ALIBABA CLOUD

阿里云

数据湖构建

数据湖构建

文档版本：20220713

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.产品简介

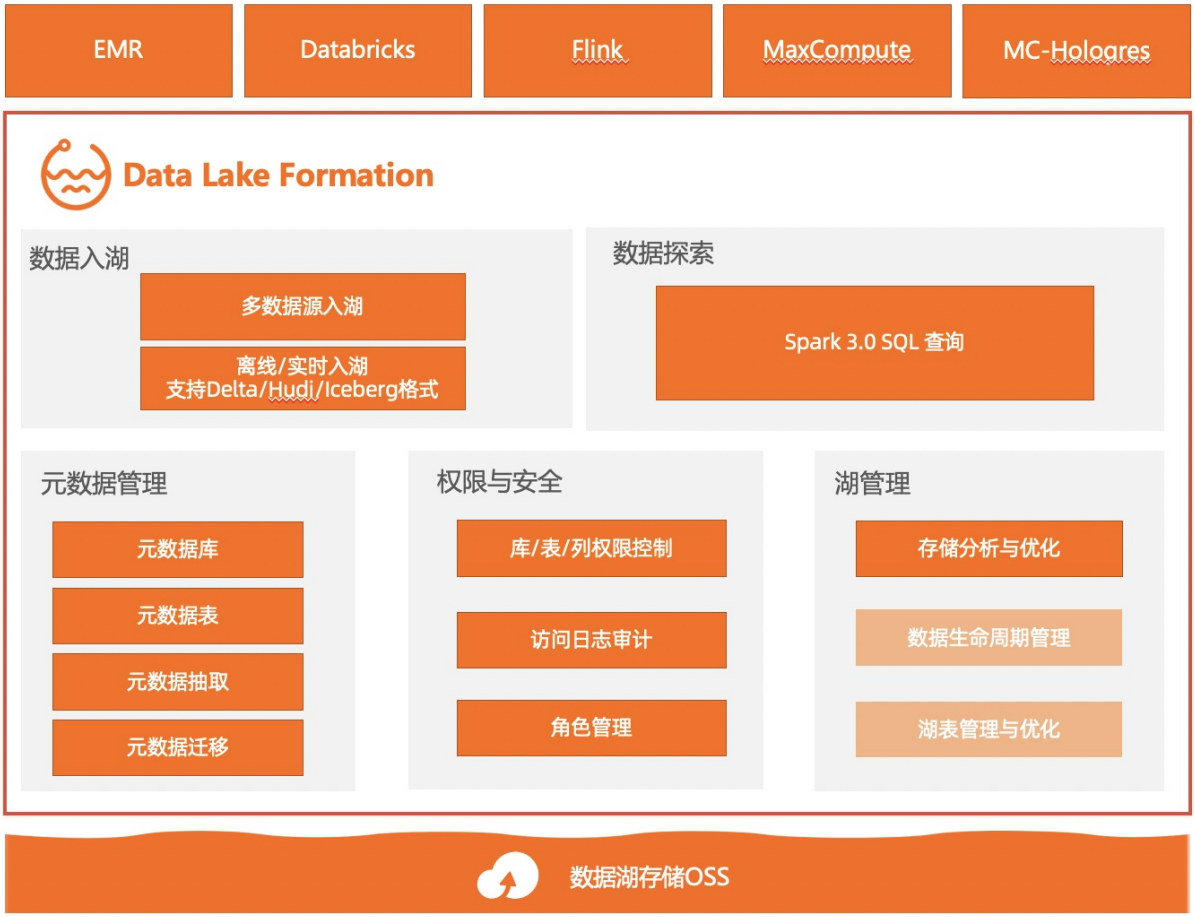
1.1. 产品简介

阿里云数据湖构建（Data Lake Formation，简称 DLF）是一款全托管的快速帮助用户构建云上数据湖及Lakehouse的服务，为客户提供了统一的元数据管理、统一的权限与安全管理、便捷的数据入湖能力以及一键式数据探索能力。DLF可以帮助用户快速完成云原生数据湖及Lakehouse方案的构建与管理，并可无缝对接多种计算引擎，打破数据孤岛，洞察业务价值。

产品定价

- 数据湖构建的数据入湖、数据探索、权限管理、湖管理功能均为公测**免费阶段**，无计费。
- 元数据管理功能按量计费，每个月元数据对象存储<=100万个以下**免费**；超过上述数量会有计费，详情请参考计费。
- 每个月API请求数量<=100万个以下**免费**。超过上述数量会有计费，详情请参考计费。

产品功能架构



- 元数据管理，通过控制台查看和管理数据湖中元数据库和表的信息，通过新增元数据库的方式操作元数据，集成到第三方应用服务。并支持多版本管理、可通过元数据发现和入湖任务自动生成元数据。

- 入湖基础操作，通过入湖任务的方式将分散在MySQL、Kafka和PolarDB等数据统一存储，入湖过程如果没有定义元数据信息，入湖任务会自动生成元数据的表信息。
- 数据权限管理，可以加强湖上数据权限控制，保障数据安全。可支持对元数据库、元数据表、元数据列三种粒度的权限。
- 数据探索，为您提供一键式数据探索能力，可支持Spark 3.0 SQL语法，可以保存历史查询，预览数据，导出结果，一键生产tpc-ds测试数据集。
- 湖管理，将为您提供对湖内数据存储的分析及优化建议，加强对数据生命周期管理，优化使用成本，方便您进行数据运维管理。

应用场景

- 数据分析场景，通过元数据发现、数据探索能力，可以快速的对OSS内结构化、半结构化数据进行分析、探索。
- 结合E-MapReduce、OSS两个产品，DLF协助客户快速构建云上数据湖。

- 结合MaxCompute、DataWorks、E-MapReduce3个产品，DLF协助客户快速构建湖仓一体架构。

- 结合Databricks、OSS产品，构建云上全托管Lakehouse数据架构。

1.2. 使用限制

您在使用数据湖构建（Data Lake Formation，简称DLF）控制台和接口时，产品做了如下限制，请在使用时注意不要超过相应的限制值，以免出现异常。

数据湖元数据

限制项	用户配额
单表QPS	500
单表分区数量	100万

数据湖入湖预处理作业

限制项	用户配额
用户入湖作业数量（每个region）	1000个
每个入湖作业最大资源量	100CU

兼容与使用限制说明

1. 以下3种Hive特性不支持，建议采用最新Delta/Hudi/Iceberg方案替代：
   i. 不支持 Hive DB Lock manager API
   ii. 不支持 Hive Db Transaction Manager API
   iii. 不支持 Hive Constraint：如Primary key/Foreign key
2. 不支持Hive SQL Standards Based Authorization
   i. 旧版Hive权限，开源社区已不再发展，建议使用数据湖构建数据权限功能替代
3. 不支持Metastore listener
   i. 建议使用阿里云操作审计API，实现对元数据操作的监听和管理。
4. 不支持Hive LLAP
   i. 建议使用Presto/Spark等引擎替代

1.3. 已开通的地域和访问域名

Region表示DLF的数据中心所在的地域，Endpoint表示DLF对外服务的访问域名。本文主要介绍Region与Endpoint的对应关系。

DLF Region和Endpoint对照表

经典网络情况下各地域均支持HTTPS访问。当前已开通的各地域和Endpoint如下。

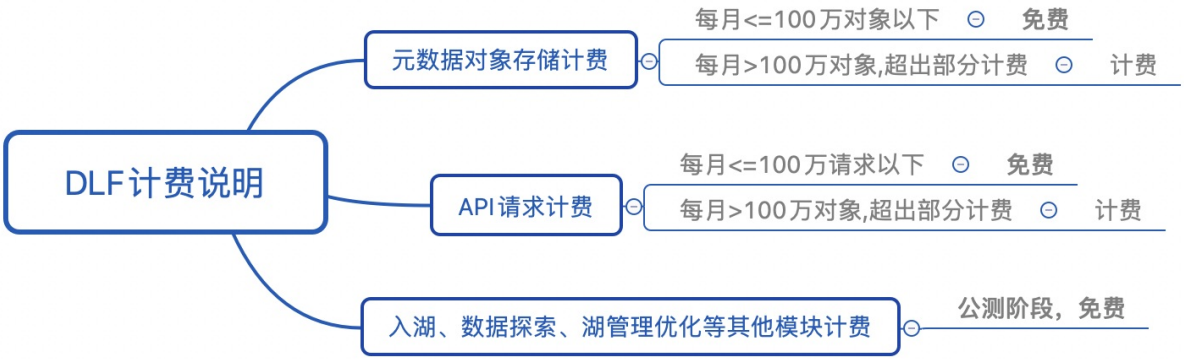
站点	Region	Region Id	公网Endpoint	VPC网络Endpoint	MaxCompute使用Endpoint
中国站	华北2（北京）	cn-beijing	dlf.cn-beijing.aliyuncs.com	dlf-vpc.cn-beijing.aliyuncs.com	dlf-share.cn-beijing.aliyuncs.com
中国站	华东2（上海）	cn-shanghai	dlf.cn-shanghai.aliyuncs.com	dlf-vpc.cn-shanghai.aliyuncs.com	dlf-share.cn-shanghai.aliyuncs.com
中国站	华东1（杭州）	cn-hangzhou	dlf.cn-hangzhou.aliyuncs.com	dlf-vpc.cn-hangzhou.aliyuncs.com	dlf-share.cn-hangzhou.aliyuncs.com
中国站	华南1（深圳）	cn-shenzhen	dlf.cn-shenzhen.aliyuncs.com	dlf-vpc.cn-shenzhen.aliyuncs.com	dlf-share.cn-shenzhen.aliyuncs.com
新加坡	新加坡	ap-southeast-1	dlf.ap-southeast-1.aliyuncs.com	dlf-vpc.ap-southeast-1.aliyuncs.com	dlf-share.ap-southeast-1.aliyuncs.com

2.产品定价

2.1. 计费模式

本文介绍数据湖构建中各资源的计费规格，包括入湖作业资源用量、数据存储对象和元数据请求三部分。

计量计费项



入湖作业资源使用量入湖作业，是指从数据源抽取数据到数据湖统一存储的入湖作业。

- 入湖作业目前是免费服务阶段，暂无计费。

元数据对象的存储量

- 元数据对象是指数据湖元数据中存储的数据库、表、分区和索引。
- 计费规则，**每月前100万个存储对象免费**，超过100万后，超过100万的部分，每10万个对象收费5元/月，不足10万的按10万个收费。
- 计费周期与出账周期，按月统计对象数量，每月1号出账并结算一次费用。

> **说明**
>
> 元数据对象存储量计费案例：
>
> 例如在1月1号，数据湖元数据中包括100个库，1万张表，共50万个分区，0.5万个索引，则
>
> - 月初时，元数据对象的存储量=0.01（库）+1（表）+50（分区）+0.5（索引）=51.51万个存储对象
> - 1月31号，增加60万个分区，库、表和索引数量没有发生变化，则元数据对象的存储量=111.51万个存储对象。
> - 2月1号您收到账单，实际产生的费用=0元（前100万存储对象免费）+10元（100万到111.51万对象存储的费用）=10元。
> - 在2月1号，会从账号中扣除10元作为1月产生的数据湖元数据对象存储费用。

元数据请求

- 元数据请求是指对数据湖元数据中的库、表、分区、索引和函数，发起创建、读取、修改和删除操作请求。
- 发起元数据请求可以通过数据湖构建对接的计算引擎（例如EMR的Hive、SparkSQL、Presto，以及MaxCompute）中执行DDL语句，也可以通过数据湖构建产品控制台或者在API中操作元数据对象。

- 计费规则，**每月前100万个请求免费**。每月请求超过100万后，超过100万的部分，每100万个请求收取5元，不足100万的，按100万请求数收费。
- 计费周期与出账周期，按月统计元数据请求数量，每月1号出账并结算一次费用。

> 说明
>
> 元数据请求计费案例：
>
> - 1月通过EMR、MaxCompute、Hologres等计算引擎创建、访问了元数据，实际访问的元数据次数为120万，则1月实际产生的费用=0元（前100万次元数据请求免费）+ 5元（100万到120万的元数据请求）=5元。
> - 在2月1号，会从账号中扣除5元作为1月产生的数据湖元数据请求费用。

欠费影响

用户在欠费的前72小时内，用户已经运行的任务和数据不受影响。但无法进行新建及资源申请相关的操作：

1. 不能新建数据源
2. 不能新建入湖任务
3. 不能新建数据库、表等
4. 不能启动入湖任务

用户在持续欠费超过72小时后，系统将进一步停止用户对资源的使用：

1. 停止运行中的运行任务
2. 禁止元数据接口的调用

用户在持续欠费超过168小时后，系统会对用户的资源进行释放和删除：

1. 删除用户账户下的数据源、入湖任务、元数据（库、表、分区等）
2. 控制台页面无法正常访问和使用

注：数据释放后无法找回，如有重要数据请提前备份。

3.快速入门

3.1. 快速入门

数据湖构建（Data Lake Formation，DLF）产品主要使用流程如下。

前提条件

注册阿里云账号，并完成实名认证。

创建数据源

创建数据湖的入湖来源，当前支持阿里云RDS MySQL和PolarDB作为数据来源。

- 您需要输入RDS MySQL连接的用户名和密码。
- 选择RDS MySQL所在的VPC、交换机和安全组。

详细操作请参见数据源管理。

创建入湖模板

- 创建入湖模板，可以定时或者手动的执行数据抽取任务，将数据源中指定的数据抽取到数据湖。
- 当前入湖模板支持5种数据抽取方式，可以根据数据抽取的场景选择并创建入湖模板。
- 入湖模板需要指定抽取数据的具体位置。
- 指定RAM角色，数据湖构建服务所代理的角色，默认为AliyunDLFWorkFlowDefaultRole。
- 选择运行抽取任务所需的资源，并指定任务运行方式。

详细操作请参见入湖模板。

创建数据湖的元数据

- 添加元数据库
- 创建元数据表，指定表中数据的存储位置和存储格式

详细操作请参见元数据管理。

4.用户指南

4.1. 概述

数据湖构建可以帮助用户快速构建云上数据湖，采用统一的管理视角治理数据湖。

本产品目前处于公测阶段，您可以随时开通使用，目前数据湖构建所有功能均为免费使用阶段。

用户使用流程

数据湖构建将帮助您快速简洁抽取源数据到统一数据湖的服务，用户使用流程如下：

1. 开通公测流程后，登录阿里云管理控制台，选择**数据湖构建**，进入数据库管理控制台。
2. 参见数据源管理章节，创建数据源，选择希望导入到数据湖的数据来源。
3. 参见入湖模板章节，创建数据湖模板，定期将数据源中的数据抽取到数据湖。
4. 参见元数据管理章节，定义数据湖的元数据库和表。

控制台概览

控制台概览分为2个部分，左侧为主要功能区，右侧为产品主要信息，帮助用户快速上手产品。

注册数据湖位置

阿里云数据湖构建采用OSS作为统一数据湖位置，用户需要注册一个OSS的Bucket或OSS路径作为数据湖位置。

元数据管理

数据湖元数据管理包括元数据库和元数据表两层结构构成。

数据源

用户从数据源抽取数据到注册的数据湖位置，数据湖构建支持多种形式的数据源，目前RDS MySQL已对外开放。

参数	描述
连接名称	数据湖构建中惟一的名称
连接类型	目前支持RDS MySQL
用户名	连接mysql数据库的用户名
密码	连接mysql数据库的密码
虚拟专有网络（VPC）	数据库所在的vpc
交换机（Switch）	数据库所在的交换机
安全组	数据库所在的安全组

入湖模板

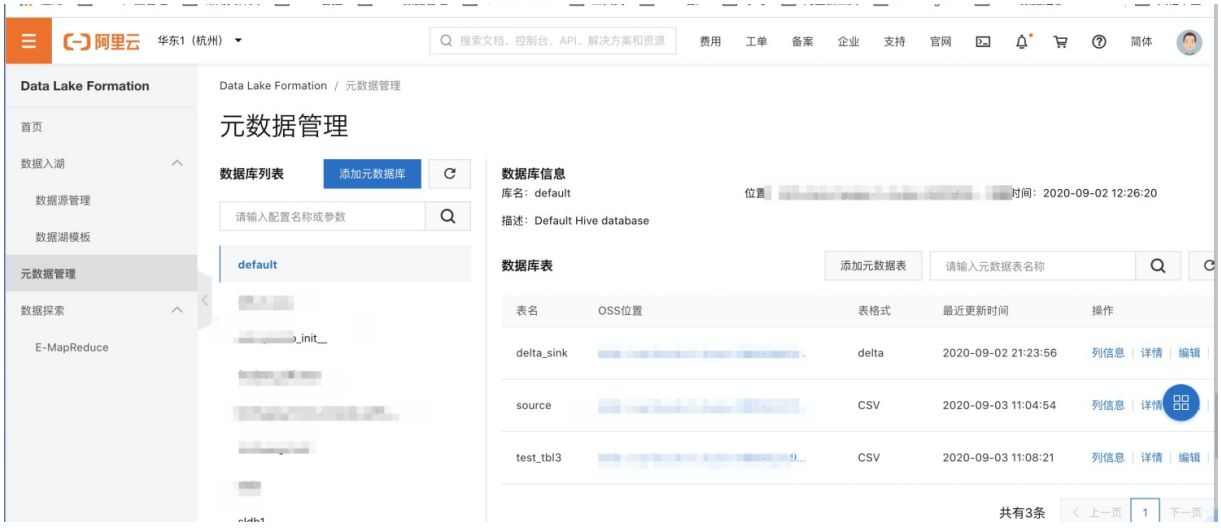
用户创建入湖模板将数据源中的数据通过手动或者定时的方式抽取到数据湖中。

4.2. 元数据

4.2.1. 元数据管理

元数据管理是构建数据湖重要一环，通过有效集中式的元数据管理，可以提升数据资产价值。您可以使用该功能管理元数据库和元数据表。

页面概览



创建元数据库

1. 登录数据湖管理控制台，选择**元数据管理**。
2. 单击**添加元数据库**。
3. 配置元数据库参数。

i. 输入元数据库名称。

ii. （选填）输入元数据库描述。

iii. （选填）输入元数据库的位置。

创建元数据表

1. 创建完成元数据表后，选择**创建元数据表**。
2. 配置元数据表参数。

i. 输入元数据表的名称。

ii. （选填）输入元数据表描述。

iii. 选择元数据表中数据存储的位置。

iv. 选择元数据表的存储格式。

v. 指定元数据表的分隔符。

vi. 手动定义元数据表的列，指定列编号、列名称、是否是分区列等信息。

删除元数据库

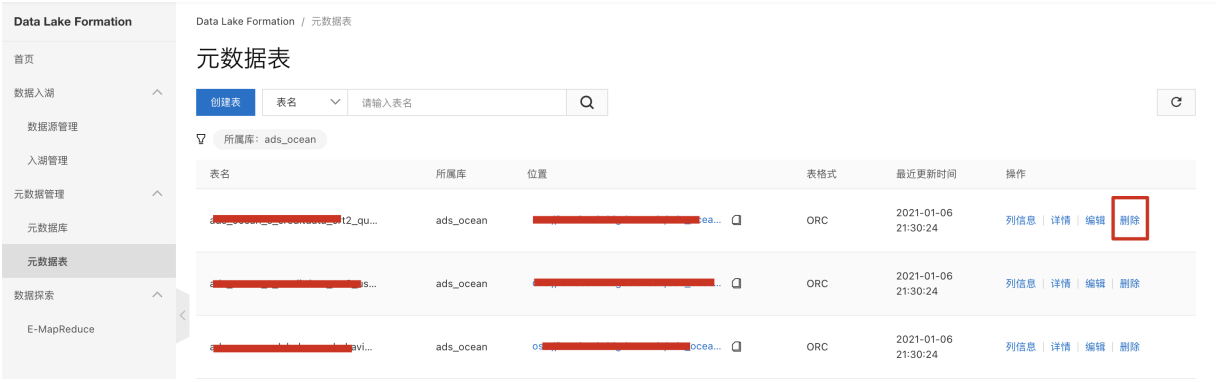
1. 进入元数据管理-元数据库页面。

2.找到想要删除的元数据库，点击右侧的删除按钮，点击确认窗口中的“确认”按钮。

删除元数据表

1. 进入元数据管理-元数据表页面。

2. 找到想要删除的元数据表，点击右侧的删除按钮，点击确认窗口中的“确认”按钮。



4.2.2. 元数据发现

在业务运行中，经常会沉淀到大量数据到数据湖中这部分数据可能是没有像数据仓库一样经过严格的数据管理流程或沉淀规范明确的元数据信息。元数据发现可以分析数据湖中特定格式的数据，并自动生成元数据信息，通过周期性或手动执行，实现数据湖分析计算shema on read目标。

使用限制

1. 抽取的数据存储仅支持数据存储在阿里云OSS标准型中的数据。
2. 目前元数据发现仅支持json，csv，parquet，orc格式。
3. 元数据发现消耗算力不收取费用。

操作步骤

新建抽取任务

1. 登入数据湖构建控制台。
2. 在左侧导航栏，单击**元数据管理> 元数据抽取**。
3. 在元数据抽取页面，单击**新建抽取任务**。
4. 输入元数据抽取任务的配置参数，详细参数说明如下：

参数配置	字段描述
抽取任务名称	元数据抽取任务的名称，输入为中英文数字和（_）。
OSS路径	指定待抽取数据的OSS目录。
排除模式	排查指定OSS目录下的文件，支持正则匹配。
解析格式	支持json，csv，parquet或orc中某一类格式进行抽取，采用自动识别模式会对数据文件自动解析。
目标元数据库	抽取获取的元数据存储的元数据库位置。
目标元数据表前缀	通过元数据抽取生成跟文件一致的表名，输入目标元数据表前缀后，会在表名前添加前缀。
抽取任务发现表字段更新时	当元数据抽取任务获取的表与现有表字段发现不一致时，采取： • 仅新增列，不会删除元数据中原有的列。 • 更新表结构，根据最新探测的表结构生成新的表结果。 • 忽略更新，不修改任何表，现有元数据保持不变。
如何处理OSS中发现已删除对象	当元数据抽取任务探测到原来表对应OSS数据已经被删除，采取： • 删除对应的元数据。 • 忽略更新,不删除任何表。
RAM角色	执行元数据抽取任务时采用的角色，默认为AliyunDLFWorkFlowDefaultRole，赋予DLF产品有作业执行的权限。
执行策略	• 手动执行，通过手动方式触发任务执行。 • 调度执行，周期性的通过指定时间执行元数据抽取任务。

5. 确认任务执行的相关参数，点击**保存并立即执行**。

4.2.3. 元数据迁移

元数据迁移提供可视化的元数据迁移能力，可以帮您快速的将Hive Metastore的元数据迁移到数据湖构建（DLF）中。

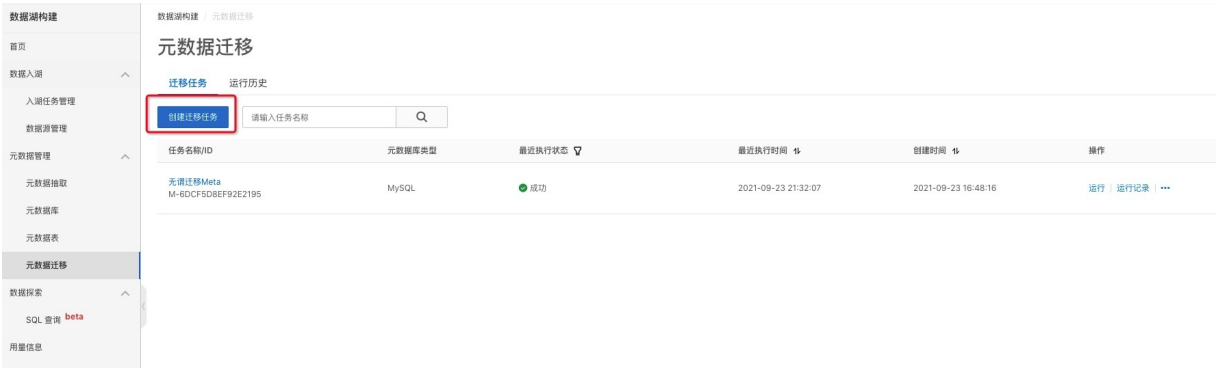
前提条件

- 支持Hive版本：2.3.x 版本。
- 支持元数据库类型：MySQL。

创建元数据迁移任务

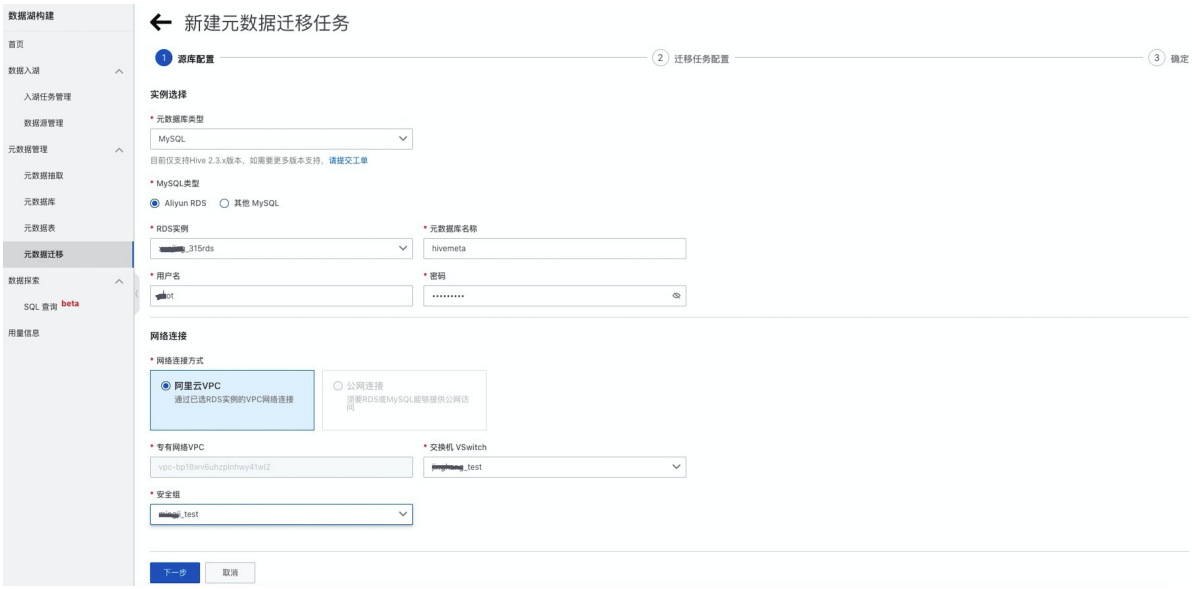
创建迁移任务

1. 打开数据湖构建控制台。
2. 选择左侧菜单“元数据管理-元数据迁移”。
3. 单击创建迁移任务，开始配置元数据迁移任务。

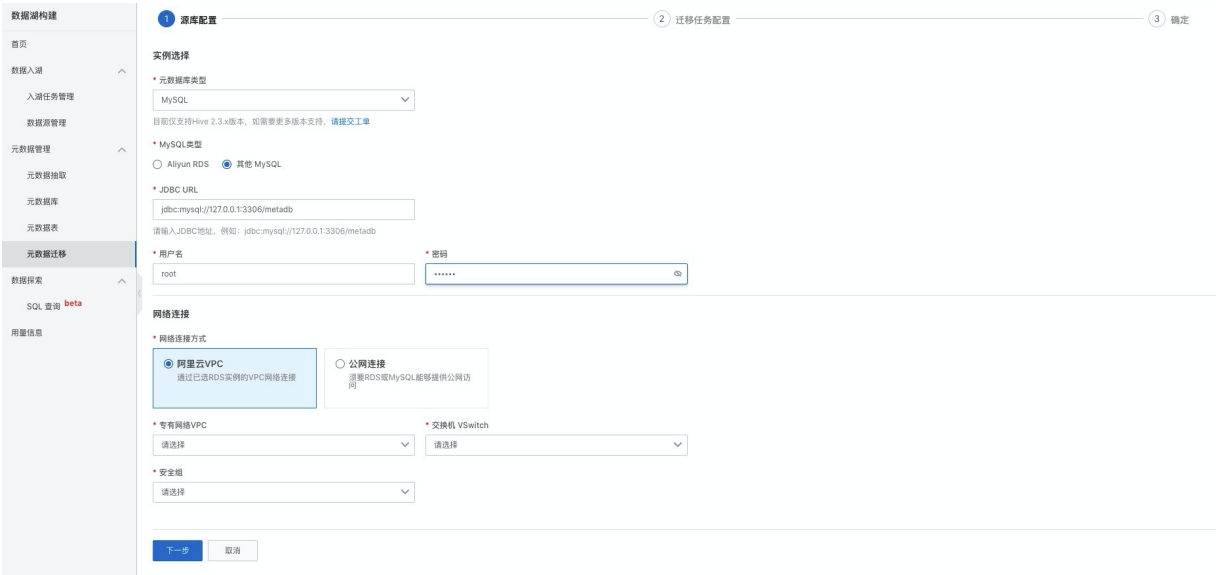


源库配置

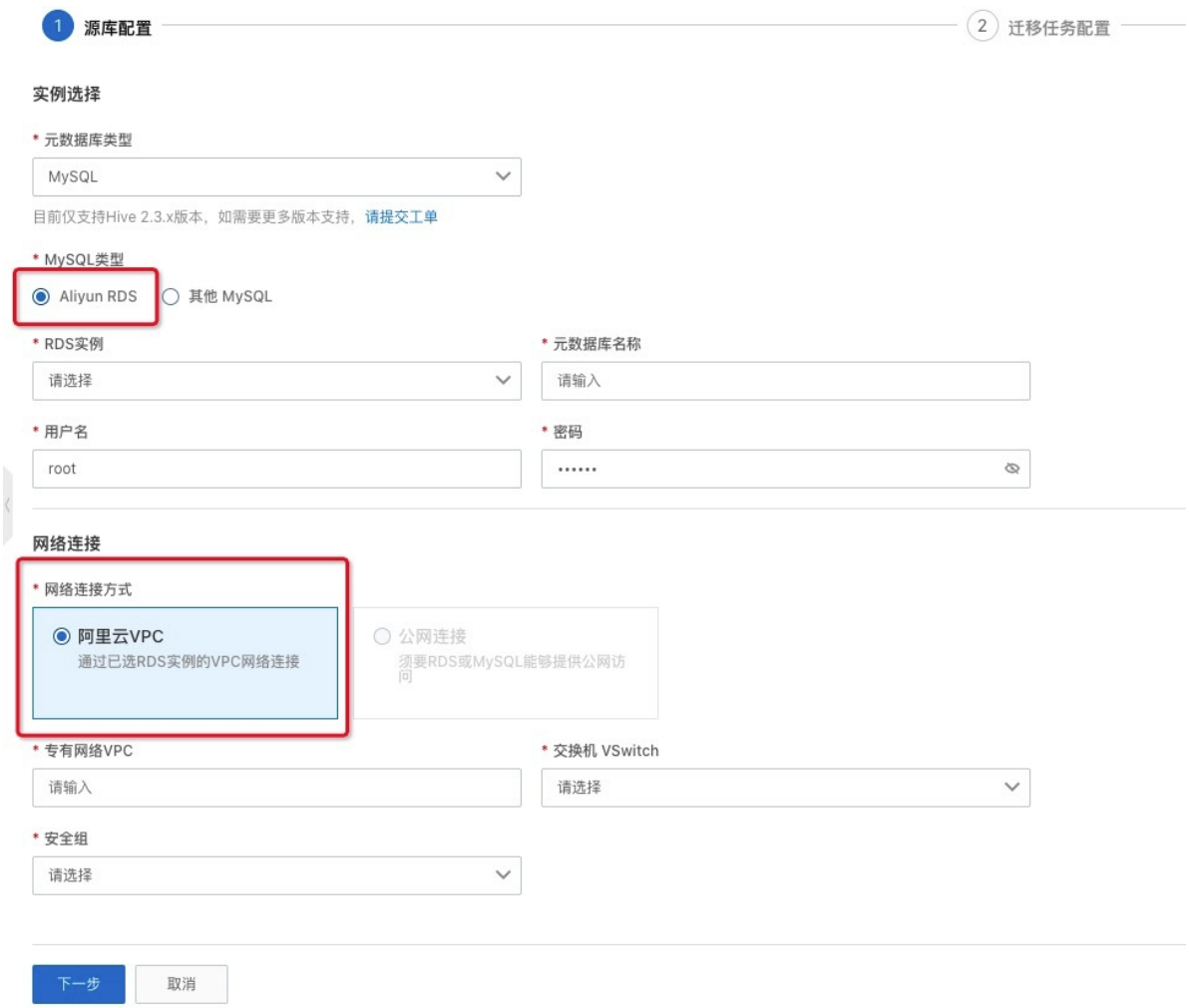
- 元数据库类型：目前仅支持MySQL类型。
- MySQL类型：
  - Aliyun RDS：阿里云官网提供的RDS，可参考云数据库RDS MySQL 版。
  - 其他MySQL：EMR内置MySQL、自建MySQL或其他MySQL数据库。
- 如果选择为Aliyun RDS，则需要填写RDS实例相关信息：
  - RDS实例。
  - 元数据库名称。
  - 用户名。
  - 密码。



