DataWorks（数据工场）

最佳实践

最佳实践

进阶示例-BI报表

示例说明

实验背景

本实验基于一份真实的日志数据，了解通过 DataWorks 操作 MaxCompute 来构建数据仓库，完成各种数据分析需求。

使用环境

本示例需要使用的环境有：大数据计算服务（MaxCompute）、大数据开发（DataWorks）、Quick BI。

数据准备

该示例基于一份真实的数据集，数据来源于酷壳（CoolShell.cn）网站上的HTTP访问日志数据。

这份数据是2014/2/12一天的访问日志（附件coolshell_20140212.log列分隔符为"`"）。

数据格式如下：

$remote_addr - $remote_user [$time_local] "$request" $status $body_bytes_sent"$http_referer" "$http_user_agent" [unknown_content];

序号	字段名称	字段说明
1	$remote_addr	发送请求的客户端IP地址。
2	$remote_user	客户端登录名

3	$time_local	服务器本地时间
4	$request	请求，包括HTTP请求类型+请求URL+HTTP协议版本号
5	$status	服务端返回状态码
6	$body_bytes_sent	返回给客户端的字节数（不含header）
7	$http_referer	该请求的来源URL
8	$http_user_agent	发送请求的客户端信息，如使用的浏览器等

一条真实的数据如下：

14.136.107.2482014-02-12 03:08:03`GET /feed HTTP/1.1`200`92446Mozilla/5.0 (Linux; Android 4.4.2; Nexus 4 Build/KOT49H) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/30.0.0.0 Mobile Safari/537.36

需求分析

基于这份网络日志，完成如下分析需求：

1.统计网站的PV（浏览次数）、UV（独立访客），按用户的终端类型（如Android、iPad、iPhone、PC等）分别统计，并给出这一天的统计报表；

2.网站的访问来源，了解网站的流量从哪里来。

【说明】网站统计指标：

浏览次数（PV）和独立访客（UV）是衡量网站流量的两项最基本指标。用户每打开一个网站页面，记录一个PV，多次打开同一个页面PV累计多次。独立访客是指一天内，访问网站的不重复用户数，一天内同一个访客多次访问网站只计算一次。
Referer表示该请求日志从哪里来，它可以分析完整的访问来源，还可以用于分析用户和偏好等，是网站广告投放评估的重要指标。

要实现这两个需求，整个流程如下：

1）日志数据导入到ODPS表，从数据仓库角度该表属于ODS层，因此导入的ODPS表名为ods_log_tacker；

2）数据加工。从数据说明章节中我们看到日志数据中$request字段包含“HTTP请求类型+请求URL+HTTP协议版本号”，由于后续分析中经常会分别查询统计如GET的请求统计、URL进行分析等，所以我们需要把原始表的request字段拆解，并把拆解后的数据写入表 dw_log_parser（数据仓库层表）。

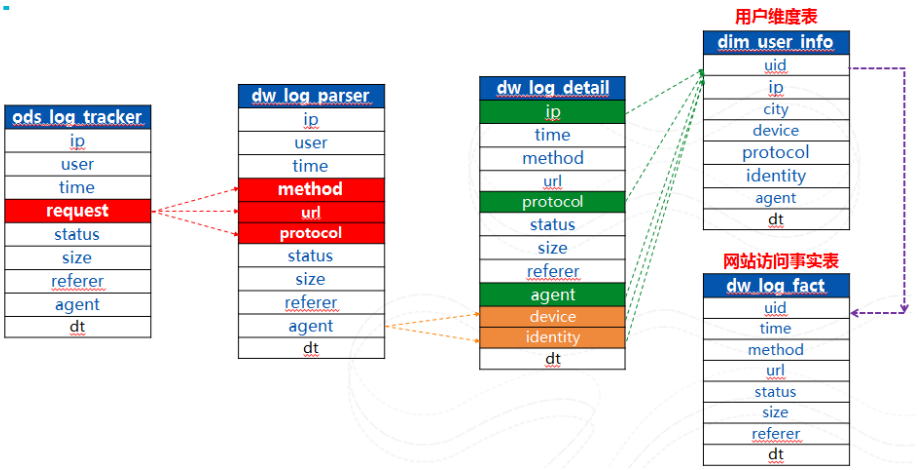
3）数据分析。一般网站日志数据按用户身份可以划分为真实用户请求和程序发送请求（订阅程序、爬虫等）两

