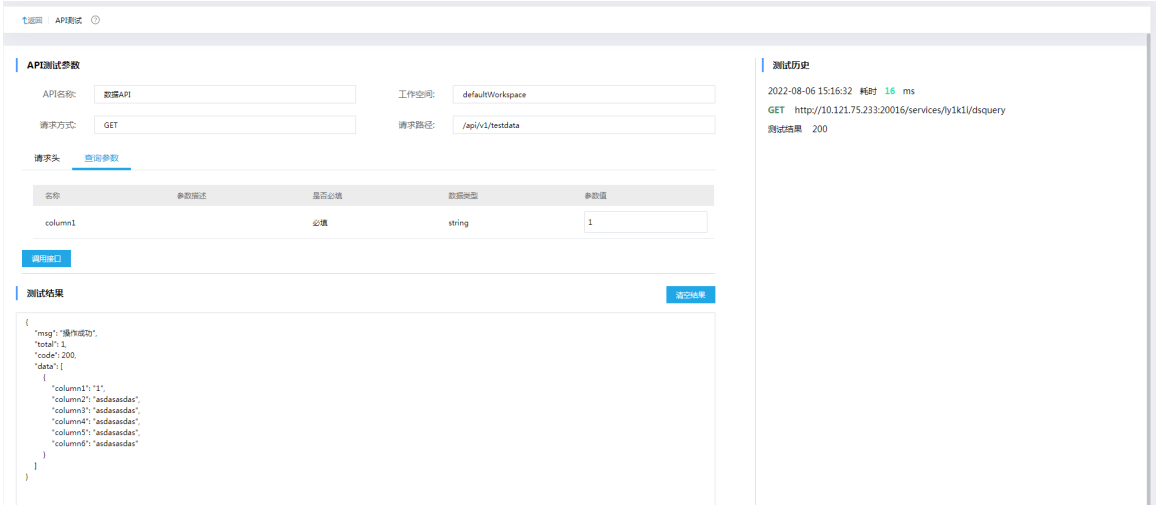


(7) 查看预览信息无误后，单击<保存>按钮，返回 API 管理页面。在 API 管理页面可以看到新增的 API。

图5-7 查看 API

(8) 单击<测试>按钮，进入 API 测试页面，填写输入输出参数，单击<调用接口>，即可查看测试结果。

图5-8 测试 API



(9) API 测试通过后，返回[API 管理]页面，单击<API 部署>，选择部署的网关节点，将 API 部署到网关。

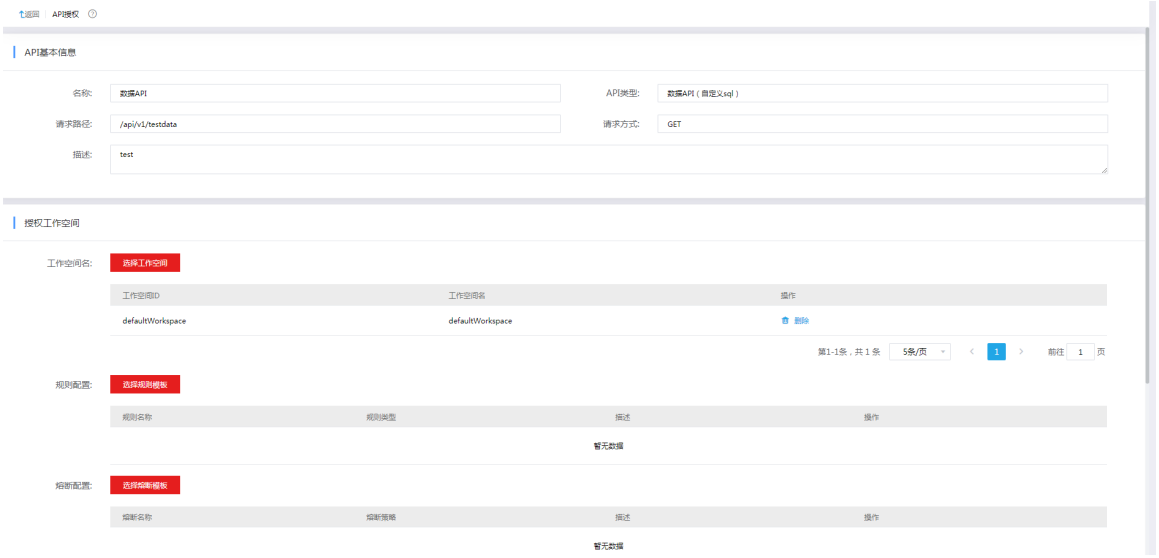
图5-9 API 部署

(10) 部署完成后，在[服务集成/API 网关/API 列表]中可以查看部署到网关的 API。

图5-10 查看部署到网关的 API

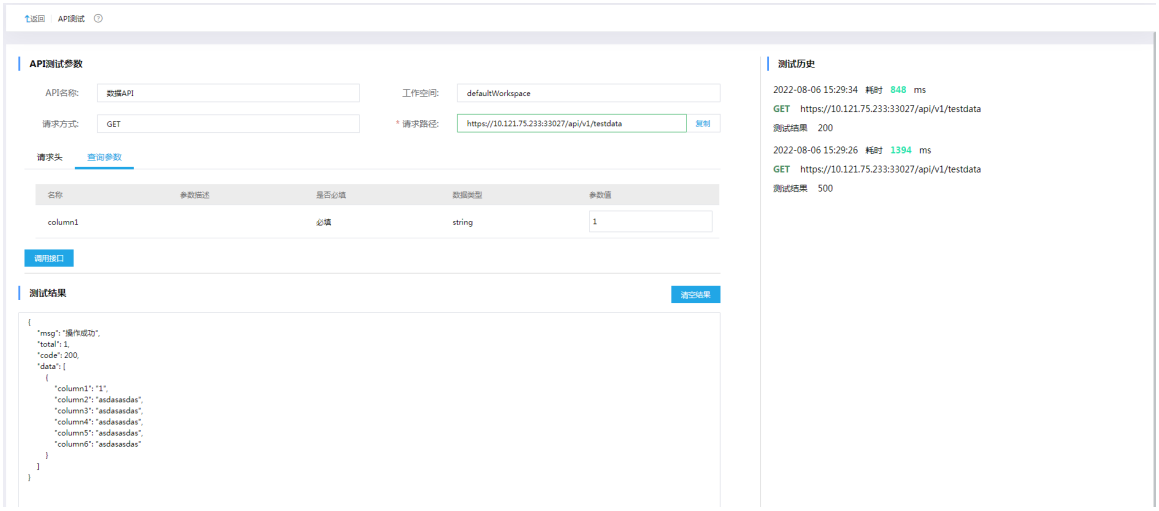
(11) [API 列表]页面，单击<授权>按钮，进入 API 授权页面，将 API 授权给需要的工作空间、认证方式等。

图5-11 API 授权



(12) [授权的 API]页面，可以查看授权给当前工作空间的 API 列表，单击<测试>按钮，进行接口测试。

图5-12 API 测试



(13) 测试页面的请求路径即网关代理后的地址，配合请求头、输入参数，可以在外部进行接口调用。

5.2 接入第三方系统接口场景-通用API

1. 场景描述

存在多个业务系统，如系统 A、B、C ……，各个业务系统间存在接口调用情况，这时首先需要打通各个系统之间的网络连接，其次一个系统调用其他业务系统时，需要和各个系统进行认证，错综

复杂，这时就需要一个公共的平台将所有系统的业务接口统一管理，对外提供统一的出口，每个业务系统只需要和接口管理平台来统一认证对接即可，简化服务间的对接工作。

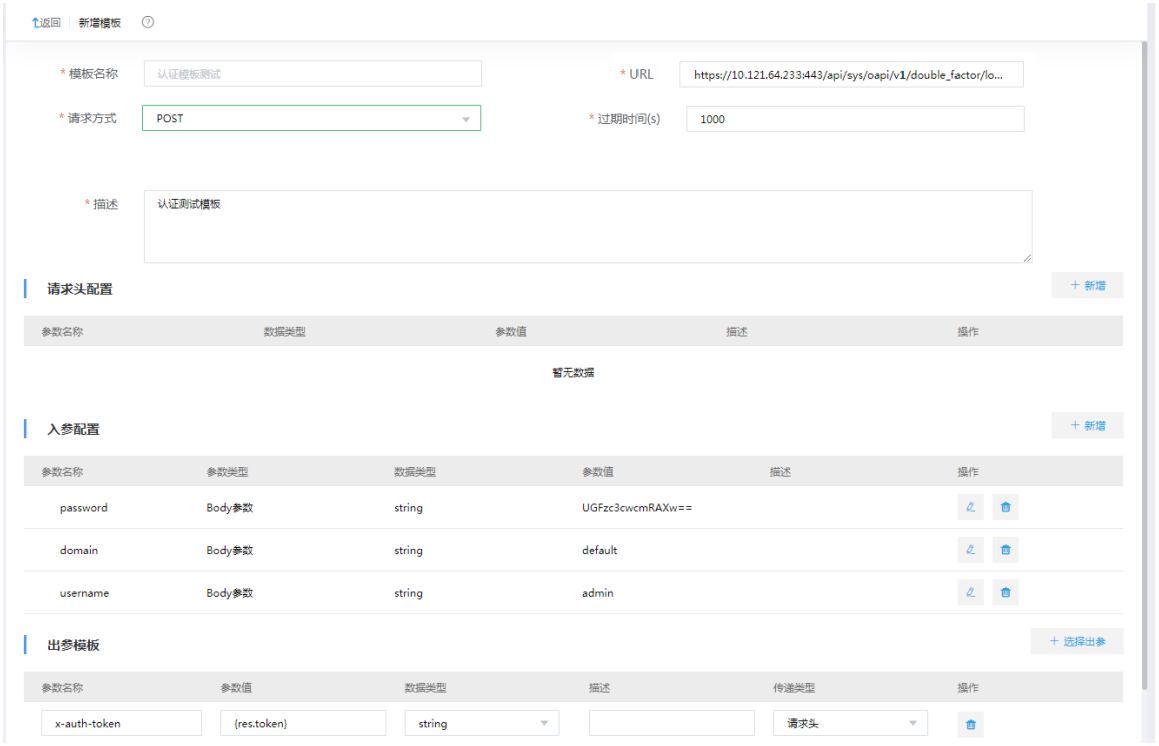
2. 场景分析

服务集成可以通过通用 API 接入第三方业务系统的接口，实现接口的注册、代理和转发。

3. 示例前置条件

第三方接口对应的接口文档，如果是带动态认证的第三方接口，需要提前添加认证模板。[服务集成/API 工厂/认证模板]页面，单击<新增>按钮，添加认证模板。

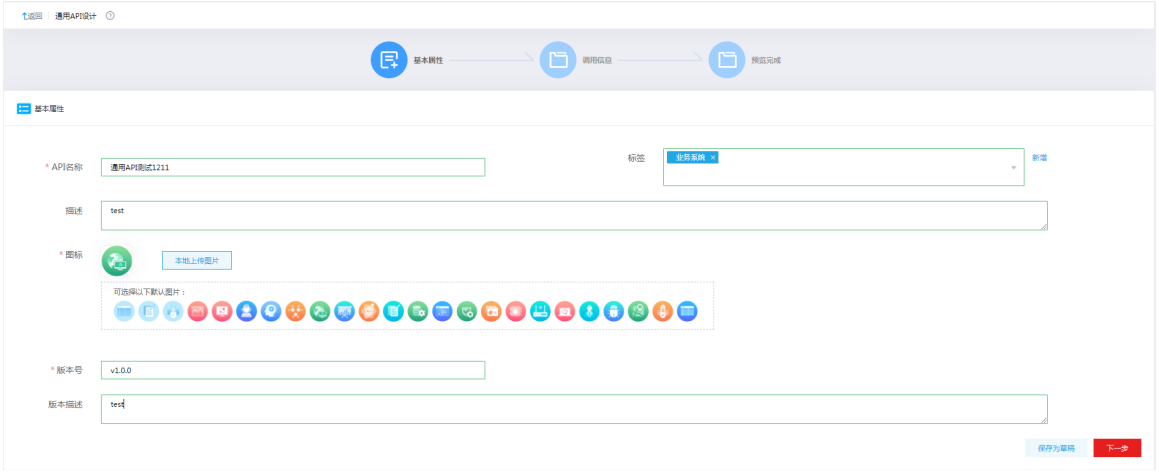
图5-13 配置认证模板



4. 示例详细步骤

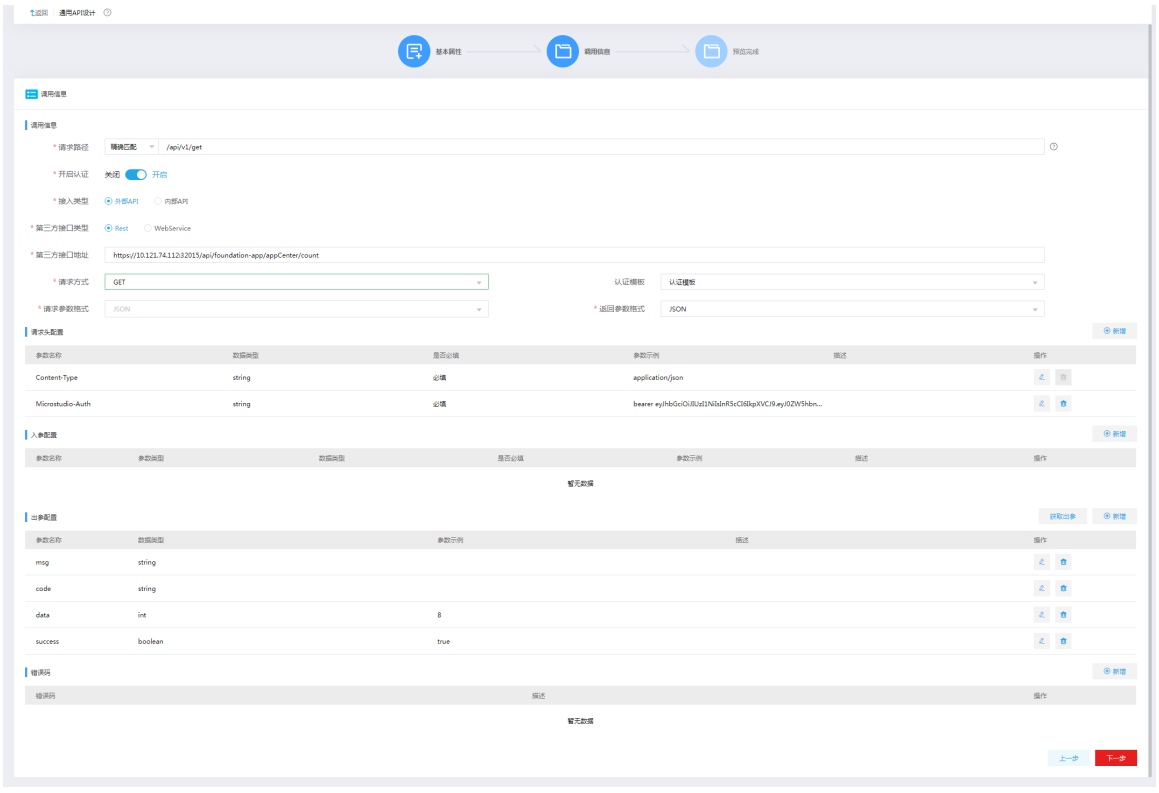
(1) [服务集成/API 工厂/API 管理]页面，单击<API 注册>按钮，选择注册类型为“通用 API”，进入通用 API 设计页面，配置通用 API 基本属性。

图5-14 配置基本属性



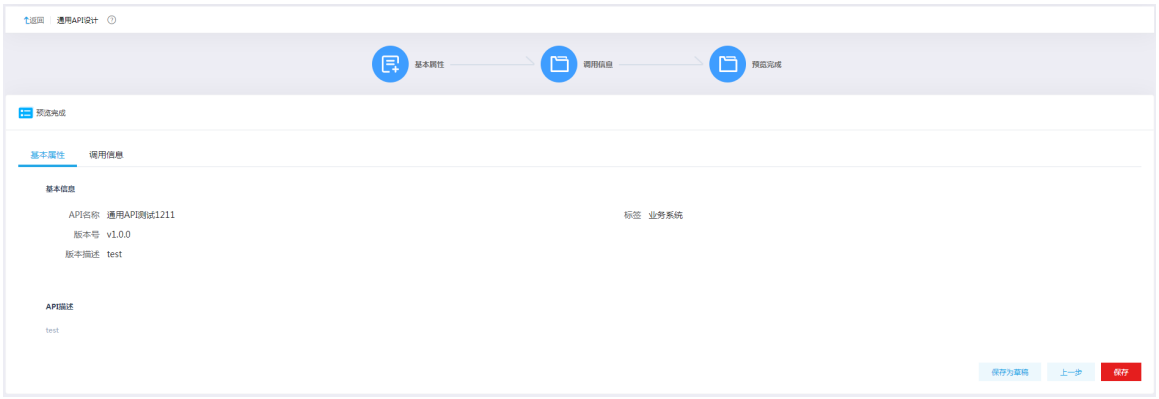
(2) 配置完基本属性后，单击<下一步>按钮，配置接口的调用信息。配置接口的请求路径，接入类型选择外部 API，配置第三方接口类型及地址，配置请求方式，如果接口带认证，则认证模板关联提前添加好的模板，配置请求头信息、输入输出参数信息，单击<下一步>。