- 如果选择为其他MySQL，则需要填写MySQL连接的相关信息：
  - JDBC URL。
  - 用户名。
  - 密码。



- 网络连接配置
  - 当MySQL类型=Aliyun RDS时，此时网络连接方式仅可选择“阿里云VPC”。当您选择VPC连接时，建议选择与RDS或MySQL匹配的VPC，VSwitch与安全组，避免网络出现问题。

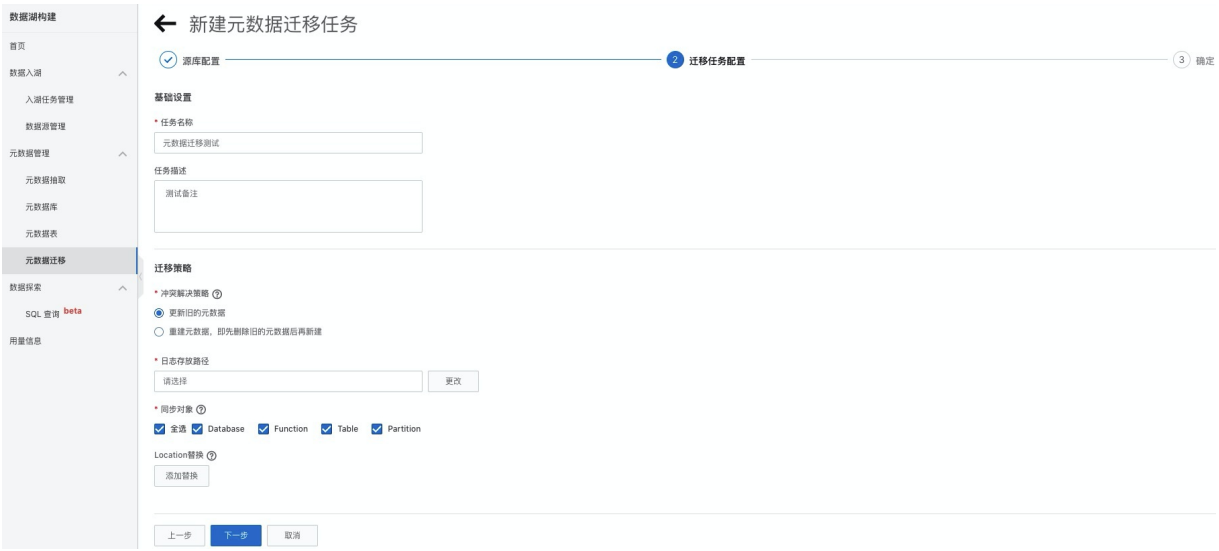


- 当MySQL类型=其他MySQL时，此时网络连接方式可选择“阿里云VPC”，“公网连接”两种方式。

> **说明**
>
> 当您选择公网连接时，请确保公网MySQL允许远程访问，并且端口对弹性公网IP 121.41.166.235 放开（DLF元数据迁移会使用该IP访问您的MySQL）。

迁移任务配置

- 任务名称：输入元数据迁移任务的名称。
- 任务描述（可选）：输入您的一些任务备注信息。
- 冲突解决策略：
  - 更新旧的元数据：旧数据不会被删除，在原有基础上更新元数据。
  - 重建元数据，即先删除旧的元数据后再新建：旧数据将会被全部删除，重新同步元数据并新建。
- 日志存放路径：所有任务日志将会存放的OSS位置。
- 同步对象：包括Database、Function、Table、Partition 四种对象，一般为全选。
- Location替换：适用于迁移过程中需要对table/database的location进行替换修改的情况。比如从传统HDFS架构，迁移到OSS存算分离架构，则需要将hdfs://路径，替换为oss://路径等情况。



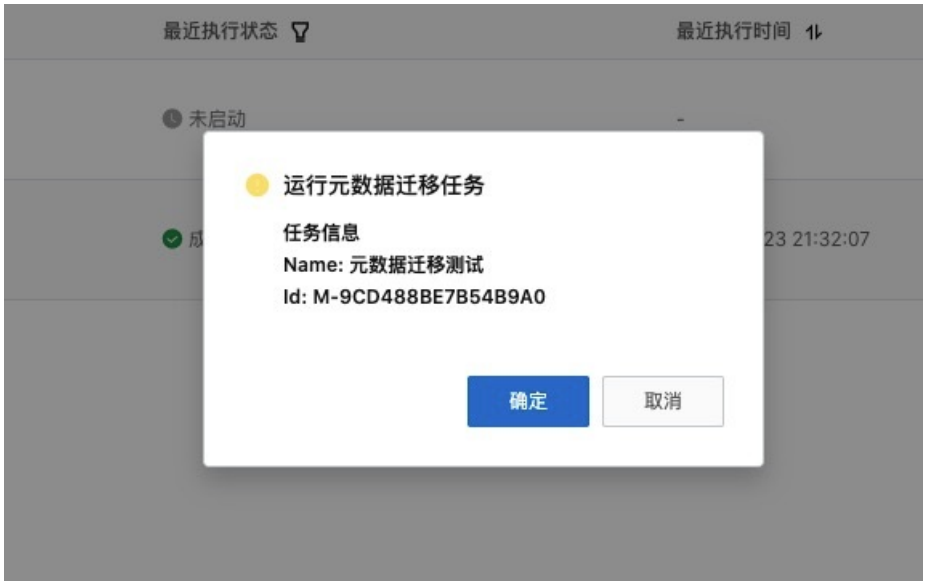
保存任务

确认任务配置信息无误，单击确定按钮，创建任务完成。

运行元数据迁移任务

- 单击每行迁移任务右侧操作“运行”，运行当前元数据迁移任务。

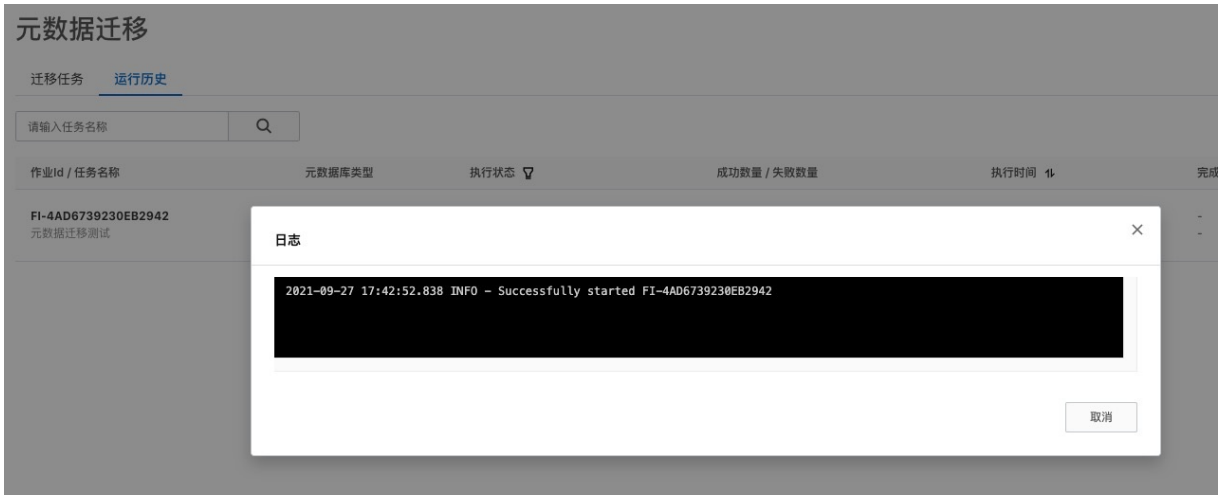
- 在弹出提示框中，单击确定后，开始执行元数据迁移任务。



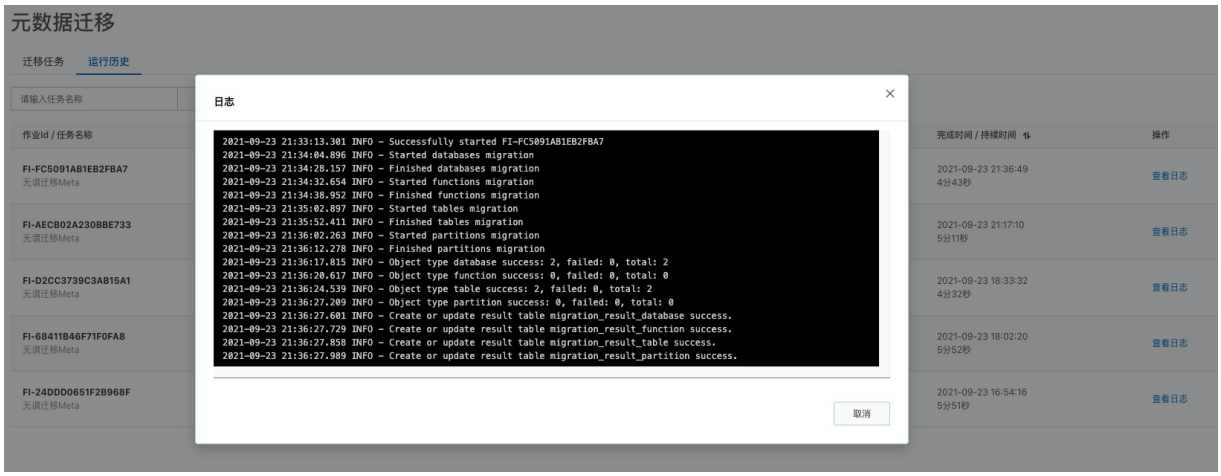
- 任务开始运行中，此时单击右侧 “停止”操作，可以停止当前任务。

- 单击右侧操作 “运行记录”，可以查看运行的详情信息。

- 单击查看日志，可以查看运行中的日志信息。

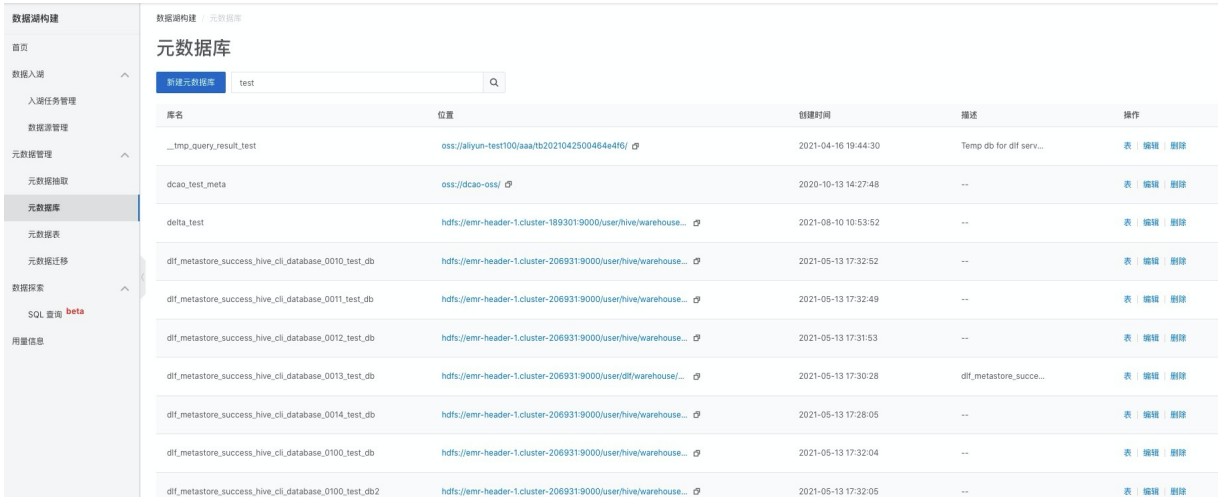


- 元数据迁移完成后，可以从日志中看到成功的结果信息。

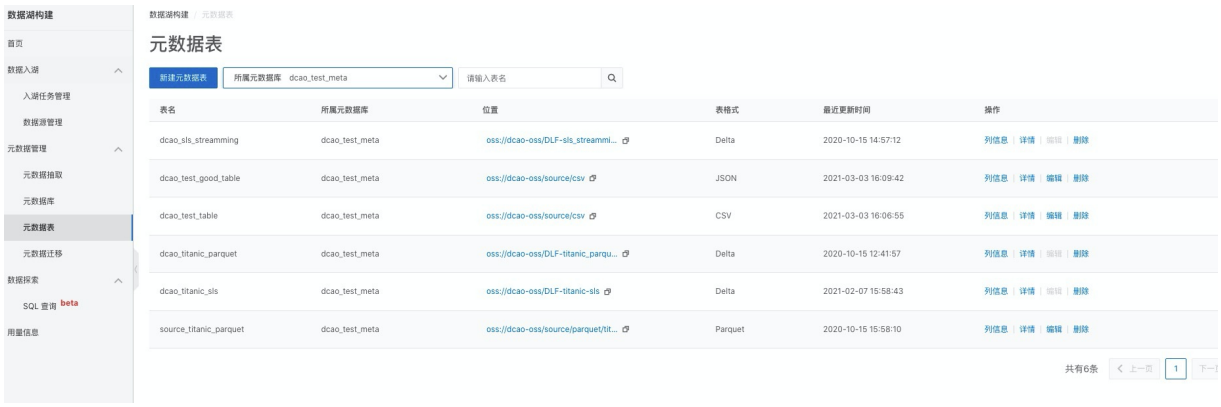


验证元数据同步结果

- 在元数据管理-元数据库中，查询期望同步的数据库信息，可以查询到相应的数据库信息。



- 在元数据管理-元数据表中，查询期望同步的数据表信息，可以查询到相应的数据表信息。



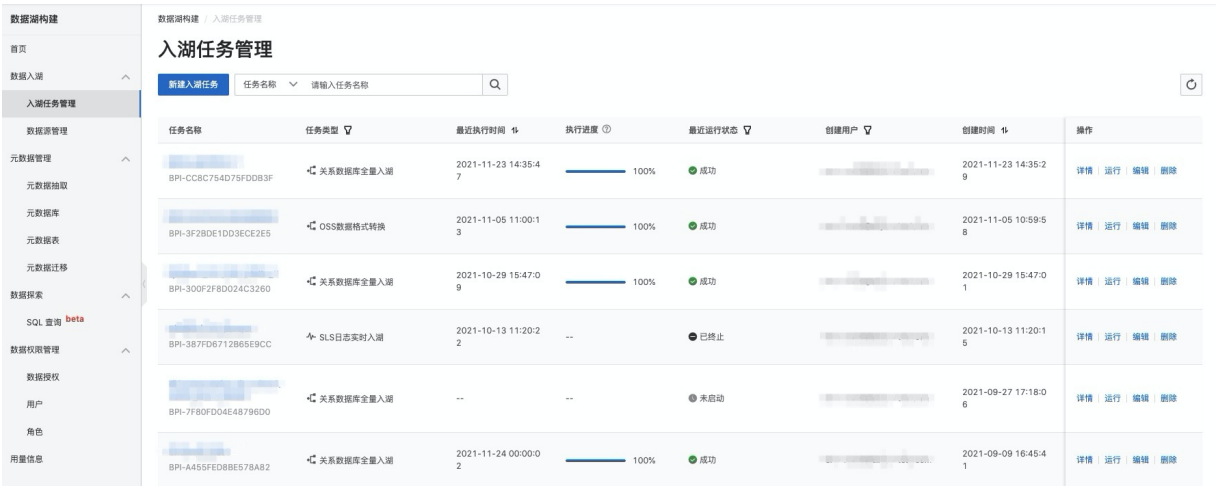
4.3. 数据入湖

4.3.1. 入湖基础操作

数据湖构建（Data Lake Formation，简称DLF）支持多种入湖任务类型用于快速构建数据湖，通过入湖任务，您可以定义数据入湖的方式和所消耗的资源。本文为您介绍入湖任务的基础操作。

页面概览

访问入湖任务管理页面，可以查看入湖任务的当前运行状态、执行进度、创建时间与修改时间等信息，手动停止、启动或删除一个入湖任务。



创建入湖任务

您可以参考如下步骤创建一个入湖任务。

1. 登录数据湖构建控制台，选择**数据入湖 > 入湖任务管理**。
2. 单击**新建入湖任务**，选择入湖任务类型。DLF目前支持6种类型的入湖任务，用于满足不同的数据入湖场景。

- 关系数据库全量入湖：可以实现RDS MySQL或PolarDB数据库表数据全量同步到数据湖，选择在数据源管理中配置好的数据源，将数据源对应数据库中选定表的数据全量同步到OSS中，如果待同步表中数据量大，则所消耗的资源也会比较大，建议在业务低谷期执行此类任务，避免对业务连续性产生影响。待抽取的数据表须确认包含主键，否则抽取任务会报错。
- 关系数据库实时入湖：可以实现RDS MySQL或PolarDB数据库表数据增量同步到数据湖，选择在数据源管理中配置好的数据源，将数据源对应数据库中选定表的数据抽取binlog的方式将数据库表数据实时同步回放到数据湖中。待同步的数据表需确认包含主键，否则抽取任务会报错。

- SLS日志实时入湖：可以实现阿里云日志服务SLS数据实时同步到数据湖，您可以选择当前账号下的SLS Project，该Project下对应的Log Store，实时的将数据同步到数据湖中。
- 表格存储(TableStore)实时入湖：可以实现将阿里云表格存储TableStore（原OTS）数据同步到数据湖中，入湖任务通过实时读取binlog方式，实时的将TableStore指定表数据同步到数据湖中。
- OSS数据格式转换：可以实现对阿里云对象存储OSS中已有数据进行格式转换，如CSV转Parquet、Parquet转Delta等。
- Kafka实时入湖：可以实现将阿里云消息队列Kafka或EMR Kafka的数据实时同步到数据湖中，支持用户自定义预处理算子。

3. 配置数据源与目标数据湖信息。
   - 配置数据源。不同类型的入湖任务配置项有所差异。以关系数据库全量入湖为例，如下图所示。

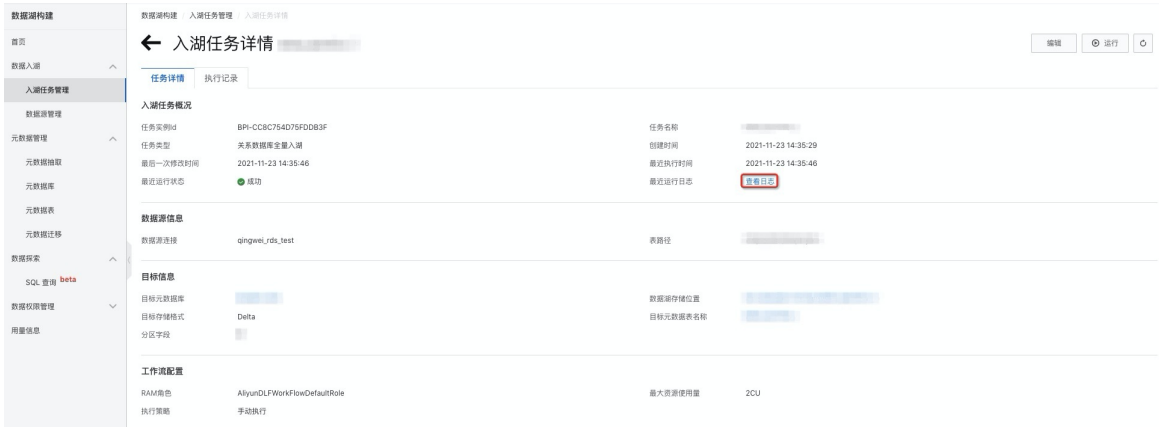
   - 配置目标数据湖信息。主要包括：
     - 目标元数据库：目标表所在的元数据库。
     - 目标元数据表名称：定义目标表名称。
     - 存储格式：选择数据在数据湖中的存储格式，目前支持Delta，Iceberg，Hudi，Parquet，ORC。
     - 数据湖存储位置：数据存储的OSS路径，入湖任务会自动创建此处填写的空文件夹来存储数据。

4. 配置任务信息。主要包括：
   - 任务实例名称：设置入湖任务名称。
   - RAM角色：设置数据湖构建服务所代理的角色，默认角色为AliyunDLFWorkFlowDefaultRole。您可以根据业务需要在RAM中自定义一个Role。
   - 最大资源使用量：设置运行入湖任务所需要的资源。数据湖构建采用计算单元为计算单位，1个计算单元（CU，Computing Unit）包含2 vCPU，8GiB内存的计算资源。
   - 执行策略：设置入湖任务触发方式，手动方式或定时调度的方式。只有全量入湖任务需要设置。

查看入湖日志

您可以参考如下步骤查看一个入湖任务的日志。

1. 登录数据湖构建控制台，选择**数据入湖 > 入湖任务管理**。
2. 找到想要查看日志的入湖任务，点击右侧的“详情”按钮，进入如下入湖任务详情页面。

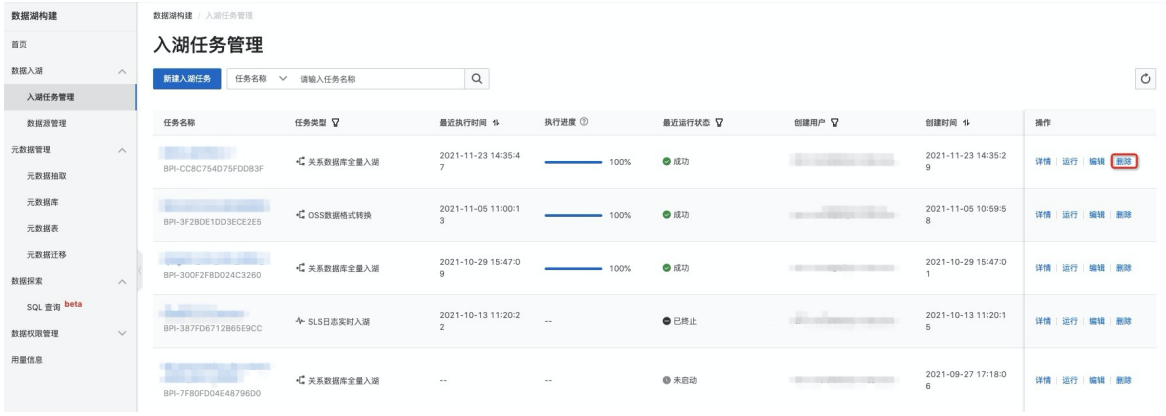


3. 单击如上任务详情页面中的“查看日志”按钮，会弹出如下日志详情窗口。

删除入湖任务

您可以参考如下步骤删除一个入湖任务。

1. 登录数据湖构建控制台，选择**数据入湖 > 入湖任务管理**。
2. 找到想要删除的入湖任务，点击右侧的“删除”按钮，在弹框中点击“确认”按钮。



4.3.2. Kafka实时入湖

本文为您介绍如何在数据湖构建（Data Lake Formation，简称DLF）中使用Kafka实时入湖任务将数据同步到数据湖中。

前提条件

已开通数据湖构建服务，尚未开通请点击开通数据湖构建。

操作步骤

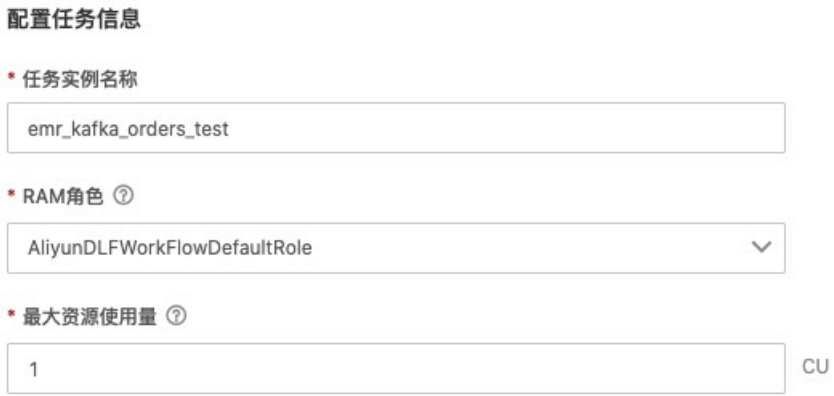
1. 登录数据湖构建控制台，选择**数据入湖 > 入湖任务管理**。
2. 单击**新建入湖任务**，选择“Kafka实时入湖”并单击下一步。

3. 配置数据源。
   - 数据源连接：Kafka数据源需要提前在数据源管理中创建，目前支持阿里云消息队列Kafka与EMR Kafka集群。
   - 订阅Topic：Kafka Topic的名称，一个入湖任务仅支持订阅一个Topic，不支持同时订阅多个Topic。

4. 数据预处理。目前Kafka入湖任务中需要通过定义预处理算子的方式，对Kafka Topic中的消息进行解析、过滤等处理。详情请参见数据预处理。

5. 配置目标数据湖信息。配置项包括目标数据库、目标数据表名称、存储格式、数据湖存储位置等。

6. 配置任务信息。配置项包括任务实例名称、RAM角色、最大资源使用量等。



数据预处理

Kafka实时入湖提供了数据预处理功能对Kafka数据在入湖前进行处理，目前需要用户自定义预处理算子实现。

Kafka默认字段列表

在使用数据预处理功能之前，我们需要清楚Kafka入湖过程中目标schema包含哪些字段，字段列表如下。

字段	类型	说明
key	binary	Kafka消息key
value	binary	Kafka消息体

字段	类型	说明
topic	string	Kafka topic
partition	int	Kafka分区值
offset	long	Kafka消息偏移量
timestamp	timestamp	时间戳
timestampType	int	时间戳类型

数据预处理含义

数据预处理是Kafka实时入湖提供的用于对入湖数据预先处理的功能。数据预处理支持使用Spark SQL函数定义预处理算子，目前支持map与filter两种算子。定义预处理算子需要注意以下几点：

- 第一个预处理算子需要基于Kafka入湖的schema来编写，请参考如上字段列表。
- 每一个预处理算子的处理逻辑可以看作一个SQL的子查询。算子按照定义顺序执行，后面算子需要基于前面算子的输出字段来编写SQL函数。
- map算子类似SQL的select操作，由若干个SQL表达式逗号隔开组成，每个表达式必须包含通过as指定表达式别名。filter算子类似SQL的where语句，对前面的算子输出的字段进行过滤。

数据预处理示例

您可以参考以下示例编写自己的预处理算子。

- 提取Kafka消息体与时间戳字段。通过定义一个map算子实现，表达式参考如下。

```
cast(value as string) as content, from_unixtime(cast(timestamp as bigint), 'yyyy-MM-dd') as dt
```

- 展开标准Json格式日志数据。通过定义一个map算子实现，表达式参考如下。

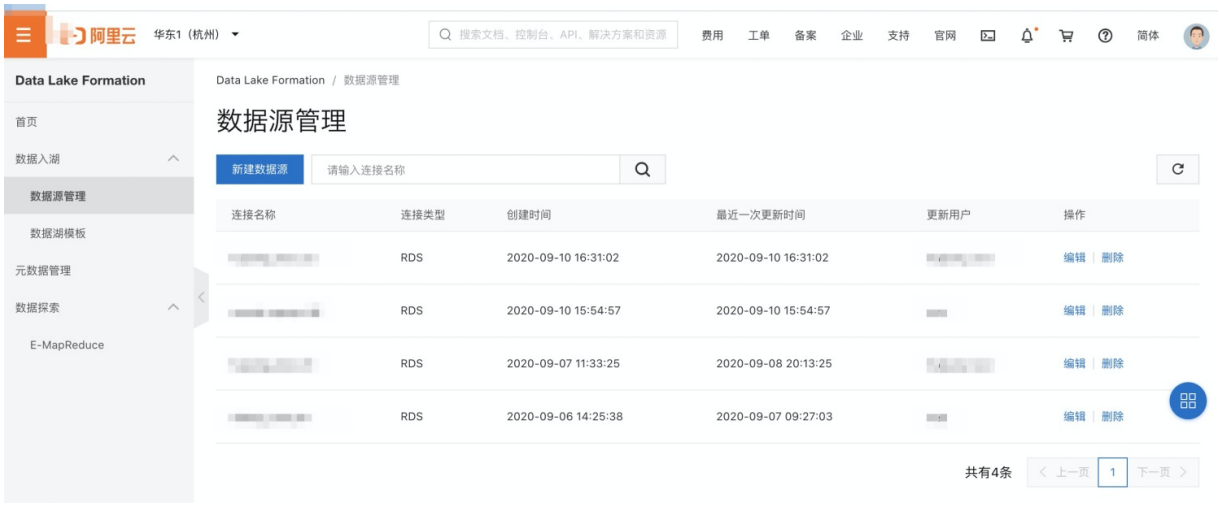
```
get_json_object(cast(value as string), '$.id') as id, get_json_object(cast(value as string), '$.eventType') as eventType, get_json_object(cast(value as string), '$.bizCode') as bizCode
```

- 过滤ID字段值大于1000的数据。通过一个filter算子实现，表达式参考如下。

```
id > 1000
```

4.3.3. 数据源管理

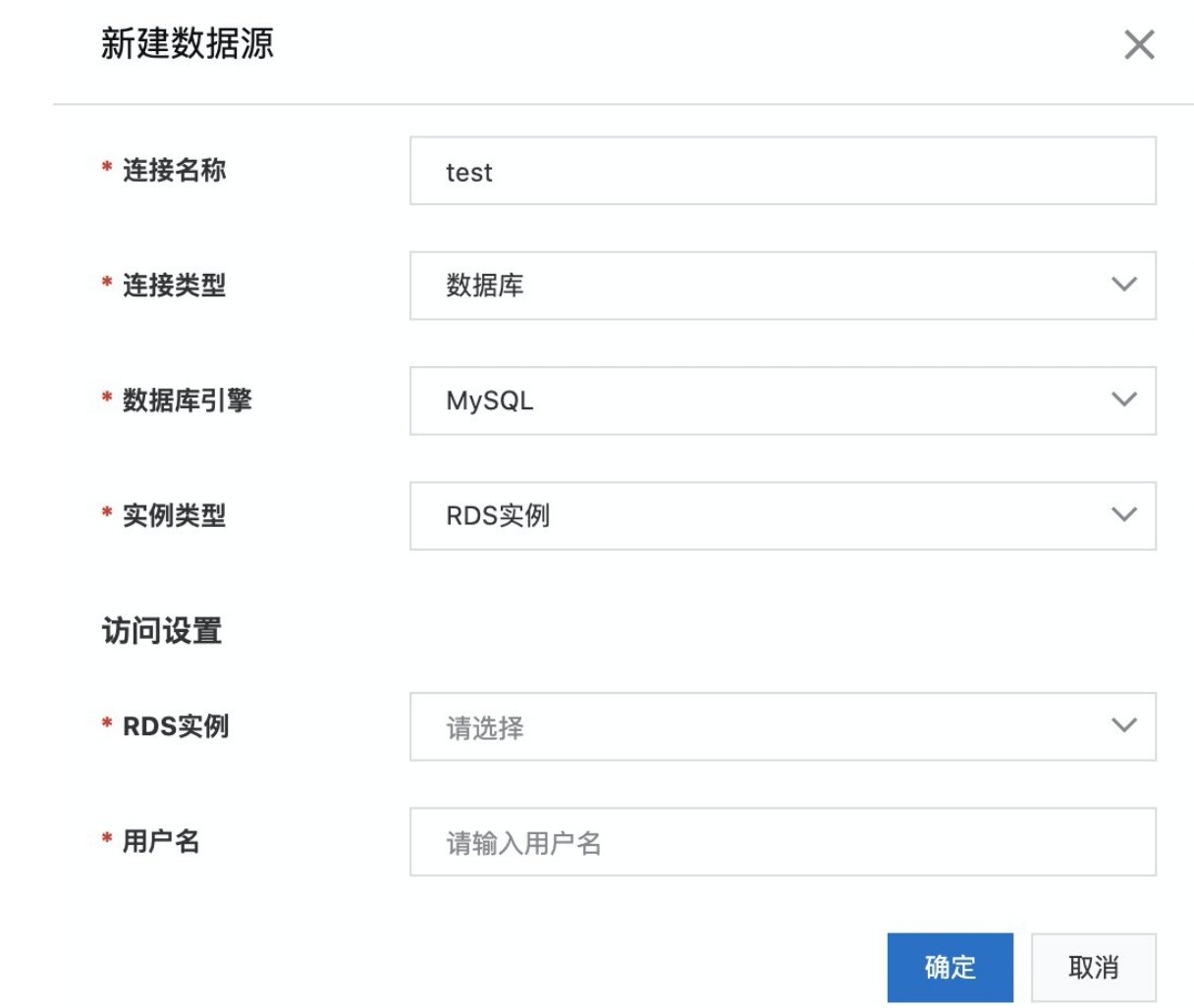
数据源管理是管理入湖数据来源的入口，公测阶段已经支持RDS MySQL作为数据湖的来源。您可以新建、编辑和删除数据源。