大类，在统计流量（PV、UV）等指标时往往是基于真实用户请求的日志进行统计，此外，用户访问页面时，往往会有除页面本身请求外的其他如js、图片发送的请求，这些都是做真实统计时需要过滤的请求。因此需要把这些分析的处理结果写到新表dw_log_detail（数据仓库层表）。

4）在数据仓库中，随着数据分析的深入，往往会构建维度表和事实表，本实验中，我们可以构建用户维度表dim_user_info和网站访问事实表dw_log_fact。

5）根据数据仓库层中的维度表和事实表，生成基于终端设备信息的PV和UV统计表adm_user_measures（应用数据集市层）、生成网站请求的来源URL统计表adm_refer_info（应用数据集市层）。

以上1-4过程涉及到的各个表逻辑关系如下图：



创建表&导入数据

1）创建ODPS表。在此我们不再赘述，创建过程可以参考章节 **快速开始>创建删除表**，相关建表语句请看附件章节。

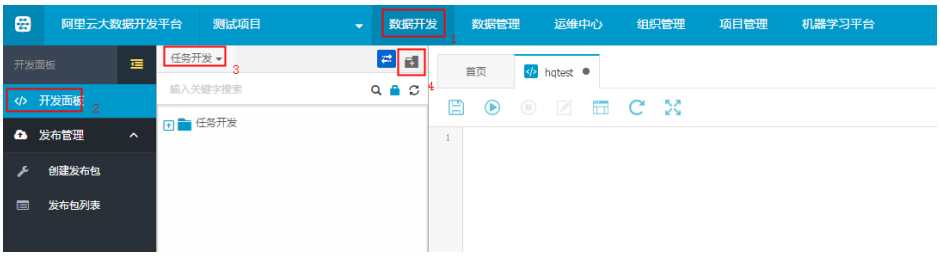
2)数据导入ODPS表。在此也不再赘述数据如何导入到ODPS目标表，数据导入ODPS相关步骤请参考章节，**快速开始>本地数据导入**(适用目标为非分区表，本地文件小于10M)或者**快速开始>创建数据同步任务**(需把日志文件放到RDS等云存储服务)或者**大数据计算服务 ODPS > 快速开始 > 导入导出数据**（需要自己本地安装console客户端）