图5-15 配置调用信息



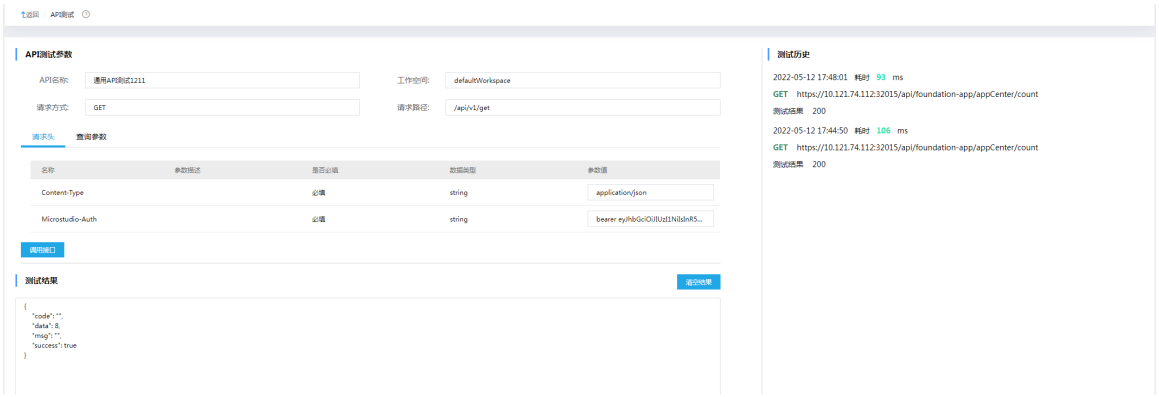
(3) 单击<下一步>，查看配置的通用 API 的整体信息。

图5-16 API 预览



(4) 单击<保存>按钮，返回 API 管理页面。在 AP 管理页面可以看到新增的 API。单击<测试>按钮，可进行接口测试。

图5-17 API 测试



(5) 接下来的部署、授权操作，可参考 5.1 3. (9)中数据 API 的操作步骤。

5.3 复杂场景下第三方接口对接-函数API

1. 场景描述

第三方接口简单的代理转发可以通过通用 API 来实现，如果涉及参数的转换或者多个接口的编排调用，通用 API 则无法实现，这时需要一种自定义编排的方法来注册接口。

2. 场景分析

函数 API 可以通过编写函数脚本来实现复杂场景下的参数转化和多接口调用编排，利用内部实现的工具类，通过 JS 脚本可以实现灵活的编排调用。

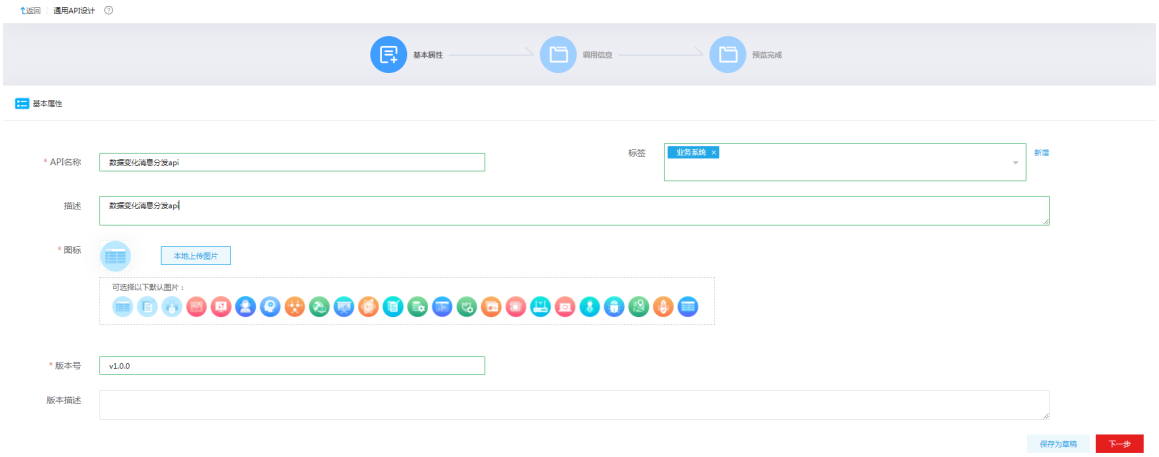
3. 示例前置条件

在[服务集成/API 工厂/环境配置/密码箱]页面及[服务集成/API 工厂/环境配置/环境变量]页面分别添加编写函数 API 时需要用到的环境变量和密码箱。

4. 示例详细步骤

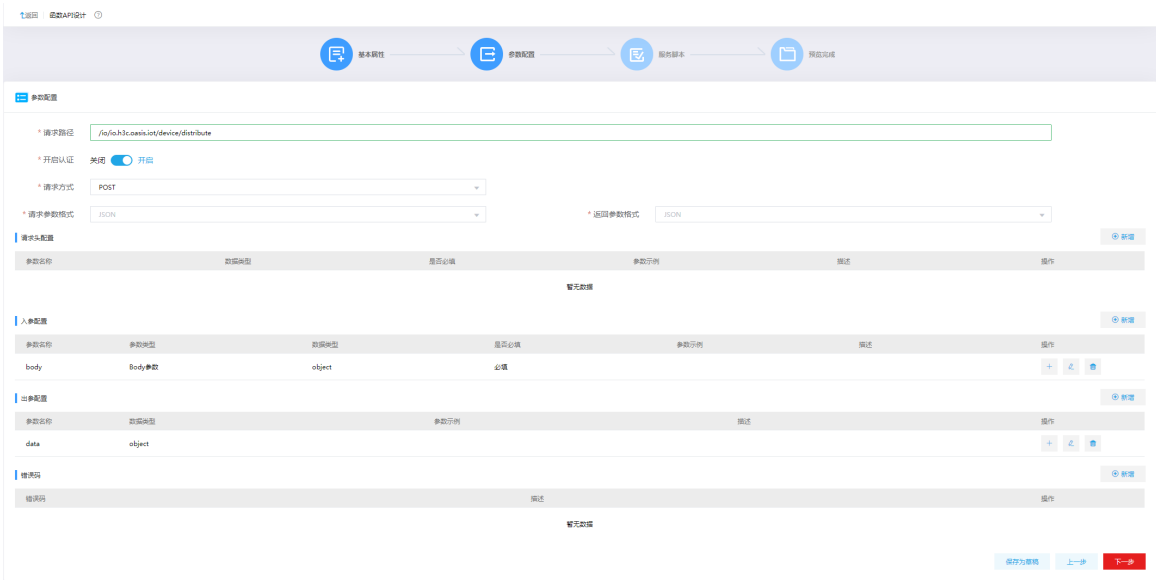
(1) [服务集成/API 工厂/API 管理]页面，单击<API 注册>按钮，选择注册类型为“函数 API”，进入函数 API 设计页面，配置函数 API 基本属性。

图5-18 配置函数 API 基本属性



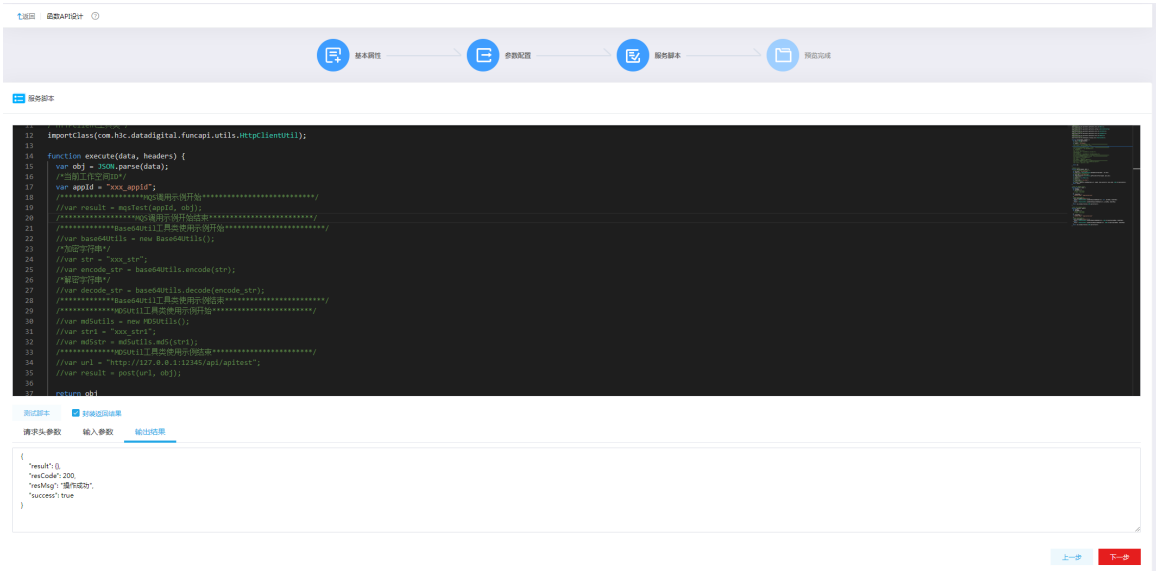
(2) 配置完基本属性后，单击<下一步>按钮，进入函数 API 的参数配置页面。

图5-19 参数配置



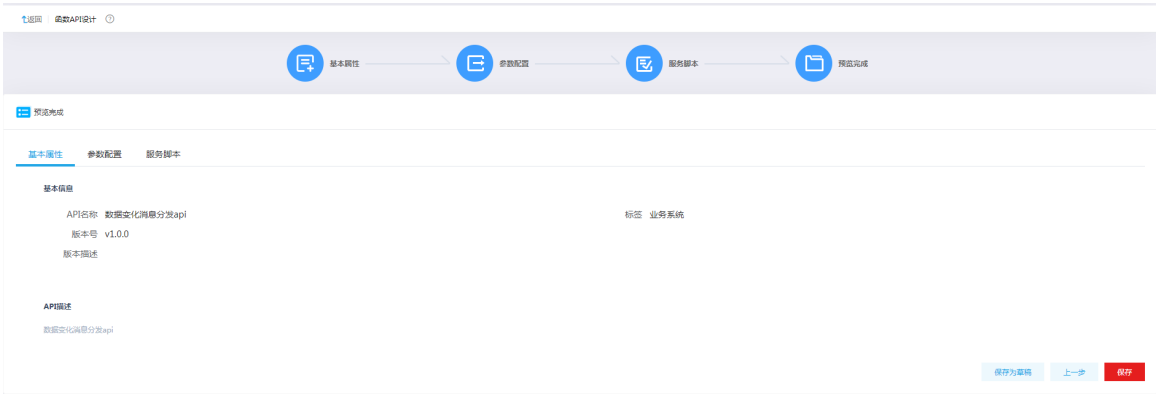
(3) 单击<下一步>，编写 JS 脚本

图5-20 编写 JS 脚本



(4) 单击<下一步>，查看配置的函数 API 的整体信息。

图5-21 预览信息



(5) 单击<保存>按钮，返回 API 管理页面。在 API 管理页面可以看到新增的 API。接下来的测试、部署、授权操作，可参考 5.1　3. (8)中数据 API 的操作步骤。

5.4 画布方式实现多接口编排场景

1. 场景描述

对于一个业务接口，如果想对接口的输出参数进行过滤，或者将多个接口的执行编排在一起，函数 API 可以实现这种场景，但是需要编写 JS 脚本，对人员要求较高，此时需要一种简便快速的方法来实现接口的编排。

2. 场景分析

服务集成的服务编排可以通过画布的方式，在页面利用拖拉拽来实现输出参数的过滤和多个接口的编排。

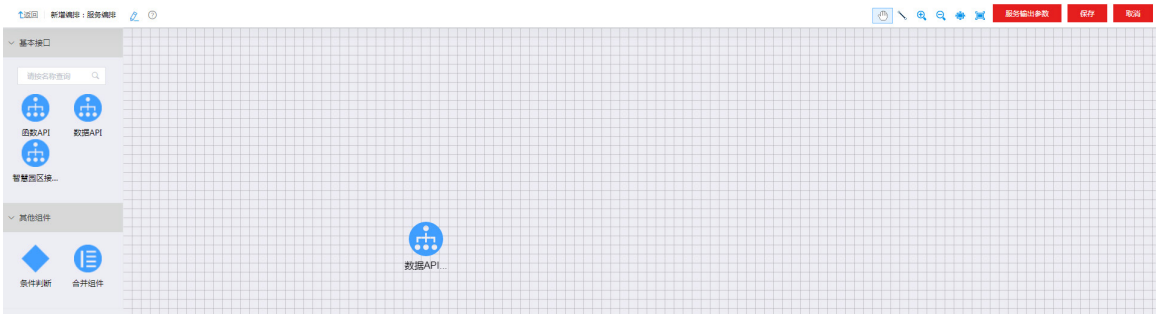
3. 示例前置条件

在[API 工厂/API 管理]页面提前注册需要进行服务编排的 API 并测试通过。

4. 示例详细步骤

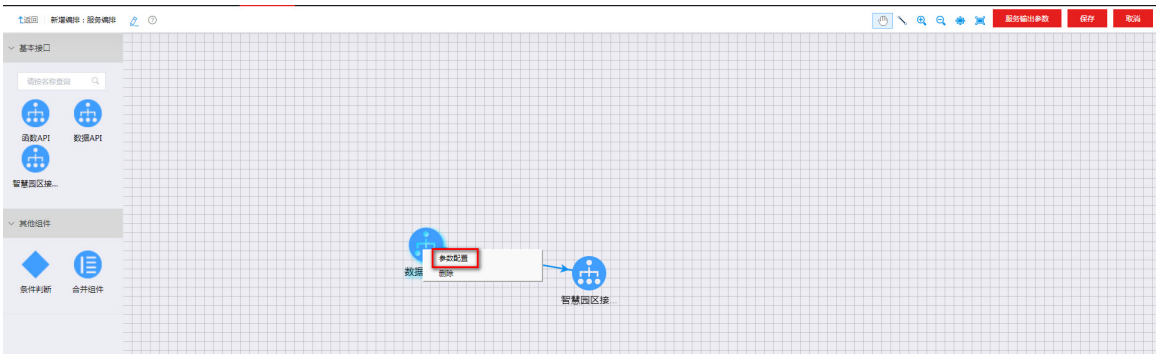
(1) [服务集成/API 工厂/服务编排]页面，单击<新增>按钮，填写基本信息，跳转到设计页面。

图5-22 服务编排设计页面



(2) 在设计页面，将需要进行服务编排的 API 拖拽到画布中央，对两个接口进行编排，一个接口的输出作为另一个接口的输入。双击接口组件进行参数配置；或者右键单击接口组件，弹出框中选择<参数配置>按钮进行输入参数配置。

图5-23 参数配置



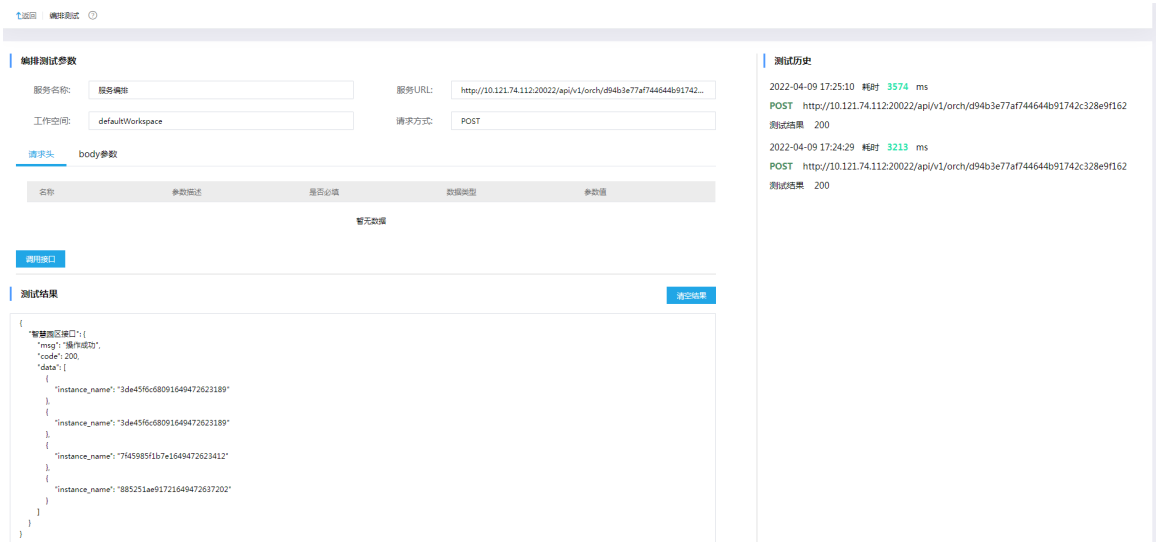
(3) 右键单击第二个 API，弹出参数配置对话框，配置第一个接口的出参为第二个接口的入参。入参配置完成后，单击页面右上角<服务输出参数>，配置服务出参。