创建数据源

创建一个数据源，需要指定如下要素：

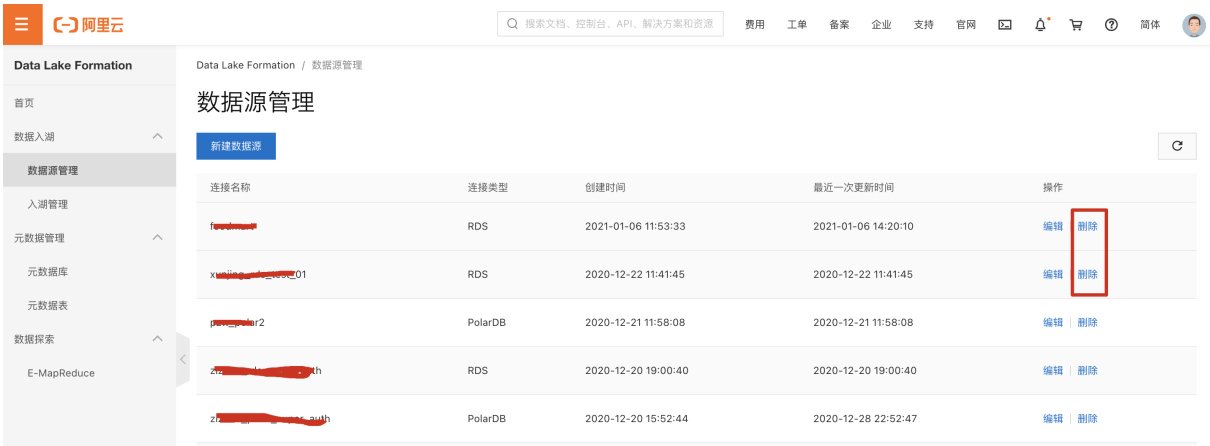
1. 连接名称，数据源连接名称在子账号维度是唯一的，即数据源连接名称不能重复。
2. 数据源连接的类型和方式，目前仅支持阿里云RDS MySQL的数据源连接。
3. 根据不同的数据源连接方式选择连接访问方式。



删除数据源

1. 进入数据源管理页面。

2. 找到需要删除的数据源， 点击右侧的“删除”按钮，在弹出框中点击“确认”。



4.4. 数据探索

4.4.1. 数据探索简介

数据探索是一种线上的交互式查询服务，开通即用。它是完全托管的，并且具备了高性能、弹性、易用等特点，无需申请任何资源即可直接使用。 用户可以对入湖后的数据使用Spark SQL快速的进行数据探索，以便对湖内数据进行审核，质量检查，分类等。 支持数据湖内多种存储格式，包括Delta、Hudi、CSV、Parquet、JSON、ORC等数据格式。

准备工作

1. 已开通OSS。
2. 已开通DLF，并创建了元数据库。
3. 通过元数据发现 /入湖任务管理 或者API等方式创建了元数据表。

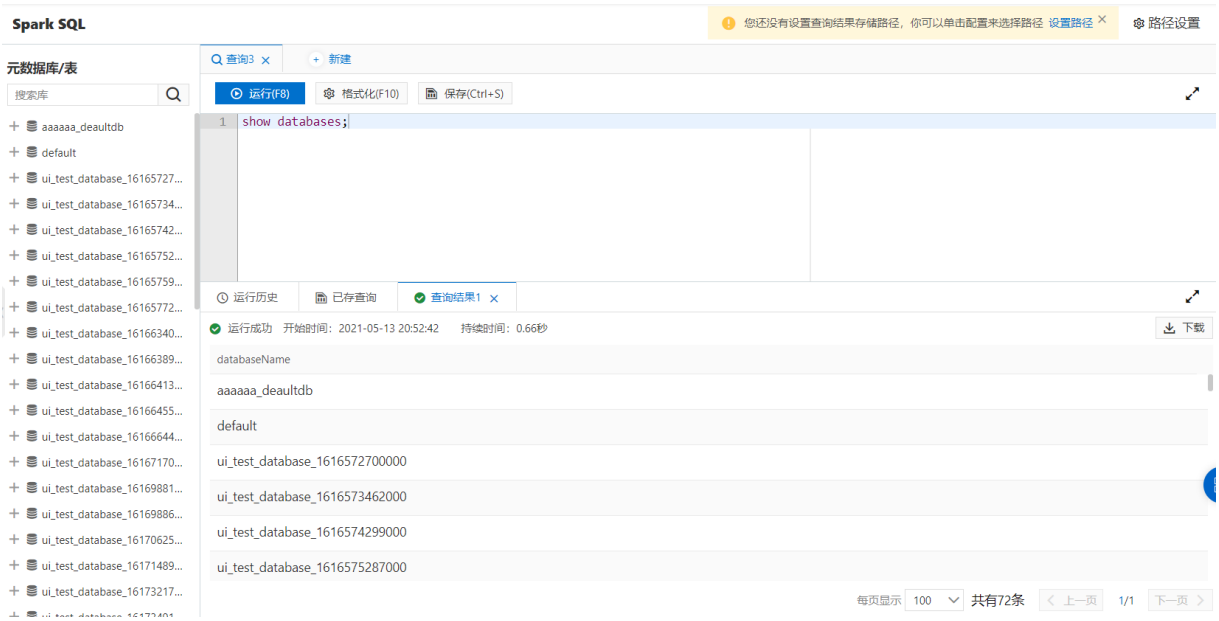
运行查询

1. 登录数据湖构建控制台，在左侧菜单中选择**数据探索**。
2. 左侧元数据库/表区域，会列举出当前账户下所有元数据库和元数据表。您可以在此区域查看元数据表的基础信息，或者生成数据预览SQL语句。
3. 在右侧SQL编辑器区域，输入SQL语句。本功能基于EMR Spark 2.4版本，更多特性详情请参见Spark SQL Guide。示例如下：

```
-- SQL语句示例
show databases;
```

4. 点击**运行**（或快捷键F8），下方会展示查询进度状态，当查询完毕时会直接显示查询结果。查询结果会分页展示，受前端限制目前最多展示10000条数据。如需获取全部查询结果，可以在配置存储路径之后进入OSS查看，或直接点击下载。

> **注意**
>
> DLF-Spark SQL不会在您的SQL语句后面自动加limit限制，请避免不必要的全表扫描，以免造成资源浪费。



使用限制：

1. SQL执行超时时间：60分钟
2. SQL长度限制：不超过6000字符
3. 查询结果展示：最多10000行
4. 同一个账号，最大使用Spark Driver内存：4G
5. 同一个账号，最大使用CU限制：200CU （1CU=1核4GB）

结果路径设置

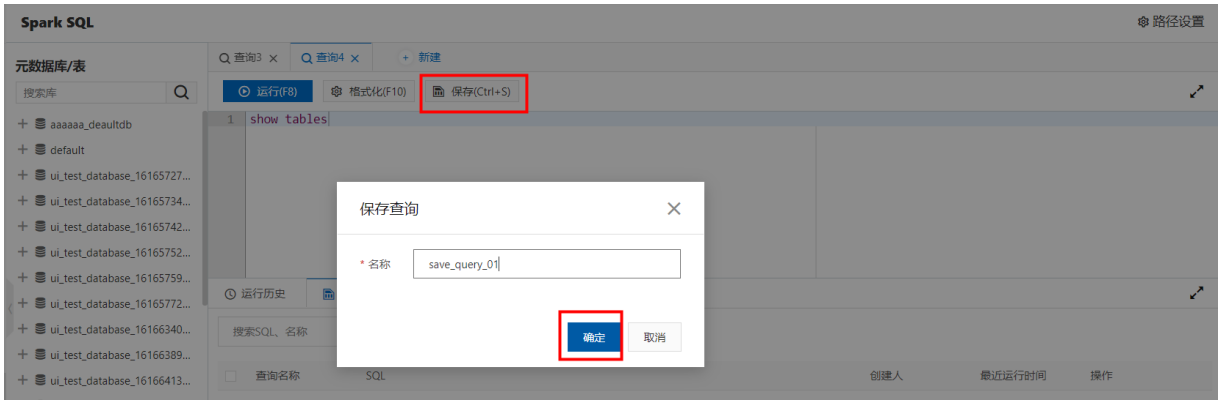
您可以通过路径设置，把每次查询结果保存在OSS上，以便于全量结果的下载和归档。仅当设置了保存路径之后，才可以使用结果下载功能。保存的结果文件没有时间限制。

1. 进入**数据探索-Spark SQL**，点击右上方**路径设置**按钮。
2. 在弹出的OSS输入框中，选择用于保存查询结果的OSS路径。并点击**确定**。
3. 设置成功之后，执行的查询结果会自动全量写入您设置的OSS路径中，目前默认以**CSV格式**保存。如果您的查询结果很大时，下载导出可能需要几分钟，请耐心等待。

保存查询

对于常用的查询，您可以直接保存。

1. 在SQL编辑器中输入SQL语句，点击上方的**保存**按钮，或者Ctrl+S快捷键。
2. 在弹出的输入框中，输入本次保存的查询名称。
3. 保存成功后，您可以在下方的**已存查询**中，恢复保存过的查询。



运行历史

当前用户下，每次执行的查询都会记录在运行历史列表中。

1. 打开下方**运行历史**TAB页。
2. 运行历史列表中，展示每次查询的运行开始时间、原始SQL语句、执行耗时和状态，如果您已经配置过结果路径，可以看到每次查询结果保存的OSS路径，或者直接进行下载操作。

4.4.2. 使用数据探索查询Iceberg表

本文为您介绍如何使用DLF数据探索查询Iceberg表。目前DLF数据探索可以直接支持Delta、Hudi、CSV、Parquet、JSON、ORC等格式的表；受限于Spark和Iceberg的底层设计，在DLF数据探索中查询Iceberg表需要指定特定的Catalog。

准备工作

1. 已开通DLF，并创建了元数据库。
2. 通过元数据发现 /入湖任务管理 或者API等方式创建了Iceberg元数据表。

操作步骤

1. 登录数据湖构建控制台，在左侧菜单中选择**数据探索**。
2. 在SQL输入框中，输入查询语句。针对Iceberg表，需要在指定的元数据库和表之前，**加上dlf_catalog.前缀**。例如：

```
SELECT * FROM dlf_catalog.database_name.iceberg_table limit 100;
```

4.4.3. 快速使用TPC-DS测试数据集

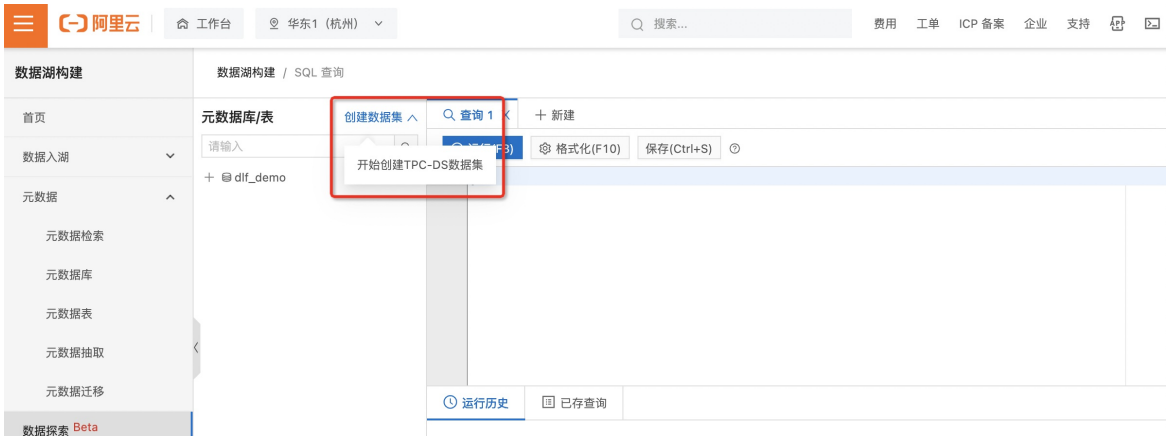
TPC-DS是一套决策支持系统测试基准，提供99个SQL查询（SQL99或2003），分析数据量大，测试数据与实际商业数据高度相似，同时具有各种业务模型（分析报告型，数据挖掘型等等）。使用DLF数据探索，可以便捷地快速创建TPC-DS数据集，便于用户快速上手和测试数据探索的功能。

准备工作

1. 已开通DLF。

操作步骤

1. 登录数据湖构建控制台，在左侧菜单中选择**数据探索**。
2. 在左侧点击**创建数据集-开始创建TPC-DS数据集**按钮。点击后，会自动执行一系列初始化SQL：在您的元数据中创建名为sampledata_tpcds_3g的库，并在库中创建24个TPC-DS的数据表。



3. 无需点击运行，在下方运行历史列表中，可以看到初始化语句的执行情况，刷新列表等待全部执行完成。
4. 执行完成后，就可以在查询输入框中，对新生成的TPC-DS元数据库进行SQL查询了。

4.5. 数据权限

数据湖权限支持配置元数据库、元数据表、元数据列三种纬度的权限。下面针对不同的数据操作，对所需要配置的权限进行详细的说明。

数据权限

- 概述
- 新增授权
- 查询授权
- 取消授权
- 用户管理
- 角色管理

4.5.1. 概述

数据湖权限支持配置元数据库、元数据表、元数据列三种纬度的权限。下面针对不同的数据操作，对所需要配置的权限进行说明，并对Owner权限的定义和权限鉴定方式进行详细说明。

> **说明**
>
> 1. 数据权限功能目前公测中，如需使用，请您提交工单。
> 2. 目前支持的引擎：
>   i. E-MapReduce产品（EMR-3.40.0及后续版本或EMR-5.6.0及后续版本），包括的计算引擎如下：
>   a. Spark
>   b. Hive
>   c. Presto
>   ii. Databricks产品。

背景信息

配置权限时需要包含如下三个要素。

权限要素	说明
主体（Principal）	被授予权限的用户或角色。用户必须为阿里云RAM用户或RAM Role；角色为数据湖角色管理中创建的角色。 Principal具有如下几种格式： • 阿里云主账户：acs:ram::<阿里云账号uid>:root，例如 `acs:ram::123456:root` 。 • 阿里云RAM子账户：acs:ram::<阿里云账号uid>:user/<RAM用户名>，例如 `acs:ram::123456:user/user_a` 。 • 阿里云RAM Role：acs:ram::<阿里云账号uid>:role/<RAM Role名称>，例如 `acs:ram::123456:role/role_a` 。
资源（Resource）	数据湖中管理的资源。 • 库（Database）：数据湖元数据中管理的库 • 表（Table）：数据湖元数据中管理的表 • 列（Column）：数据湖元数据中管理的列
访问资源的方式（Access）	访问资源的方式与资源类型有关，不同资源类型支持的访问方式各不相同。如库支持CreateTable、List等权限，表支持Select、Update等权限，列只支持Select权限。

权限总览

数据湖权限支持的权限列表如下：

资源（Resource）	访问资源的方式（Access）	说明
Database	Describe	查看Database的元数据信息或切换Database。例如 `desc database <database_name>` 、`use <database_name>`等。
	Alter	修改Database。例如 `alter database <database_name> set location '<path>'` 等。
	Drop	删除Database。例如 `drop database <database>` 。
	Create Table	在Database中创建Table。例如 `create table <database_name>.<table_name> ...` 。

资源（Resource）	访问资源的方式（Access）	说明
	List	查看Database下资源列表。例如 `show tables` 。 **注意** • E-MapReduce Spark引擎不支持List权限的鉴定。 • 资源列表暂不支持根据用户权限进行过滤展示，所有资源都将显示出来。
Table	Describe	查看Table的元数据信息。例如 `desc formatted <table_name>` 等。
	Alter	修改Table。例如 `alter table <table_name> add columns ...` 、 `alter table <table_name> drop partition ...` 等。
	Drop	删除Table。例如 `drop table <table_name>` 。

资源（Resource）	访问资源的方式（Access）	说明
	Select	查看Table的数据。例如 `select * from <table_name>` 。
	Update	更新Table的数据。例如 `insert into table <table_name>` 、 `delete from <table_name> where ...` 等。
Column	Select	查看Column的数据。例如 `select <column_name1>, <column_name2> from <table_name>` 。

Owner权限

Owner定义

资源（Resource）的创建者称为资源的Owner。您可以在元数据库或元数据表基本信息中，查看到元数据库或元数据表的Owner信息。

- 使用阿里云RAM用户在数据湖元数据管理中新建元数据库或元数据表时，该RAM用户即为元数据库或元数据表资源的Owner，Owner以用户对应的Principal格式表示。
- 在E-MapReduce引擎中使用Linux或LDAP用户执行SQL命令创建资源时，该Linux或LDAP用户为资源的Owner。
- Databricks引擎暂不支持资源Owner。

> **注意**
>
> - 为了打通阿里云RAM用户与开源大数据用户体系，数据湖权限定义了当RAM用户与Linux或LDAP用户具有相同用户名时，两者作为Owner是等价的，例如Owner=acs:ram::<阿里云账号uid>:user/user_a与Owner=user_a等价。
> - 阿里云主账户作为资源Owner时，没有等价的Linux或LDAP用户。需要特别注意的是Owner=acs:ram::<阿里云账号uid>:root与Owner=root不等价。
> - 您可以在数据湖构建-数据权限-用户功能中点击RAM用户查看用户名信息。在使用E-MapReduce引擎时，建议使用用户管理添加与RAM用户同名的Linux或LDAP用户。

Owner权限的定义

资源Owner拥有对应资源的所有权限。例如元数据库的Owner为RAM用户user_a时，user_a能够有权限执行Alter Database、Drop Database等操作。

> 注意
>
> 资源的Owner权限不能向子资源扩展，例如元数据库的Owner只能获取元数据库的Owner权限，没有该元数据库下所有元数据表的Owner权限。

Owner权限的鉴定

- 使用阿里云RAM用户登录数据湖构建控制台时，用户身份为RAM用户，可以获取Owner为当前用户身份（或等价身份）的资源的Owner权限。
- 使用E-MapReduce引擎访问数据湖元数据资源时，用户身份为Linux或LDAP用户，可以获取Owner为当前用户身份（或等价身份）的资源的Owner权限。
- Databricks引擎暂不支持Owner权限的鉴定。

4.5.2. 新增授权

本文档主要为您说明如何进行新增数据授权。

> 说明
>
> 1. 数据权限功能目前公测中，如需使用，请您提交工单。
> 2. 目前支持的引擎：
>   i. E-MapReduce产品（EMR-3.40.0及后续版本或EMR-5.6.0及后续版本），包括的计算引擎如下：
>   a. Spark
>   b. Hive
>   c. Presto
>   ii. Databricks产品。

打开新增授权页面

1. 打开数据授权控制台，并以数据湖管理员身份或已有授权权限的用户身份登录。
2. 单击左侧菜单：数据权限-数据授权。
3. 单击“新增授权”按钮 ，打开新增授权页面。

指定授权主体

1. 主体类型：可以选择RAM用户/角色或DLF角色。
   - RAM用户/角色：可以选择已有RAM用户/RAM角色，关于RAM用户/RAM角色的管理，可以在RAM控制台进行配置。
   - DLF角色：指在数据湖构建（DLF）中定义的角色，关于角色的管理，可以在数据湖构建控制台进行配置。
2. 主体选择：可以选择一个或多个RAM用户/角色或DLF角色。

选择授权资源

1. 授权方式：目前仅支持资源授权，资源授权指对元数据库、元数据表、元数据列这类资源进行数据权限配置。
2. 资源类型：可选元数据库、元数据表、元数据列。
3. 选择资源实体，可以进行模糊搜索，找到您想授权的库/表/列，并选中。

4.5.3. 查询授权

本文档主要为您说明如何进行查询已有的数据授权信息。

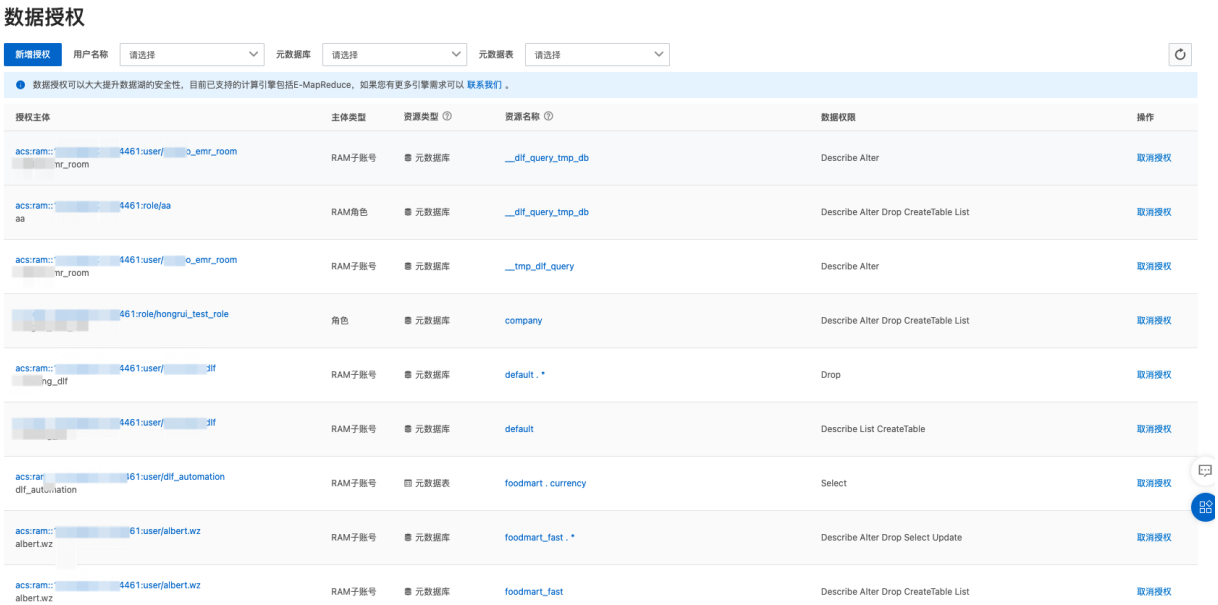
操作步骤

1. 打开数据授权控制台，并以数据湖管理员身份或已有授权权限的用户身份登录。

2. 查看数据授权信息。

相关字段说明如下：

- 授权主体：指被授权的主体ID及名称。
- 主体类型：目前支持RAM用户/RAM角色/DLF角色。
- 资源类型：包括元数据表、元数据库、元数据列三种类型。
- 资源名称：指定具体的资源名称，如数据库的库名。
- 数据权限：授权的权限名称，关于权限描述可参考 权限配置。

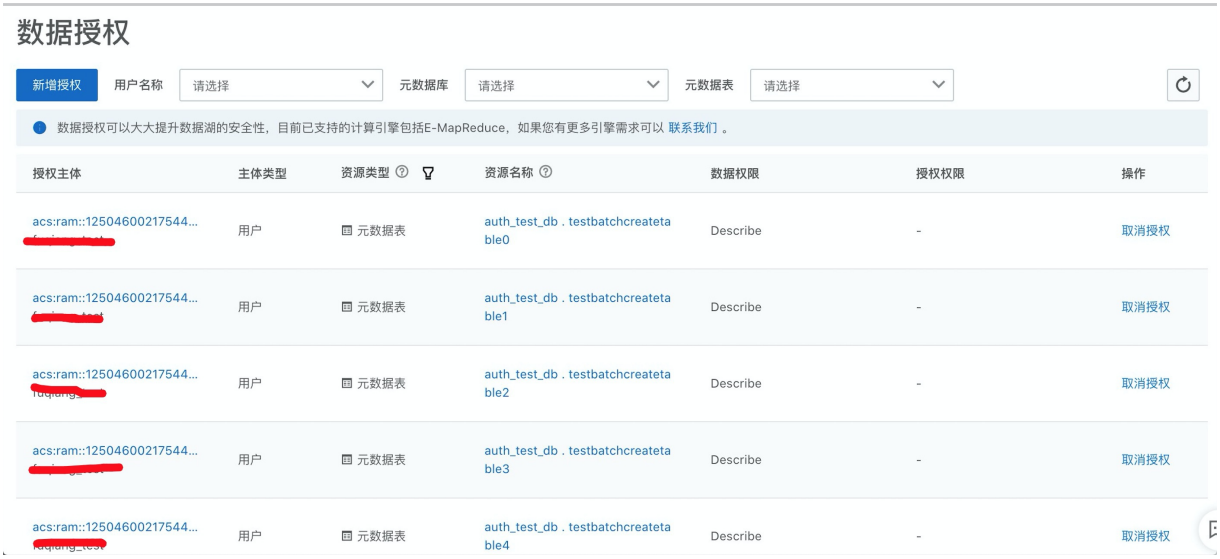


4.5.4. 取消授权

本文档主要为您说明如何取消已有授权权限。

操作步骤

1. 打开数据授权控制台，并以数据湖管理员身份或已有授权权限的用户身份登录，并打开数据授权页面，如下图所示：



2. 搜索您想要取消的授权信息。

3. 点击您要取消的授权信息后的“取消授权”按钮，完成取消授权操作。

4.5.5. 用户管理

本文档主要为您说明如何管理数据湖构建中已添加的用户。

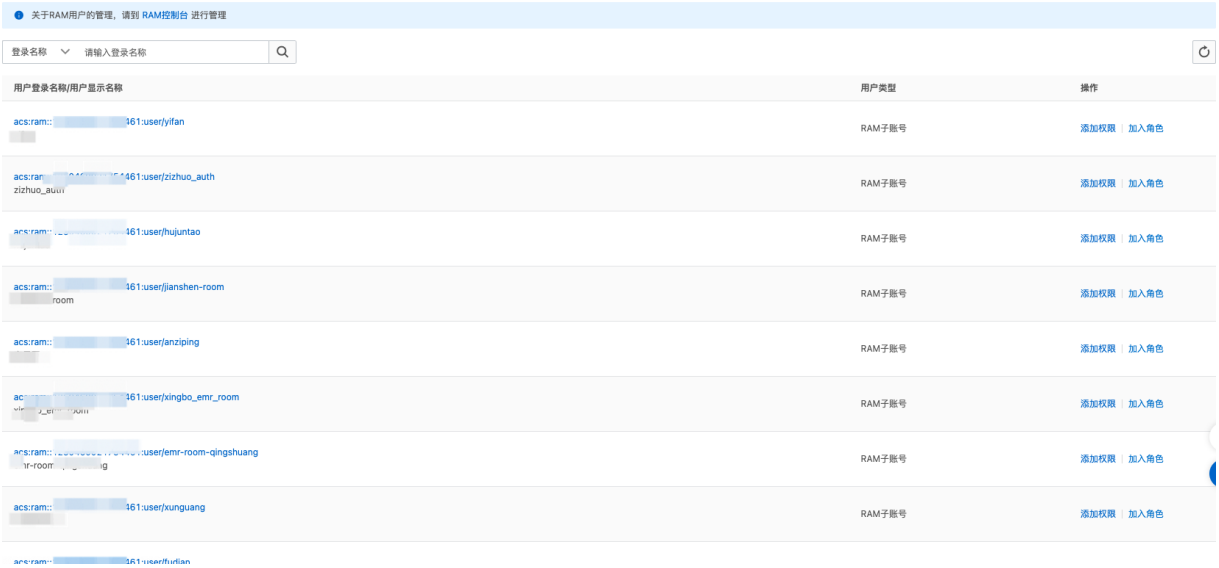
查看用户列表

1. 打开数据湖构建控制台，并打开数据权限管理-用户菜单。

2. 打开用户列表信息页面。

> **说明**
>
> 此处管理的用户为添加到数据湖构建中的用户，目前仅支持添加RAM用户到数据湖构建。如需对RAM用户管理，请前往RAM用户控制台

用户



查看用户详情

1. 点击用户名链接，可以打开用户详情信息。

2. 在用户详情信息，可以查看用户基本信息，已有角色信息，已有权限信息。

将当前用户加入角色

可以将当前用户添加到一个DLF已有角色中，使其具备所选角色的权限。

查看当前用户拥有的权限

打开“权限管理”页签，可以查看当前用户已经拥有的个人的权限及角色权限。

添加权限

打开“权限管理”页签，点击添加权限，可以为当前用户添加数据权限。

4.5.6. 角色管理

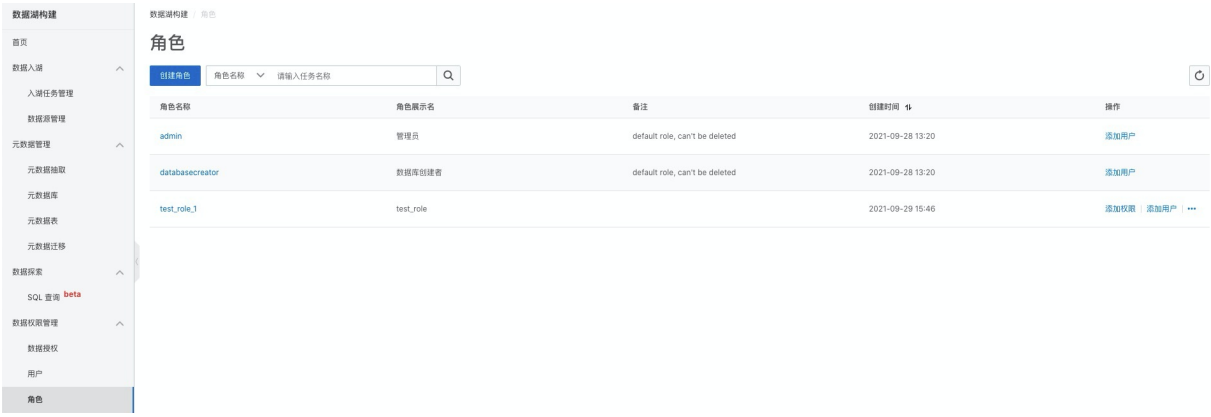
本文档主要为您说明如何管理数据湖构建中的角色。

查看角色信息

1. 打开数据湖构建控制台，并打开数据权限管理-角色菜单。
2. 打开角色列表信息页面。

系统内置角色：

- admin（数据湖管理员）：拥有数据湖构建中，所有的数据权限及授权权限。
- super_administrator（超级管理员）：拥有数据湖构建中，所有的数据权限及授权权限，可以对admin的用户进行修改。