>>>下一步：数据加工分析>>>

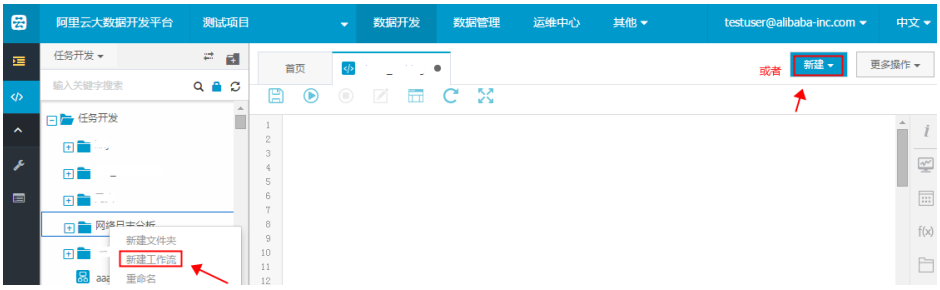
数据加工分析

进入数据开发页面，创建工作流coolshell_log。

步骤1：创建工作流文件目录。文件目录树切换到“数据开发”新建目录——网络日志分析：



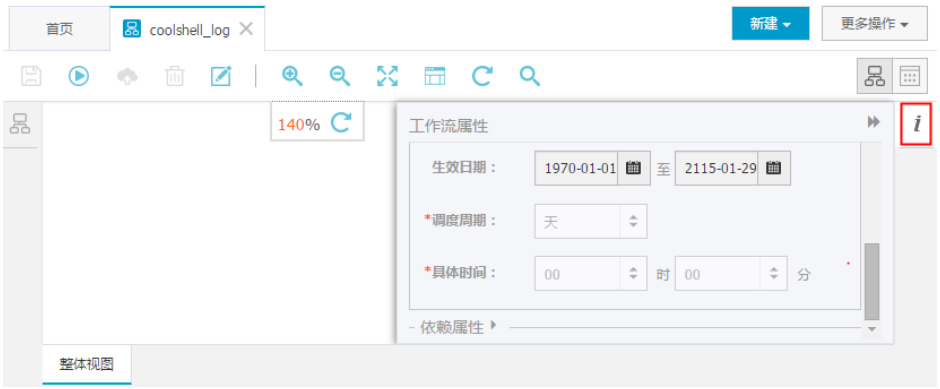
步骤2：目录文件夹上右键新建工作流或者点击工作区右上角新建按钮，下拉列表中选择新建工作流。



输入工作流名称（coolshell_log）和描述，并选择调度类型为“周期调度”。考虑到该工作流是需要后续每日自动调度生产每日报表，所以选择周期调度。

点击“创建”按钮，成功创建工作流。

步骤3：配置工作流属性。调度时间属性如下图，依赖属性本案例不用依赖，现实场景中如果有数据导入工作流则需要依赖输入导入工作流。



设计工作流节点

工作流创建好后，进入工作流设计面板添加节点进行节点设计。

步骤1：鼠标双击节点组件或者拖拽节点组件到右边画布，依次添加以下节点：

■ 数据加工（ODPS SQL）：节点名dw_log_parser，数据导入之后，对数据进行进一步的ETL过程（request字段拆解），并将数据写入dw_log_parser。

现实场景若数据导入不是在其他工作流，那么应该需要先有一个输入导入节点，本案例省略了数据导入步骤。

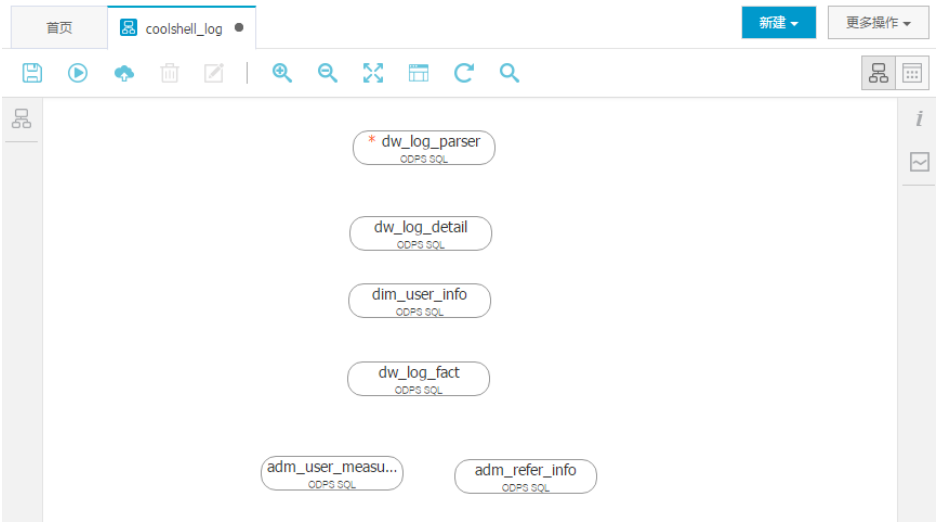
■ 数据分析（ODPS SQL）：节点名dw_log_detail，对dw_log_parser表进行进一步的数据分析、加工，得到dw_log_detail。

■ 数据分析（ODPS SQL）：基于dw_log_detail表构建用户维度表（dim_user_info）和网站访问事实表（dw_log_fact），对应节点名称dim_user_info和dw_log_fact。