(4) 另外服务编排支持“条件判断”和“合并组件”操作。

- 合并组件：目前仅支持数据 API 的结果合并，合并组件主要是对主键和外键的配置（也就是关联字段的配置）。合并组件连接两个父节点后，鼠标右键单击合并组件，弹出窗中选择“合并字段”可进行参数配置。
- 判断组件：判断组件主要是对任务执行流程的控制。判断组件连接一个父节点后，鼠标右键单击判断组件，弹出窗中选择“选择字段”进入选择字段页面，单击<参数选择>按钮进行判断字段的选择。然后鼠标右键单击判断组件和子节点之间的连线，可在连线上添加判断条件。

(5) 服务编排完成后，用户可在[服务集成/API 工厂/服务编排]页面，单击对应服务操作栏的<测试>按钮，对编排后的服务进行测试，确保编排正确性。

图5-24 对编排后的服务进行测试。



5.5 函数API多接口编排-核酸检测结果查询

1. 场景描述

查询个人核酸检测结果时，A 接口和 B 接口都可以查询检测结果，客户希望优先通过 A 接口查询检测结果，如果查询不到，则调用 B 接口查询检测结果，为了方便使用，要求将两个接口的返回字段合并在一起返回。

2. 场景分析

集成平台服务集成的函数 API 可以通过编写 JS 代码对多接口调用进行编排，如果调用第一个接口未查询到检测结果，则调用第二个接口进行查询，然后将两个接口的返回字段进行合并返回。

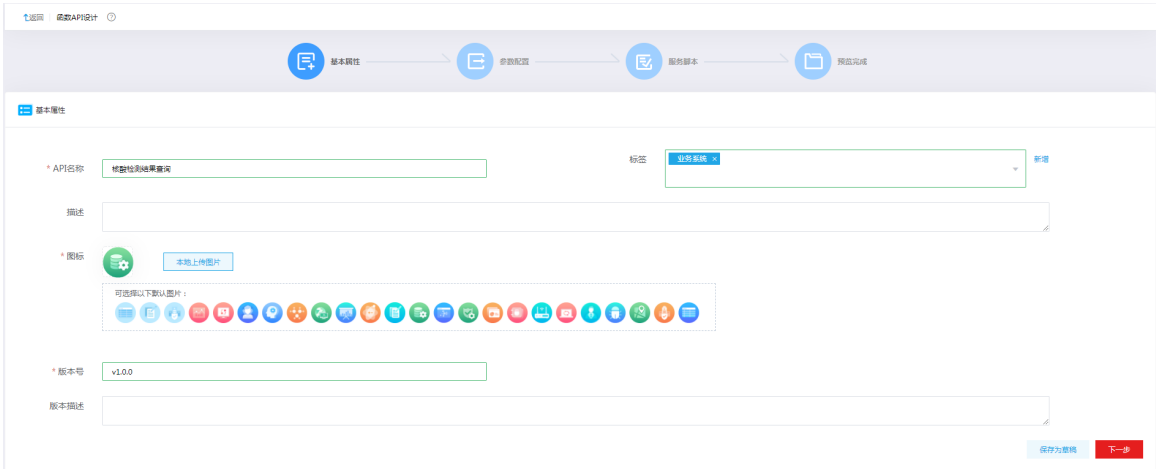
3. 示例前置条件

在[服务集成/API 工厂/密码箱管理]页面及[服务集成/API 工厂/环境配置]页面分别添加编写函数 API 时需要用到的环境变量和密码箱。

4. 示例详细步骤

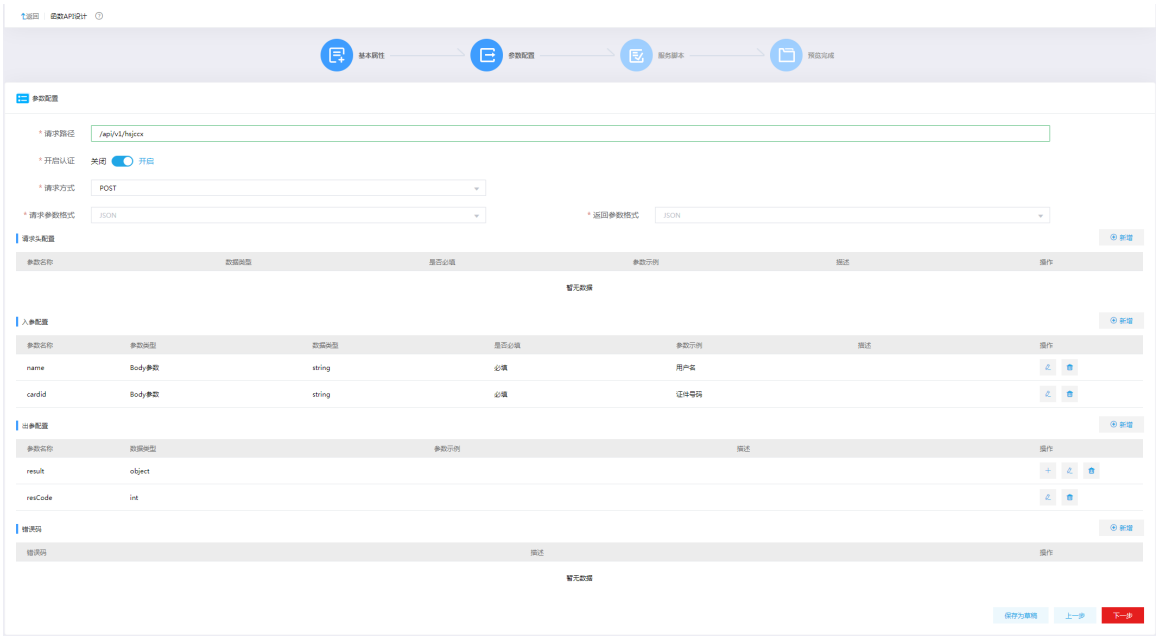
(1) [服务集成/API 工厂/API 管理]页面，单击<API 注册>按钮，选择注册类型为“函数 API”，进入函数 API 设计页面，配置函数 API 基本属性。

图5-25 配置函数 API 基本属性



(2) 配置完基本属性后，单击<下一步>按钮，进入函数 API 的参数配置页面。参数配置姓名和证件号码。

图5-26 参数配置

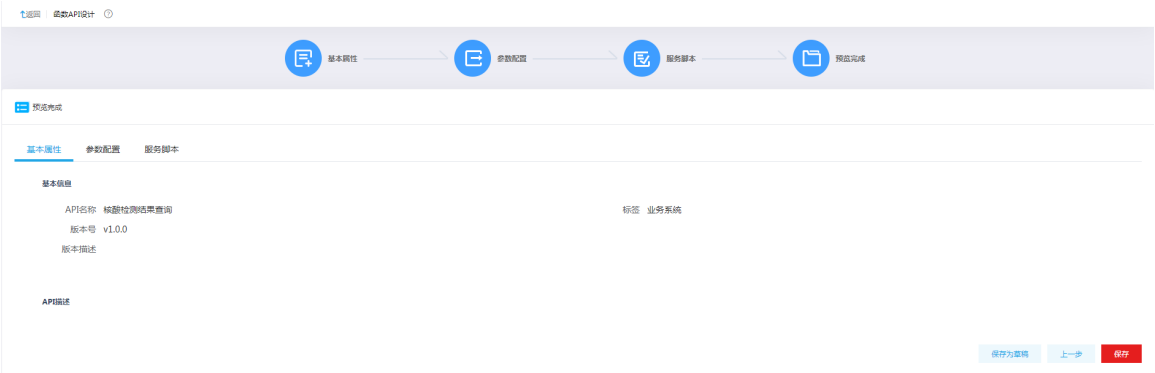


(3) 单击<下一步>，编写 JS 脚本。通过编写 JS 代码对多接口调用进行编排，如果调用第一个接口未查询到检测结果，则调用第二个接口进行查询，然后将两个接口的返回字段进行合并返回。

图5-27 编写 JS 脚本。

(4) 单击<下一步>，查看配置的函数 API 的整体信息。

图5-28 预览信息



(5) 单击<保存>按钮，返回 API 管理页面。在 API 管理页面可以看到新增的 API。接下来按照 API 的通用操作进行测试、部署、授权等操作，可参考 5.1　3. (8)中数据 API 的操作步骤。

(6) 部署并授权完成后，用户即可调用该函数 API 接口进行使用。

6 消息集成典型配置案例

6.1 作为消息中间件的生产消费场景

消息集成系统作为消息中间件使用，支持 Kafka 客户端进行消息的生产消费，这是消息集成使用最多的场景，以该场景为例，进行实操案例的描述。

1. 场景描述

用户生产、消费消息到消息集成的 Kafka 集群中。

2. 操作示例

(1) [Topic 管理/Topic 列表]页面，单击<新建>按钮，弹出新建窗口，创建 Topic。

图6-1 创建 Topic

(2) [Topic 管理/Topic 列表]页面，单击 Topic 列表中的<权限>按钮，弹出 Topic 权限配置窗口，可为指定工作空间赋予该 Topic 的读写权限。

图6-2 配置 Toiic 权限

(3) 为工作空间分配 Topic 权限后，在 Topic 权限配置窗口，单击<配置信息>按钮，可查看该 Topic 使用相关的信息。

图6-3 查看配置信息

(4) 使用 Topic 相关配置信息，用户可连接对应 Topic，然后自行构造 Kafka 生产、消费客户端。

7 资产市场典型配置案例

7.1 资产市场订阅使用接口场景

1. 场景描述

集成平台系统内存在多个组织，组织 A 发布的 API 通过主动授权时只能授权给组织 A 内的工作空间使用，组织 B 如果想要使用组织 A 发布的 API，需要主动去订阅使用。

2. 场景分析

服务集成发布的 API，部署到网关后，可以单击操作列的<上线>按钮，申请 API 上线，经过管理员审批后，完成上线。上线后的 API 可以在资产市场展示，平台内的所有用户都可以看见，并且可以进行收藏和订阅。

3. 示例前置条件

- 已完成 API 的部署，因为 API 部署后才能进行上线。
- 用户资产在上线到资产市场时需指定资产来源。系统默认配置了一些资产来源信息，系统管理员也可以根据实际需要在[资产市场/资产配置/资产来源]页面进行新建。
- 资产在数字平台中上线时需要指定资产来源，资产来源会关联组织管理员，资产上线后会展示在组织管理员的资产列表中，由对应的组织管理员进行管理。

4. 示例详细步骤

(1) [API 工厂/API 管理]页面，将已经部署好的 API 进行上线。API 管理页面，单击<更多>按钮，在下拉框中选择上线，系统会自动发出上线审批流程，API 状态变为“上线审批中”。注意：上线时需要选择资产来源，上线后该资产会被资产来源关联的组织管理员进行管理。

图7-1 API 上线

(2) 管理员可在[个人中心/待办审批]页面查看上线流程，然后进行审批。

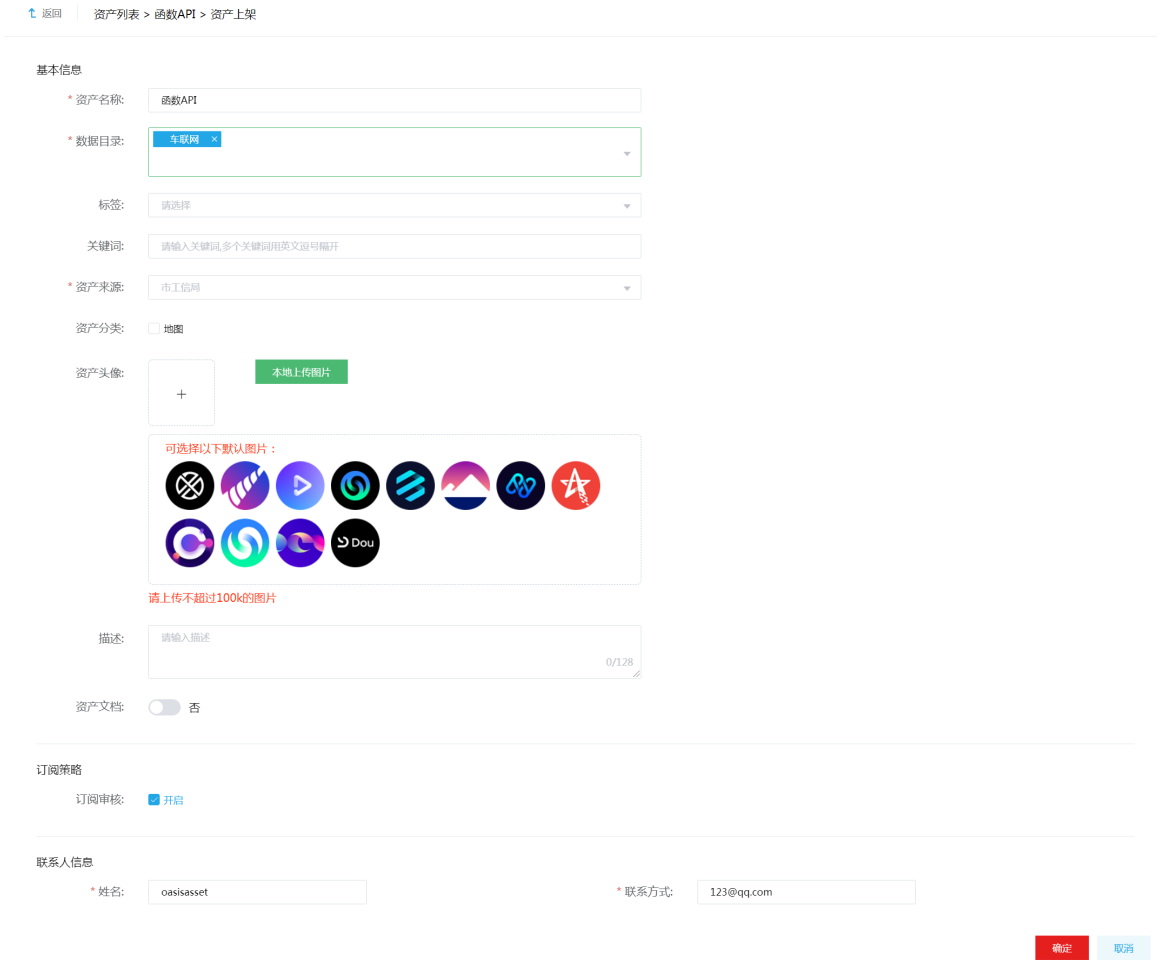
图7-2 审批上线流程

(3) 资产上线后，会展示在组织管理员的资产列表中。组织管理员可在[个人中心/我的资产/资产列表]中对资产进行上架。

图7-3 资产列表

(4) 组织管理员可将资产上架到资产市场。上架时可配置该资产标签、资产来源、是否开启订阅审核等。

图7-4 上架资产

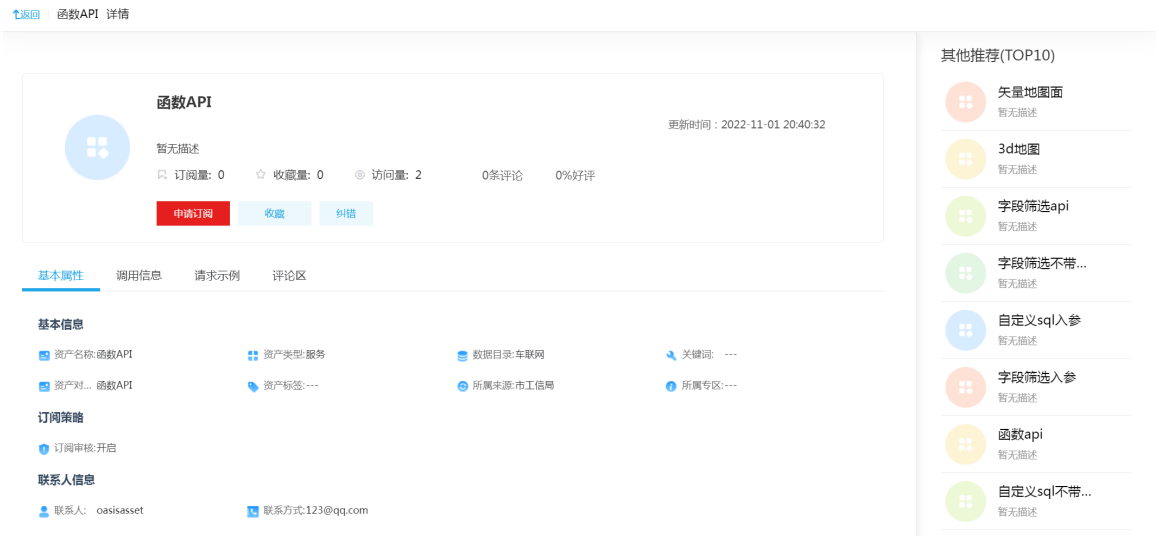


(5) 资产上架后，会展示在资产市场中，用户可在资产市场中查看已上架的资产。

图7-5 公共资产页面

(6) 单击资产，可查看资产详情。资产详情页面，可以查看资产的基本属性和调用信息。

图7-6 API 详情



(7) 单击<申请订阅>按钮，弹出订阅对话框，选择应用，配置订阅周期，填写申请理由，进行订阅申请。系统会发出一个订阅流程给组织管理员进行审批。

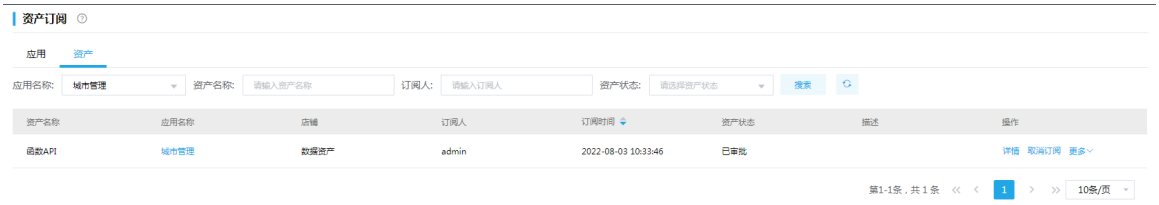
图7-7 订阅申请

(8) 组织管理员可在[个人中心/待办审批]页面查看到待审批的流程。

图7-8 待办审批

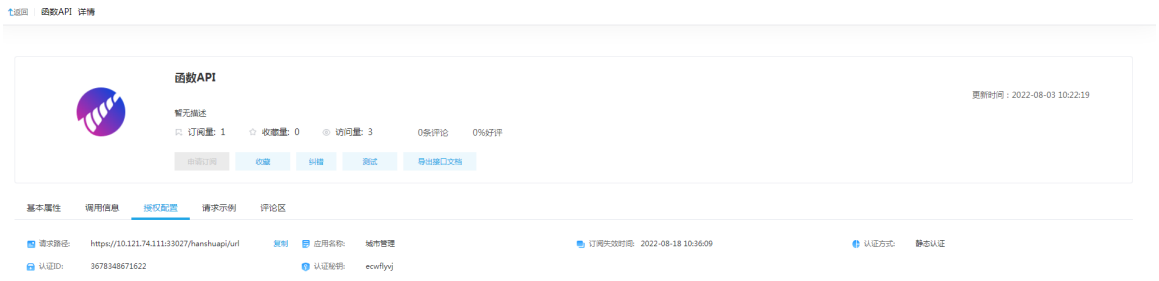
(9) 组织管理员审批通过后，订阅者在[个人中心/资产订阅]页面可以查看审批通过的资产。