创建角色

1. 打开角色列表信息页面，单击角色页面的“创建角色”按钮。
2. 输入角色相关信息。
3. 点击确定保存角色，完成角色创建。

4.6. 湖管理

4.6.1. 存储概览

存储概览主要包括存储，元数据对象的基本信息，还包括存储趋势、存储归档分布，表格式分布，小文件分布等信息，可以方便您快速了解当前存储资源使用情况及问题，并进行优化

前提条件

- 已经开通OSS产品
- 目前开通区域包括杭州，其他区域陆续开发中，如果您有需求可以提交工单反馈

存储概览开通

1. 打开数据湖构建控制台，点击左侧湖管理-存储概览菜单，点击立即启用，开启存储概览功能

> **注意**
>
> 1. 开通湖资产，元数据库的OSS地址将写入文件的统计信息到OSS中，将产生少量存储成本。
> 2. 首日开通无统计数据，需要等待第二天数据产出后可查看统计信息。

2. 点击“确定”按钮，确认同意授权

操作说明

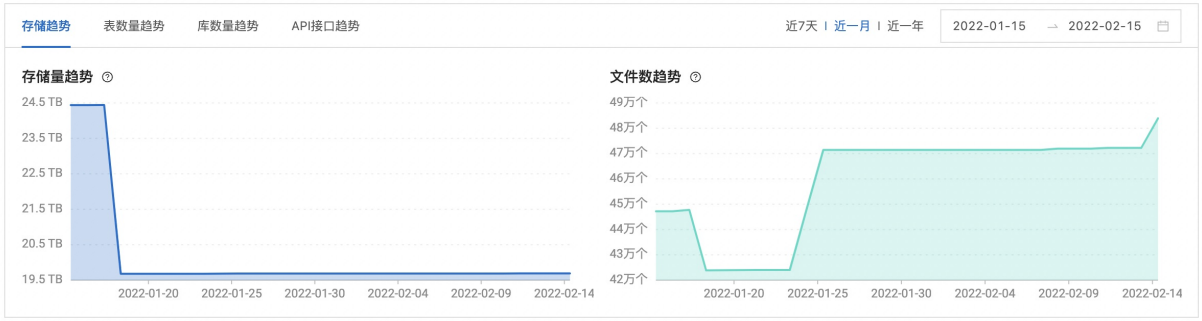
资源总计

- 总存储量及月/日变化：元数据管理下的表的总存储量（仅包括oss类型存储，不包括hdfs存储）。
- 总表数量及月/日变化：元数据管理中所有表的总数量
- 总库数量级月/日变化：元数据管理中所有库的总数量
- API月/日访问量：当前月（自然月）的API月访问量

趋势变化

存储量、表数量、库数量、API接口趋势 的趋势变化图。

您可以按照时间段，选择要查询的时间段。



表/库存储排名

统计表/库所占OSS存储量的大小排名，根据业务需要对排名靠前的表/库进行优化。

表存储排名

排名	表名称	表存储大小
1	tpcds_text_parquet_10000.store_sales	4.28 TB
2	tpcds_text_parquet_10000.catalog_sales	3.24 TB
3	tpcds_bin_partitioned_parquet_10000.store_sales	2.13 TB
4	tpcds_text_parquet_10000.web_sales	1.62 TB
5	tpcds_bin_partitioned_parquet_10000.catalog_sales	1.43 TB

共有 100 条 < 1 2 3 4 5 ··· 20 >

库存储排名

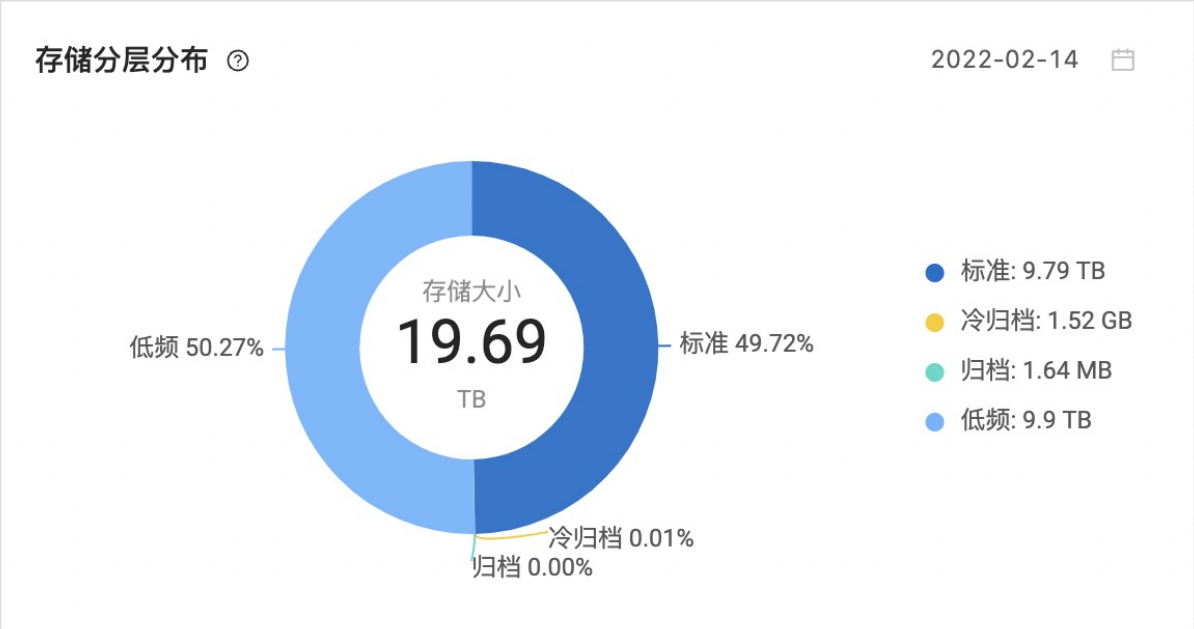
排名	库名称	库存储大小
1	tpcds_text_parquet_10000	9.9 TB
2	tpcds_bin_partitioned_parquet_10000	4.74 TB
3	xj_zorder	4.7 TB
4	tpcds_bin_partitioned_parquet_1000	258.3 GB
5	default	57.2 GB

共有 100 条 < 1 2 3 4 5 ··· 20 >

存储分层分布

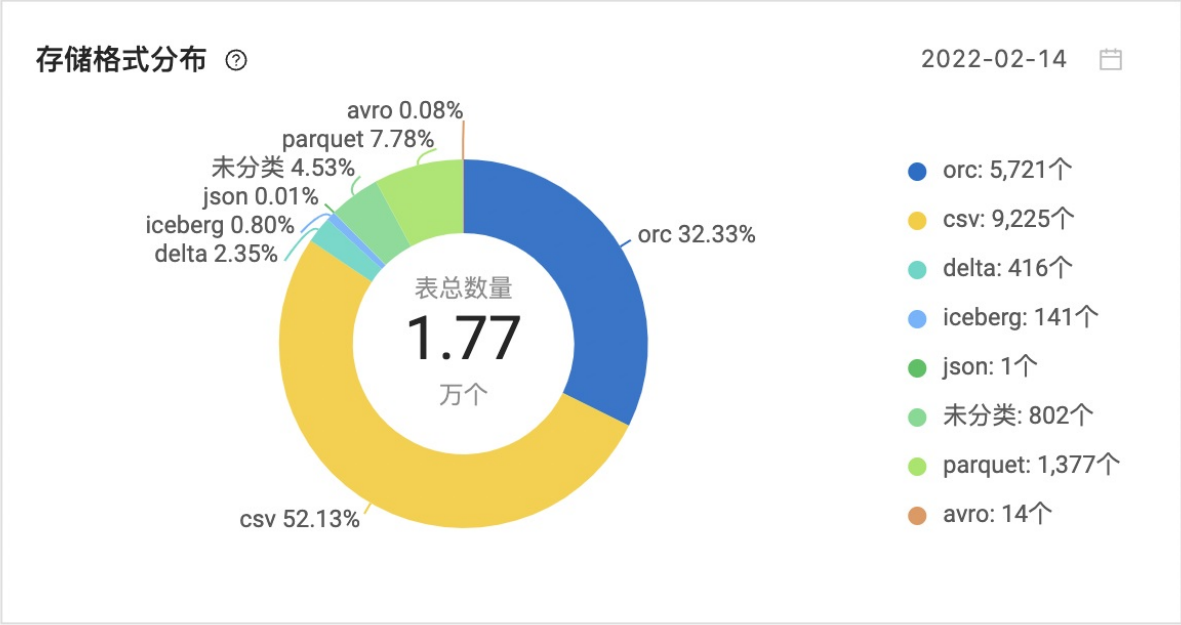
可以查看OSS上存储归档的分布情况，OSS存储包括标准存储、低频存储、归档存储、冷归档存储。您可以根据需要对不同业务数据选择合适的存储方式，优化存储成本。

数据湖构建DLF后续还将推出生命周期管理功能，帮助您对湖内数据进行自动归档。目前功能开发中。



存储格式分布

统计表的存储格式分布情况。



大小文件分布和排名

统计大小文件分布及排名情况，根据业务情况，针对小文件较多的表进行优化，提高查询性能。

4.6.2. 生命周期管理

4.6.2.1. 生命周期管理介绍

生命周期管理支持多种类型的生命周期管理规则，通过建立不同的规则，您可以方便的管理数据湖内的数据生命周期，以便节约存储成本。本文为您介绍生命周期管理规则的基础操作。

功能说明

您可以通过生命周期管理对数据湖中的数据库、数据表配置数据管理规则，可以基于分区/表创建时间、分区/表最近修改时间、分区值三种规则类型，对数据定期进行存储类型转换，从而节省数据存储成本。

适用场景说明

- 数据湖中存在大量数据库/表的历史数据，这些历史数据随着时间变化，不再被业务使用，期望转为成本更低的低频、归档，冷归档类型存储。例如：
  - 订单表（按pt分区，如pt=20220101），业务诉求为仅分析近3年数据，而历史分区数据期望转为冷归档，降低存储使用成本。此类场景，可以配置按分区值规则类型进行定期归档。
  - 业务A的数据库A，因为业务A不再发展，历史数据暂时封存，可以配置该数据库的定期转为冷归档，将整个数据库转为冷归档。

使用限制

1. 元数据管理使用数据湖构建（DLF），且数据存储在OSS中。
2. 暂时无法支持非结构化数据管理，如有这方面需求请参考OSS的生命周期管理。

费用说明

使用生命周期管理功能，涉及两部分费用，如下：

1. 数据湖构建（DLF）产品的生命周期管理功能，目前公测中，当前阶段免费。
2. OSS生命周期费用说明，请参考 OSS生命周期费用说明

注意事项

1. 如果数据被转为归档、冷归档的数据将无法被计算引擎所访问，您必须手工对其进行解冻恢复才可继续使用，且解冻有相关费用产生。详细介绍参考：
   i. OSS存储类型介绍
   ii. OSS类型转换

请您结合自己业务情况，充分评估后，再进行规则配置。

2. 如果数据被转为低频的数据，被计算引擎访问时性能将会下降。请您结合自己业务情况，充分评估后，再进行规则配置。
3. 生命规则任务，开启调度执行时，每天晚上定时执行，在第二天早上8点前生效。手动执行的任务，执行完成后即生效。

操作说明

前提条件

1. 您已经开通OSS产品，如未开通请前往OSS控制台。
2. 生命周期管理的库表权限，受到DLF的数据权限管控，所以用户仅能对其权限内的数据库/表进行生命周期规则配置。

创建生命周期规则

您可以参考如下步骤创建一条生命周期规则。

1. 登录 数据湖构建控制台，选择**湖管理>生命周期管理**。
2. 单击“**新建规则**”，**进行规则配置**。

a) 填写规则名称、描述、资源类型。

资源类型可以选型“库”，“表”两种类型，分别对应对元数据库、元数据表进行生命周期规则配置。

b) 选择规则类型，目前DLF支持以下三种规则类型：

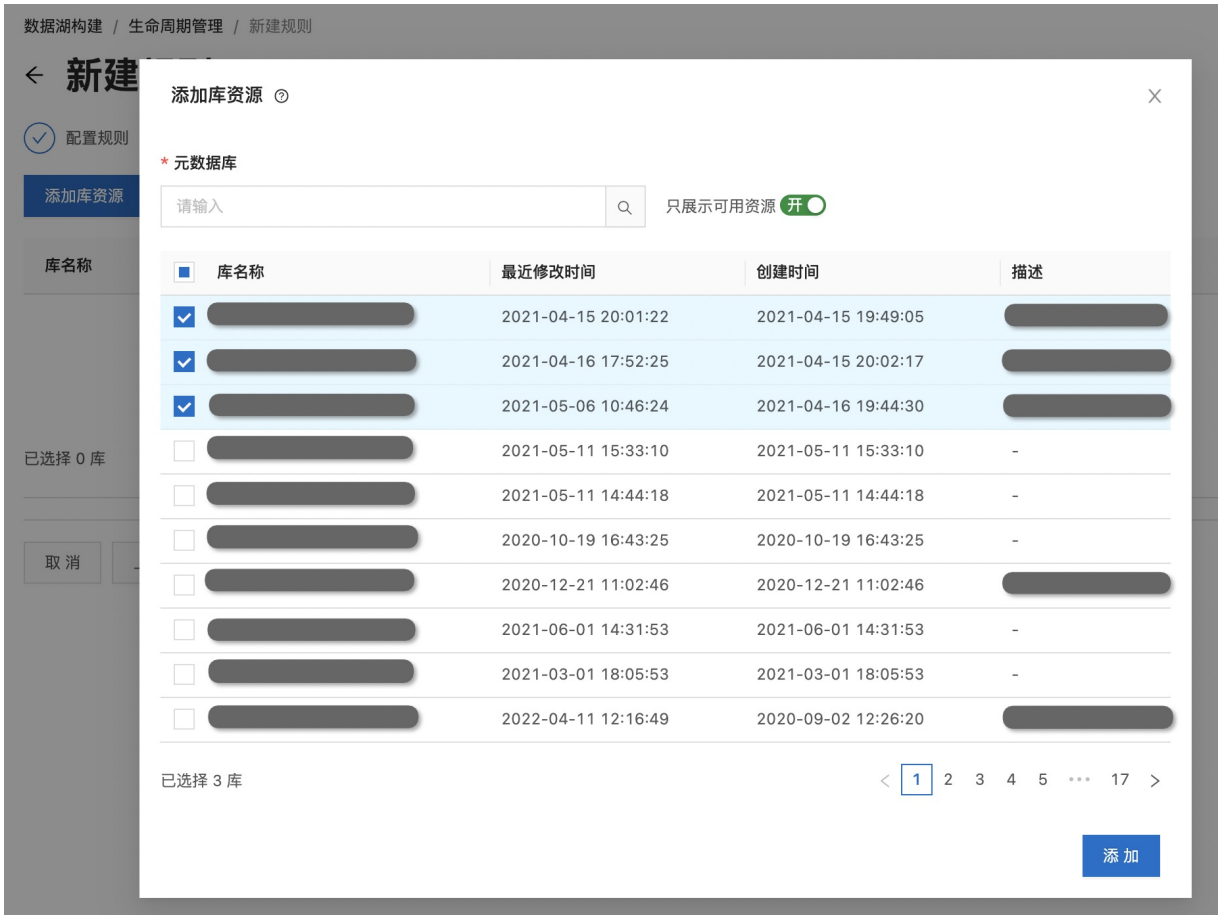
- 分区/表创建时间：可以实现按分区/表创建时间来界定生命周期。如果表有分区，则按最细粒度分区创建时间，否则按表创建时间。
- 分区/表最近修改时间：可以实现按分区/表最近修改时间来界定生命周期。如果表有分区，则按按最细粒度分区最近修改时间，否则按表最近修改时间。
- 分区值(按时间)：可以实现按分区值来界定生命周期。目前只适用于一级分区中包含时间格式的表。

c) 选择转换至低频访问存储/归档存储/冷归档存储的时间间隔。

d）配置规则执行机制。如果您希望DLF每天自动帮您执行当前规则，可打开调度开关。如果不需要每天自动执行，可建立规则后，在页面概览页手动执行。调度执行会在每天8点前执行完成。

3. 单击“下一步”，选择要归档的元数据库或元数据表。

a）点击添加资源按钮，选择需要绑定的资源，支持搜索和跨页选择。



b）添加资源后，点击确定，即可看到资源绑定结果。

如果绑定成功，可看到成功绑定资源数；

如果绑定失败，可看到失败原因。

说明

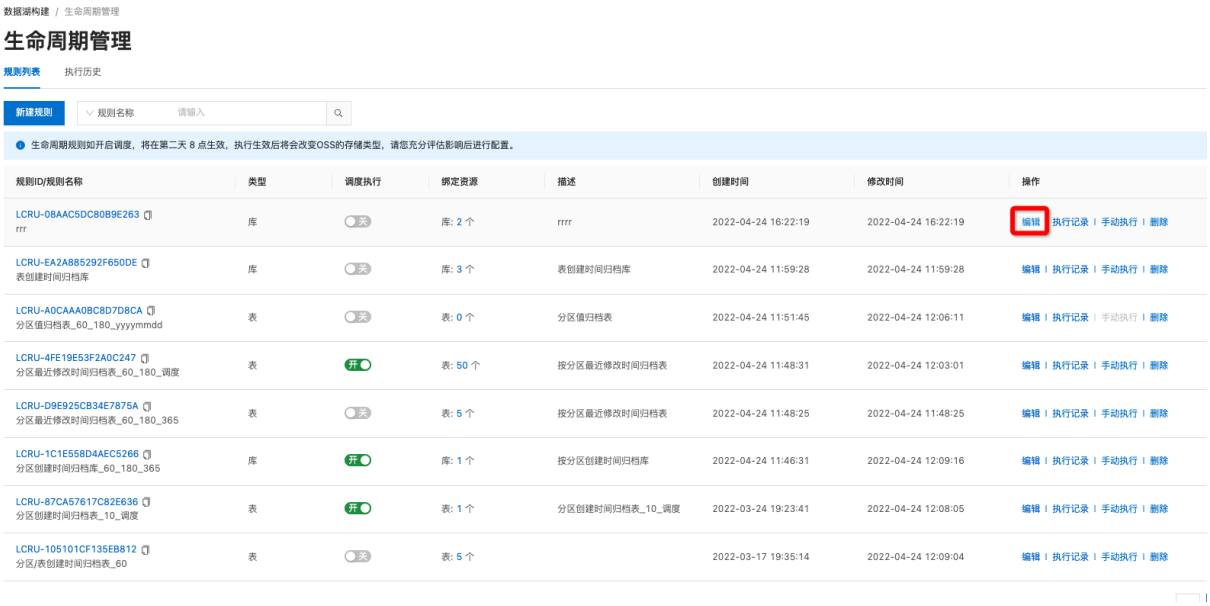
1. 资源类型为库时，可以绑定库资源；资源类型为表时，可以绑定表资源。
2. 表规则优先级大于库规则，如果某表已经绑定库规则，则该操作会覆盖表上的原有库规则。
3. 每个库/每个表仅支持同时绑定一个规则。
4. 每个规则最多绑定1000个资源。
5. 支持仅配置规则，后续再为规则绑定资源；直接点击保存即可。

编辑生命周期规则

如果您要对当前的生命周期规则进行修改或编辑，可以在列表页，单击右侧“编辑”按钮。

注意

1. 规则被修改后，如果调度执行是开启的，其在第二天执行时才会生效。
2. 规则被修改后，当其再次被执行，将会影响其绑定的所有资源，影响情况如下：
   i. 如该数据已经被历史规则判定转为低频/归档/冷归档，当继续保持低频/归档/冷归档状态。
   ii. 如该数据未被转为低频/归档/冷归档，将会按照新规则生效。

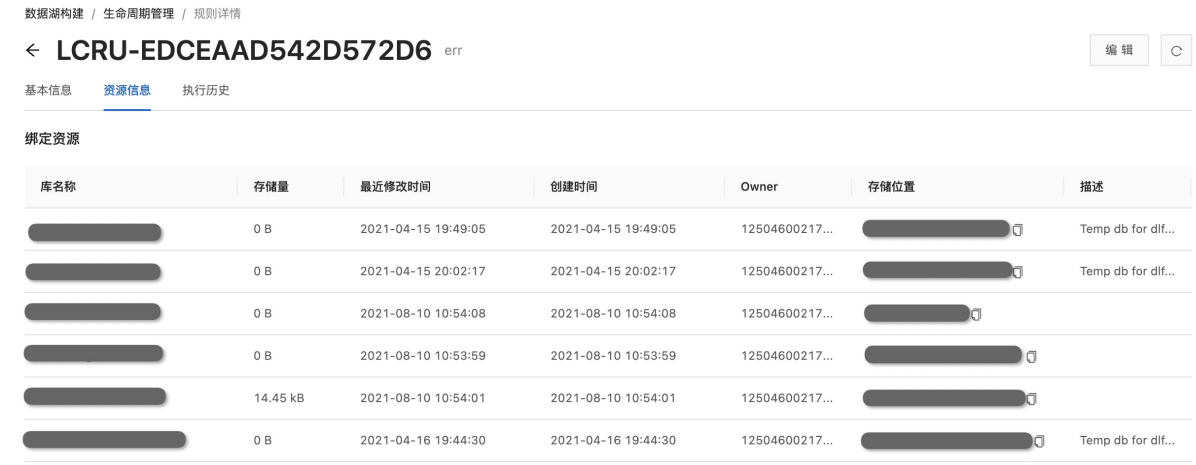


查看生命周期信息

1. 登录数据湖构建控制台，选择**湖管理>生命周期管理**。
2. 选择一条规则，点击规则ID进入，可以查看规则的当前信息。

- 基本信息：包括规则基础信息、规则详情、执行机制。

- 资源信息：规则所绑定的库或表信息。



- 执行历史：规则手动执行/调度执行的历史信息。

删除生命周期规则

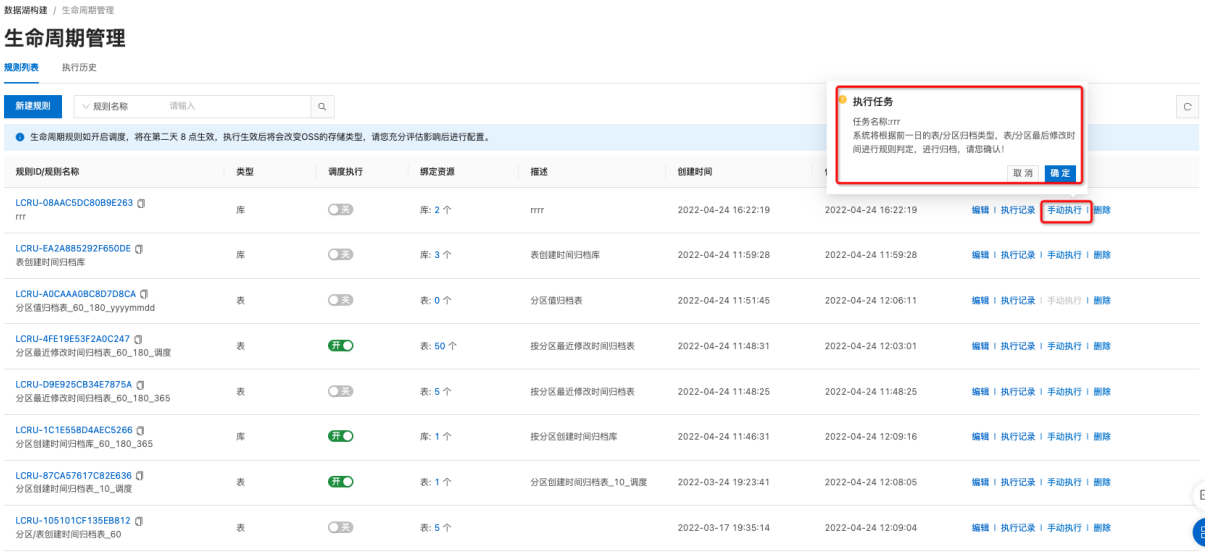
1. 登录数据湖构建控制台，选择**湖管理>生命周期管理**。
2. 找到想要删除的生命周期规则，点击右侧的“删除”按钮，在弹框中点击“确认”按钮。

> **说明**
>
> 1. 删除生命周期规则将无法再次被手工执行，以及被调度执行。
> 2. 删除生命周期规则后，之前被规则影响的数据将会保持当前现状。

手工执行任务

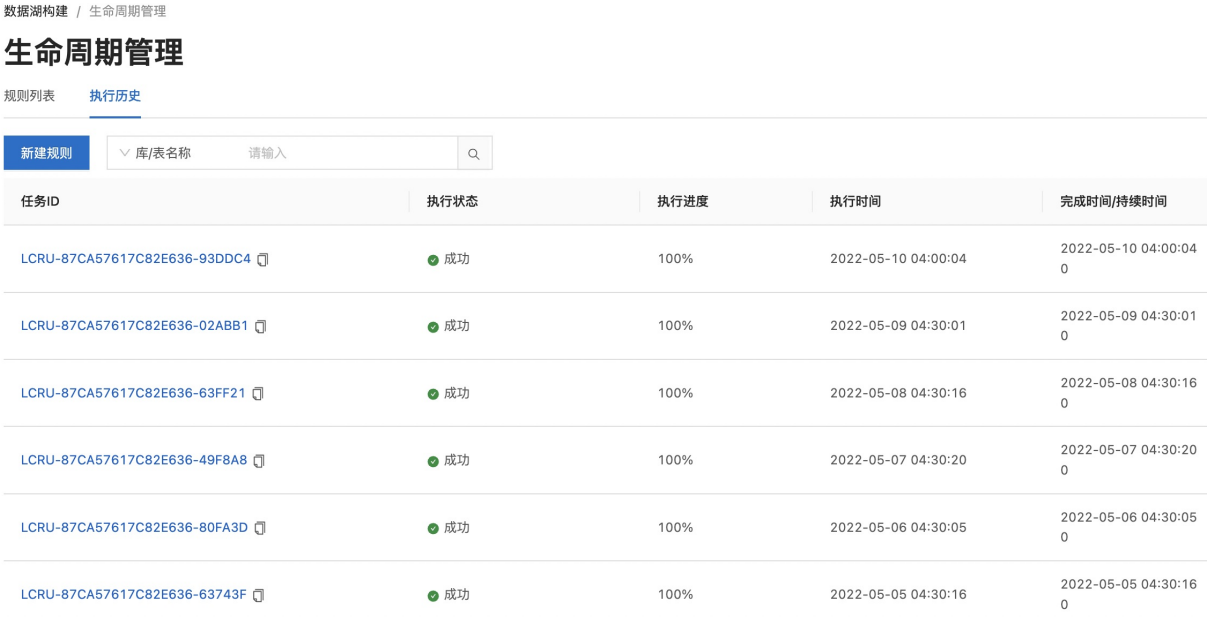
1. 登录数据湖构建控制台，选择**湖管理>生命周期管理**。
2. 找到想要手动执行的生命周期规则，点击右侧的“手动执行”按钮，仔细阅读弹出提示后，确认无误，单击“确定”按钮任务开始执行。

> **注意**
>
> 手动执行的操作将会立即执行，并对当前绑定资源的数据产生影响，可能影响业务访问，请您提前准确评估风险后再执行。

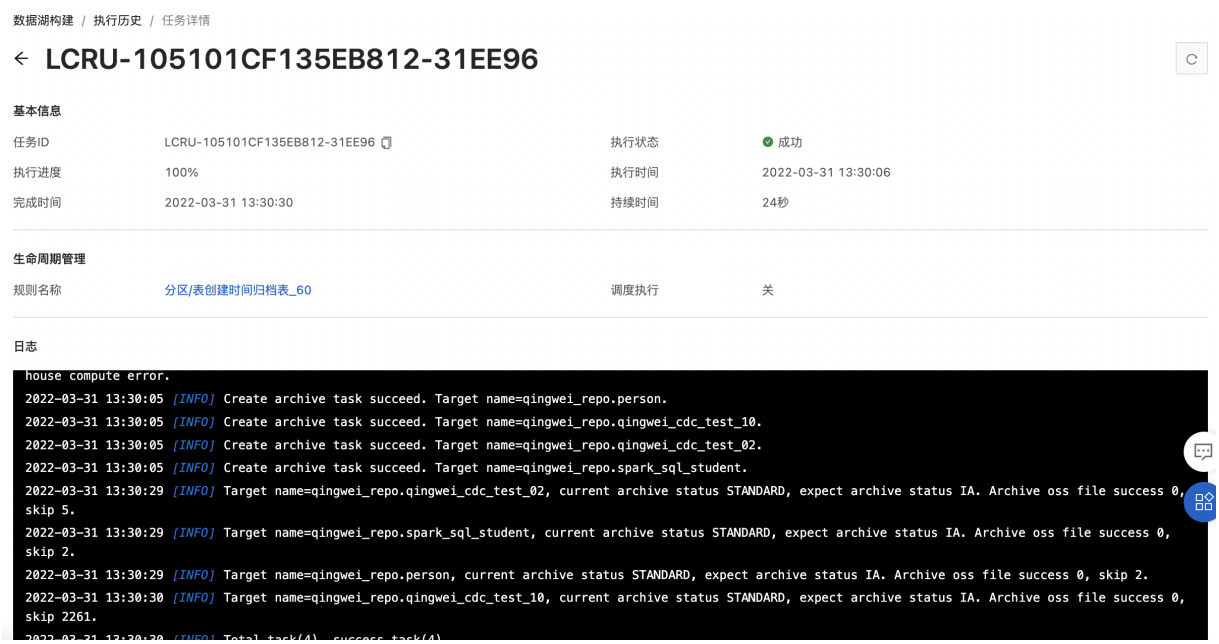


查看任务执行记录

1. 登录数据湖构建控制台，选择**湖管理>生命周期管理**。
2. 选择页签“执行历史”，可以对所有历史执行的归档任务进行查询，并查看执行日志。



3. 点击任务名称，可以查看任务执行信息及执行日志


数据湖构建 / 执行历史 / 任务详情
LCRU-105101CF135EB812-31EE96
基本信息
任务ID
LCRU-105101CF135EB812-31EE96
执行状态
成功
执行进度
100%
执行时间
2022-03-31 13:30:06
完成时间
2022-03-31 13:30:30
持续时间
24秒
生命周期管理
规则名称
分区/表创建时间归档表_60
调度执行
关
日志
house compute error.
2022-03-31 13:30:05 [INFO] Create archive task succeed. Target name=qingwei_repo.person.
2022-03-31 13:30:05 [INFO] Create archive task succeed. Target name=qingwei_repo.qingwei_cdc_test_10.
2022-03-31 13:30:05 [INFO] Create archive task succeed. Target name=qingwei_repo.qingwei_cdc_test_02.
2022-03-31 13:30:05 [INFO] Create archive task succeed. Target name=qingwei_repo.spark_sql_student.
2022-03-31 13:30:29 [INFO] Target name=qingwei_repo.qingwei_cdc_test_02, current archive status STANDARD, expect archive status IA. Archive oss file success 0, skip 5.
2022-03-31 13:30:29 [INFO] Target name=qingwei_repo.spark_sql_student, current archive status STANDARD, expect archive status IA. Archive oss file success 0, skip 2.
2022-03-31 13:30:29 [INFO] Target name=qingwei_repo.person, current archive status STANDARD, expect archive status IA. Archive oss file success 0, skip 2.
2022-03-31 13:30:30 [INFO] Target name=qingwei_repo.qingwei_cdc_test_10, current archive status STANDARD, expect archive status IA. Archive oss file success 0, skip 2261.