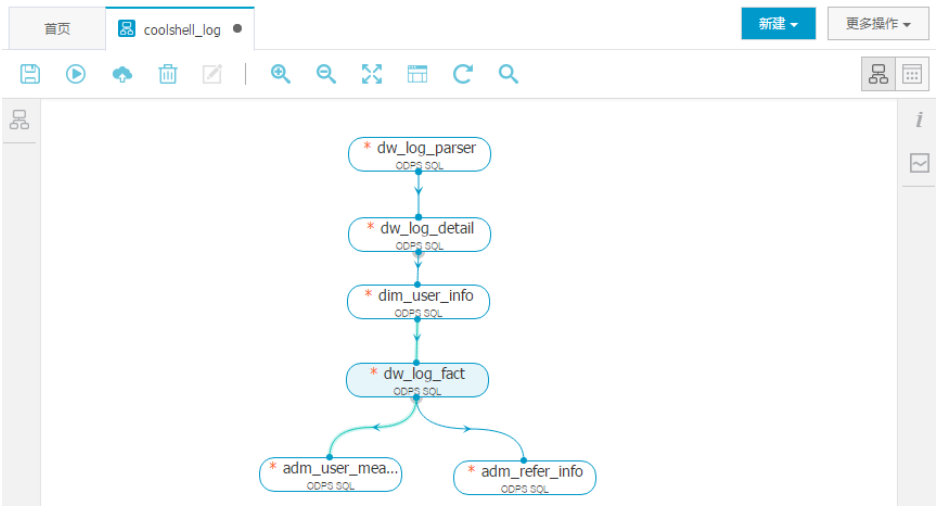
■ 数据应用（ODPS SQL）：基于前面的用户维度表和网站访问事实表，完成本实验“需求分析”中提出的业务需求，产出按用户终端类型统计网站的PV/UV表（adm_user_measueres）和网站访问来源表(adm_refer_info)。

> 数据应用面向业务需求，正常情况下，有可能这一层开发会是另一个团队同学，任何会在另外项目另外工作流里，本实验为了方便完整的走通，这一层的两个任务dm_user_measueres、 adm_refer_info也放在这个工作流中。

这些节点在画布上散布如图：

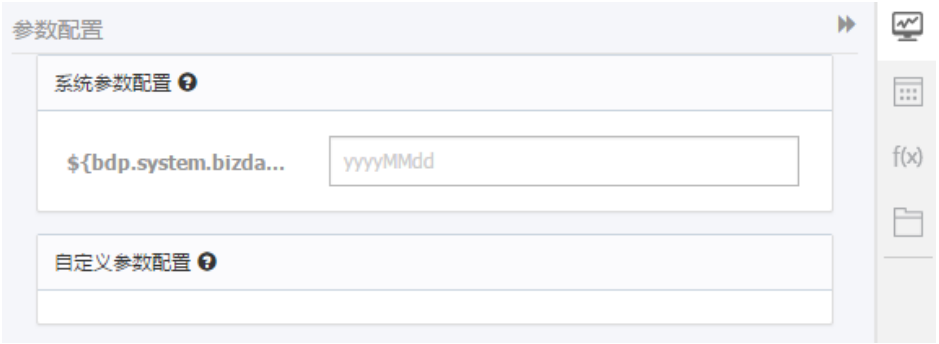


3）需要根据节点内在的逻辑，通过连线体现出相互之间关系。鼠标移到节点上时该节点下沿中部有一个小半圆，然后把鼠标移到这个半圆上，鼠标即变为十字形，此时，按下左键可以划出连线，把鼠标拖至下一个节点，即形成了两节点之间的连线。连线体现了节点的依赖关系，箭头体现的是顺序。 对节点进行连线， 连线后点击保存按钮，保存我们的设计。 连线后各节点的情况如下图：