图7-9 订阅的资产



(10) 在我的订阅页面的列表中，单击待查看详情资产对应操作列中的<详情>按钮，进入资产详情页面。未订阅前，用户可以查看资产相关介绍信息；订阅后用户可以查看资产具体的使用信息，用户可通过这些信息使用资产。

图7-10 资产详情



8 典型应用案例

8.1 医保云案例

8.1.1 应用现状

在全国医保的建设要求中，省级医疗保障局将根据国家医疗保障局信息化建设指导意见，坚持医疗保障信息化建设“一盘棋”的原则，依托省级医疗保障平台与国家医疗保障平台之间的协作联通。建设医保云面临如下挑战：

- 时效：数据中台需实现在医保决策分析系统中，支撑实时监控和 T+1 两种类型医保指标的统计和显示
- 数据库响应：需满足高并发、低延迟、实时计算要求，为典型的互联网 OLTP 场景
- 数据质量：需按照 4000+规范要求进行适配和代码转变，以保证提交上级的数据符合质量要求
- 复杂业务场景：数据库需满足海量数据场景下交易型事务处理

8.1.2 解决方案

使用数字平台构建数据中台，从各医疗保障相关数据源中采集或接入数据，并由数据管控和数据开发功能对数据进行处理，然后通过 API 接口和数据库接口进行数据服务集成，实现对上层数据应用提供数据服务。

其中，数据处理部分通过实时计算从实时数据流提取出高价值密度的结构化数据，通过离线计算从结构化、半结构化和非结构化基础数据中提取出高价值密度的结构化数据，并通过数据仓库的分层建模整理数据，最终实现数据整合，统一标准。

医保数据处理全流程：

(1) 历史数据迁移：集成平台创建 DataX 作业，将原医保平台的历史数据迁移至新的医保云平台。
   - Oracle->生产库

(2) 建仓：在 BDP 大数据平台上建立统一的 Hive 数据仓库对所有生产数据进行管理，最大化发挥数据的价值。
   - 生产库->Hive（历史全量）
   - 生产库->Hive（T+1）

(3) 交换：生产数据导入 Hive 数仓后，通过在集成平台中创建 ETL 作业，定期从仓库中抽取数据，向省交换库做数据同步。
   - Hive->省交换库（T+1）
   - 生产库->省交换库（5 分钟定时）

8.1.3 示例详细流程

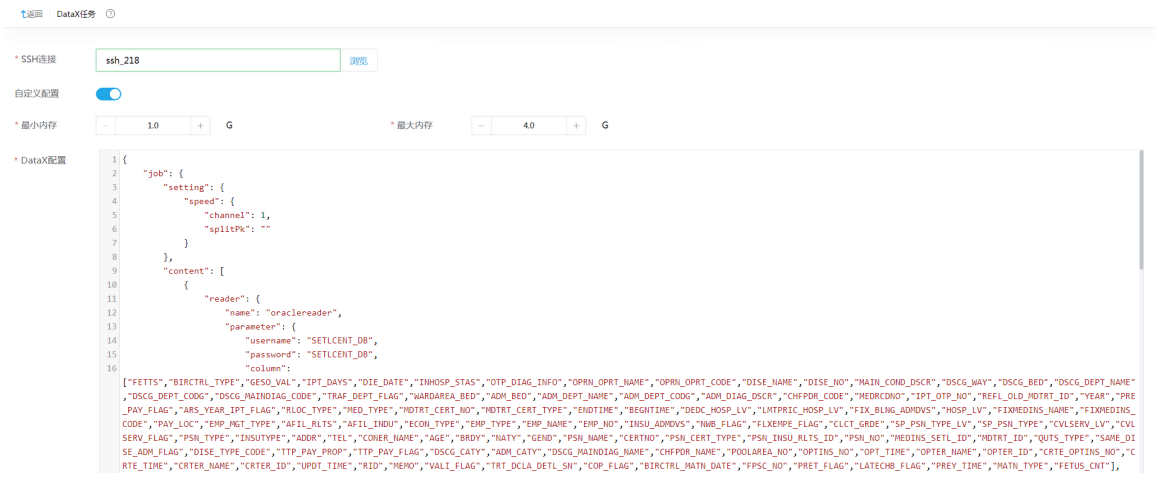
1. 步骤一：历史数据迁移

原医保平台的历史数据需要迁移至新的医保云平台，一般情况下的数据流向为：Oracle -> DRDS。

集成平台支持创建 DataX 作业，适用于关系型数据库间大宗数据的迁移。

(1) 进入[作业管理/作业定义]页面，单击<新建作业>按钮，新建作业。将 DataX 任务添加到作业设计画布中，双击任务名称后跳转至 DataX 任务设计页面，用户可根据实际需要进行任务设计。

图8-1 DataX 任务



DataX 任务配置参数说明如下，用户可根据实际情况进行配置。

- SSH 连接：单击<浏览>按钮，弹出数据源连接窗口，选在需要进行操作的 SSH 连接。
- 自定义配置：自定义配置关闭时，页面输入项主要划分数据来源、数据去向、字段映射、高级配置四部分内容。自定义配置开启时，页面输入项为最小内存、最大内存及 DataX 配置。
  - 最小/最大内存：配置任务最小/最大内存，最小内存默认 1G，最大内存默认 4G，最小可选 1G，最大可选 32G，允许填写小数。
  - DataX 配置：根据需要在 DataX 配置内容框中填写 DataX 执行的 JSON 内容。
- DataX 配置：根据需要在 DataX 配置内容框中填写 DataX 执行的 JSON 内容。

DataX 配置示例（仅供参考）：

```
{
    "job": {
        "setting": {
            "speed": {
                "channel": 1,
                "splitPk": ""
            }
        },
        "content": [
            {
                "reader": {
                    "name": "oraclereader",
                    "parameter": {
                        "username": "SETLCENT_DB",
                        "password": "SETLCENT_DB",
                        "column":
["FETTS","BIRCTRL_TYPE","GESO_VAL","IPT_DAYS","DIE_DATE","INHOSP_STAS","OTP_DIAG_INFO","
OPRN_OPRT_NAME","OPRN_OPRT_CODE","DISE_NAME","DISE_NO","MAIN_COND_DSCR","DSCG_W
AY","DSCG_BED","DSCG_DEPT_NAME","DSCG_DEPT_CODG","DSCG_MAINDIAG_CODE","TRAF_DEPT
_FLAG","WARDAREA_BED","ADM_BED","ADM_DEPT_NAME","ADM_DEPT_CODG","ADM_DIAG_DSCR","
CHFPDR_CODE","MEDRCDNO","IPT_OTP_NO","REFL_OLD_MDTRT_ID","YEAR","PRE_PAY_FLAG","AR
S_YEAR_IPT_FLAG","RLOC_TYPE","MED_TYPE","MDTRT_CERT_NO","MDTRT_CERT_TYPE","ENDTIME
","BEGNTIME","DEDC_HOSP_LV","LMTPRIC_HOSP_LV","FIX_BLNG_ADMDVS","HOSP_LV","FIXMEDINS
_NAME","FIXMEDINS_CODE","PAY_LOC","EMP_MGT_TYPE","AFIL_RLTS","AFIL_INDU","ECON_TYPE","
EMP_TYPE","EMP_NAME","EMP_NO","INSU_ADMDVS","NWB_FLAG","FLXEMPE_FLAG","CLCT_GRDE","
SP_PSN_TYPE_LV","SP_PSN_TYPE","CVLSERV_LV","CVLSERV_FLAG","PSN_TYPE","INSUTYPE","ADD
R","TEL","CONER_NAME","AGE","BRDY","NATY","GEND","PSN_NAME","CERTNO","PSN_CERT_TYPE","P
SN_INSU_RLTS_ID","PSN_NO","MEDINS_SETL_ID","MDTRT_ID","QUTS_TYPE","SAME_DISE_ADM_FLA
G","DISE_TYPE_CODE","TTP_PAY_PROP","TTP_PAY_FLAG","DSCG_CATY","ADM_CATY","DSCG_MAIN
DIAG_NAME","CHFPDR_NAME","POOLAREA_NO","OPTINS_NO","OPT_TIME","OPTER_NAME","OPTER_
ID","CRTE_OPTINS_NO","CRTE_TIME","CRTER_NAME","CRTER_ID","UPDT_TIME","RID","MEMO","VALI_
FLAG","TRT_DCLA_DETL_SN","COP_FLAG","BIRCTRL_MATN_DATE","FPSC_NO","PRET_FLAG","LATEC
HB_FLAG","PREY_TIME","MATN_TYPE","FETUS_CNT"],
                        "splitPk": "",
                        "connection": [
                            {
                                "table": [
                                    "SETLCENT_DB.MDTRT_D"
                                ],
                                "jdbcUrl": [
                                    "jdbc:oracle:thin:@//101.12.54.35:1521/orcl"
```

2. 步骤二：建仓

医保系统包含有核心经办、基金监管在内的多个业务子系统，数据分散在各个业务数据库中。需要建立统一的数据仓库对所有生产数据进行管理，最大化发挥数据的价值。

集成平台支持创建 Sqoop 任务，适用于关系型数据库和 HDFS、Hive、HBase 等大数据组件之间的数据迁移。同时，集成平台的 ETL 任务也对 HDFS、Hive、HBase 等做了适配，通过创建 ETL 任务，进行一些简单的拖拉拽和配置，既可实现大数据组件中数据与关系型数据库或者文本数据的相互转换。

目标

建仓是数字平台承载的核心业务之一，分为初始全量数据导入和 T+1 增量导入。通过数字平台创建 Hive 数仓（分层），并准确、准时将生产数据抽取并写入 Hive 数仓。

示例前置条件

已提前部署好 BDP 大数据平台，并在大数据平台上安装好了 Hive。

建库建表

数据仓库使用 Hive，数据分层管理。主要包括：STG、ODS、DWD、ADS 等。

- STG 层：临时数据层，存放原始数据，直接加载原始数据，数据保持原貌不做处理。数据根据 updt_time 字段按天分区（同一条数据在 STG 层可能会有多条记录，分布在不同的分区中）。
- ODS 层：结构和粒度与原始表保持一致，对 STG 层数据进行简单清洗（去除空值、脏数据、超过极限范围的数据）。数据根据 crte_time 字段按年或月分区（同一条数据在 ODS 层只有一条，经过行级质检、表级质检）。
- DWD 层：数据来源于 ODS，根据子系统进一步分区管理，以 ODS 层为基础伴随业务需要进行轻度汇总。
- ADS 层：数据集市层，为各种统计报表提供数据，供上层应用使用。

(1) 建库

登录任意 DE 集群节点，连接到 Hive，若集群开启了 Kerberos 认证，需要使用集群超级用户进行认证。使用如下命令创建几层数据库（仅为示例，用户根据实际情况进行创建）：

```
create database stg_prd;
create database ods_prd;
create database dwd_prd;
```

(2) 登录任意 DE 集群节点，连接到 Hive（或者通过数据管理平台进行可视化建表）。使用如下语句创建 Hive 表（仅为示例，用户根据实际情况创建需要的表，本应用案例中创建了 stg_prd.stg_cep_stcdb_mdtrt_d、ods_prd.ods_cep_stcdb_mdtrt_d、dwd_prd.dwd_dgn_mdtrt_d。如下为 ods 的建表语句，ods、stg 及 dwd 的建表语句具体内容可参见 10.3　ods、stg 及 dwd 建表语句）：

```
CREATE TABLE `ods_prd.ods_cep_stcdb_mdtrt_d`(
  `mdtrt_id` string COMMENT '就诊ID',
  `medins_setl_id` string COMMENT '医药机构结算ID',
  `psn_no` string COMMENT '人员编号',
  `psn_insu_rlts_id` string COMMENT '人员参保关系ID',
----------用户可根据实际需要自由扩展字段----------
  `emp_type` string COMMENT '单位类型',
  `dty_flag` string COMMENT '是否脏数据(0否1是)',
  `local_dty_flag` string COMMENT '本地规则-脏数据标识(0否1是)',
  `exch_updt_time` timestamp COMMENT '入仓时间')
COMMENT ''
PARTITIONED BY (
  `dt` string,
  `region` string);
```

全量迁移

为了支撑医保云上层的监管、决策类子系统的正常运行，需要将近几年乃至全量的医保生产数据一次性导入到 Hive 仓库中。

关键点：部分表的历史数据较多，覆盖的时间范围较大，STG 层要求基于数据的更新时间按天分区，所以在执行历史数据初始导入时涉及的分区数会很多，需要重点关注是否存在数据倾斜，并相应调整运行时参数。如：某地医保项目在做历史数据导入时，单表 1.4 亿数据，以 updt_time 字段按天分区后发现位于某一天的数据达到了 5000w+，Sqoop 单个 map 无法在默认的 SQL 执行超时（1 小时）之前完成这一天的数据抽取，因此在运行前需要优化一部分参数，包括数据库服务端调大超时时间（DRDS 中参数为 sqlTimeout）、Sqoop 命令的 jdbc url 添加自定义参数。如：socketTimeout=3600000。

数据流向：生产 -> STG 层 -> ODS 层 -> DWD 层 -> ADS 层。整体操作思路如下：

(1) 执行 Sqoop 命令从生产库抽取全量数据，导入 STG。

(2) 执行 Spark SQL 对 STG 数据进行质检，数据分流至 ODS 和脏库。

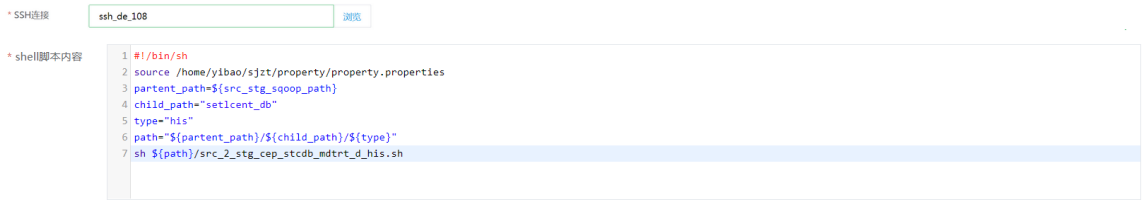
(3) 执行 Spark SQL 将 ODS 数据导入 DWD。

具体步骤如下：

(1) 从生产库抽取全量数据，导入 STG 层。使用集成平台创建 Shell 任务，执行预先准备好的数据同步脚本。

   a. 进入[作业管理/作业定义]页面，单击<新建作业>按钮，新建作业。将 Shell 任务添加到作业设计画布中，配置完后双击任务图标跳转至 Shell 任务设计页面，用户可根据实际需要进行任务设计。