5.相关协议

5.1. 数据湖构建服务条款

提示条款

欢迎您与阿里云计算有限公司（以下简称“阿里云”）共同签署本《阿里云产品服务协议》（下称“本协议”）并使用阿里云服务！

协议中条款前所列索引关键词仅为帮助您理解该条款表达的主旨之用，不影响或限制本协议条款的含义或解释。为维护您自身权益，建议您仔细阅读各条款具体表述。

【审慎阅读】您在同意本协议之前，应当认真阅读本协议。请您务必审慎阅读、充分理解各条款的内容，特别是免除或者限制责任的条款、法律适用和争议解决条款，这些条款将以粗体下划线标识，您应重点阅读。**如您对协议有任何疑问，可以向客服和相关业务部门进行咨询。**

【签约动作】当您阅读并点击同意本协议或以其他方式选择接受本协议后，即表示您已充分阅读、理解并接受本协议的全部内容，并与阿里云达成一致。本协议自您通过网络页面点击确认或以其他方式选择接受本协议之日起成立。阅读本协议的过程中，如果您不同意本协议或其中任何条款约定，请勿进行签约动作。

通用服务条款

1. 签约主体及协议范围

本服务协议是**您与阿里云计算有限公司**就您使用**阿里云服务**所签署的服务协议。

2. 服务内容

本条款中“服务”指：阿里云www.aliyun.com 网站和客户端（以下单独或统称“阿里云网站”）所展示的、您申请订购且阿里云同意向您提供的服务。

3. 服务费用

3.1. 服务费用将在您订购页面予以列明公示，您可自行选择具体服务类型并按列明的价格予以支付。您可选择先付费或后付费的服务。

3.2. 先付费：

3.2.1. 在您付费之后，阿里云才开始为您提供服务。您未在下单后立即付费的，订单将为您保留7天，7天届满您仍未付费或者7天内阿里云服务售罄的，订单失效，订单失效后阿里云与您就服务所达成的合意失效。

3.2.2. 服务期满双方愿意继续合作的，您至少应在服务期满前7天内支付续费款项，以使服务得以继续进行。

3.3. 后付费：您可先使用后付费。具体扣费规则请查看阿里云网站上的页面展示且以页面公布的后付费服务当时有效的计费模式和标准。

3.4. 阿里云保留在您未按照约定支付全部费用之前不向您提供服务和/或技术支持，或者终止服务和/或技术支持的权利。同时，阿里云保留对您的欠费要求您按日承担万分之五的违约金以及追究其他法律责任的权利。

3.5. 您完全理解阿里云价格体系中所有的赠送服务项目或优惠活动均为阿里云在正常服务价格之外的一次性特别优惠，赠送的服务项目或优惠活动不可折价、冲抵服务价格。

4. 您的权利和义务

4.1. 成功订购服务后，您有权要求阿里云按照本服务协议以及阿里云网站相关页面所展示的服务说明、技术规范等内容向您提供服务。

4.2. 您订购阿里云的服务后，您可享受免费的售后服务。除此之外阿里云并提供其他付费的技术服务。

4.3. 您应按照阿里云的网站页面提示及本服务协议的约定支付相应服务费用。

4.4. 就阿里云服务的使用应符合阿里云的《服务使用规则》以及本服务协议。

4.5. 您对自己存放在阿里云云平台上的数据以及进入和管理阿里云云平台上各类产品与服务的口令、密码的完整性和保密性负责。因您维护不当或保密不当致使上述数据、口令、密码等丢失或泄漏所引起的损失和后果均由您承担。

4.6. 您须依照《网络安全法》、《互联网信息服务管理办法》等法律法规的规定保留自己网站的访问日志记录，包括发布的信息内容及其发布时间、互联网地址（IP）、域名等，国家有关机关依法查询时应配合提供。您将承担未按规定保留相关记录而引起的相应法律责任。

4.7. 为了数据的安全，您应负责您数据的备份工作。阿里云的产品或服务可能会为您配置数据备份的功能或工具，您负责操作以完成备份。

4.8. 您应对您的用户业务数据的来源及内容负责，阿里云提示您谨慎判断数据来源及内容的合法性。您将承担因您的用户业务数据内容违反法律法规、部门规章或国家政策而造成的相应结果及责任。

4.9. 您理解并同意，中华人民共和国的国家秘密受法律保护，您有保守中华人民共和国的国家秘密的义务；您使用阿里云服务应遵守相关保密法律法规的要求，并不得危害中华人民共和国国家秘密的安全。

4.10. 您还应仔细阅读并遵守阿里云在网站页面上展示的相应服务说明、技术规范、使用流程、操作文档等内容（以上简称“操作指引”），依照相关操作指引进行操作。您将承担违反相关操作指引所引起的后果；同时，阿里云郑重提示您，请把握风险谨慎操作。

5. 阿里云的权利、义务

5.1. 阿里云应按照约定提供服务。

5.2. 服务期限内，阿里云将为您提供如下售后服务：

5.2.1. 阿里云将提供7×24电话以及在线工单咨询服务，解答您在使用中的问题；

5.2.2. 阿里云将为您提供故障支持服务，您应通过在线工单申报故障；阿里云将及时就您非人为操作所出现的故障提供支持，但因您的人为原因和/或不可抗力、以及其他非阿里云控制范围内的事项除外。

5.3. 您还可通过阿里云获得其他付费的售后服务，具体详见阿里云的网站相关页面的收费售后服务内容。

5.4. 阿里云仅负责操作系统以下的底层部分及阿里云提供的软件的运营维护，即服务的相关技术架构及阿里云提供的操作系统等。操作系统之上部分（如您在系统上安装的应用程序）由您负责。此外，您自行升级操作系统可能会造成宕机等不良影响，请把握风险并谨慎操作。

5.5. 您了解阿里云无法保证其所提供的服务毫无瑕疵（如阿里云安全产品并不能保证您的硬件或软件的绝对安全），但阿里云承诺不断提升服务质量及服务水平。所以您同意：即使阿里云提供的服务存在瑕疵，但上述瑕疵是当时行业技术水平所无法避免的，其将不被视为阿里云违约。您同意和阿里云一同合作解决上述瑕疵问题。

5.6. 阿里云的某些服务可能具备账户授权管理功能，即您可将您对服务的全部或部分操作权限授权给您指定的一个或多个被授权账户，此种情况下，任一被授权账户下进行的所有操作行为，均将被视为您通过本人账户所进行的行为，都将由您承担相应的责任和由此产生的服务费用。

5.7. 您理解并认可，阿里云将为您提供基于某些服务的安全防护（如“云盾安骑士服务”）以及管理与监控的相关功能及服务（如“云监控”），尽管阿里云对该等服务经过详细的测试，但并不能保证其与所有的软硬件系统完全兼容，亦不能保证其软件及服务的完全准确性。如果出现不兼容及软件错误的情况，您应立即关闭或停止使用相关功能，并及时联系阿里云，获得技术支持。

5.8. 您应理解并同意，阿里云在对服务进行公测、邀测等您免费试（使）用服务期间或免费服务额度内，虽然阿里云会对服务可用性和可靠性提供支撑，但将不对任何服务可用性、可靠性做出承诺，阿里云亦不对您使用或不能使用阿里云服务的工作或结果承担任何责任。阿里云保留日后对该等免费服务收取费用的可能性，收取服务费用之前，阿里云将另行通知您。

6. 用户业务数据

6.1. 阿里云理解并认可，您通过阿里云提供的服务，加工、存储、上传、下载、分发以及通过其他方式处理的数据，均为您的用户业务数据，您完全拥有您的用户业务数据。

6.2．就用户业务数据，阿里云除执行您的服务要求外，不进行任何未获授权的使用及披露；但以下情形除外：

6.2.1． 在国家有关机关依法查询或调阅用户业务数据时，阿里云具有按照相关法律法规或政策文件要求提供配合，并向第三方或者行政、司法等机构披露的义务；

6.2.2． 您和阿里云另行协商一致。

6.3. 您可自行对您的用户业务数据进行删除、更改等操作。如您释放服务或删除数据的，阿里云将删除您的数据，按照您的指令不再保留该等数据。就数据的删除、更改等操作，您应谨慎操作。

6.4. 当服务期届满、服务提前终止（包括双方协商一致提前终止，其他原因导致的提前终止等）或您发生欠费时，除法律法规明确规定、主管部门要求或双方另有约定外，阿里云仅在一定的缓冲期（以您所订购的服务适用的专有条款、产品文档、服务说明等所载明的时限为准）内继续存储您的用户业务数据（如有），缓冲期届满阿里云将删除所有用户业务数据，包括所有缓存或者备份的副本，不再保留您的任何用户业务数据。

6.5. 用户业务数据一经删除，即不可恢复；您应承担数据因此被删除所引发的后果和责任，您理解并同意，阿里云没有继续保留、导出或者返还用户业务数据的义务。

6.6. 根据您与阿里云协商一致，阿里云在您选定的数据中心存储用户业务数据。阿里云恪守对用户的安全承诺，根据适用的法律保护用户存储在阿里云数据中心的数据。

7. 知识产权

7.1. 在本协议项下一方向对方提供的任何资料、技术或技术支持、软件、服务等的知识产权均属于提供一方或其合法权利人所有； 除提供方或合法权利人明示同意外，另一方无权复制、传播、转让、许可或提供他人使用上述知识成果，否则应承担相应的责任。

7.2. 您应保证提交阿里云的素材、对阿里云服务的使用及使用阿里云服务所产生的成果未侵犯任何第三方的合法权益。阿里云应保证向您提供的服务未侵犯任何第三方的合法权益。

7.3. 如果第三方机构或个人对您使用阿里云服务所涉及的相关素材的知识产权归属提出质疑或投诉，或对您使用的阿里云服务的知识产权的归属提出质疑或投诉，您和阿里云均有责任出具相关知识产权证明材料，并配合对方的相关投诉处理工作。对于因此引起的索赔、诉讼或可能向其提起诉讼,违约方应负责解决，承担费用和损失，以及使另一方免责。

8. 保密条款

8.1．本服务条款所称保密信息，是指一方（以下简称“接受方”）从对方（以下简称“披露方”）取得的、获知的、或因双方履行本协议而产生的商业秘密（包括财务秘密）、技术秘密、经营诀窍和（或）其他应予保密的信息和资料（包括产品资料，产品计划，价格，财务及营销规划，业务战略，客户信息，客户数据，研发，软件，硬件，API应用数据接口，技术说明，设计，特殊公式，特殊算法等），无论上述信息和资料以何种形式或载于何种载体，无论披露方在披露时是否以口头、图像或书面等方式表明其具有保密性。

8.2．双方应采取适当措施妥善保存对方提供的保密信息，措施的审慎程度不少于其保护自身的保密信息时的审慎程度。双方仅能将保密信息用于与本协议项下的有关用途或目的。

8.3．双方保证保密信息仅可在各自一方从事该业务的负责人和雇员范围内知悉，并严格限制接触上述保密信息的员工遵守本条之保密义务。

8.4．本条上述限制条款不适用于以下情况：

8.4.1．在签署本协议之时或之前，该保密信息已以合法方式属接受方所有；

8.4.2．保密信息在通知给接受方时，已经公开或能从公开领域获得；

8.4.3．保密信息是接受方从与其没有保密或不透露义务的第三方获得的；

8.4.4．在不违反本协议约定责任的前提下，该保密信息已经公开或能从公开领域获得；

8.4.5．该保密信息是接受方或其关联或附属公司独立开发，而且未从通知方或其关联或附属公司获得的信息中获益；

8.4.6．接受方应法院或其它法律、行政管理部门要求（通过口头提问、询问、要求资料或文件、传唤、民事或刑事调查或其他程序）因而透露保密信息；

8.4.7．接受方为向行政管理部门、行业协会等机构申请某项业务资质、获得某项认定、或符合国家、行业标准/认证，需结合对方情况向前述机构提交材料或进行说明的而披露的信息，在该等情况下，接受方应秉持必要情况下最少披露原则及要求因此获知保密信息的机构按不低于本协议的标准予以保密。

8.5．您和阿里云都应尽最大的努力保护上述保密信息不被泄露。一旦发现有上述保密信息泄露事件，双方应合作采取一切合理措施避免或者减轻损害后果的产生。如因此给对方造成损失的，应赔偿因此给对方造成的直接经济损失。

9. 服务的开通、终止与变更

9.1. 先付费的服务：

9.1.1. 您付费后服务即开通，开通后您获得阿里云向您发送的登录、使用服务的密钥、口令即可使用服务，服务期限自开通之时起算（而非自您获得登录、使用服务的密钥、口令时起算）；

9.1.2. 以包年包月等固定期限形式售卖的服务，服务期限至订购的期限届满为止；以资源包（或套餐包）形式售卖的服务，服务期限则至您订购的资源包服务期限到期或资源包中的服务被使用完毕为止（以前述二者早发生为准）；

9.1.3. 您应在服务期限内将资源包的服务数量使用完毕，如资源包的服务期限届满，您已订购但未使用完毕的服务将被作废且阿里云将不提供其他替代或补充。

9.1.4. 您对于服务的使用将优先消耗订购的资源包，除法定及双方另行约定外，如资源包中的各项服务使用完毕或者服务期限到期，且您未继续订购资源包服务但持续使用此项服务的，阿里云将视为您使用阿里云以后付费形式售卖的该服务（如有），阿里云将持续计费并根据计费结果予以扣划服务费用。

9.2. 后付费的服务：

除非另有其他约定或您未结清其他应付款项的，您开通服务即可使用阿里云的服务；您应确保您的账户余额充足，以便持续使用服务至法律规定或本服务条款约定的终止情形出现时为止。

9.3. 发生下列情形之一的，服务期限提前终止：

9.3.1. 双方协商一致提前终止的；

9.3.2. 您严重违反本协议（包括，您严重违反相关法律法规规定，或您严重违反本协议项下之任一承诺内容等），阿里云有权提前终止服务**直至清除您的全部数据**；

9.3.3. 您理解并充分认可，虽然阿里云已经建立（并将根据技术的发展不断完善）必要的技术措施来防御包括计算机病毒、网络入侵和攻击破坏（包括DDoS）等危害网络安全事项或行为（以下统称该等行为），**但鉴于网络安全技术的局限性、相对性以及该等行为的不可预见性，因此如因您网站遭遇该等行为而给阿里云或者阿里云的其他网络或服务器（包括本地及外地和国际的网络、服务器等）带来危害，或影响阿里云与国际互联网或者阿里云与特定网络、服务器及阿里云内部的通畅联系，阿里云可决定暂停或终止服务。如果终止服务的，将按照实际提供服务月份计算（不足一个月的按天计）服务费用，将剩余款项（如有）返还。**

9.3.4. 阿里云可提前30天在阿里云网站上通告或给您发网站内通知或书面通知的方式终止本服务服务协议；届时阿里云应将您已预付但未消费的款项退还至您的阿里云账户。

9.4．您理解并认可，为技术升级、服务体系升级、或因经营策略调整或配合国家重大技术、法规政策等变化，阿里云不保证永久的提供某种服务，并有权变更所提供服务的形式、规格或其他方面（如服务的价格和计费模式），在终止该种服务或进行此种变更前，阿里云将尽最大努力且提前以网站公告、站内信、邮件或短信等一种或多种方式进行事先通知。

10. 违约责任

10.1．您违反本协议中的承诺、保证条款、服务使用规则或义务的任一内容，或阿里云根据其判断认为您的使用行为存在异常的，阿里云均有权就其情节，根据独立判断并单方采取以下措施中的一种或多种：（1）限制、中止使用服务；（2）终止提供服务，终止本协议；（3）追究您的法律责任；（4）其他阿里云认为适合的处理措施。阿里云依据前述约定采取中止服务、终止服务等措施而造成的用户损失将由您承担。

10.2．如因您违反有关法律法规或者本协议、相关规则之规定，使阿里云遭受任何损失、受到其他用户、任何第三方的索赔或任何行政管理部门的处罚，您应对阿里云、其他用户或相关第三方的实际损失进行全额赔偿，包括合理的律师费用。

10.3．您理解且同意，鉴于计算机、互联网的特殊性，下述情况不属于阿里云违约：

10.3.1. 阿里云在进行系统及服务器配置、维护、升级时，需要短时间中断服务；

10.3.2. 由于Internet上的通路阻塞造成您网站访问速度下降。

10.4．如果因阿里云原因造成您连续72小时不能正常使用服务的，您可终止接受服务，但非阿里云控制之内的原因引起的除外。

10.5．在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性的损害，包括您使用阿里云服务而遭受的利润损失承担责任（即使您已被告知该等损失的可能性）。

10.6. 在法律允许的情况下，阿里云在本协议项下所承担的损失赔偿责任不超过就该服务过往12个月所缴纳的服务费用的总和。

11.通知

11.1. 您在使用阿里云服务时，您应该向阿里云提供真实有效的联系方式（包括您的电子邮件地址、联系电话、联系地址等），对于联系方式发生变更的，您有义务及时更新有关信息，并保持可被联系的状态。您接收站内信、系统消息的会员账号（包括子账号），也作为您的有效联系方式。

11.2. 阿里云将向您的上述联系方式的其中之一或其中若干向您送达各类通知，而此类通知的内容可能对您的权利义务产生重大的有利或不利影响，请您务必及时关注。

11.3. 阿里云通过上述联系方式向您发出通知，其中以电子的方式发出的书面通知，包括公告，向您提供的联系电话发送手机短信，向您提供的电子邮件地址发送电子邮件，向您的账号发送系统消息以及站内信信息，在发送成功后即视为送达；以纸质载体发出的书面通知，按照提供联系地址交邮后的第五个自然日即视为送达。

11.4. 你应当保证所提供的联系方式是准确、有效的，并进行实时更新。如果因提供的联系方式不确切，或不及时告知变更后的联系方式，使法律文书无法送达或未及时送达，将由您承担由此可能产生的法律后果。

12. 不可抗力

12.1. 因不可抗力或者其他意外事件，使得本服务条款的履行不可能、不必要或者无意义的，遭受不可抗力、意外事件的一方不承担责任。

12.2. 不可抗力、意外事件是指不能预见、不能克服并不能避免且对一方或双方当事人造成重大影响的客观事件，包括自然灾害如洪水、地震、瘟疫流行等以及社会事件如战争、动乱、政府行为、电信主干线路中断、黑客、网路堵塞、电信部门技术调整和政府管制等。

13. 法律适用及争议解决

13.1. 本协议之订立、生效、解释、修订、补充、终止、执行与争议解决均适用中华人民共和国大陆法律。

13.2. 您因使用阿里云服务所产生及与阿里云服务有关的争议，由阿里云与您协商解决。协商不成时，任何一方均可向杭州市西湖区人民法院提起诉讼。

14. 附则

14.1. 本协议的附件，以及阿里云在阿里云网站相关页面上的服务说明、价格说明和您确认同意的订购页面（包括产品的专用条款、服务说明、操作文档等）均为本协议不可分割的一部分。如遇不一致之处，以（1）服务说明、价格说明、其他订购页面，（2）专用条款和附件，（3）本协议通用条款的顺序予以适用。

14.2. 如本协议内容发生变动，阿里云应通过提前30天在阿里云网站的适当版面公告向您提示修改内容；如您继续使用阿里云服务，则视为您接受阿里云所做的相关修改。

14.3. 阿里云有权经提前通知将本协议的权利义务全部或者部分转移给阿里云的关联公司。

14.4. 阿里云于您过失或违约时放弃本协议规定的权利，不应视为其对您的其他或以后同类之过失或违约行为弃权。

14.5. 本协议任一条款被视为废止、无效或不可执行，该条应视为可分的且并不影响本协议其余条款的有效性及可执行性。

14.6. 本协议项下之保证条款、保密条款、知识产权条款、法律适用及争议解决条款等内容，不因本协议的终止而失效。

专用服务条款

1.1.数据湖构建采用按量付费形式计费，阿里云将每小时自您的阿里云账户中扣划前一小时的服务费用，您应确保账户余额充足，以保证服务的持续使用。

1.2.当您的账户余额不足以支付账单金额，您将处于欠费状态，则阿里云将按本条约定停止您的部分或全部服务，同时未停止的服务将持续计费：

1.2.1.自您欠费之日起7日内，您仍可使用数据湖构建的元数据服务；如您自欠费之日起的7日届满，仍未能补缴所有欠费账单的，阿里云将停止为您提供数据湖元数据服务。

1.2.2.自您欠费之日起，您将无法执行新的入湖作业，已经运行的入湖作业仍可继续执行7日；如您自欠费之日起的7日届满，仍未能补缴所有欠费账单的，阿里云将停止您执行所有入湖作业。

1.2.3.自您欠费之日起，您将无法创建新的入湖模板；如您自欠费之日起的30日届满仍未能补缴所有欠费账单的，阿里云将清空您的全部入湖模板。

5.2. 服务等级协议

自2021年1月起，数据湖构建（DLF）服务等级协议（SLA）生效。详细内容参考数据湖构建服务等级协议。

6.最佳实践

6.1. EMR+DLF数据湖解决方案

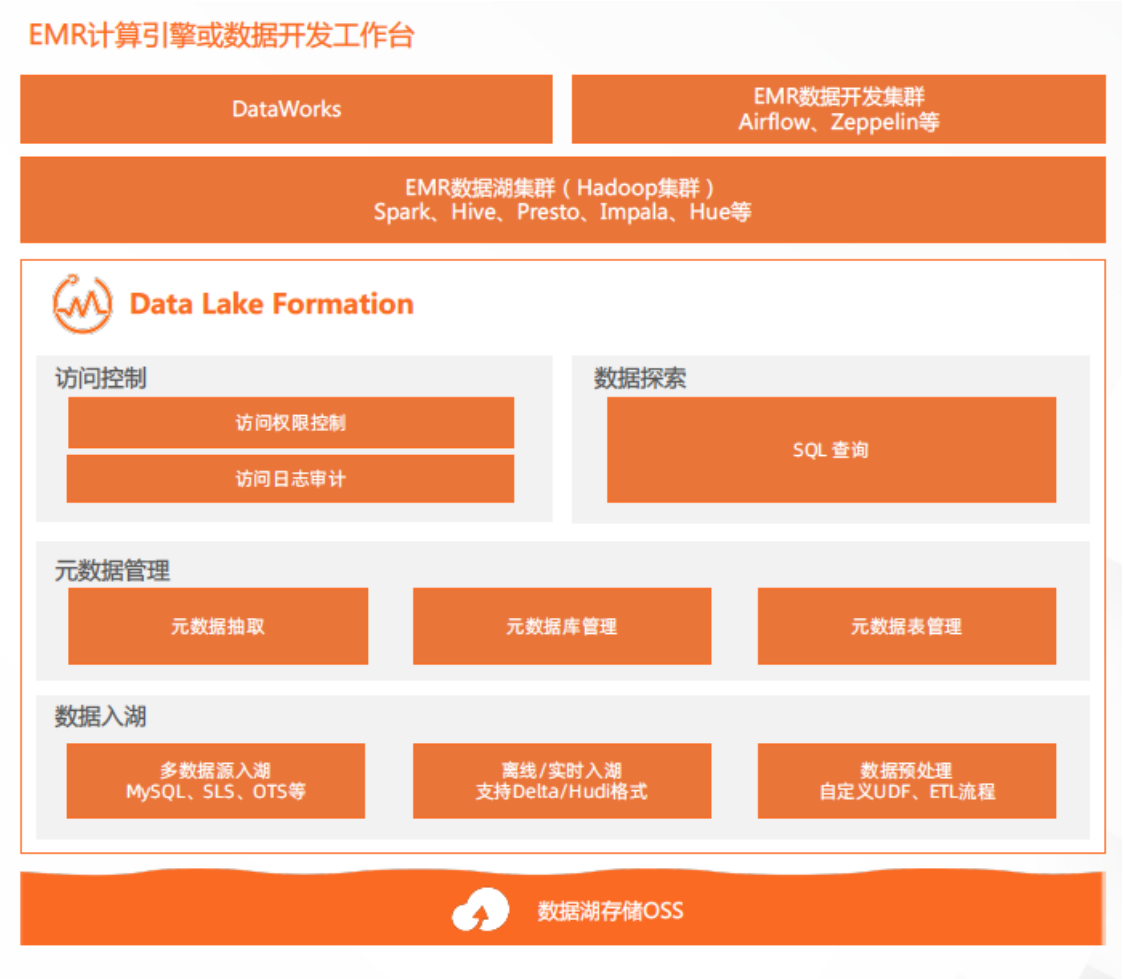
在EMR+DLF数据湖方案中，可以为企业提供数据湖内的统一的元数据管理，统一的权限管理，支持多源数据入湖以及一站式数据探索的能力。本文为您介绍EMR+DLF数据湖方案具体步骤。

背景信息

在EMR数据湖方案中，结合DLF，可以为企业提供数据湖内的统一的元数据管理，统一的权限管理，支持多源数据入湖以及一站式数据探索的能力。采用EMR+DLF数据湖解决方案，相对传统EMR数据湖方案有下列优点：

- DLF提供了统一的、服务化的元数据和权限管理工具，使元数据和权限变得更为透明，减少了元数据和权限不一致性，降低了管理成本
- DLF提供了多套常见的数据入湖方案，包括RDS全量、RDS实时、SLS实时、OTS实时、Kafka实时数据入湖模板。用户可以基于DLF数据入湖能力，高效搭建自己的数据入湖解决方案。

EMR+DLF数据湖解决方案架构



限制条件

目前DLF服务仅在以下区域开通，仅华东1（杭州）、华东2（上海）、华北2（北京）、华南1（深圳）。

操作流程

具体操作

- 步骤一：创建采用DLF为元数据服务的EMR集群
- 步骤二：在DLF中创建元数据库和元数据表
- 步骤三：通过DLF入湖功能创建RDS到数据湖的入湖流程
- 步骤四：通过EMR的Spark、Presto引擎查询DLF表

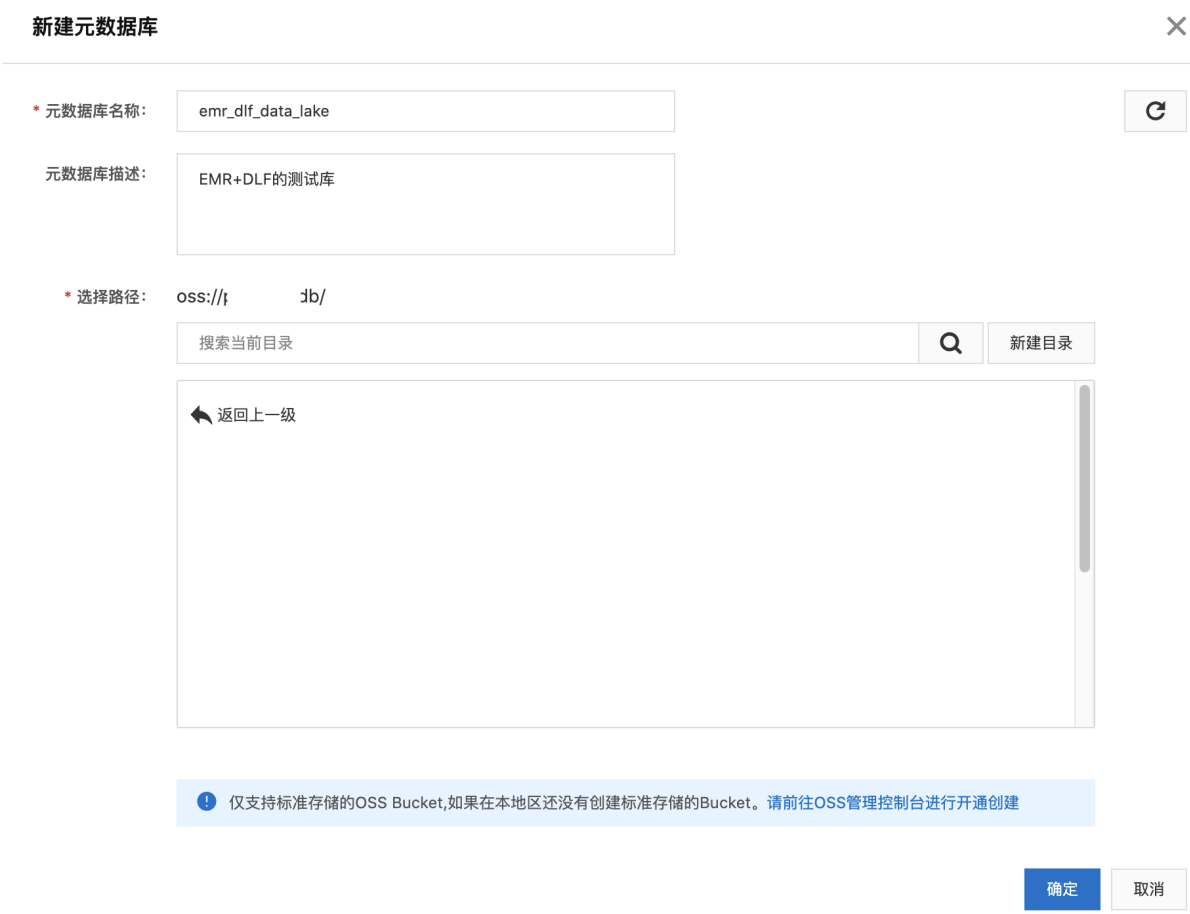
步骤一：创建采用DLF为元数据服务的EMR集群

在创建EMR集群的流程中，注意在配置“基础配置”步骤时，选择“数据湖元数据”。如果，您没有开通DLF，会提醒您先开通DLF产品。

步骤二：在DLF中创建元数据库

基于准备好的OSS信息，在DLF中创建元数据库和元数据表，用于存放OSS数据的元数据信息。操作步骤如下：

1. 登录元数据库管理控制台，在左上角选择与OSS相同的地域，例如华东1（杭州）。
2. 创建元数据库。例如emr_dlf_data_lake。
   i. 在左侧导航栏，单击元数据库。
   ii. 在元数据库页面，单击新建元数据库。
   iii. 在新建元数据库对话框，配置元数据库参数。
   iv. 单击**确定**，完成元数据库创建。



步骤三：通过DLF入湖功能创建RDS到数据湖的入湖流程

可以利用DLF的入湖功能，创建从RDS数据库中到数据湖的数据同步任务作业。具体操作如下：

1. 登入入湖任务管理工作台，在数据源管理中添加您需要同步数据的RDS数据源信息，主要包括您的RDS实例信息，连接身份信息，网络信息等；
2. 在入湖任务管理界面中单击**新建入湖任务**按钮；
3. 在入湖任务创建向导中，选择关系数据库全量入湖，进入**配置数据源和目标信息步骤**；
4. 在**配置数据源和目标步骤**时，选择刚才创建的数据源连接信息，和步骤二中创建的目标元库信息；
5. 完成任务创建，并在**入湖任务管理**界面运行任务，待任务执行完成。

步骤四：通过EMR的Spark、Presto引擎查询DLF表

通过ssh登录到EMR集群的emr-header-1节点。

1. 通过spark-sql查询表departments：

```
a:409) finished in 0.608 s
21/06/22 17:30:31 INFO [main] DAGScheduler: Job 6 finished: processCmd at CliDriver.java:409, took 0.610911
 s
21/06/22 17:30:31 INFO [main] SparkSQLQueryListener: execution is called
21/06/22 17:30:31 INFO [main] SparkSQLQueryListener: Spark user root executed on 1624354231553 with spark s
ql successfully.
21/06/22 17:30:31 INFO [main] SparkSQLQueryListenerHelper: Unparitioned table:emr_dlf_data_lake.departments
;cols:dept_no,dept_name;path:oss://preview-db/emr_dlf_data_lake/departments.
21/06/22 17:30:31 INFO [main] NativeClient: JindoTable put 0 records.
21/06/22 17:30:31 INFO [Thread-36] JbootLogger: active_standby_channel.cpp:324] Selected Channel[192.168.12
0.154:8101], socket 1
d009    Customer Service
d005    Development
d002    Finance
d003    Human Resources
d001    Marketing
d004    Production
d006    Quality Management
d008    Research
d007    Sales
Time taken: 11.924 seconds, Fetched 9 row(s)
21/06/22 17:30:31 INFO [main] SparkSQLCLIDriver: Time taken: 11.924 seconds, Fetched 9 row(s)
spark-sql> 21/06/22 17:30:40 INFO [Thread-37] JbootLogger: ClientMetricsService.cpp:123] Memory buffer usag
e for IO stream 0%, used 0B, capacity 6442450944B, watermark 0.3
```