图8-2 Shell 任务类型



b. Shell 任务配置参数说明如下，用户可根据实际情况进行配置。
   - SSH 连接：单击<浏览>按钮，弹出数据源连接窗口，选在需要进行操作的 SSH 连接。
   - Shell 脚本内容：在 Shell 脚本内容框中输入脚本内容。

c. Shell 脚本中 property.properties 用于定义一些脚本执行参数，如：数据库连接信息、公共脚本路径等，示例如下：

```
#生产库连接信息
setlcent_db_src_db_url=jdbc:mysql://101.12.60.51:3323
setlcent_db_src_db_user=test
setlcent_db_src_db_pwd=passwd
#stg层的数据库名
target_stg_db=stg_prd
#各类预置脚本的存放目录
run_job_shell_path=/home/yibao/sjzt/shell
#封装了通过beeline客户端执行hive sql的命令
run_job_shell_name=run_job.sh
#封装了通过spark客户端执行spark sql的命令
run_spark_shell_name=stg_ods_spark_job.sh
#封装了通过spark客户端执行spark sql的命令，申请的资源较多
run_big_spark_shell_name=stg_ods_big_spark_job.sh
#封装了通过beeline客户端执行hive sql的命令
run_beeline_shell_name=run_beeline_job.sh
#存放将生产数据同步至stg的sqoop脚本路径
src_stg_sqoop_path=/home/yibao/sjzt/sqoop
#stg同步至ods的sql文件目录名
stg_ods_sql_path=stg_ods_sql
#ods同步至dwd的sql文件目录名
ods_dwd_sql_path=ods_dwd_sql
#所有入仓相关脚本、配置文件根目录
path=/home/yibao/sjzt
```

d. src_2_stg_cep_stcdb_mdtrt_d_his.sh 内容示例如下：

```
#!/bin/sh
v_time=`date "+%Y-%m-%d %H:%M:%S"`
v_date=`date -d "$v_time" +%Y%m%d`
v_date_ago_1=`date -d "$v_date -1 day" +%Y-%m-%d`
etl_date=${v_date_ago_1}
source /home/yibao/sjzt/property/property.properties
url=${setlcent_db_src_db_url}
username=${setlcent_db_src_db_user}
password=${setlcent_db_src_db_pwd}
hive_db=${target_stg_db}
sql="alter table ${hive_db}.stg_cep_stcdb_mdtrt_d drop if exists partition (dt='${etl_date}');"
query_sql="select
mdtrt_id,medins_setl_id,psn_no,psn_insu_rlts_id,psn_cert_type,certno,psn_name,gend,naty,brdy,age,coner_n
ame,tel,addr,insutype,psn_type,cvlserv_flag,cvlserv_lv,sp_psn_type,sp_psn_type_lv,clct_grde,flxempe_flag,nw
b_flag,insu_admdvs,emp_no,emp_name,emp_type,econ_type,afil_indu,afil_rlts,emp_mgt_type,pay_loc,fixmedi
ns_code,fixmedins_name,hosp_lv,fix_blng_admdvs,lmtpric_hosp_lv,dedc_hosp_lv,begntime,endtime,mdtrt_cer
t_type,mdtrt_cert_no,med_type,rloc_type,ars_year_ipt_flag,pre_pay_flag,year,refl_old_mdtrt_id,ipt_otp_no,med
rcdno,chfpdr_code,adm_diag_dscr,adm_dept_codg,adm_dept_name,adm_bed,wardarea_bed,traf_dept_flag,d
scg_maindiag_code,dscg_dept_codg,dscg_dept_name,dscg_bed,dscg_way,main_cond_dscr,dise_no,dise_na
me,oprn_oprt_code,oprn_oprt_name,otp_diag_info,inhosp_stas,die_date,ipt_days,geso_val,birctrl_type,fetts,fet
us_cnt,matn_type,prey_time,latechb_flag,pret_flag,fpsc_no,birctrl_matn_date,cop_flag,trt_dcla_detl_sn,vali_fla
g,memo,rid,updt_time,crter_id,crter_name,crte_time,crte_optins_no,opter_id,opter_name,opt_time,optins_no,p
oolarea_no,chfpdr_name,dscg_maindiag_name,adm_caty,dscg_caty,ttp_pay_flag,ttp_pay_prop,dise_type_cod
e,same_dise_adm_flag,quts_type,'cep' as subsys_codg_src,'0' dty_flag from mdtrt_d where updt_time
<'${v_date}' and \$CONDITIONS"
beeline -e "${sql}" -e "!exit"
sqoop import \
  --connect ${url}/SETLCENT_DB \
  --username ${username} \
  --password ${password} \
  --hcatalog-database ${hive_db} \
  --hcatalog-table stg_cep_stcdb_mdtrt_d \
  --hcatalog-partition-keys dt \
  --hcatalog-partition-values ${etl_date} \
  --query "${query_sql}" --split-by updt_time -m 12
  --boundary-query "select min(updt_time), max(updt_time) from mdtrt_d where updt_time <'${v_date}' "
```

(2) STG 到 ODS（质检、分流）

对 STG 层数据进行行级质检、表级质检，并根据质检结果将数据分流至 ODS 和脏数据库。使用集成平台创建 Shell 作业，执行预先准备好的质检和数据分流脚本。其中

stg_2_ods_cep_stcdb_mdtrt_d.sql 内容可参见 10.1 stg_2_ods_cep_stcdb_mdtrt_d.sql 脚本内容。

图8-3 STG 到 ODS

(3) ODS 到 DWD

基于 ODS 层数据进行数据重分布，按照子系统进一步分区管理。使用集成平台创建 Shell 作业，执行脚本。ods_2_dwd_cep_stcdb_mdtrt_d.sql 脚本内容可参见 10.2 ods_2_dwd_cep_stcdb_mdtrt_d.sql 脚本内容。

图8-4 ODS 到 DWD

(4) DWD 到 ADS

该部分内容与相应的医保子系统业务强相关，集成平台主要提供作业管理、运行和调度。执行的脚本无需关注。

T+1 入仓

一般情况下，全量入仓只需要在系统初始化时做一次。后续的持续建仓过程需要通过在数字平台创建周期调度的作业，定期执行预先准备好的 T+1 入仓脚本来完成。

增量流程与全量入仓类似，只是在相应的环节中执行的脚本、SQL 语句稍微有一些区别。

3. 步骤三：交换

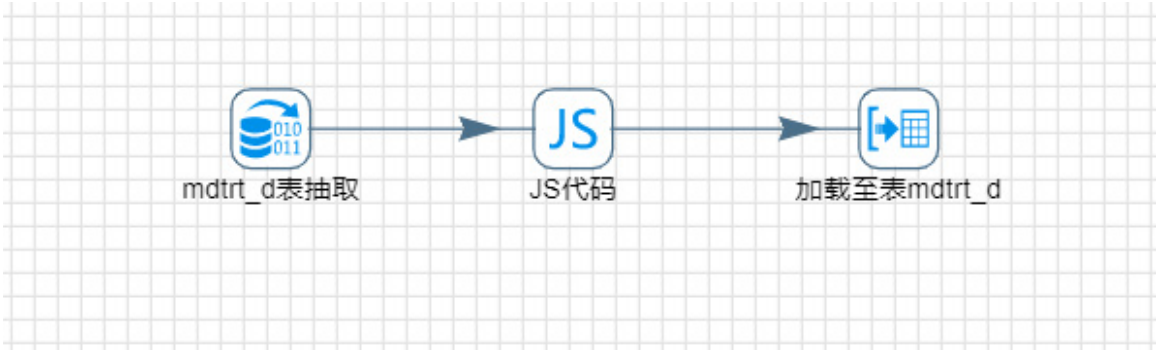
生产数据导入 Hive 数仓后，需要定期从仓库中抽取数据，向省交换库做数据同步。

T+1 交换

(1) 进入[作业管理/作业定义]页面，单击<新建作业>按钮，新建作业。将 ETL 任务添加到作业设计画布中，配置完后双击任务图标跳转至 ETL 任务设计页面，用户可根据实际需要进行任务设计。

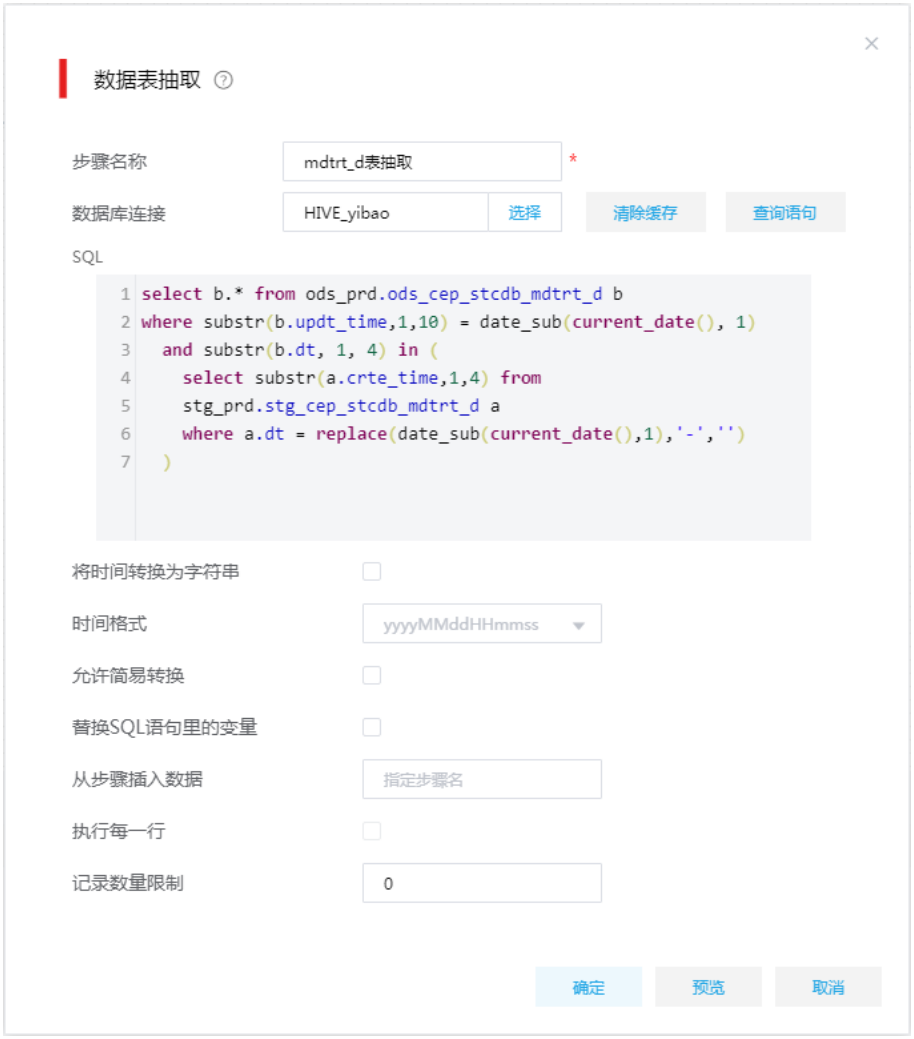
(2) 用户可根据实际需要进行 ETL 任务设计，每天将增量数据同步至省交换库。

图8-5 ETL 任务设计



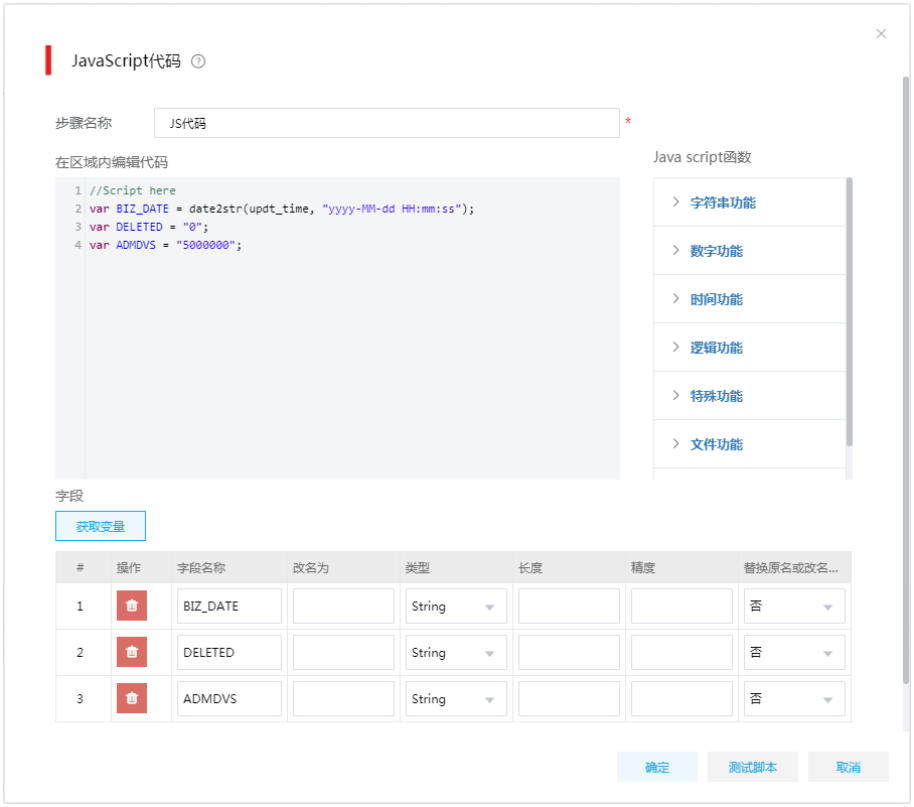
(3) 其中，“mdtrt_d 抽取”步骤负责从 Hive 的 ods 库中抽取 mdtrt_d 表的 T+1 增量数据。

图8-6 mdtrt_d 抽取



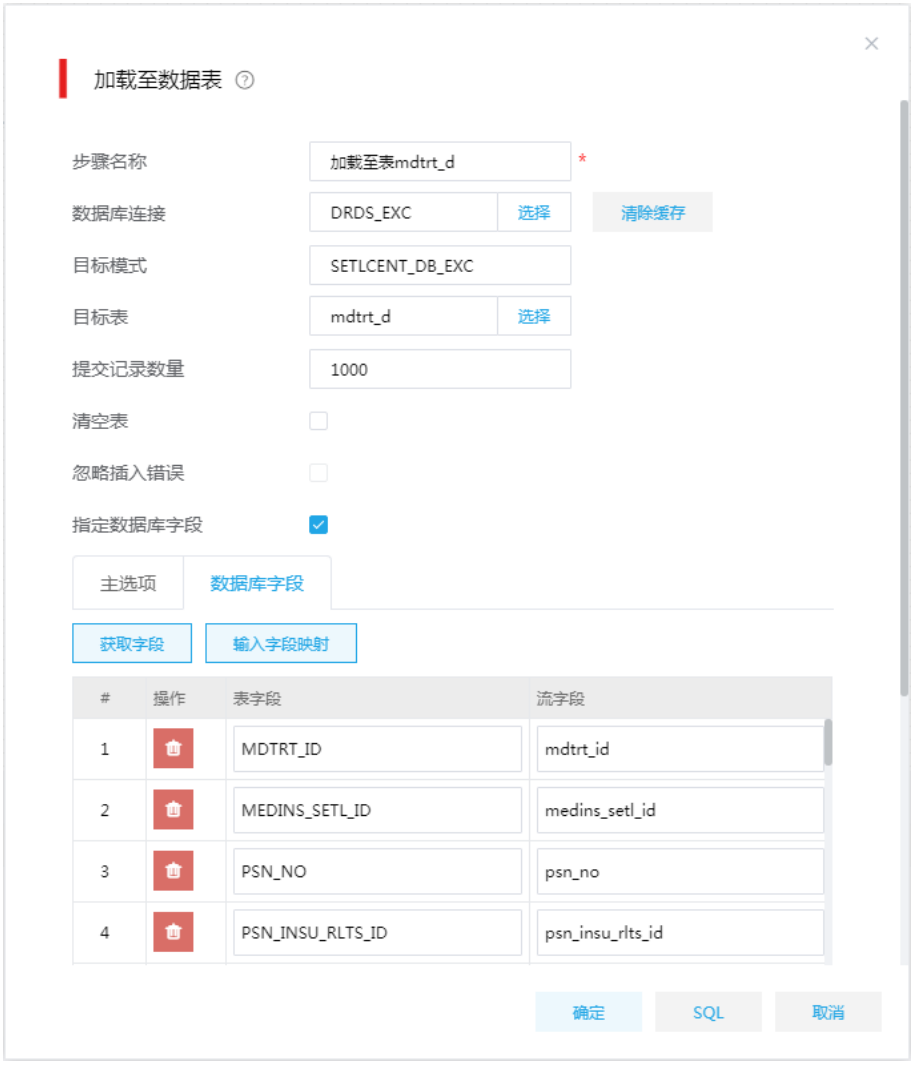
(4) “JS 代码”步骤用于添加几个交换库专有字段，如：业务日期、医保区划等。

图8-7 JS 代码



(5) “加载至表 mdtrt_d”步骤用于向交换库加载数据，需要配置正确的字段映射。

图8-8 加载至数据表

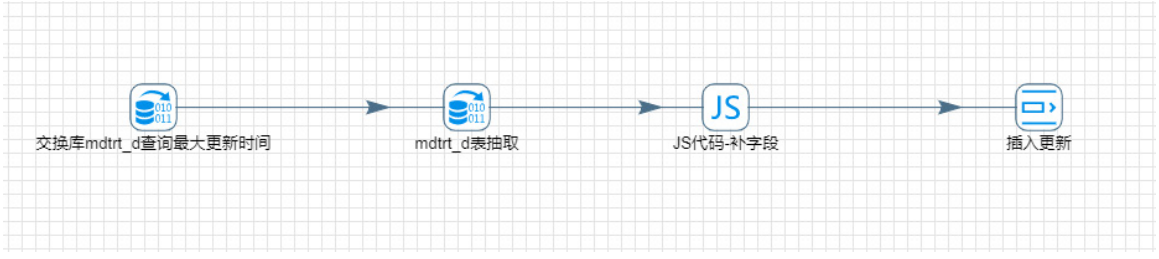


准实时交换

生产库中部分表数据需要准实时同步至交换库，数据流向：生产库 –> 交换库。

(1) 通过集成平台创建 ETL 作业，每隔 5 分钟将增量数据同步至省交换库。

图8-9 准实时交换 ETL 任务



(2) 其中，“交换库 mdtrt_d 查询最大更新时间”步骤用于从省交换库中抽取 mdtrt_d 表的最大更新时间。

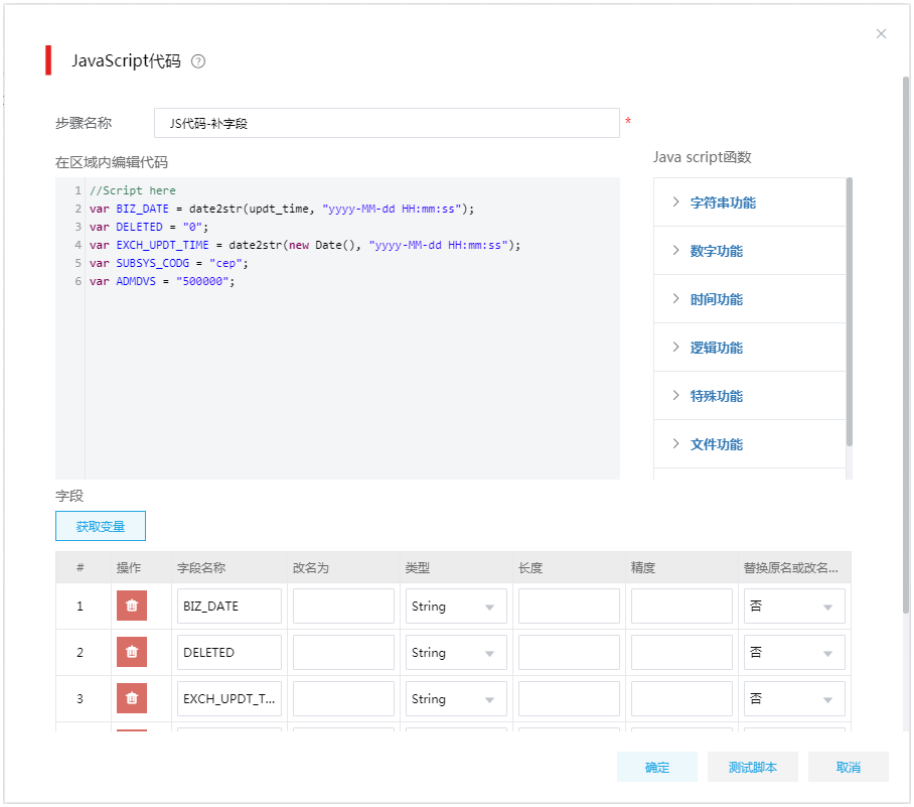
图8-10 抽取最大更新时间

(3) “mdtrt_d 表抽取”步骤使用上一步骤查询的最大时间，从生产库中查询增量数据。“从步骤插入数据”配置为“交换库 mdtrt_d 查询最大更新时间”。

图8-11 查询增量数据

(4) “JS 代码-补字段”步骤用于添加几个交换库专有字段，如：医保区划、删除标记、交换时间、子系统代码等。

图8-12 添加交换库专有字段



(5) “插入更新”步骤用于向交换库做数据同步，这里配置查询字段为表的主键（联合主键），更新字段页签中添加所有表字段。为了避免丢数据，在从生产库中查询增量数据时，还包含了上次同步的部分数据（上次同步的最大更新时间对应的数据），因此这里使用了插入更新组件，来处理重复的数据。

图8-13 插入更新


插入更新
步骤名称
插入更新
数据库连接
DRDS_EXC_RT
选择
清除缓存
目标模式
SETLCENT_DB_EXC_RT
目标表
mdtrt_d
选择
提交记录数量
0
不执行任何更新
查询字段
更新字段
获取字段
操作 表中字段 比较符 流中字段1 流中字段2
1 rid = rid
2 biz_date = BIZ_DATE
3 subsys_codg = SUBSYS_CO...
增加
确定
取消

9 常见问题解答

9.1 HBase、Hive、HDFS、Kafka等大数据组件开启了Kerberos认证，连接这些数据源时如何配置Kerberos认证信息

(1) 登录大数据集群任一节点。

(2) 拷贝 krb5.conf 文件到本地，krb5.conf 文件路径：/etc/krb5.conf。

(3) 拷贝 keytab 文件到本地，keytab 文件路径：/etc/security/keytabs/xxx.service.keytab。

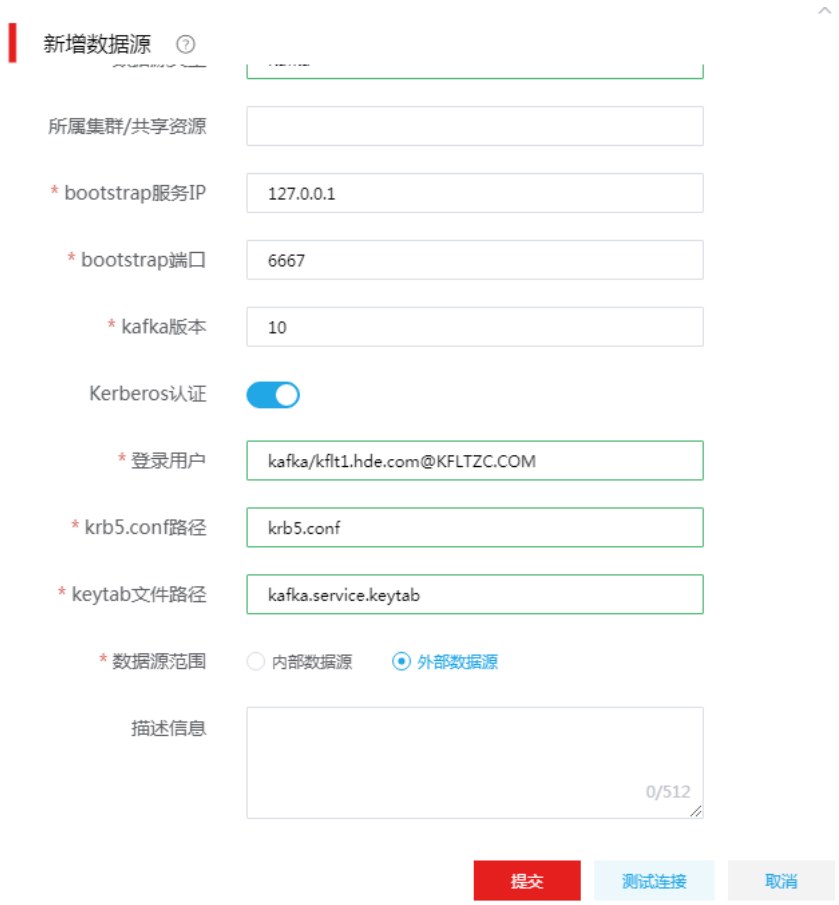
(4) 获取 kerberos 用户名。以 Kafka 为例：执行 ***klist –kt /etc/security/keytabs/kafka.service.keytab*** 命令，查询出的 Principal(kafka/kflt1.hde.com@KFLTZC.COM)即为 Kerberos 用户名，如图 9-1。

图9-1 获取 Kerberos 用户名

```
[root@kflt1 config]# klist -kt /etc/security/keytabs/kafka.service.keytab
Keytab name: FILE:/etc/security/keytabs/kafka.service.keytab
KVNO Timestamp           Principal
---- ------------------- ------------------------------------------------------
   2 11/19/2020 00:13:09 kafka/kflt1.hde.com@KFLTZC.COM
   2 11/19/2020 00:13:09 kafka/kflt1.hde.com@KFLTZC.COM
   2 11/19/2020 00:13:09 kafka/kflt1.hde.com@KFLTZC.COM
   2 11/19/2020 00:13:09 kafka/kflt1.hde.com@KFLTZC.COM
   2 11/19/2020 00:13:09 kafka/kflt1.hde.com@KFLTZC.COM
```

(5) 配置 Kerberos 信息时，填写 Kerberos 用户名，上传 krb5.conf 和 keytab 文件，如图 9-2。

图9-2 配置 Kerberos 认证相关信息



(6) 配置完成后，测试连接并提交即可完成数据源的新增。

10 附录

10.1 stg_2_ods_cep_stcdb_mdtrt_d.sql脚本内容

```
set hive.exec.max.dynamic.partitions=100000;
set hive.exec.max.dynamic.partitions.pernode=100000;
set hive.exec.dynamic.partition.mode=nonstrict;

use ods_prd;
CACHE TABLE STG_CEP_STCDB_MDTRT_D_cached as
  with ETLDATETEMP as (
select
      date_format(CRTE_TIME,'yyyy-MM') as ETL_DATE
      ,case when POOLAREA_NO is null or POOLAREA_NO = '' or length(POOLAREA_NO) <> 6 or
substr(POOLAREA_NO,1,2) <> '50' then '500000' else POOLAREA_NO end    as REGION
from stg_prd.STG_CEP_STCDB_MDTRT_D
where DT ='${etl_date}'
group by
      date_format(CRTE_TIME,'yyyy-MM')
      ,case when POOLAREA_NO is null or POOLAREA_NO = '' or length(POOLAREA_NO) <> 6 or
substr(POOLAREA_NO,1,2) <> '50' then '500000' else POOLAREA_NO end
)
,STG_QUALITY_CONTROLL as (
SELECT
      A.MDTRT_ID,
      A.MEDINS_SETL_ID,
      A.PSN_NO,
      A.PSN_INSU_RLTS_ID,
      A.PSN_CERT_TYPE,
      A.CERTNO,
      A.PSN_NAME,
      A.GEND,
      A.NATY,
      A.BRDY,
      A.AGE,
      A.CONER_NAME,
```

A.TEL,

A.ADDR,

A.INSUTYPE,

A.PSN_TYPE,

A.CVLSERV_FLAG,

A.CVLSERV_LV,

A.SP_PSN_TYPE,

A.SP_PSN_TYPE_LV,

A.CLCT_GRDE,

A.FLXEMPE_FLAG,

A.NWB_FLAG,

A.INSU_ADMDVS,

A.EMP_NO,

A.EMP_NAME,

A.EMP_TYPE,

A.ECON_TYPE,

A.AFIL_INDU,

A.AFIL_RLTS,

A.EMP_MGT_TYPE,

A.PAY_LOC,

A.FIXMEDINS_CODE,

A.FIXMEDINS_NAME,

A.HOSP_LV,

A.FIX_BLNG_ADMDVS,

A.LMTPRIC_HOSP_LV,

A.DEDC_HOSP_LV,

A.BEGNTIME,

A.ENDTIME,

A.MDTRT_CERT_TYPE,

A.MDTRT_CERT_NO,

A.MED_TYPE,

A.RLOC_TYPE,

A.ARS_YEAR_IPT_FLAG,

A.PRE_PAY_FLAG,

A.YEAR,

A.REFL_OLD_MDTRT_ID,

A.IPT_OTP_NO,

A.MEDRCDNO,

A.CHFPDR_CODE,

A.ADM_DIAG_DSCR,

A.ADM_DEPT_CODG,

A.ADM_DEPT_NAME,

A.ADM_BED,

A.WARDAREA_BED,

A.TRAF_DEPT_FLAG,

A.DSCG_MAINDIAG_CODE,

A.DSCG_DEPT_CODG,

A.DSCG_DEPT_NAME,

A.DSCG_BED,

A.DSCG_WAY,

A.MAIN_COND_DSCR,

A.DISE_NO,

A.DISE_NAME,

A.OPRN_OPRT_CODE,

A.OPRN_OPRT_NAME,

A.OTP_DIAG_INFO,

A.INHOSP_STAS,

A.DIE_DATE,

A.IPT_DAYS,

A.GESO_VAL,

A.BIRCTRL_TYPE,

A.FETTS,

A.FETUS_CNT,

A.MATN_TYPE,

A.PREY_TIME,

A.LATECHB_FLAG,

A.PRET_FLAG,

A.FPSC_NO,

A.BIRCTRL_MATN_DATE,

A.COP_FLAG,

A.TRT_DCLA_DETL_SN,

A.VALI_FLAG,

A.MEMO,

A.RID,

A.UPDT_TIME,

A.CRTER_ID,

```
    A.CRTER_NAME,
    A.CRTE_TIME,
    A.CRTE_OPTINS_NO,
    A.OPTER_ID,
    A.OPTER_NAME,
    A.OPT_TIME,
    A.OPTINS_NO,
    A.POOLAREA_NO,
    A.CHFPDR_NAME,
    A.DSCG_MAINDIAG_NAME,
    A.ADM_CATY,
    A.DSCG_CATY,
    A.TTP_PAY_FLAG,
    A.TTP_PAY_PROP,
    A.DISE_TYPE_CODE,
    A.SAME_DISE_ADM_FLAG,
    A.QUTS_TYPE,
    A.SUBSYS_CODG_SRC,
    CONCAT_WS(chr(1)
        ,CHECK_COLUMN_LOGIC(CAST( CASE WHEN (A.BEGNTIME) IS NULL THEN '' ELSE
UNIX_TIMESTAMP(A.BEGNTIME) END AS STRING), '<=', CAST( CASE WHEN (A.ENDTIME) IS NULL THEN ''
ELSE UNIX_TIMESTAMP(A.ENDTIME) END AS STRING), 'DATETIME', CONCAT( A.BEGNTIME,'|',A.ENDTIME),
'S08202102260432', '开始时间不能大于结束时间', '行级', '1')
        ,CHECK_COLUMN_DICT( CASE WHEN A.AFIL_INDU IS NULL THEN '' ELSE A.AFIL_INDU END ,
'AFIL_INDU', 'T', 'SJZD202103011542', '就诊信息表(MDTRT_D)中所属行业(AFIL_INDU)数据值应在数据
字典【所属行业 AFIL_INDU】范围内', '行级', '1')
        ,CHECK_COLUMN_NULL( CASE WHEN A.CRTE_TIME IS NULL THEN '' ELSE A.CRTE_TIME END ,
'L0000000001', '创建时间不能为空', '行级', '3')
        ,CHECK_COLUMN_NULL( CASE WHEN A.RID IS NULL THEN '' ELSE A.RID END , 'L0000000002', '
数据唯一编号不能为空', '行级', '3')
    ) AS DQ_RESULT
FROM stg_prd.STG_CEP_STCDB_MDTRT_D A
WHERE A.DT = '${etl_date}'
)
-- 分流
select
    A.MDTRT_ID,
    A.MEDINS_SETL_ID,
    A.PSN_NO,
```