2. 通过presto查询表departments：

```
 hive-acc
 jmx
 kudu
 system
(8 rows)

Query 20210622_093213_00000_i5rdb, FINISHED, 2 nodes
Splits: 36 total, 36 done (100.00%)
1.45 [0 rows, 0B] [0 rows/s, 0B/s]

presto> use hive.emr_dlf_data_lake;
USE
presto:emr_dlf_data_lake> show tables;
    Table
-------------
 departments
(1 row)

Query 20210622_093228_00003_i5rdb, FINISHED, 3 nodes
Splits: 36 total, 36 done (100.00%)
0.93 [1 rows, 38B] [1 rows/s, 41B/s]

presto:emr_dlf_data_lake> select * from departments;

Query 20210622_093235_00005_i5rdb, FAILED, 1 node
```

6.2. 数据湖构建之MaxCompute湖仓一体最佳实践

MaxCompute + DLF湖仓一体方案打破数据湖与数据仓库割裂的体系，架构上将数据湖的灵活性、生态丰富与数据仓库的企业级能力进行融合，构建数据湖和数据仓库融合的数据管理平台。本文介绍湖仓一体的具体方案。

背景信息

大数据计算服务MaxCompute（原名ODPS）是一种快速、完全托管的EB级数据仓库解决方案。与数据湖相比数据仓库具备易优化、易治理等优点，但同时面临数据种类单一，灵活性低，仅向特定引擎开放等不足，提高了数据存储和加工的成本，不利于数据的共享。依托数据湖构建产品（DLF）提供的企业级元数据能力，MaxCompute数仓可以通过共享元数据的方式访问湖上的存储，与湖上引擎打通，实现MaxCompute湖仓一体。

方案详情

详细内容请参考：《数据湖构建之MaxCompute湖仓一体最佳实践》

6.3. DLF数据探索快速入门-淘宝用户行为分析

DLF产品（数据湖构建）提供数据发现和数据探索的功能，本文介绍如何通过DLF完成对淘宝用户行为样例的分析。

操作流程

1. 服务开通：开通阿里云账号及DLF和OSS相关服务。
2. 样例数据集下载和导入：下载样例数据（csv文件），并上传至OSS。
3. DLF数据发现：使用DLF自动识别文件Schema并创建元数据表。
4. DLF数据探索：使用DLF数据探索，对用户行为进行分析，包括用户活跃度、漏斗模型等。

数据说明

本次测试的数据集来自阿里云天池比赛中使用的淘宝用户行为数据集，为了提高性能，我们做了一定的裁剪。数据集中以csv的格式存储了用户行为及商品样例数据。

淘宝用户行为数据集介绍：https://tianchi.aliyun.com/dataset/dataDetail?dataId=46

数据范围：2014年12月1日 - 2014年12月7日

数据格式：

user表：

Column	Description	Comment
user_id	Identity of users	Sampled&desensitized
item_id	Identity of items	Desensitized
behavior_type	The user behavior type	Including click, collect,add-to-cart and payment, the corresponding values are 1, 2, 3 and 4,respectively.

user_geohash	Latitude(user location when the behavior occurs, whichmay be null)	Subject to fuzzing
item_category	The category id of the item	Desensitized
time	The time of the behavior	To the nearest hours

item表：

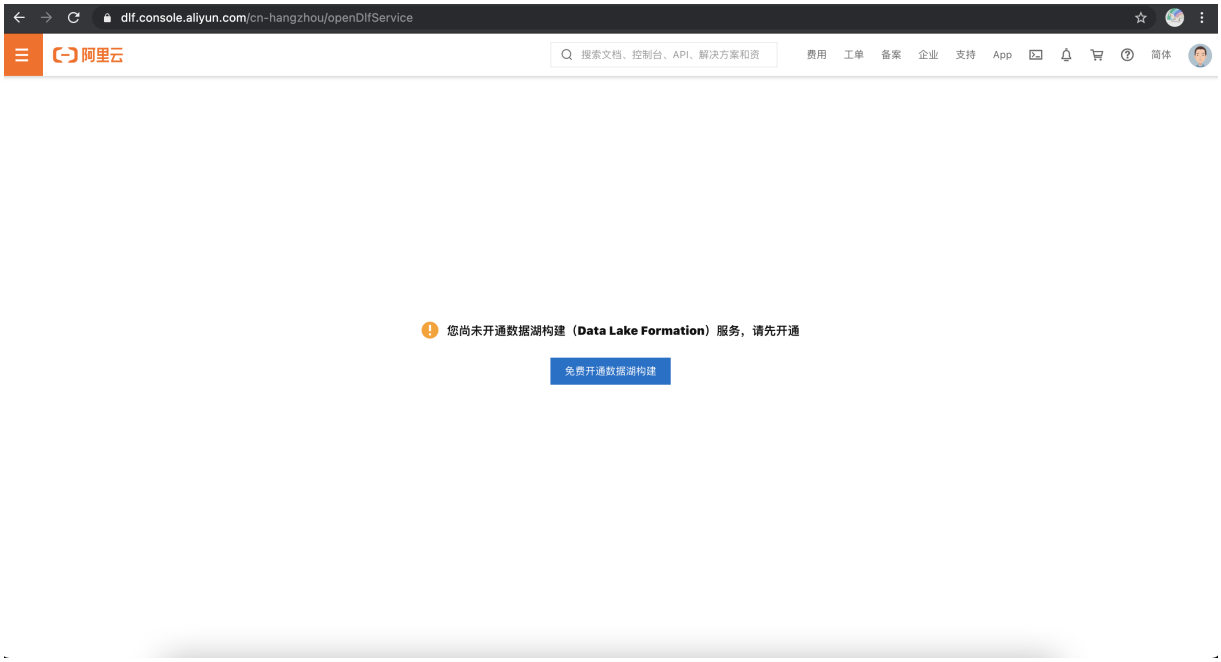
Column	Description	Comment
item_id	Identity of items	Sampled & desensitized
item_ geohash	user location where the behavior occurs(may be null)	generated by longitude and altitude through acertain privacy-preserving algorithm
item_category	The category id of the item	Desensitized

详细流程

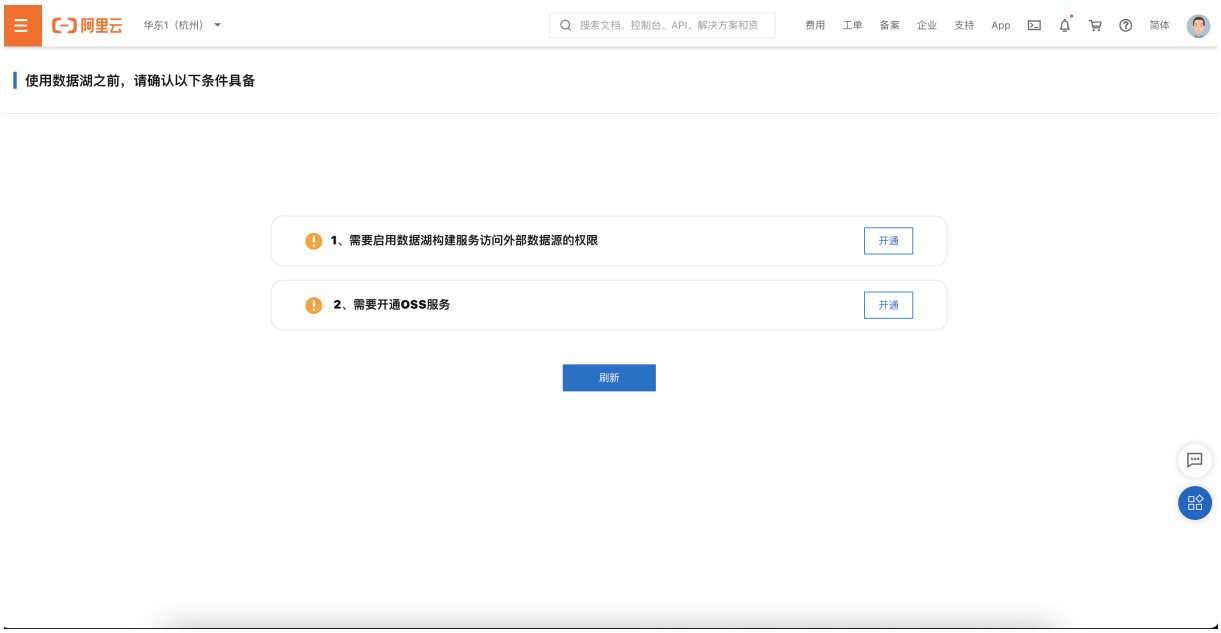
第一步：开通DLF和OSS服务

1.1 登录到DLF控制台页面。
1.2 开通DLF及其依赖OSS服务，并完成授权。（如果已开通可直接跳过）。

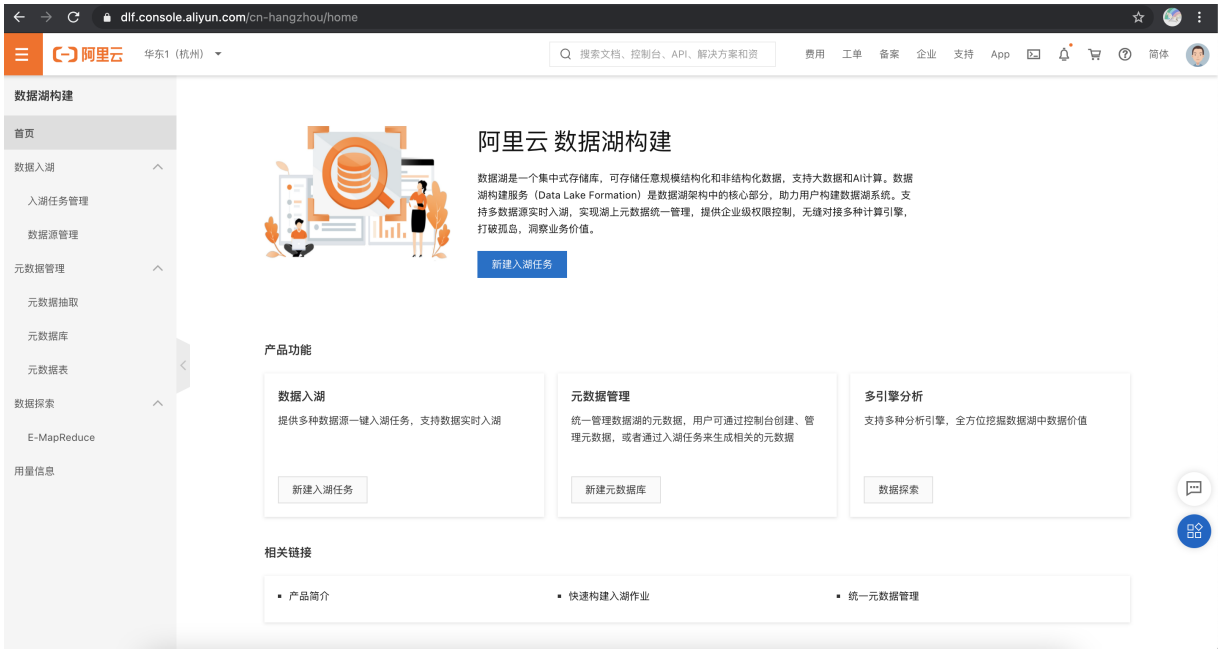
若之前未开通过DLF服务，会提示用户开通服务。单击**免费开通数据湖构建**。



开通服务后，返回DLF控制台页面。会提示开通OSS服务，以及授予DLF访问依赖数据源的权限。点击按钮完成OSS开通及授权。回到DLF控制台页面，点击刷新检查状态。页面自动跳转至DLF控制台主页面。



开通完成后，进入DLF控制台主页：



第二步：在OSS中导入需要分析的数据

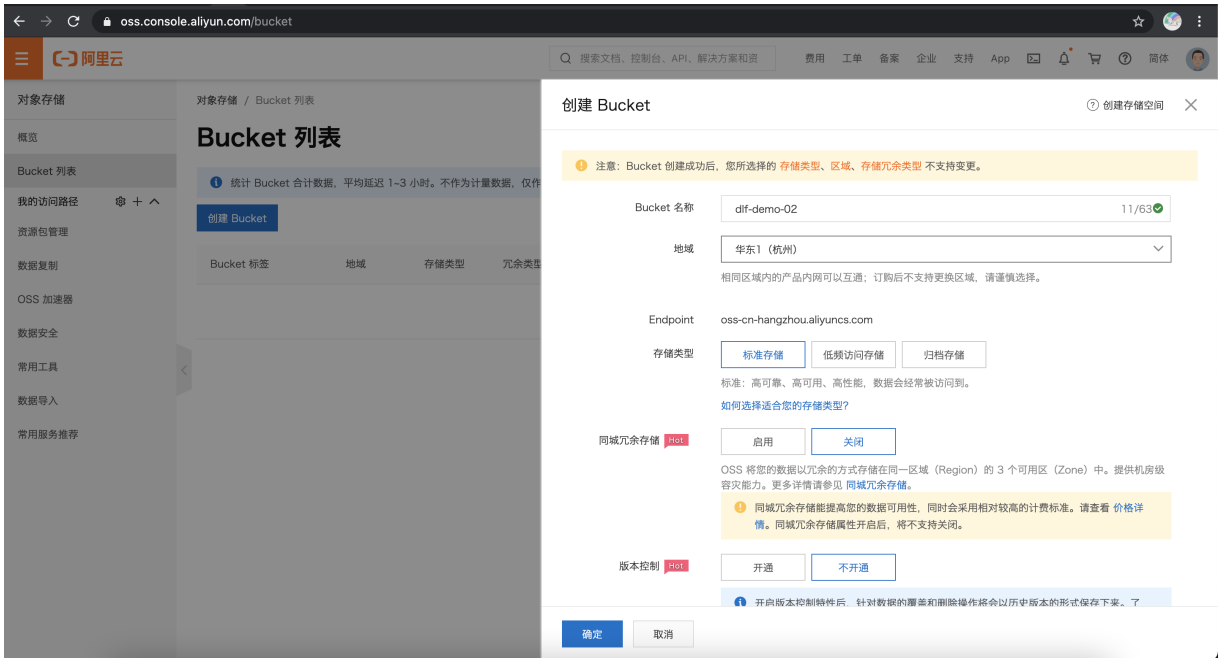
2.1 下载样例代码，放在本地磁盘。

解压后得到文件夹：user_behavior_data，包含item和user个文件夹，里面分别包含了各自的csv数据文件。本次分析主要集中在user文件中，数据内容如下。

```
user_id,item_id,behavior_type,user_geohash,item_category,time
113884016,70627757,1,,5271,2014-12-04 20
120690949,367355690,1,,13381,2014-12-07 11
106105154,291140936,1,,1863,2014-12-02 10
105410696,271557759,1,,12189,2014-12-07 23
127321491,394130643,1,9qo7r9d,2513,2014-12-05 23
105315608,287328215,1,,13230,2014-12-05 21
109130675,236349544,1,,5271,2014-12-02 19
117144062,217590155,1,,7117,2014-12-06 21
101982646,81372085,1,,13455,2014-12-04 23
```

2.2 将文件上传至OSS。

进入OSS控制台，上传文件使用已有的Bucket，或创建新的Bucket。



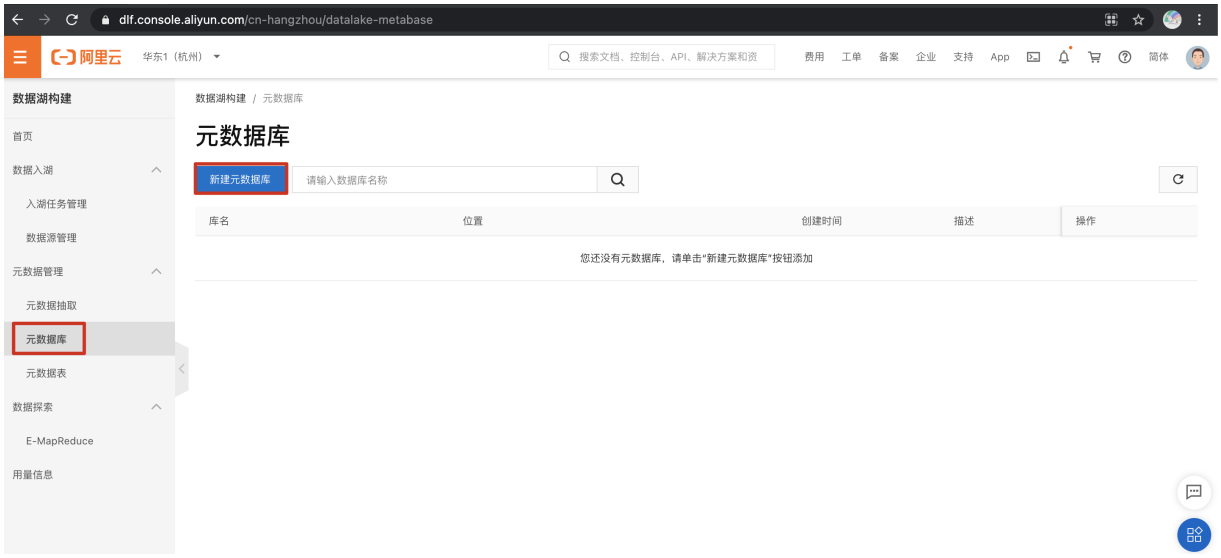
上传解压后的user_behavior_data文件夹。上传后目录结构如下所示，item和user为两个表的数据文件夹。

第三步：在DLF上抽取元数据

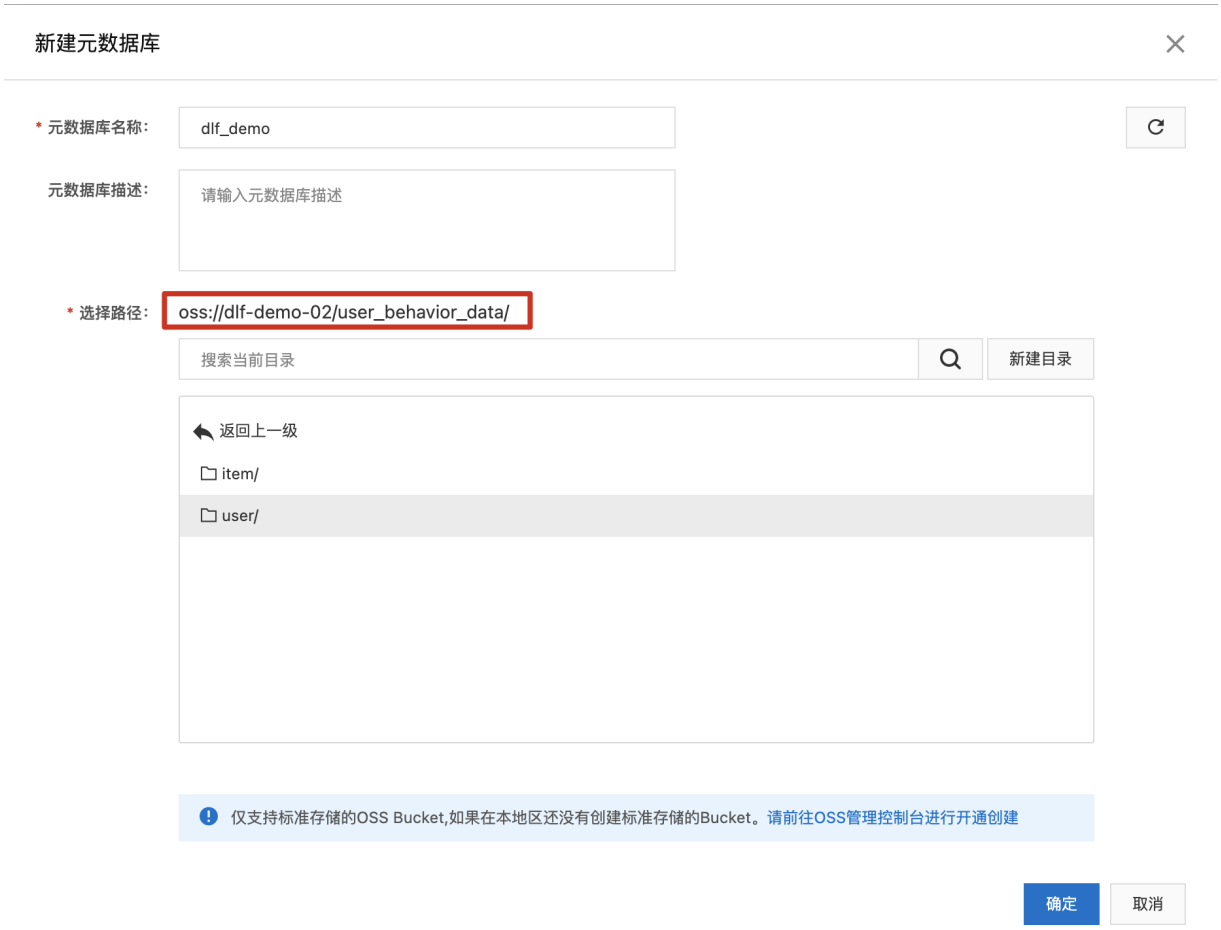
3.1 创建元数据表

DLF中元数据库可以理解为在关系型数据库中的Database，其下一级为Table。

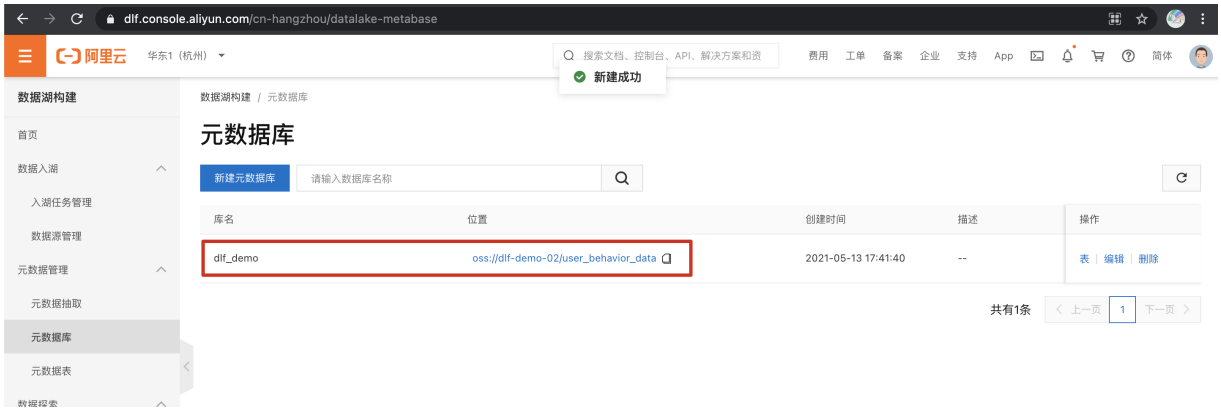
a. 在DLF控制台中，进入元数据库页面，创建元数据库。



b. 填入数据库名称。并选择刚才存有用户行为分析的

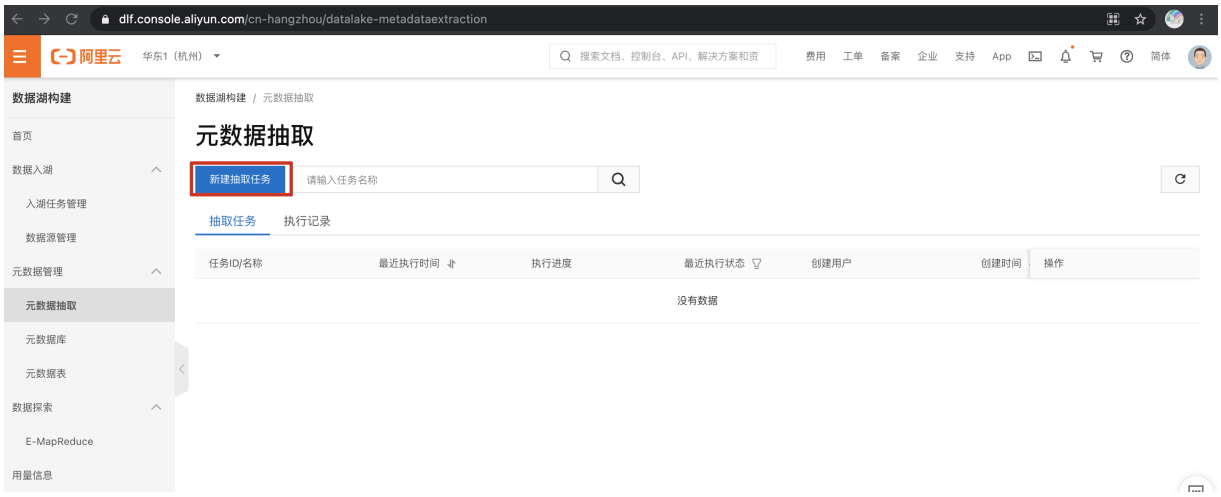


如下图所示，元数据库创建成功。

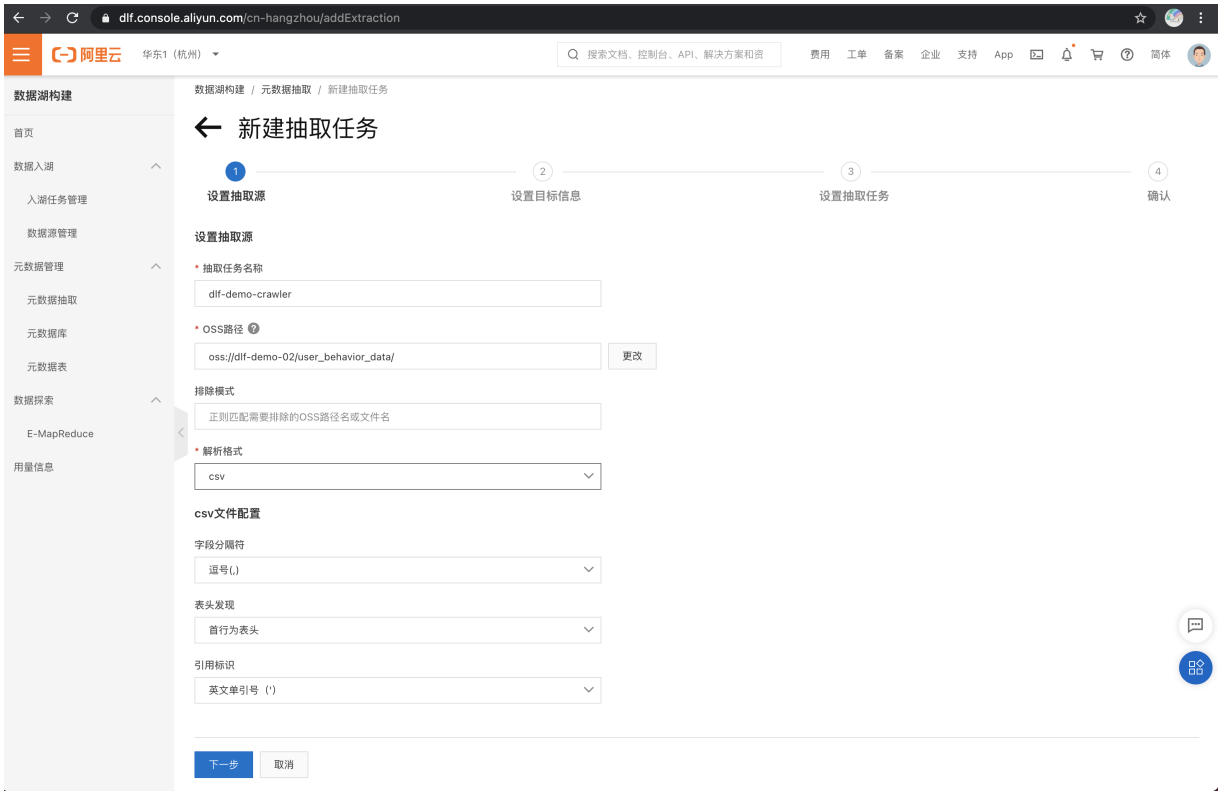


3.2 发现OSS文件中的元数据表信息

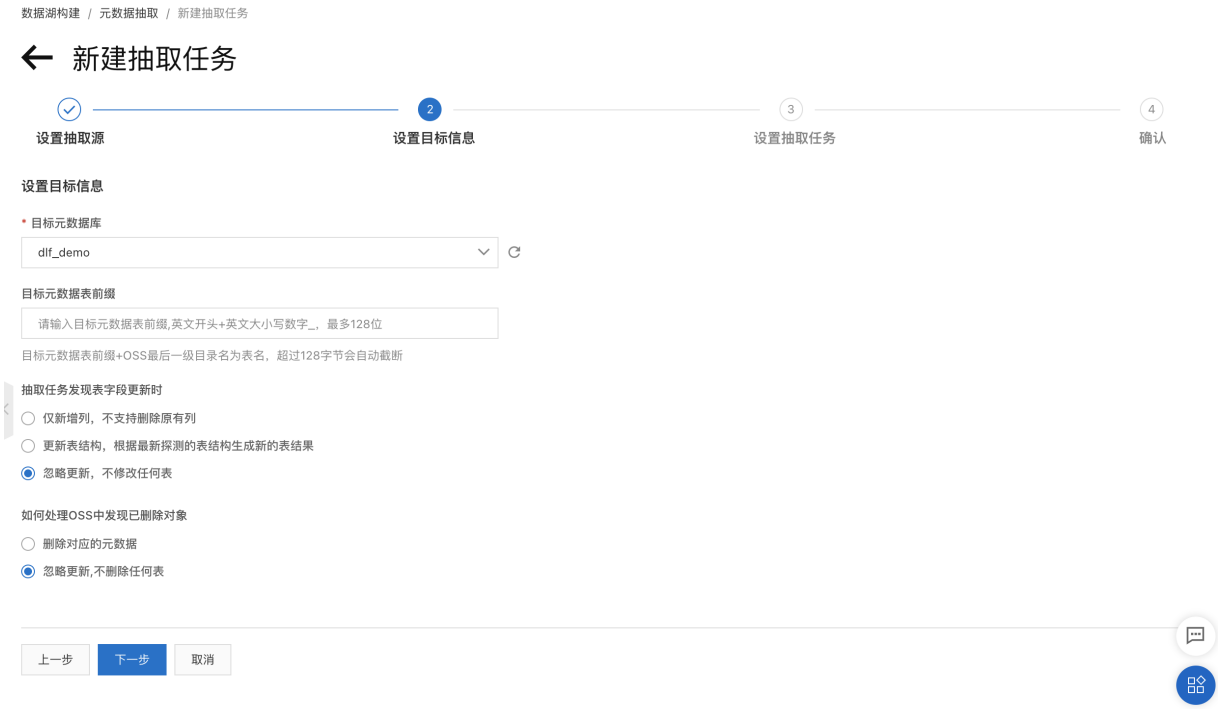
a. 进入DLF元数据抽取页面，点击“新建抽取任务”。



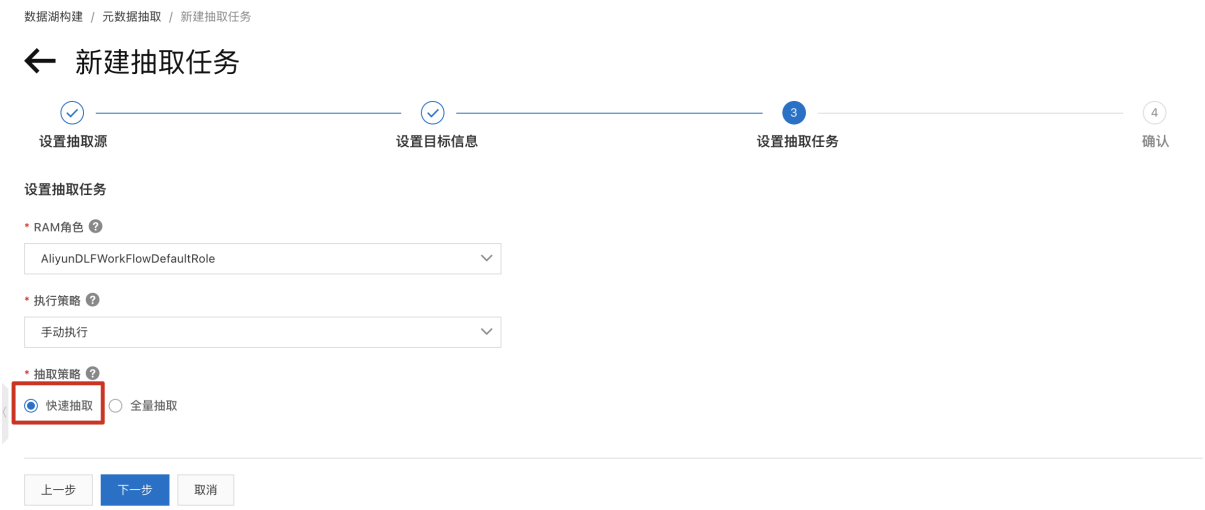
b. 填写数据源相关配置，点击下一步。



c. 选择要使用的目标元数据库，点击下一步。

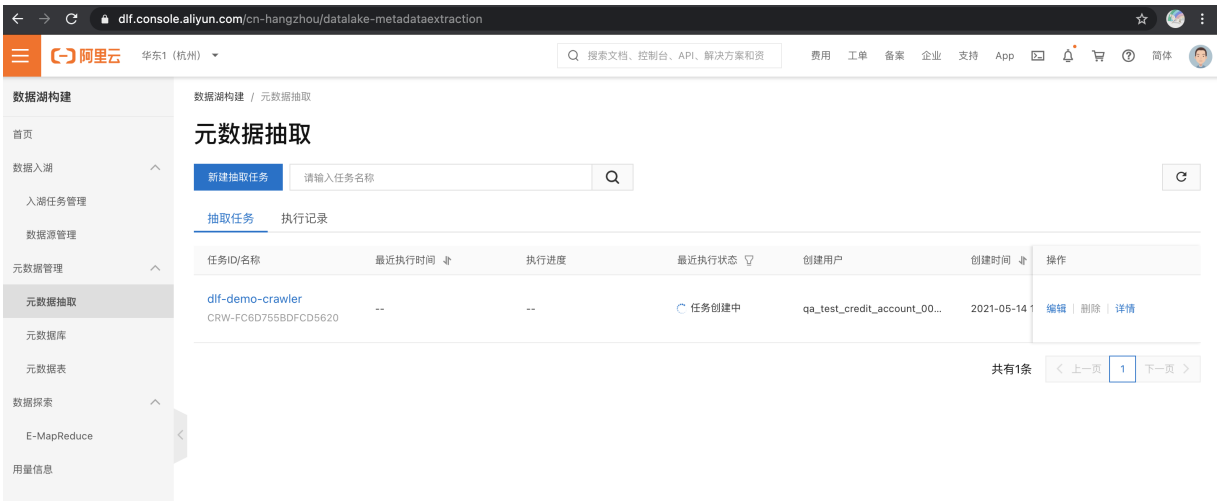


d. 选择过程中需要用到的RAM角色，默认为开通阶段已经授权的“AliyunDLFWorkFlowDefaultRole”。运行模式选择“手动执行”。抽取策略选择“快速模式”以最快的速度完成元数据发现。