A.PSN_INSU_RLTS_ID,

A.PSN_CERT_TYPE,

A.CERTNO,

A.PSN_NAME,

A.GEND,

A.NATY,

A.BRDY,

A.AGE,

A.CONER_NAME,

A.TEL,

A.ADDR,

A.INSUTYPE,

A.PSN_TYPE,

A.CVLSERV_FLAG,

A.CVLSERV_LV,

A.SP_PSN_TYPE,

A.SP_PSN_TYPE_LV,

A.CLCT_GRDE,

A.FLXEMPE_FLAG,

A.NWB_FLAG,

A.INSU_ADMDVS,

A.EMP_NO,

A.EMP_NAME,

A.EMP_TYPE,

A.ECON_TYPE,

A.AFIL_INDU,

A.AFIL_RLTS,

A.EMP_MGT_TYPE,

A.PAY_LOC,

A.FIXMEDINS_CODE,

A.FIXMEDINS_NAME,

A.HOSP_LV,

A.FIX_BLNG_ADMDVS,

A.LMTPRIC_HOSP_LV,

A.DEDC_HOSP_LV,

A.BEGNTIME,

A.ENDTIME,

A.MDTRT_CERT_TYPE,

A.MDTRT_CERT_NO,

A.MED_TYPE,

A.RLOC_TYPE,

A.ARS_YEAR_IPT_FLAG,

A.PRE_PAY_FLAG,

A.YEAR,

A.REFL_OLD_MDTRT_ID,

A.IPT_OTP_NO,

A.MEDRCDNO,

A.CHFPDR_CODE,

A.ADM_DIAG_DSCR,

A.ADM_DEPT_CODG,

A.ADM_DEPT_NAME,

A.ADM_BED,

A.WARDAREA_BED,

A.TRAF_DEPT_FLAG,

A.DSCG_MAINDIAG_CODE,

A.DSCG_DEPT_CODG,

A.DSCG_DEPT_NAME,

A.DSCG_BED,

A.DSCG_WAY,

A.MAIN_COND_DSCR,

A.DISE_NO,

A.DISE_NAME,

A.OPRN_OPRT_CODE,

A.OPRN_OPRT_NAME,

A.OTP_DIAG_INFO,

A.INHOSP_STAS,

A.DIE_DATE,

A.IPT_DAYS,

A.GESO_VAL,

A.BIRCTRL_TYPE,

A.FETTS,

A.FETUS_CNT,

A.MATN_TYPE,

A.PREY_TIME,

A.LATECHB_FLAG,

A.PRET_FLAG,

```
        A.FPSC_NO,
        A.BIRCTRL_MATN_DATE,
        A.COP_FLAG,
        A.TRT_DCLA_DETL_SN,
        A.VALI_FLAG,
        A.MEMO,
        A.RID,
        A.UPDT_TIME,
        A.CRTER_ID,
        A.CRTER_NAME,
        A.CRTE_TIME,
        A.CRTE_OPTINS_NO,
        A.OPTER_ID,
        A.OPTER_NAME,
        A.OPT_TIME,
        A.OPTINS_NO,
        A.POOLAREA_NO,
        A.CHFPDR_NAME,
        A.DSCG_MAINDIAG_NAME,
        A.ADM_CATY,
        A.DSCG_CATY,
        A.TTP_PAY_FLAG,
        A.TTP_PAY_PROP,
        A.DISE_TYPE_CODE,
        A.SAME_DISE_ADM_FLAG,
        A.QUTS_TYPE,
        A.SUBSYS_CODG_SRC,
        A.DTY_FLAG,
        A.LOCAL_DTY_FLAG,
        A.EXCH_UPDT_TIME,
        '' AS DQ_RESULT,
        A.DT as ETL_DATE,
        A.REGION as REGION
FROM ods_prd.ODS_CEP_STCDB_MDTRT_D A
INNER JOIN ETLDATETEMP B
        ON A.DT = B.ETL_DATE
        AND A.REGION = B.REGION
LEFT JOIN STG_QUALITY_CONTROLL C
```

```
ON A.MDTRT_ID = C.MDTRT_ID AND A.PSN_NO = C.PSN_NO
WHERE C.MDTRT_ID is null AND C.PSN_NO is null
union ALL
select
    MDTRT_ID,
    MEDINS_SETL_ID,
    PSN_NO,
    PSN_INSU_RLTS_ID,
    PSN_CERT_TYPE,
    CERTNO,
    PSN_NAME,
    GEND,
    NATY,
    BRDY,
    AGE,
    CONER_NAME,
    TEL,
    ADDR,
    INSUTYPE,
    PSN_TYPE,
    CVLSERV_FLAG,
    CVLSERV_LV,
    SP_PSN_TYPE,
    SP_PSN_TYPE_LV,
    CLCT_GRDE,
    FLXEMPE_FLAG,
    NWB_FLAG,
    INSU_ADMDVS,
    EMP_NO,
    EMP_NAME,
    EMP_TYPE,
    ECON_TYPE,
    AFIL_INDU,
    AFIL_RLTS,
    EMP_MGT_TYPE,
    PAY_LOC,
    FIXMEDINS_CODE,
    FIXMEDINS_NAME,
```

HOSP_LV,
FIX_BLNG_ADMDVS,
LMTPRIC_HOSP_LV,
DEDC_HOSP_LV,
BEGNTIME,
ENDTIME,
MDTRT_CERT_TYPE,
MDTRT_CERT_NO,
MED_TYPE,
RLOC_TYPE,
ARS_YEAR_IPT_FLAG,
PRE_PAY_FLAG,
YEAR,
REFL_OLD_MDTRT_ID,
IPT_OTP_NO,
MEDRCDNO,
CHFPDR_CODE,
ADM_DIAG_DSCR,
ADM_DEPT_CODG,
ADM_DEPT_NAME,
ADM_BED,
WARDAREA_BED,
TRAF_DEPT_FLAG,
DSCG_MAINDIAG_CODE,
DSCG_DEPT_CODG,
DSCG_DEPT_NAME,
DSCG_BED,
DSCG_WAY,
MAIN_COND_DSCR,
DISE_NO,
DISE_NAME,
OPRN_OPRT_CODE,
OPRN_OPRT_NAME,
OTP_DIAG_INFO,
INHOSP_STAS,
DIE_DATE,
IPT_DAYS,
GESO_VAL,

```
        BIRCTRL_TYPE,
        FETTS,
        FETUS_CNT,
        MATN_TYPE,
        PREY_TIME,
        LATECHB_FLAG,
        PRET_FLAG,
        FPSC_NO,
        BIRCTRL_MATN_DATE,
        COP_FLAG,
        TRT_DCLA_DETL_SN,
        VALI_FLAG,
        MEMO,
        RID,
        UPDT_TIME,
        CRTER_ID,
        CRTER_NAME,
        CRTE_TIME,
        CRTE_OPTINS_NO,
        OPTER_ID,
        OPTER_NAME,
        OPT_TIME,
        OPTINS_NO,
        POOLAREA_NO,
        CHFPDR_NAME,
        DSCG_MAINDIAG_NAME,
        ADM_CATY,
        DSCG_CATY,
        TTP_PAY_FLAG,
        TTP_PAY_PROP,
        DISE_TYPE_CODE,
        SAME_DISE_ADM_FLAG,
        QUTS_TYPE,
        SUBSYS_CODG_SRC,
        CASE WHEN INSTR(DQ_RESULT, '"natResult":"FAIL"') > 0 THEN '1' ELSE '0' END DTY_FLAG,
        CASE WHEN INSTR(DQ_RESULT, '"localResult":"FAIL"') > 0 THEN '1' ELSE '0' END LOCAL_DTY_FLAG,
        concat(date_add(current_timestamp(),-1),' ',date_format(current_timestamp(),'HH:mm:ss')) AS
EXCH_UPDT_TIME,
```

```
    DQ_RESULT,
    date_format(CRTE_TIME, 'yyyy-MM') AS ETL_DATE,
    case when POOLAREA_NO is null or POOLAREA_NO = '' or length(POOLAREA_NO) <> 6 or
substr(POOLAREA_NO,1,2) <> '50' then '500000' else POOLAREA_NO end    AS REGION
FROM STG_QUALITY_CONTROLL;

-- 写 ODS 表
REFRESH TABLE ods_prd.ODS_CEP_STCDB_MDTRT_D;
FROM    STG_CEP_STCDB_MDTRT_D_cached
INSERT OVERWRITE TABLE ods_prd.ODS_CEP_STCDB_MDTRT_D PARTITION(DT,REGION)
SELECT
    MDTRT_ID,
    MEDINS_SETL_ID,
    PSN_NO,
    PSN_INSU_RLTS_ID,
    PSN_CERT_TYPE,
    CERTNO,
    PSN_NAME,
    GEND,
    NATY,
    BRDY,
    AGE,
    CONER_NAME,
    TEL,
    ADDR,
    INSUTYPE,
    PSN_TYPE,
    CVLSERV_FLAG,
    CVLSERV_LV,
    SP_PSN_TYPE,
    SP_PSN_TYPE_LV,
    CLCT_GRDE,
    FLXEMPE_FLAG,
    NWB_FLAG,
    INSU_ADMDVS,
    EMP_NO,
    EMP_NAME,
    EMP_TYPE,
```

ECON_TYPE,
AFIL_INDU,
AFIL_RLTS,
EMP_MGT_TYPE,
PAY_LOC,
FIXMEDINS_CODE,
FIXMEDINS_NAME,
HOSP_LV,
FIX_BLNG_ADMDVS,
LMTPRIC_HOSP_LV,
DEDC_HOSP_LV,
BEGNTIME,
ENDTIME,
MDTRT_CERT_TYPE,
MDTRT_CERT_NO,
MED_TYPE,
RLOC_TYPE,
ARS_YEAR_IPT_FLAG,
PRE_PAY_FLAG,
YEAR,
REFL_OLD_MDTRT_ID,
IPT_OTP_NO,
MEDRCDNO,
CHFPDR_CODE,
ADM_DIAG_DSCR,
ADM_DEPT_CODG,
ADM_DEPT_NAME,
ADM_BED,
WARDAREA_BED,
TRAF_DEPT_FLAG,
DSCG_MAINDIAG_CODE,
DSCG_DEPT_CODG,
DSCG_DEPT_NAME,
DSCG_BED,
DSCG_WAY,
MAIN_COND_DSCR,
DISE_NO,
DISE_NAME,

OPRN_OPRT_CODE,
OPRN_OPRT_NAME,
OTP_DIAG_INFO,
INHOSP_STAS,
DIE_DATE,
IPT_DAYS,
GESO_VAL,
BIRCTRL_TYPE,
FETTS,
FETUS_CNT,
MATN_TYPE,
PREY_TIME,
LATECHB_FLAG,
PRET_FLAG,
FPSC_NO,
BIRCTRL_MATN_DATE,
COP_FLAG,
TRT_DCLA_DETL_SN,
VALI_FLAG,
MEMO,
RID,
UPDT_TIME,
CRTER_ID,
CRTER_NAME,
CRTE_TIME,
CRTE_OPTINS_NO,
OPTER_ID,
OPTER_NAME,
OPT_TIME,
OPTINS_NO,
POOLAREA_NO,
CHFPDR_NAME,
DSCG_MAINDIAG_NAME,
ADM_CATY,
DSCG_CATY,
TTP_PAY_FLAG,
TTP_PAY_PROP,
DISE_TYPE_CODE,

```
        SAME_DISE_ADM_FLAG,
        QUTS_TYPE,
        SUBSYS_CODG_SRC,
        DTY_FLAG,
        LOCAL_DTY_FLAG,
        EXCH_UPDT_TIME,
        ETL_DATE,
        REGION
-- 写脏数据日志
INSERT OVERWRITE TABLE ods_prd.ODS_DTY_DETAIL_X PARTITION(DT,STG_TABLE_NAME)
SELECT
        case when POOLAREA_NO is null or POOLAREA_NO = '' or length(POOLAREA_NO) <> 6 or
substr(POOLAREA_NO,1,2) <> '50' then '500000' else POOLAREA_NO end    AS REGION
        ,'MDTRT_D' AS TABLE_NAME
        ,'CEP' AS SUBSYS_CODG
        ,GET_JSON_OBJECT(dq_result_t, '$.checkLv') AS VIO_DQ_LVL
        ,'${etl_date}' BIZ_DATE
        ,EXCH_UPDT_TIME
        ,RID AS RID
        ,CRTE_TIME
        ,GET_JSON_OBJECT(dq_result_t, '$.rawData') RAW_DATA
        ,GET_JSON_OBJECT(dq_result_t, '$.ruleCode') VIO_DQ_CODE
        ,GET_JSON_OBJECT(dq_result_t, '$.checkRule') MEMO
        ,'${etl_date}' AS ETL_DATE
        ,'STG_CEP_STCDB_MDTRT_D' AS STG_TABLE_NAME
lateral view explode(split(dq_result,chr(1))) num AS dq_result_t
WHERE (DTY_FLAG = '1' or LOCAL_DTY_FLAG = '1')
        and dq_result_t is not null and dq_result_t <> ''
;

DROP TABLE STG_CEP_STCDB_MDTRT_D_cached;
REFRESH TABLE ods_prd.ODS_CEP_STCDB_MDTRT_D;
REFRESH TABLE ods_prd.ODS_DTY_DETAIL_X;
```

10.2 ods_2_dwd_cep_stcdb_mdtrt_d.sql脚本内容

```
set hive.optimize.sort.dynamic.partition=true;
set hive.exec.dynamic.partition = true;
set hive.exec.dynamic.partition.mode = nonstrict;
```

```
set hive.exec.max.dynamic.partitions=100000;
set hive.exec.max.dynamic.partitions.pernode=100000;
set hive.compute.query.using.stats=false;

use dwd_prd;

create temporary table if not exists tmp_yibao_ods_dwd as
select
date_format(CRTE_TIME,'yyyy-MM') as cdate
,case when POOLAREA_NO is null or POOLAREA_NO = '' or length(POOLAREA_NO) <> 6 or
substr(POOLAREA_NO,1,2) <> '50'    then '500000' else POOLAREA_NO end    as area
from stg_prd.stg_cep_stcdb_mdtrt_d
where dt ='${etl_date}'
group by date_format(CRTE_TIME,'yyyy-MM'),case when POOLAREA_NO is null or POOLAREA_NO = '' or
length(POOLAREA_NO) <> 6 or substr(POOLAREA_NO,1,2) <> '50'    then '500000' else POOLAREA_NO
end ;

insert overwrite table dwd_prd.dwd_dgn_mdtrt_d partition(dt,region,subs_code)
select
mdtrt_id, medins_setl_id, psn_no, psn_insu_rlts_id, psn_cert_type, certno, psn_name, gend, naty, brdy,
age, coner_name, tel, addr, insutype, psn_type, cvlserv_flag, cvlserv_lv, sp_psn_type, sp_psn_type_lv,
clct_grde, flxempe_flag, nwb_flag, insu_admdvs, emp_no, emp_name, emp_type, econ_type, afil_indu,
afil_rlts, emp_mgt_type, pay_loc, fixmedins_code, fixmedins_name, hosp_lv, fix_blng_admdvs,
lmtpric_hosp_lv, dedc_hosp_lv, begntime, endtime, mdtrt_cert_type, mdtrt_cert_no, med_type,
rloc_type, ars_year_ipt_flag, pre_pay_flag, year, refl_old_mdtrt_id, ipt_otp_no, medrcdno, chfpdr_code,
adm_diag_dscr, adm_dept_codg, adm_dept_name, adm_bed, wardarea_bed, traf_dept_flag,
dscg_maindiag_code, dscg_dept_codg, dscg_dept_name, dscg_bed, dscg_way, main_cond_dscr, dise_no,
dise_name, oprn_oprt_code, oprn_oprt_name, otp_diag_info, inhosp_stas, die_date, ipt_days, geso_val,
birctrl_type, fetts, fetus_cnt, matn_type, prey_time, latechb_flag, pret_flag, fpsc_no, birctrl_matn_date,
cop_flag, trt_dcla_detl_sn, vali_flag, memo, rid, updt_time, crter_id, crter_name, crte_time,
crte_optins_no, opter_id, opter_name, opt_time, optins_no, poolarea_no, chfpdr_name,
dscg_maindiag_name, adm_caty, dscg_caty, ttp_pay_flag, ttp_pay_prop, dise_type_code,
same_dise_adm_flag, quts_type, subsys_codg_src, dty_flag
,concat(date_add(current_timestamp(),-1),' ',date_format(current_timestamp(),'HH:mm:ss')) as
exch_updt_time
,date_format(CRTE_TIME,'yyyy-MM') as dt
,region
,'cep_stcdb' as subs_code
from ods_prd.ods_cep_stcdb_mdtrt_d odstb join tmp_yibao_ods_dwd on
odstb.dt=tmp_yibao_ods_dwd.cdate and odstb.region=tmp_yibao_ods_dwd.area
```

```
where local_dty_flag='0'
cluster BY dt,mdtrt_id,psn_no ;
```

10.3 ods、stg及dwd建表语句

1. stg_prd.stg_cep_stcdb_mdtrt_d

```
CREATE TABLE `stg_prd.stg_cep_stcdb_mdtrt_d`(
    `mdtrt_id` string COMMENT '就诊 ID',
    `medins_setl_id` string COMMENT '医药机构结算 ID',
    `psn_no` string COMMENT '人员编号',
    `psn_insu_rlts_id` string COMMENT '人员参保关系 ID',
    `psn_cert_type` string COMMENT '人员证件类型',
    `certno` string COMMENT '证件号码',
    `psn_name` string COMMENT '人员姓名',
    `gend` string COMMENT '性别',
    `naty` string COMMENT '民族',
    `brdy` date COMMENT '出生日期',
    `age` string COMMENT '年龄',
    `coner_name` string COMMENT '联系人姓名',
    `tel` string COMMENT '联系电话',
    `addr` string COMMENT '联系地址',
    `insutype` string COMMENT '险种类型',
    `psn_type` string COMMENT '人员类别',
    `cvlserv_flag` string COMMENT '公务员标志',
    `cvlserv_lv` string COMMENT '公务员等级',
    `sp_psn_type` string COMMENT '特殊人员类型',
    `sp_psn_type_lv` string COMMENT '特殊人员类型等级',
    `clct_grde` string COMMENT '缴费档次',
    `flxempe_flag` string COMMENT '灵活就业标志',
    `nwb_flag` string COMMENT '新生儿标志',
    `insu_admdvs` string COMMENT '参保所属医保区划',
    `emp_no` string COMMENT '单位编号',
    `emp_name` string COMMENT '单位名称',
    `emp_type` string COMMENT '单位类型',
    `econ_type` string COMMENT '经济类型',
    `afil_indu` string COMMENT '所属行业',
    `afil_rlts` string COMMENT '隶属关系',
    `emp_mgt_type` string COMMENT '单位管理类型',
    `pay_loc` string COMMENT '支付地点类别',
```

`fixmedins_code` string COMMENT '定点医药机构编号',
`fixmedins_name` string COMMENT '定点医药机构名称',
`hosp_lv` string COMMENT '医院等级',
`fix_blng_admdvs` string COMMENT '定点归属医保区划',
`lmtpric_hosp_lv` string COMMENT '限价医院等级',
`dedc_hosp_lv` string COMMENT '起付线医院等级',
`begntime` string COMMENT '开始时间',
`endtime` string COMMENT '结束时间',
`mdtrt_cert_type` string COMMENT '就诊凭证类型',
`mdtrt_cert_no` string COMMENT '就诊凭证编号',
`med_type` string COMMENT '医疗类别',
`rloc_type` string COMMENT '异地安置类别',
`ars_year_ipt_flag` string COMMENT '跨年度住院标志',
`pre_pay_flag` string COMMENT '先行支付标志',
`year` string COMMENT '年度',
`refl_old_mdtrt_id` string COMMENT '转诊前就诊 ID',
`ipt_otp_no` string COMMENT '住院/门诊号',
`medrcdno` string COMMENT '病历号',
`chfpdr_code` string COMMENT '主诊医师代码',
`adm_diag_dscr` string COMMENT '入院诊断描述',
`adm_dept_codg` string COMMENT '入院科室编码',
`adm_dept_name` string COMMENT '入院科室名称',
`adm_bed` string COMMENT '入院床位',
`wardarea_bed` string COMMENT '病区床位',
`traf_dept_flag` string COMMENT '转科室标志',
`dscg_maindiag_code` string COMMENT '住院主诊断代码',
`dscg_dept_codg` string COMMENT '出院科室编码',
`dscg_dept_name` string COMMENT '出院科室名称',
`dscg_bed` string COMMENT '出院床位',
`dscg_way` string COMMENT '离院方式',
`main_cond_dscr` string COMMENT '主要病情描述',
`dise_no` string COMMENT '病种编号',
`dise_name` string COMMENT '病种名称',
`oprn_oprt_code` string COMMENT '手术操作代码',
`oprn_oprt_name` string COMMENT '手术操作名称',
`otp_diag_info` string COMMENT '门诊诊断信息',
`inhosp_stas` string COMMENT '在院状态',
`die_date` date COMMENT '死亡日期',

```
  `ipt_days` string COMMENT '住院天数',
  `geso_val` string COMMENT '孕周数',
  `birctrl_type` string COMMENT '计划生育手术类别',
  `fetts` string COMMENT '胎次',
  `fetus_cnt` string COMMENT '胎儿数',
  `matn_type` string COMMENT '生育类别',
  `prey_time` string COMMENT '妊娠时间',
  `latechb_flag` string COMMENT '晚育标志',
  `pret_flag` string COMMENT '早产标志',
  `fpsc_no` string COMMENT '计划生育服务证号',
  `birctrl_matn_date` string COMMENT '计划生育手术或生育日期',
  `cop_flag` string COMMENT '伴有并发症标志',
  `trt_dcla_detl_sn` string COMMENT '待遇申报明细流水号',
  `vali_flag` string COMMENT '有效标志',
  `memo` string COMMENT '备注',
  `rid` string COMMENT '数据唯一记录号',
  `updt_time` string COMMENT '数据更新时间',
  `crter_id` string COMMENT '创建人 ID',
  `crter_name` string COMMENT '创建人姓名',
  `crte_time` string COMMENT '数据创建时间',
  `crte_optins_no` string COMMENT '创建机构编号',
  `opter_id` string COMMENT '经办人 ID',
  `opter_name` string COMMENT '经办人姓名',
  `opt_time` string COMMENT '经办时间',
  `optins_no` string COMMENT '经办机构编号',
  `poolarea_no` string COMMENT '统筹区编号',
  `chfpdr_name` string COMMENT '主诊医师姓名',
  `dscg_maindiag_name` string COMMENT '住院主诊断名称',
  `adm_caty` string COMMENT '入院科别',
  `dscg_caty` string COMMENT '出院科别',
  `ttp_pay_flag` string COMMENT '第三方赔付标志',
  `ttp_pay_prop` string COMMENT '第三方赔付比例',
  `dise_type_code` string COMMENT '病种类型代码',
  `same_dise_adm_flag` string COMMENT '同病种入院标志',
  `quts_type` string COMMENT '编制类型',
  `subsys_codg_src` string COMMENT '子系统编码',
  `dty_flag` string COMMENT '是否脏数据(0 否 1 是)')
COMMENT ''
```

```
PARTITIONED BY (
  `dt` string);
```

2. ods_prd.ods_cep_stcdb_mdtrt_d

```
CREATE TABLE `ods_prd.ods_cep_stcdb_mdtrt_d`(
  `mdtrt_id` string COMMENT '就诊 ID',
  `medins_setl_id` string COMMENT '医药机构结算 ID',
  `psn_no` string COMMENT '人员编号',
  `psn_insu_rlts_id` string COMMENT '人员参保关系 ID',
  `psn_cert_type` string COMMENT '人员证件类型',
  `certno` string COMMENT '证件号码',
  `psn_name` string COMMENT '人员姓名',
  `gend` string COMMENT '性别',
  `naty` string COMMENT '民族',
  `brdy` date COMMENT '出生日期',
  `age` decimal(4,1) COMMENT '年龄',
  `coner_name` string COMMENT '联系人姓名',
  `tel` string COMMENT '联系电话',
  `addr` string COMMENT '联系地址',
  `insutype` string COMMENT '险种类型',
  `psn_type` string COMMENT '人员类别',
  `cvlserv_flag` string COMMENT '公务员标志',
  `cvlserv_lv` string COMMENT '公务员等级',
  `sp_psn_type` string COMMENT '特殊人员类型',
  `sp_psn_type_lv` string COMMENT '特殊人员类型等级',
  `clct_grde` string COMMENT '缴费档次',
  `flxempe_flag` string COMMENT '灵活就业标志',
  `nwb_flag` string COMMENT '新生儿标志',
  `insu_admdvs` string COMMENT '参保所属医保区划',
  `emp_no` string COMMENT '单位编号',
  `emp_name` string COMMENT '单位名称',
  `emp_type` string COMMENT '单位类型',
  `econ_type` string COMMENT '经济类型',
  `afil_indu` string COMMENT '所属行业',
  `afil_rlts` string COMMENT '隶属关系',
  `emp_mgt_type` string COMMENT '单位管理类型',
  `pay_loc` string COMMENT '支付地点类别',
  `fixmedins_code` string COMMENT '定点医药机构编号',
  `fixmedins_name` string COMMENT '定点医药机构名称',
```

`hosp_lv` string COMMENT '医院等级',
`fix_blng_admdvs` string COMMENT '定点归属医保区划',
`lmtpric_hosp_lv` string COMMENT '限价医院等级',
`dedc_hosp_lv` string COMMENT '起付线医院等级',
`begntime` timestamp COMMENT '开始时间',
`endtime` timestamp COMMENT '结束时间',
`mdtrt_cert_type` string COMMENT '就诊凭证类型',
`mdtrt_cert_no` string COMMENT '就诊凭证编号',
`med_type` string COMMENT '医疗类别',
`rloc_type` string COMMENT '异地安置类别',
`ars_year_ipt_flag` string COMMENT '跨年度住院标志',
`pre_pay_flag` string COMMENT '先行支付标志',
`year` string COMMENT '年度',
`refl_old_mdtrt_id` string COMMENT '转诊前就诊 ID',
`ipt_otp_no` string COMMENT '住院/门诊号',
`medrcdno` string COMMENT '病历号',
`chfpdr_code` string COMMENT '主诊医师代码',
`adm_diag_dscr` string COMMENT '入院诊断描述',
`adm_dept_codg` string COMMENT '入院科室编码',
`adm_dept_name` string COMMENT '入院科室名称',
`adm_bed` string COMMENT '入院床位',
`wardarea_bed` string COMMENT '病区床位',
`traf_dept_flag` string COMMENT '转科室标志',
`dscg_maindiag_code` string COMMENT '住院主诊断代码',
`dscg_dept_codg` string COMMENT '出院科室编码',
`dscg_dept_name` string COMMENT '出院科室名称',
`dscg_bed` string COMMENT '出院床位',
`dscg_way` string COMMENT '离院方式',
`main_cond_dscr` string COMMENT '主要病情描述',
`dise_no` string COMMENT '病种编号',
`dise_name` string COMMENT '病种名称',
`oprn_oprt_code` string COMMENT '手术操作代码',
`oprn_oprt_name` string COMMENT '手术操作名称',
`otp_diag_info` string COMMENT '门诊诊断信息',
`inhosp_stas` string COMMENT '在院状态',
`die_date` date COMMENT '死亡日期',
`ipt_days` decimal(16,0) COMMENT '住院天数',
`geso_val` decimal(2,0) COMMENT '孕周数',

```
  `birctrl_type` string COMMENT '计划生育手术类别',
  `fetts` decimal(3,0) COMMENT '胎次',
  `fetus_cnt` decimal(3,0) COMMENT '胎儿数',
  `matn_type` string COMMENT '生育类别',
  `prey_time` timestamp COMMENT '妊娠时间',
  `latechb_flag` string COMMENT '晚育标志',
  `pret_flag` string COMMENT '早产标志',
  `fpsc_no` string COMMENT '计划生育服务证号',
  `birctrl_matn_date` timestamp COMMENT '计划生育手术或生育日期',
  `cop_flag` string COMMENT '伴有并发症标志',
  `trt_dcla_detl_sn` string COMMENT '待遇申报明细流水号',
  `vali_flag` string COMMENT '有效标志',
  `memo` string COMMENT '备注',
  `rid` string COMMENT '数据唯一记录号',
  `updt_time` timestamp COMMENT '数据更新时间',
  `crter_id` string COMMENT '创建人 ID',
  `crter_name` string COMMENT '创建人姓名',
  `crte_time` timestamp COMMENT '数据创建时间',
  `crte_optins_no` string COMMENT '创建机构编号',
  `opter_id` string COMMENT '经办人 ID',
  `opter_name` string COMMENT '经办人姓名',
  `opt_time` timestamp COMMENT '经办时间',
  `optins_no` string COMMENT '经办机构编号',
  `poolarea_no` string COMMENT '统筹区编号',
  `chfpdr_name` string COMMENT '主诊医师姓名',
  `dscg_maindiag_name` string COMMENT '住院主诊断名称',
  `adm_caty` string COMMENT '入院科别',
  `dscg_caty` string COMMENT '出院科别',
  `ttp_pay_flag` string COMMENT '第三方赔付标志',
  `ttp_pay_prop` decimal(5,4) COMMENT '第三方赔付比例',
  `dise_type_code` string COMMENT '病种类型代码',
  `same_dise_adm_flag` string COMMENT '同病种入院标志',
  `quts_type` string COMMENT '编制类型',
  `subsys_codg_src` string COMMENT '子系统编码',
  `dty_flag` string COMMENT '是否脏数据(0 否 1 是)',
  `local_dty_flag` string COMMENT '本地规则-脏数据标识(0 否 1 是)',
  `exch_updt_time` timestamp COMMENT '入仓时间')
COMMENT ''
```

```
PARTITIONED BY (
    `dt` string,
    `region` string);
```

3. dwd_prd.dwd_dgn mdtrt_d

```
CREATE TABLE `dwd_prd.dwd_dgn_mdtrt_d`(
    `mdtrt_id` string COMMENT '就诊 ID',
    `medins_setl_id` string COMMENT '医药机构结算 ID',
    `psn_no` string COMMENT '人员编号',
    `psn_insu_rlts_id` string COMMENT '人员参保关系 ID',
    `psn_cert_type` string COMMENT '人员证件类型',
    `certno` string COMMENT '证件号码',
    `psn_name` string COMMENT '人员姓名',
    `gend` string COMMENT '性别',
    `naty` string COMMENT '民族',
    `brdy` date COMMENT '出生日期',
    `age` decimal(4,1) COMMENT '年龄',
    `coner_name` string COMMENT '联系人姓名',
    `tel` string COMMENT '联系电话',
    `addr` string COMMENT '联系地址',
    `insutype` string COMMENT '险种类型',
    `psn_type` string COMMENT '人员类别',
    `cvlserv_flag` string COMMENT '公务员标志',
    `cvlserv_lv` string COMMENT '公务员等级',
    `sp_psn_type` string COMMENT '特殊人员类型',
    `sp_psn_type_lv` string COMMENT '特殊人员类型等级',
    `clct_grde` string COMMENT '缴费档次',
    `flxempe_flag` string COMMENT '灵活就业标志',
    `nwb_flag` string COMMENT '新生儿标志',
    `insu_admdvs` string COMMENT '参保所属医保区划',
    `emp_no` string COMMENT '单位编号',
    `emp_name` string COMMENT '单位名称',
    `emp_type` string COMMENT '单位类型',
    `econ_type` string COMMENT '经济类型',
    `afil_indu` string COMMENT '所属行业',
    `afil_rlts` string COMMENT '隶属关系',
    `emp_mgt_type` string COMMENT '单位管理类型',
    `pay_loc` string COMMENT '支付地点类别',
```

`fixmedins_code` string COMMENT '定点医药机构编号',
`fixmedins_name` string COMMENT '定点医药机构名称',
`hosp_lv` string COMMENT '医院等级',
`fix_blng_admdvs` string COMMENT '定点归属医保区划',
`lmtpric_hosp_lv` string COMMENT '限价医院等级',
`dedc_hosp_lv` string COMMENT '起付线医院等级',
`begntime` timestamp COMMENT '开始时间',
`endtime` timestamp COMMENT '结束时间',
`mdtrt_cert_type` string COMMENT '就诊凭证类型',
`mdtrt_cert_no` string COMMENT '就诊凭证编号',
`med_type` string COMMENT '医疗类别',
`rloc_type` string COMMENT '异地安置类别',
`ars_year_ipt_flag` string COMMENT '跨年度住院标志',
`pre_pay_flag` string COMMENT '先行支付标志',
`year` string COMMENT '年度',
`refl_old_mdtrt_id` string COMMENT '转诊前就诊 ID',
`ipt_otp_no` string COMMENT '住院/门诊号',
`medrcdno` string COMMENT '病历号',
`chfpdr_code` string COMMENT '主诊医师代码',
`adm_diag_dscr` string COMMENT '入院诊断描述',
`adm_dept_codg` string COMMENT '入院科室编码',
`adm_dept_name` string COMMENT '入院科室名称',
`adm_bed` string COMMENT '入院床位',
`wardarea_bed` string COMMENT '病区床位',
`traf_dept_flag` string COMMENT '转科室标志',
`dscg_maindiag_code` string COMMENT '住院主诊断代码',
`dscg_dept_codg` string COMMENT '出院科室编码',
`dscg_dept_name` string COMMENT '出院科室名称',
`dscg_bed` string COMMENT '出院床位',
`dscg_way` string COMMENT '离院方式',
`main_cond_dscr` string COMMENT '主要病情描述',
`dise_no` string COMMENT '病种编号',
`dise_name` string COMMENT '病种名称',
`oprn_oprt_code` string COMMENT '手术操作代码',
`oprn_oprt_name` string COMMENT '手术操作名称',
`otp_diag_info` string COMMENT '门诊诊断信息',
`inhosp_stas` string COMMENT '在院状态',
`die_date` date COMMENT '死亡日期',

`ipt_days` decimal(16,0) COMMENT '住院天数',
`geso_val` decimal(2,0) COMMENT '孕周数',
`birctrl_type` string COMMENT '计划生育手术类别',
`fetts` decimal(3,0) COMMENT '胎次',
`fetus_cnt` decimal(3,0) COMMENT '胎儿数',
`matn_type` string COMMENT '生育类别',
`prey_time` timestamp COMMENT '妊娠时间',
`latechb_flag` string COMMENT '晚育标志',
`pret_flag` string COMMENT '早产标志',
`fpsc_no` string COMMENT '计划生育服务证号',
`birctrl_matn_date` timestamp COMMENT '计划生育手术或生育日期',
`cop_flag` string COMMENT '伴有并发症标志',
`trt_dcla_detl_sn` string COMMENT '待遇申报明细流水号',
`vali_flag` string COMMENT '有效标志',
`memo` string COMMENT '备注',
`rid` string COMMENT '数据唯一记录号',
`updt_time` timestamp COMMENT '数据更新时间',
`crter_id` string COMMENT '创建人 ID',
`crter_name` string COMMENT '创建人姓名',
`crte_time` timestamp COMMENT '数据创建时间',
`crte_optins_no` string COMMENT '创建机构编号',
`opter_id` string COMMENT '经办人 ID',
`opter_name` string COMMENT '经办人姓名',
`opt_time` timestamp COMMENT '经办时间',
`optins_no` string COMMENT '经办机构编号',
`poolarea_no` string COMMENT '统筹区编号',
`chfpdr_name` string COMMENT '主诊医师姓名',
`dscg_maindiag_name` string COMMENT '住院主诊断名称',
`adm_caty` string COMMENT '入院科别',
`dscg_caty` string COMMENT '出院科别',
`ttp_pay_flag` string COMMENT '第三方赔付标志',
`ttp_pay_prop` decimal(5,4) COMMENT '第三方赔付比例',
`dise_type_code` string COMMENT '病种类型代码',
`same_dise_adm_flag` string COMMENT '同病种入院标志',
`quts_type` string COMMENT '编制类型',
`subsys_codg_src` string COMMENT '子系统编码',
`dty_flag` string COMMENT '是否脏数据(0 否 1 是)',
`exch_updt_time` timestamp COMMENT '入仓时间')

```
COMMENT '就诊信息表'
PARTITIONED BY (
    `dt` string,
    `region` string,
    `subs_code` string);
```