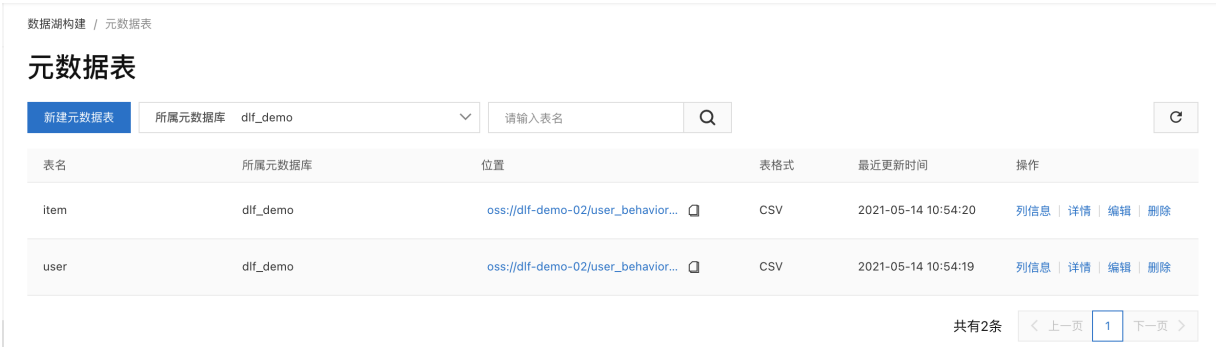
e. 核对信息后，点击“保存并立即执行”。

系统会跳转到元数据抽取列表页面，新建的任务开始创建并自动运行。

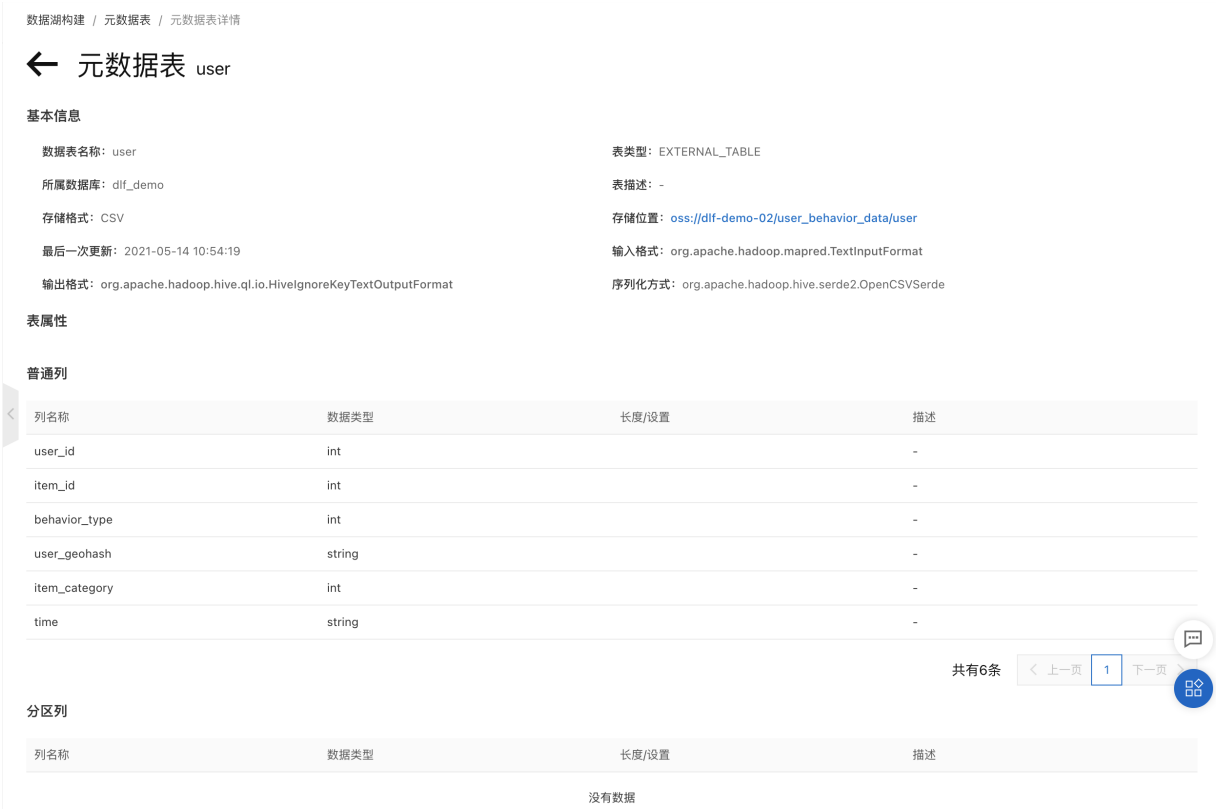


约10秒后任务运行完成。鼠标移到状态栏的问号图标，会看到已经成功创建了两张元数据表。

f. 点击浮层中的“元数据库”链接，可直接查看该库中相关的表信息。



g. 点击表详情，查看并确认抽取出来的表结构是否符合预期。

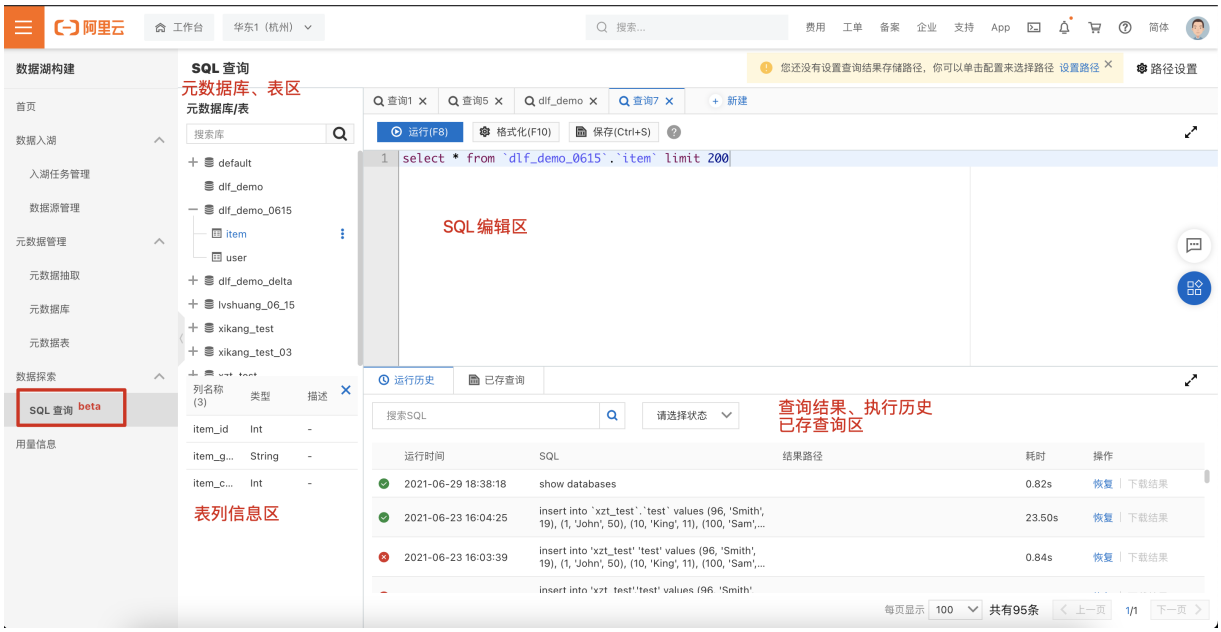


至此，我们通过DLF自动发现数据湖CSV文件Schema的过程已经完成。下一步我们开始针对数据湖内的数据做分析。

第四步：用户行为数据分析

4.1 数据分析概述

在DLF控制台页面，点击菜单“数据探索”-“SQL查询”，进入数据探索页面。



数据分析的过程主要分为三步：

1. 预览并检查数据信息。
2. 简单的数据清洗。
3. 进行用户活跃度、漏斗模型和商品热度分析。

4.2 预览并检查数据

在查询框输入下面的语句，查看文件中的数据信息。

```
-- 预览数据
select * from `dlf_demo`.`user` limit 10;
select * from `dlf_demo`.`item` limit 10;

-- 用户数 17970
select COUNT(DISTINCT user_id) from `dlf_demo`.`user`;

-- 商品数 422858
select COUNT(DISTINCT item_id) from `dlf_demo`.`item`;

-- 行为记录数 746024
select COUNT(*) from `dlf_demo`.`user`;
```

数据内容如下：


运行成功 开始时间：2021-05-14 11:45:01 持续时间：1.92秒 下载

user_id	item_id	behavior_type	user_geohash	item_category
113884016	70627757	1	\N	5271
120690949	367355690	1	\N	13381
106105154	291140936	1	\N	1863
105410696	271557759	1	\N	12189
127321491	394130643	1	9qo7r9d	2513
105315608	287328215	1	\N	13230
109130675	236349544	1	\N	5271

4.3 数据预处理

我们对原始数据进行一定的处理，已提高数据的可读性，并提高分析的性能。

将behavior_type修改成更好理解的字符串信息；将日志+时间的格式打平到两个单独的列，再加上周信息，便于分别做日期和小时级别的分析；过滤掉不必要的字段。并将数据存入新表user_log，表格式为Parquet，按日期分区。后续我们会基于新表做用户行为分析。

```
-- 数据转换耗时约40秒-- 创建新表user_log，parquet格式，按日期分区
-- 导入数据，拆分日期和小时，并将behavoior转换成可读性更好的值
-- 1-click; 2-collect; 3-cart; 4-pay
create table `dlf_demo`.`user_log`
USING PARQUET
PARTITIONED BY (date)
as select
    user_id,
    item_id,
    CASE
      WHEN behavior_type = 1 THEN 'click'
      WHEN behavior_type = 2 THEN 'collect'
      WHEN behavior_type = 3 THEN 'cart'
      WHEN behavior_type = 4 THEN 'pay'
    END as behavior,
    item_category,
    time,
    date_format(time, 'yyyy-MM-dd') as date,
    date_format(time, 'H') as hour,
    date_format(time, 'u') as day_of_week
  from `dlf_demo`.`user`;

 -- 查看运行后的数据
select * from `dlf_demo`.`user_log` limit 10;
```

4.4 用户行为分析

首先，我们基于漏斗模型，对所有用户从点击到加购/收藏到购买的转化情况。

```
-- 漏斗分析耗时13秒
SELECT
behavior, COUNT(*) AS total
FROM `dlf_demo`.`user_log`
GROUP BY behavior
ORDER BY total DESC
```

结果如下：

运行成功 开始时间：2021-05-14 14:20:18 持续时间：12.55秒 下载

behavior	total
click	702534
cart	21257
collect	15330
pay	6903

然后我们一周内每天的用户行为做统计分析

```
-- 用户行为分析耗时14秒
SELECT date, day_of_week,
COUNT(DISTINCT(user_id)) as uv,
SUM(CASE WHEN behavior = 'click' THEN 1 ELSE 0 END) AS click,
SUM(CASE WHEN behavior = 'cart' THEN 1 ELSE 0 END) AS cart,
SUM(CASE WHEN behavior = 'collect' THEN 1 ELSE 0 END) AS collect,
SUM(CASE WHEN behavior = 'pay' THEN 1 ELSE 0 END) AS pay
FROM `dlf_demo`.`user_log`
GROUP BY date, day_of_week
ORDER BY date
```

分析结果如下：（由于数据集经过裁剪，对于工作日和非工作日的结果有失真）

运行成功 开始时间：2021-05-14 14:03:25 持续时间：23.65秒 下载

date	day_of_week	uv	click	cart	collect	pay
2014-12-01	1	10981	100029	3009	2059	1028
2014-12-02	2	11064	100996	3235	2157	1071
2014-12-03	3	11171	106266	3231	2329	1079
2014-12-04	4	11064	100401	3006	2259	1039
2014-12-05	5	10799	93287	2831	2095	816
2014-12-06	6	10841	98821	2900	2139	980
2014-12-07	7	11154	102734	3045	2292	890

最后，我们结合商品表，分析出数据集中最受欢迎的是个商品品类

```
-- 销售最多的品类耗时1分10秒
SELECT item.item_category, COUNT(*) AS times
FROM `dlf_demo`.`item` item
  JOIN `dlf_demo`.`user_log` log
  ON item.item_id = log.item_id
WHERE log.behavior='pay'
GROUP BY item.item_category
ORDER BY times DESC
LIMIT 10;
```

结果如下：

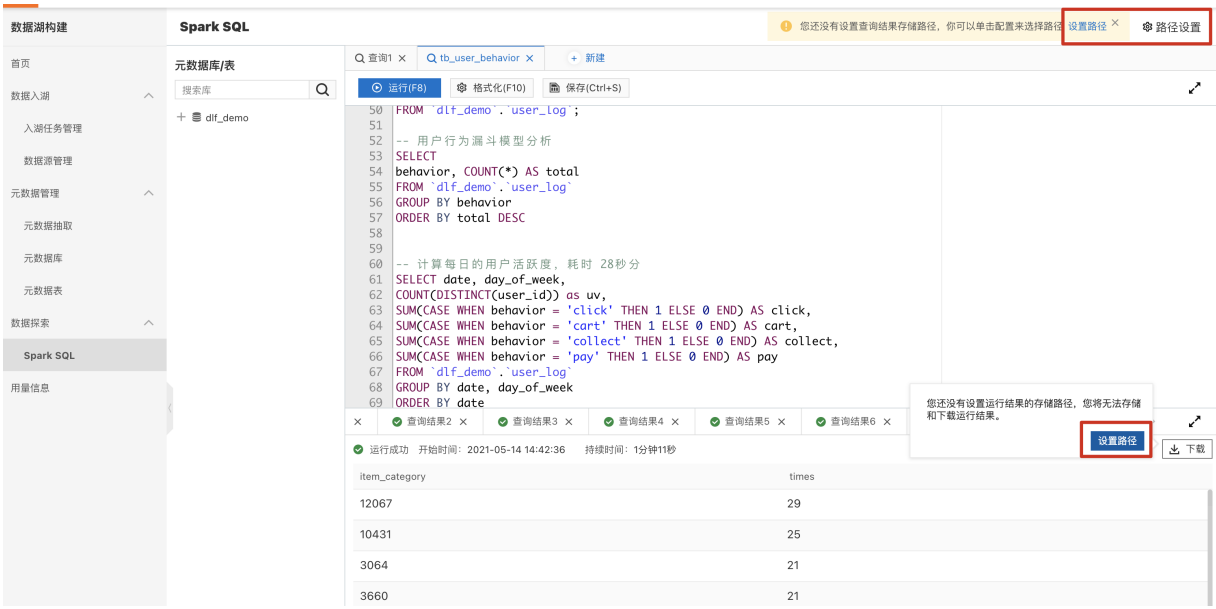
运行成功 开始时间：2021-05-14 14:42:36 持续时间：1分钟11秒 下载

item_category	times
12067	29
10431	25
3064	21
3660	21
9614	19
3942	18
3368	17
6648	17
12326	14

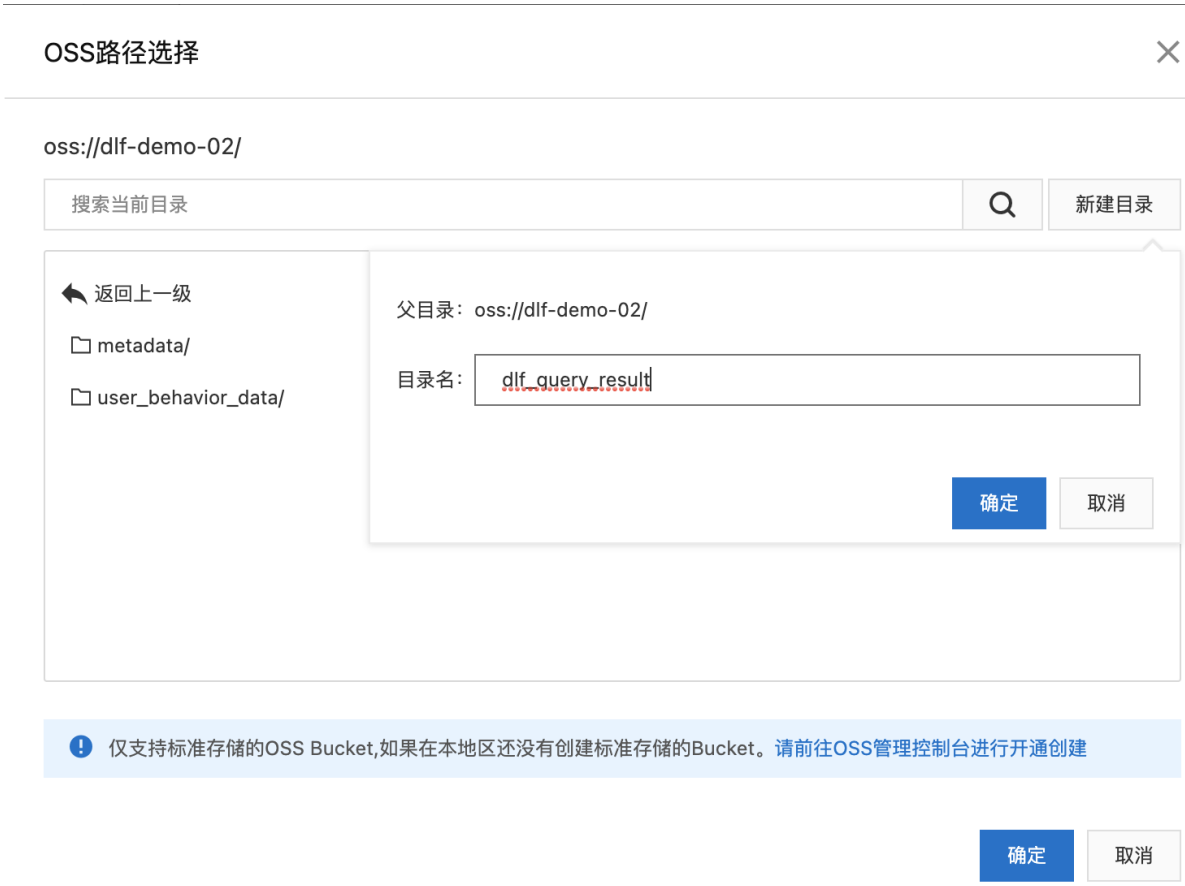
4.5 下载分析结果

DLF提供将分析结果以CSV文件的形式下载的功能，启用该功能需要提前设置分析结果的保存路径（OSS路径）。设置后，查询结果会被保存到该路径下。用户可以通过DLF页面上的“下载”按钮直接下载，也可以直接通过OSS访问和下载该文件。

通过点击页面上的“路径设置”按钮进行设置。

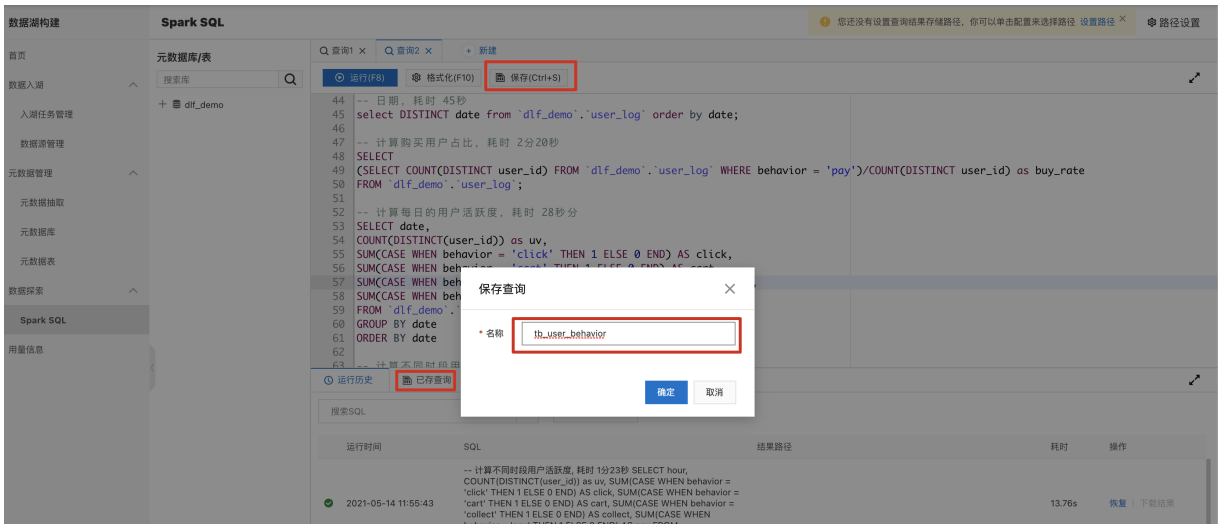


可以选择已有文件夹或者新建文件夹。



4.6 保存SQL

通过点击“保存”按钮，可以将该次分析用到的sql保存，后续可以直接在“已存查询”中打开做进一步的调用及修改。



总结

本文通过一个简单的淘宝用户行为分析案例，介绍并实践了数据湖构建（DLF）产品的元数据发现和数据探索功能。

有任何问题，或希望进一步探讨数据湖技术，欢迎扫码加入数据湖技术群，关注数据湖技术圈。

数据湖技术群（钉钉扫描）

数据湖技术圈（微信扫描）

6.4. 基于Delta lake的一站式数据湖构建与分析实战

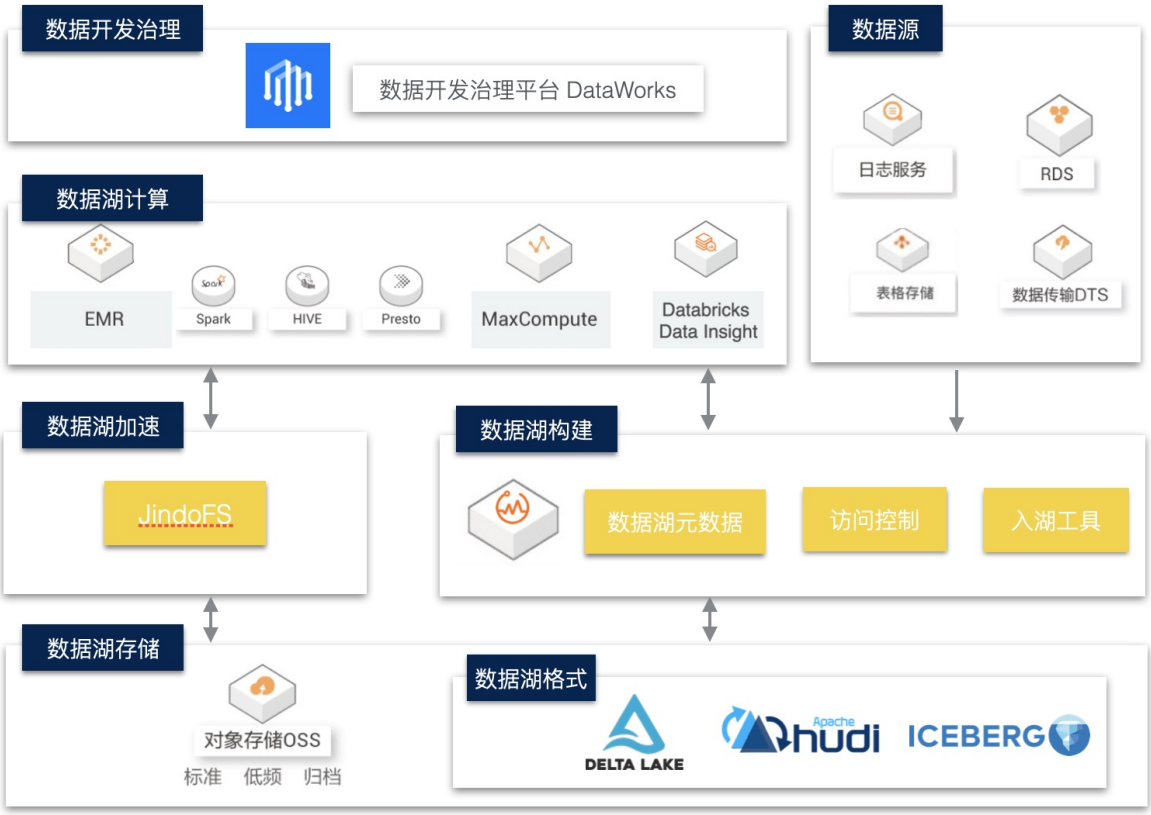
企业构建和应用数据湖一般需要经历数据入湖、数据湖存储与管理、数据湖探索与分析等几个过程。本文主要介绍基于阿里云数据湖构建（DLF）构建一站式的数据入湖与分析实战。

背景信息

随着数据时代的不断发展，数据量爆发式增长，数据形式也变的更加多样。传统数据仓库模式的成本高、响应慢、格式少等问题日益凸显。于是拥有成本更低、数据形式更丰富、分析计算更灵活的数据湖应运而生。

数据湖作为一个集中化的数据存储仓库，支持的数据类型具有多样性，包括结构化、半结构化以及非结构化的数据，数据来源上包含数据库数据、binglog 增量数据、日志数据以及已有数仓上的存量数据等。数据湖能够将这些不同来源、不同格式的数据集中存储管理在高性价比的存储如 OSS等对象存储中，并对外提供统一的数据目录，支持多种计算分析方式，有效解决了企业中面临的数据孤岛问题，同时大大降低了企业存储和使用数据的成本。

企业级数据湖架构



数据湖存储与格式

数据湖存储主要以云上对象存储作为主要介质，其具有低成本、高稳定性、高可扩展性等优点。

数据湖上我们可以采用支持ACID的数据湖存储格式，如Delta Lake、Hudi、Iceberg。这些数据湖格式有自己的数据meta管理能力，能够支持Update、Delete等操作，以批流一体的方式解决了大数据场景下数据实时更新的问题。

数据湖构建与管理

1. 数据入湖

企业的原始数据存在于多种数据库或存储系统，如关系数据库MySQL、日志系统SLS、NoSQL存储HBase、消息数据库Kafka等。其中大部分的在线存储都面向在线事务型业务，并不适合在线分析的场景，所以需要将数据以无侵入的方式同步至成本更低且更适合计算分析的对象存储。

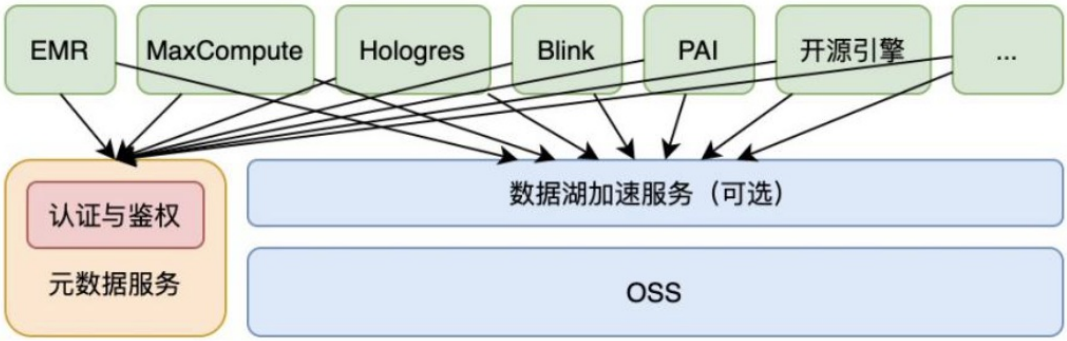
常用的数据同步方式有基于DataX、Sqoop等数据同步工具做批量同步；同时在对于实时性要求较高的场景下，配合使用Kafka+spark Streaming / flink等流式同步链路。目前很多云厂商提供了一站式入湖的解决方案，帮助客户以更快捷更低成本的方式实现数据入湖，如阿里云DLF数据入湖。

2. 统一元数据服务

对象存储本身是没有面向大数据分析的语义的，需要结合Hive Metastore Service等元数据服务为上层各种分析引擎提供数据的Meta信息。

数据湖计算与分析

相比于数据仓库，数据湖以更开放的方式对接多种不同的计算引擎，如传统开源大数据计算引擎Hive、Spark、Presto、Flink等，同时也支持云厂商自研的大数据引擎，如阿里云MaxCompute、Hologres等。在数据湖存储与计算引擎之间，一般还会提供数据湖加速的服务，以提高计算分析的性能，同时减少带宽的成本和压力。

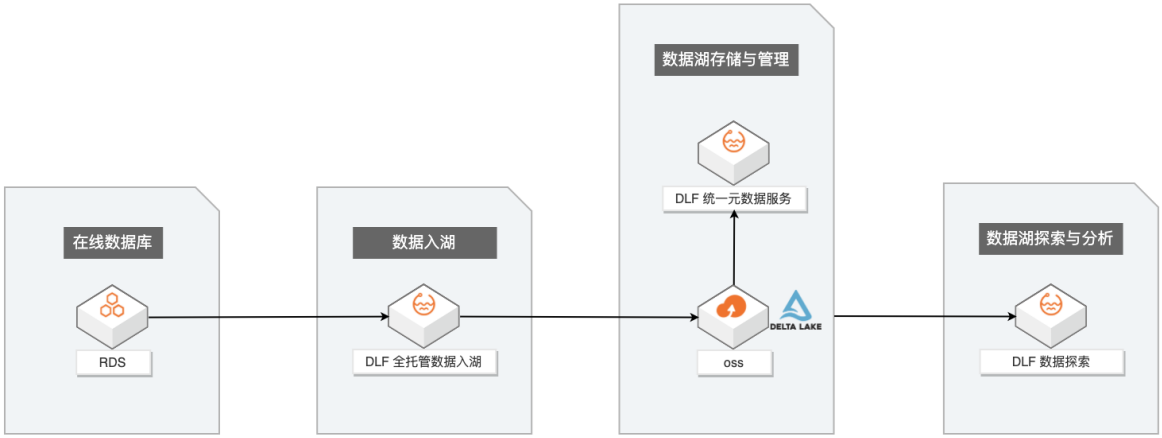


操作流程

数据湖构建与分析链路

企业构建和应用数据湖一般需要经历数据入湖、数据湖存储与管理、数据湖探索与分析等几个过程。本文主要介绍基于阿里云数据湖构建（DLF）构建一站式的数据入湖与分析实战。

其主要数据链路如下：



步骤一：服务开通并准备数据

1. 服务开通

确保DLF、OSS、DDI、RDS、DTS等云产品服务已开通。注意DLF、RDS、DDI实例均需在同一Region下。

2. 数据准备

RDS数据准备，在RDS中创建数据库dlf-demo。在账户中心创建能够读取employees数据库的用户账号，如dlf_admin。

通过DMS登录数据库，运行一下语句创建employees表，及插入少量数据。

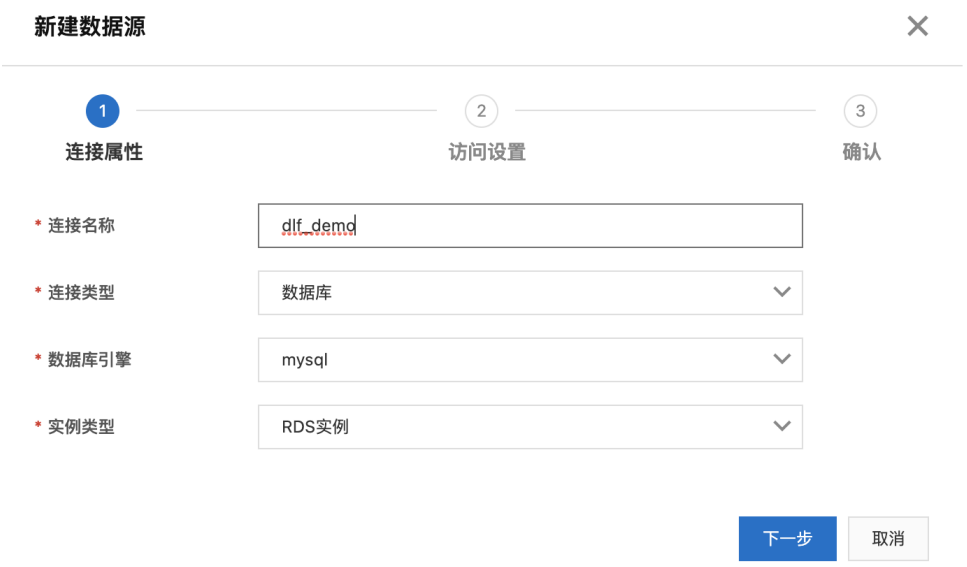
```
CREATE TABLE `employees` (
  `emp_no` int(11) NOT NULL,
  `birth_date` date NOT NULL,
  `first_name` varchar(14) NOT NULL,
  `last_name` varchar(16) NOT NULL,
  `gender` enum('M','F') NOT NULL,
  `hire_date` date NOT NULL,
  `create_time` DATETIME NOT NULL,
  `update_time` DATETIME NOT NULL,
  PRIMARY KEY (`emp_no`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8

INSERT INTO `employees` VALUES (10001,'1953-09-02','Georgi','Facello','M','1986-06-26', now
(), now());
INSERT INTO `employees` VALUES (10002,'1964-06-02','Bezalel','Simmel','F','1985-11-21', now
(), now());
```

步骤二：数据入湖

1. 创建数据源

a. 进入DLF控制台界面：https://dlf.console.aliyun.com/cn-hangzhou/home，点击菜单“数据入湖 -> 数据源管理”。

b. 点击“新建数据源”。填写连接名称，选择数据准备中的使用的RDS实例，填写账号密码，点击“连接测试”验证网络连通性及账号可用性。

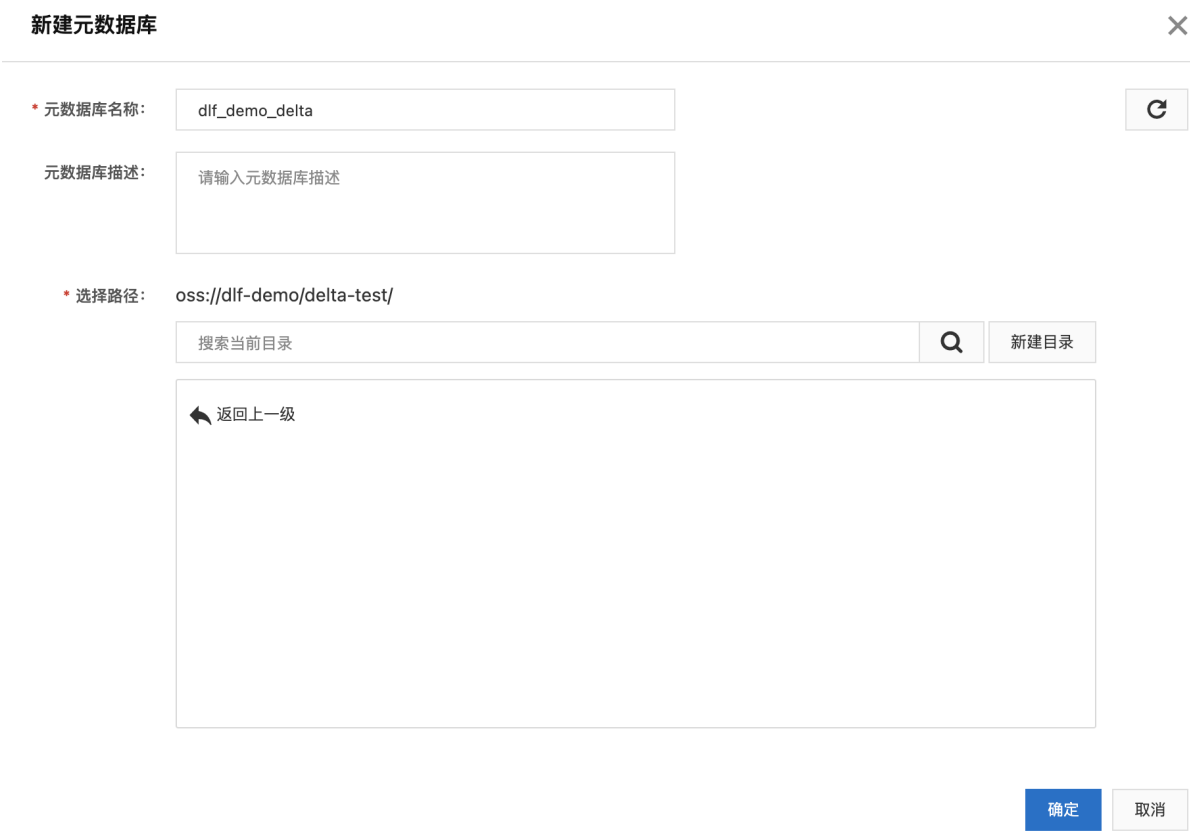


c. 点击下一步，确定，完成数据源创建。

2. 创建元数据库

a. 在OSS中新建Bucket，dlf-demo；

b. 点击左侧菜单“元数据管理”->“元数据库”，点击“新建元数据库”。填写名称，新建目录delta-test，并选择。

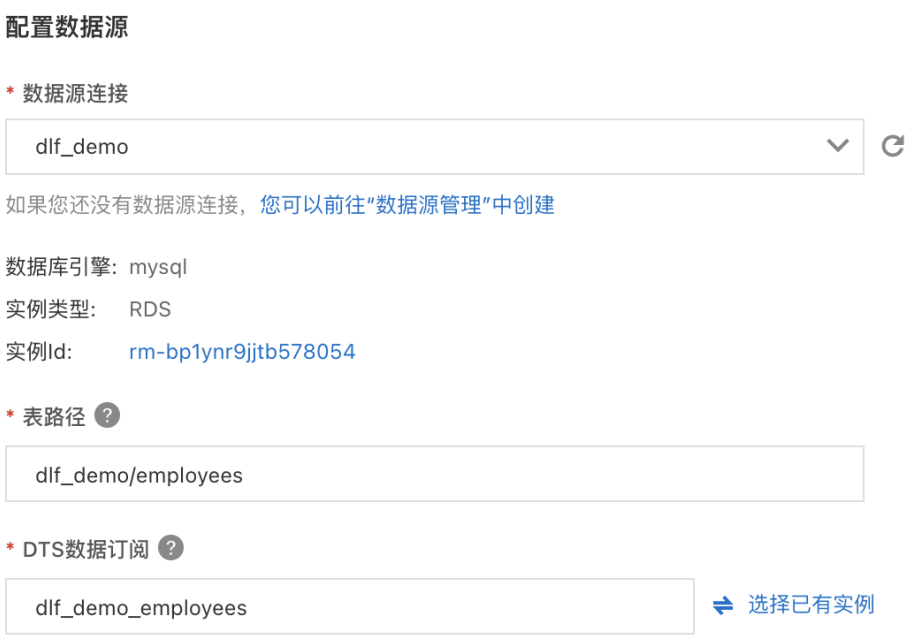


3. 创建入湖任务

a. 点击菜单“数据入湖”->“入户任务管理”，点击“新建入湖任务”。

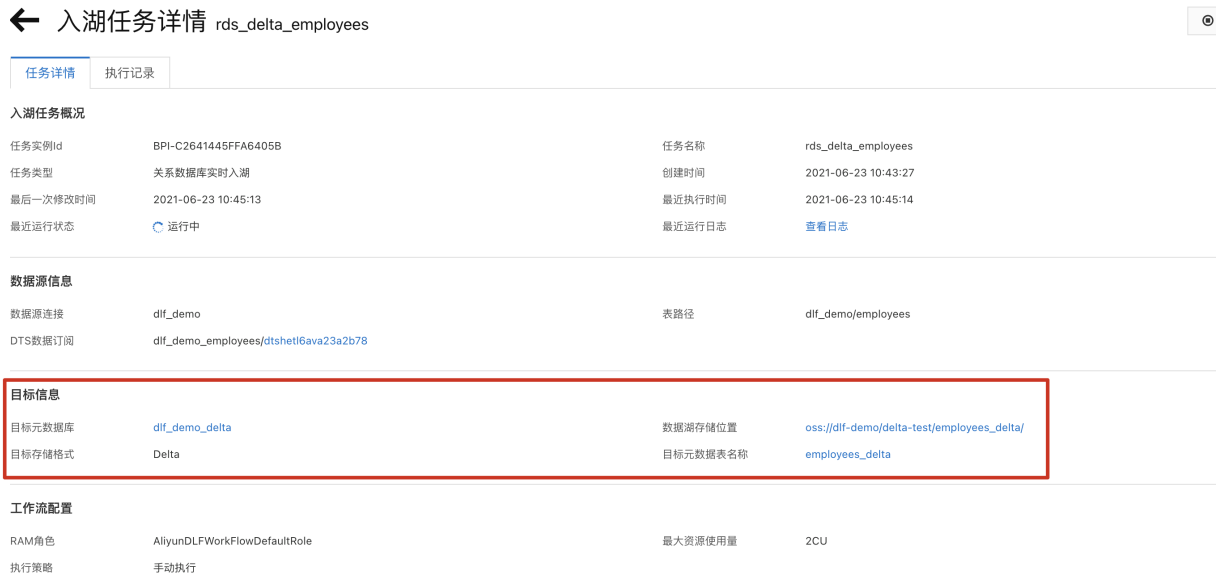
b. 选择“关系数据库实时入湖”，按照下图的信息填写数据源、目标数据湖、任务配置等信息。并保存。

c. 配置数据源，选择刚才新建的“dlf_demo”连接，使用表路径 “dlf_demo/employees”，选择新建dts订阅，填写名称。



d. 回到任务管理页面，点击“运行”新建的入湖任务。就会看到任务进入“初始化中”状态，随后会进入“运行”状态。

e. 点击“详情”进入任务详情页，可以看到相应的数据库表信息。

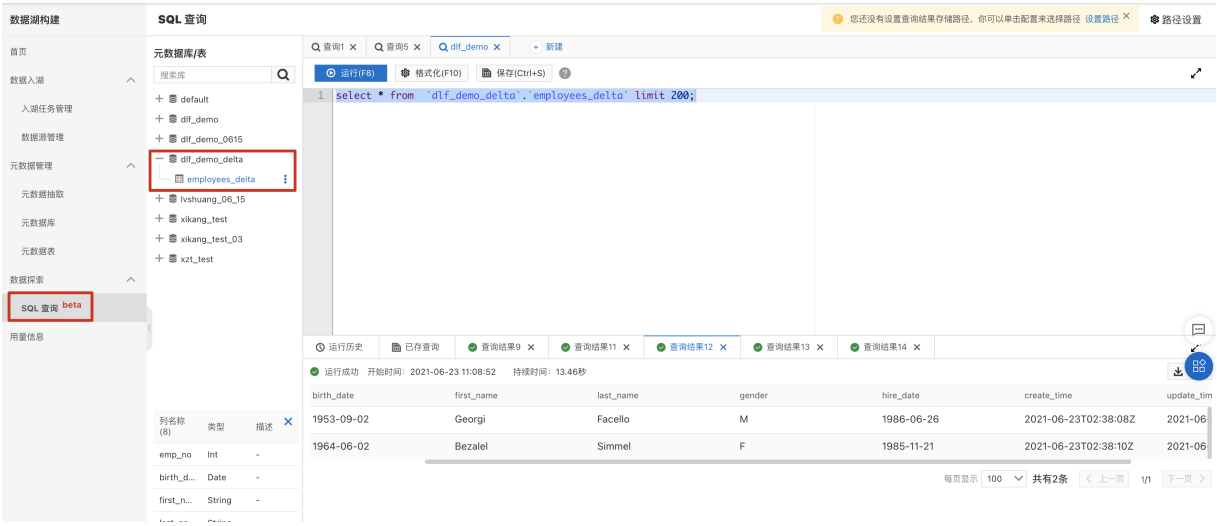


该数据入湖任务，属于全量+增量入湖，大约3至5分钟后，全量数据会完成导入，随后自动进入实时监听状态。如果有数据更新，则会自动更新至Delta Lake数据中。

步骤三：数据湖探索与分析

DLF产品提供了轻量级的数据预览和探索功能，点击菜单“数据探索”->“SQL查询”进入数据查询页面。

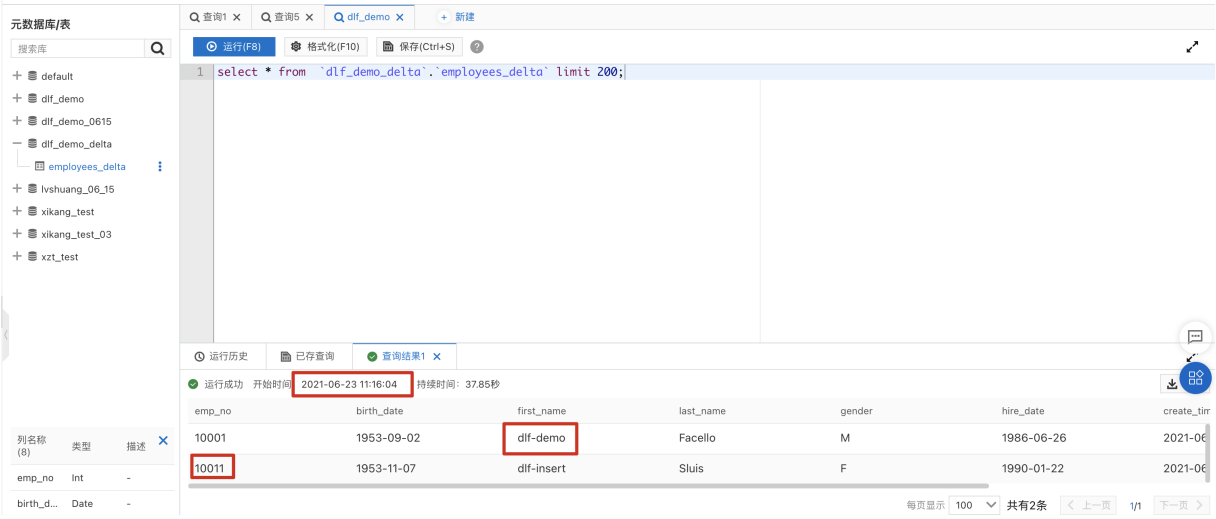
a. 在元数据库表中，找到“dlf_demo_delta”，展开后可以看到employees表已经自动创建完成。双击该表名称，右侧sql编辑框会出现查询该表的sql语句，点击“运行”，即可获得数据查询结果。



b. 回到DMS控制台，运行下方update、delete和insert SQL语句。

```
update `employees` set `first_name` = 'dlf-demo', `update_time` = now() where `emp_no` =100
01;
delete FROM `employees` where `emp_no` = 10002;
INSERT INTO `employees` VALUES (10011,'1953-11-07','dlf-insert','Sluis','F','1990-01-22', n
ow(), now());
```

c. 大约1至3分钟后，在DLF 数据探索再次执行刚才的select语句，所有的数据更新已经同步至数据湖中。



6.5. EMR元数据迁移数据湖DLF最佳实践

通过EMR+DLF数据湖方案，可以为企业提供数据湖内的统一的元数据管理，统一的权限管理，支持多源数据入湖以及一站式数据探索的能力。本方案支持已有EMR集群元数据库使用RDS或内置MySQL数据库迁移DLF，通过统一的元数据管理，多种数据源入湖，搭建高效的数据湖解决方案。

最佳实践详细步骤，请参考：EMR元数据迁移数据湖DLF最佳实践

7.常见问题

7.1. 常见问题

本文为您介绍数据湖构建的常见问题。

如何申请数据湖构建产品的公测资格？

请使用阿里云**主账号**进行公测资格申请（请如实填写公司信息），审批通过后即可访问数据湖构建的控制台。注意，子账号无法申请产品公测权限。

入湖模板中CU数如何评估？

入湖模板使用的CU数量是数据抽取任务执行时，每个计算单元消耗的资源量，1CU为2 vCPU 8GiB内存，公测阶段最多申请40个CU，如需使用更多计算资源，请提价工单到数据湖构建产品团队。

数据湖构建如何收费？

公测期产品完全免费，商业化后会根据数据入湖所消耗的计算资源收取资源费用，根据元数据表和对象设置每月的免费额度，超过额度的部分等额收取费用。

数据湖构建与OSS有什么区别？

数据湖后端存储使用OSS，用户使用数据湖构建面向大数据分析和机器学习场景可以获得集中式权限管理和统一的元数据视图，更容易对接云上大数据和分析产品。

什么情况下我需要使用数据湖构建？

在云上有数据分析和机器学习需求，希望构建云上的数据湖架构，降低运维和管理成本。

8.SDK参考

8.1. DataLake SDK for Java 使用参考

欢迎使用阿里云开发者工具套件（Alibaba Cloud SDK for Java），DataLake SDK for Java让您不用复杂编程即可访问数据湖构建。本教程介绍如何安装并开始使用DataLake SDK for Java。

前提条件

- 已创建AccessKey。
- 已安装Java环境。DataLake SDK for Java要求使用JDK1.7或更高版本。

安装Alibaba Cloud SDK for Java

您可以在Maven Repository中获取数据湖构建最新的SDK包，获取地址Maven SDK地址。

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>datalake20200710</artifactId>
    <version>1.0.0</version>
</dependency>
```

请求步骤

1. 初始化请求客户端，其中accessKey和accessKeySecret请按上方文档替换。

```
        Config authConfig = new Config();
        authConfig.accessKeyId= "xxxx";
        authConfig.accessKeySecret= "xxxx";
        authConfig.type= "access_key";
        authConfig.endpoint= "dlf.cn-shanghai.aliyuncs.com";
        authConfig.regionId= "cn-shanghai";
        Client authClient = new Client(authConfig);
```

2. 创建请求对象，构造参数。下述代码以CreateDatabase（新增元数据库）为例

```
        CreateDatabaseRequest request = new CreateDatabaseRequest();
        request.catalogId = "";

        DatabaseInput input = new DatabaseInput();
        input.description = "";
        input.locationUri = "oss://test";
        input.name = "example";

        request.databaseInput = input;
```

3. 执行调用，获取返回结果

```
        CreateDatabaseResponseBody response = authClient.createDatabase(request).body;
```

参考示例

以创建一个元数据库为例

```
package com.aliyun.datalake.examples;

import com.aliyun.datalake20200710.Client;
import com.aliyun.datalake20200710.models.CreateDatabaseRequest;
import com.aliyun.datalake20200710.models.CreateDatabaseResponseBody;
import com.aliyun.datalake20200710.models.DatabaseInput;
import com.aliyun.teaopenapi.models.Config;
import com.google.gson.Gson;

public class SchemaExample {

    public static void main(String[] args) throws Exception {
        // 1 Create and initialize a Config instance.
        Config authConfig = new Config();
        authConfig.accessKeyId = "your accessKeyId ";
        authConfig.accessKeySecret = "your accessKeySecret ";
        authConfig.type = "access_key";
        authConfig.endpoint = "dlf.cn-shanghai.aliyuncs.com";
        authConfig.regionId = "cn-shanghai";

        Client authClient = new Client(authConfig);

        // 2 Create an API request and set parameters.
        CreateDatabaseRequest request = new CreateDatabaseRequest();
        request.catalogId = "";

        DatabaseInput input = new DatabaseInput();
        input.description = "";
        input.locationUri = "oss://test";
        input.name = "example";

        request.databaseInput = input;
        // 3 Initiate the request and handle the response or exceptions.
        CreateDatabaseResponseBody response = authClient.createDatabase(request).body;
        System.out.println(new Gson().toJson(response));
    }

}
```

调用成功时，返回结果response示例：

```
{"code":"OK","message":"","requestId":"1739F0B0-A94E-49AC-95FC-C1CE5E4171FA","success":true
}
```

调用出现异常时，SDK会将状态码和错误信息包装成异常抛出给调用方，示例：

```
Exception in thread "main" com.aliyun.tea.TeaException: code: 409, Database example already
exists request id: 598B1E2F-9AEF-4B13-AE4D-EB8733B643EB
 at com.aliyun.teaopenapi.Client.doROARequest(Client.java:303)
 at com.aliyun.datalake20200710.Client.createDatabaseWithOptions(Client.java:790)
 at com.aliyun.datalake20200710.Client.createDatabase(Client.java:772)
 at com.aliyun.datalake.examples.SchemaExample.main(SchemaExample.java:34)
```

调用出现网络等未知异常时，SDK会直接抛出，示例：

```
Exception in thread "main" com.aliyun.tea.TeaException
 at com.aliyun.tea.Tea.doAction(Tea.java:67)
 at com.aliyun.teaopenapi.Client.doROARequest(Client.java:292)
 at com.aliyun.datalake20200710.Client.createDatabaseWithOptions(Client.java:790)
 at com.aliyun.datalake20200710.Client.createDatabase(Client.java:772)
 at com.aliyun.datalake.examples.SchemaExample.main(SchemaExample.java:34)
Caused by: java.net.UnknownHostException: dlf.cn-shanghai.aliyuncs.com
 at java.net.Inet6AddressImpl.lookupAllHostAddr(Native Method)
 at java.net.InetAddress$2.lookupAllHostAddr(InetAddress.java:928)
 at java.net.InetAddress.getAddressesFromNameService(InetAddress.java:1323)
 at java.net.InetAddress.getAllByName0(InetAddress.java:1276)
 at java.net.InetAddress.getAllByName(InetAddress.java:1192)
 at java.net.InetAddress.getAllByName(InetAddress.java:1126)
 at okhttp3.Dns$1.lookup(Dns.java:39)
 at okhttp3.internal.connection.RouteSelector.resetNextInetSocketAddress(RouteSelector.java
:171)
 at okhttp3.internal.connection.RouteSelector.nextProxy(RouteSelector.java:137)
 at okhttp3.internal.connection.RouteSelector.next(RouteSelector.java:82)
 at okhttp3.internal.connection.StreamAllocation.findConnection(StreamAllocation.java:171)
 at okhttp3.internal.connection.StreamAllocation.findHealthyConnection(StreamAllocation.jav
a:121)
 at okhttp3.internal.connection.StreamAllocation.newStream(StreamAllocation.java:100)
 at okhttp3.internal.connection.ConnectInterceptor.intercept(ConnectInterceptor.java:42)
 at okhttp3.internal.http.RealInterceptorChain.proceed(RealInterceptorChain.java:92)
 at okhttp3.internal.http.RealInterceptorChain.proceed(RealInterceptorChain.java:67)
 at okhttp3.internal.cache.CacheInterceptor.intercept(CacheInterceptor.java:93)
 at okhttp3.internal.http.RealInterceptorChain.proceed(RealInterceptorChain.java:92)
 at okhttp3.internal.http.RealInterceptorChain.proceed(RealInterceptorChain.java:67)
 at okhttp3.internal.http.BridgeInterceptor.intercept(BridgeInterceptor.java:93)
 at okhttp3.internal.http.RealInterceptorChain.proceed(RealInterceptorChain.java:92)
 at okhttp3.internal.http.RetryAndFollowUpInterceptor.intercept(RetryAndFollowUpInterceptor
.java:120)
 at okhttp3.internal.http.RealInterceptorChain.proceed(RealInterceptorChain.java:92)
 at okhttp3.internal.http.RealInterceptorChain.proceed(RealInterceptorChain.java:67)
 at okhttp3.RealCall.getResponseWithInterceptorChain(RealCall.java:185)
 at okhttp3.RealCall.execute(RealCall.java:69)
 at com.aliyun.tea.Tea.doAction(Tea.java:64)
 ... 4 more
```

最佳实践

为了便于返回结果的统一处理，获取到和API文档一致的异常结果，我们可以通过一些固定的写法，来处理SDK正常或异常的返回结果。

例如，我们把API调用层用统一的方法来包装：

```
public class AbstractAPI {

    protected final Client client;

    public AbstractAPI(Client client) {
        this.client = client;
    }

    public  <M, V extends ResultModel<M>> ResultModel<M> call(Callable<V> c) throws Excepti
on {
        try {
            return c.call();
        } catch (TeaException e) {
            Map<String, Object> data = e.getData();
            if (data!= null && data.get("Code") != null) {
                return TeaModel.toModel(data, new ResultModel<M>());
            } else {
                throw e;
            }
        }
    }
}
```

对于具体的API，可以继承上面的类，用适合自己的参数构造调用方法。

```
public class DatabaseAPI extends AbstractAPI {

    public DatabaseAPI(Client client) {
        super(client);
    }

    public ResultModel<Void> createDatabase(String catalogId, String databaseName, String d
escription,
                                            String locationUri, Map<String, String> paramet
ers,
                                            String ownerName, String ownerType, PrincipalPr
ivilegeSet privileges) throws Exception {
        return call(()-> {
            CreateDatabaseRequest request = new CreateDatabaseRequest();
            request.catalogId = catalogId;

            DatabaseInput input = new DatabaseInput();
            input.description = description;
            input.locationUri = locationUri;
            input.parameters = parameters;
            input.name = databaseName;
            input.ownerName = ownerName;
            input.ownerType = ownerType;
            input.privileges = privileges;

            request.databaseInput = input;

            CreateDatabaseResponseBody response = client.createDatabase(request).body;
            return new ResultModel<>(response.success, response.code, response.message,
                    response.requestId);
        });
    }
}
```

这样，在使用每个API的时候，就可以直接拿到标准的Response了：

```
public class SchemaExample {

    public static void main(String[] args) throws Exception {
        // 1 Create and initialize a Config instance.
        Config authConfig = new Config();
        authConfig.accessKeyId = "your accessKeyId ";
        authConfig.accessKeySecret = "your accessKeySecret";
        authConfig.type = "access_key";
        authConfig.endpoint = "dlf.cn-shanghai.aliyuncs.com";
        authConfig.regionId = "cn-shanghai";

        Client authClient = new Client(authConfig);

        // 2 Initiate the request and handle the response or exceptions.
        ResultModel<Void> response = new DatabaseAPI(authClient).createDatabase("", "exampl
e3", "",
                "oss://test", null,
                null, null, null);
        System.out.println(new Gson().toJson(response));
    }

}
```

创建成功时的response示例：

```
{"success":true,"code":"OK","message":"","requestId":"50778D55-696D-45FA-8328-B01983F6CEB1"
}
```

创建出错时的response示例：

```
{"success":false,"code":"AlreadyExists","message":"Database example3 already exists","reque
stId":"94617169-DA17-4020-9027-7D8F89160682","httpStatusCode":409}
```

更多信息

- DLF目前支持的地域（Region）和域名（Endpoint），请参考已开通的地域和访问域名。
- 在线调试和生成SDK示例。OpenAPI Explorer提供在线调用云产品API、动态生成SDK示例代码和快速检索接口等功能，能显著降低使用API的难度，推荐您使用。

8.2. 数据探索Java SDK示例

您可以通过SDK提交DLF数据探索任务。

前提条件

- 已创建AccessKey。
- 已安装Java环境。DataLake SDK for Java要求使用JDK1.7或更高版本。

安装DataLake SDK

您可以在Maven Repository中获取数据湖构建最新的SDK包，获取地址Maven SDK地址。

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>datalake20200710</artifactId>
    <version>2.0.6</version>
</dependency>
```

参考示例

您可以通过数据探索相关的API：SubmitQuery、GetQueryResult、CancelQuery来执行和管理SQL查询任务。API详情参考数据探索API

```
import com.aliyun.datalake20200710.Client;
import com.aliyun.datalake20200710.models.GetQueryResultRequest;
import com.aliyun.datalake20200710.models.GetQueryResultResponse;
import com.aliyun.datalake20200710.models.SubmitQueryRequest;
import com.aliyun.datalake20200710.models.SubmitQueryResponse;
import com.aliyun.teaopenapi.models.Config;

public class QueryExample {

    public static void main(String[] args) throws Exception {
        Config authConfig = new Config();
        authConfig.accessKeyId= "xxxxxxxxx";
        authConfig.accessKeySecret= "xxxxxxxxx";
        authConfig.endpoint= "dlf.cn-hangzhou.aliyuncs.com";
        authConfig.regionId= "cn-hangzhou";
        Client authClient = new Client(authConfig);

        // 提交查询
        SubmitQueryRequest queryRequest = new SubmitQueryRequest();
        queryRequest.setSql("show databases;");
        SubmitQueryResponse queryResponse = authClient.submitQuery(queryRequest);

        String queryId = queryResponse.getBody().getData();


        // 取消查询示例
//        CancelQueryRequest cancelRequest  = new CancelQueryRequest();
//        cancelRequest.setQueryId(queryId);
//        CancelQueryResponse cancelResult = authClient.cancelQuery(cancelRequest);
//        System.out.println(cancelResult.getBody().getSuccess());

        // 获取查询结果
        GetQueryResultRequest queryResultRequest = new GetQueryResultRequest();
        queryResultRequest.setQueryId(queryId);
        queryResultRequest.setPageNumber(1);
        queryResultRequest.setPageSize(100);
        GetQueryResultResponse result = authClient.getQueryResult(queryResultRequest);

        while(!result.getBody().getJobCompleted()){
            // fetch query result until it is ready
            Thread.sleep(1000);
            result = authClient.getQueryResult(queryResultRequest);
        }

        if ("AVAILABLE".equals(result.getBody().getStatus())) {
            System.out.println(result.getBody().getSchema());
            System.out.println(result.getBody().getRows());
        } else {
            System.out.println(result.getBody().getErrorMessage());
        }

    }
}
```