配置节点

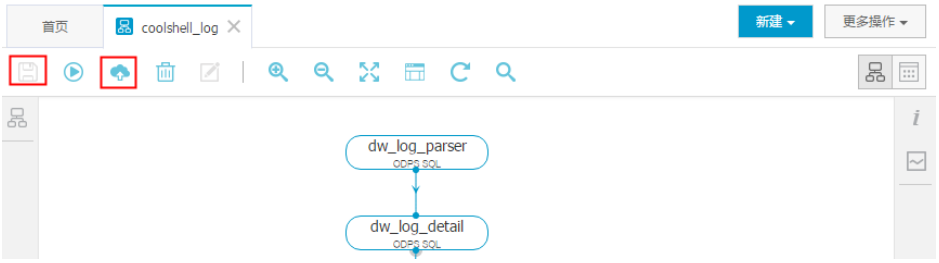
在开发面板整体视图中对分别对所有节点双击进入节点代码编辑区，输入对应的sql语句（具体sql语句请看附件），并完成对应参数配置。 几个sql节点代码没有用到自定义变量，所以参数不需要配置，默认即可。



代码和参数都配置完成，保存节点。

执行工作流节点

要执行整个工作流，需要先保存提交工作流。



工作流提交后，可以通过调度测试整个工作流所有节点运行情况。

本实验原始数据提供的是20140212一天的数据，所以在此我们业务日期选择2014-02-12。 创建冒烟工作流后可调整到运维中心查看工作流测试具体情况

双击工作流进入具体节点实例，看节点运行状态，最终确认所有表数据正常产出。

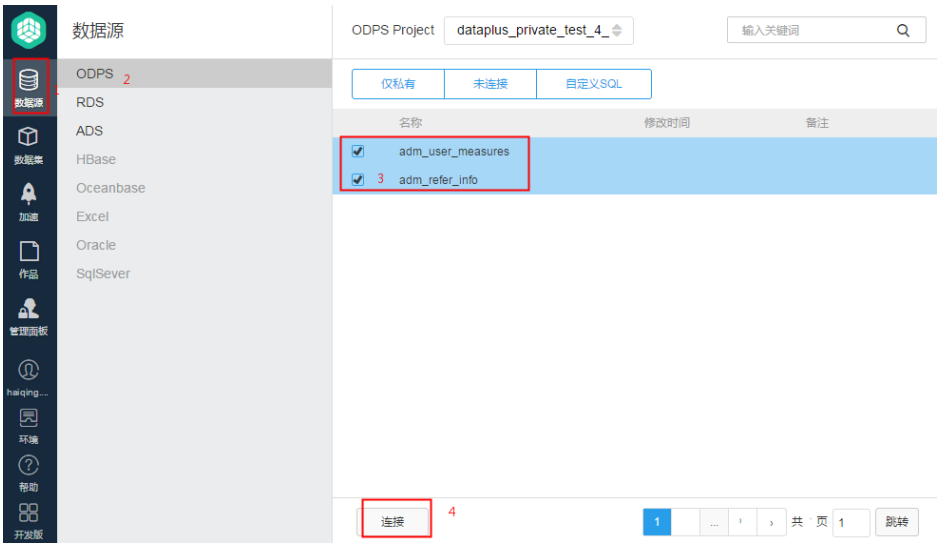
>>>下一步：BI报表制作>>>

BI报表制作

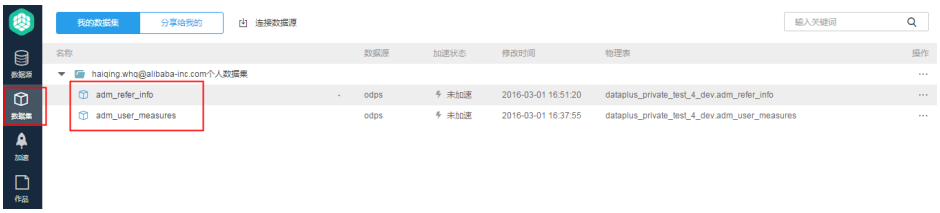
应用数据层表产出后，我们可以直接通过“BI报表”系统对数据进行报表统计，本实验我们将在BI报表上完成实验需求，产出简单的报表图。

回到数加管理控制台，左边导航点击 BI报表，选择项目“测试项目”，进入BI报表页面。

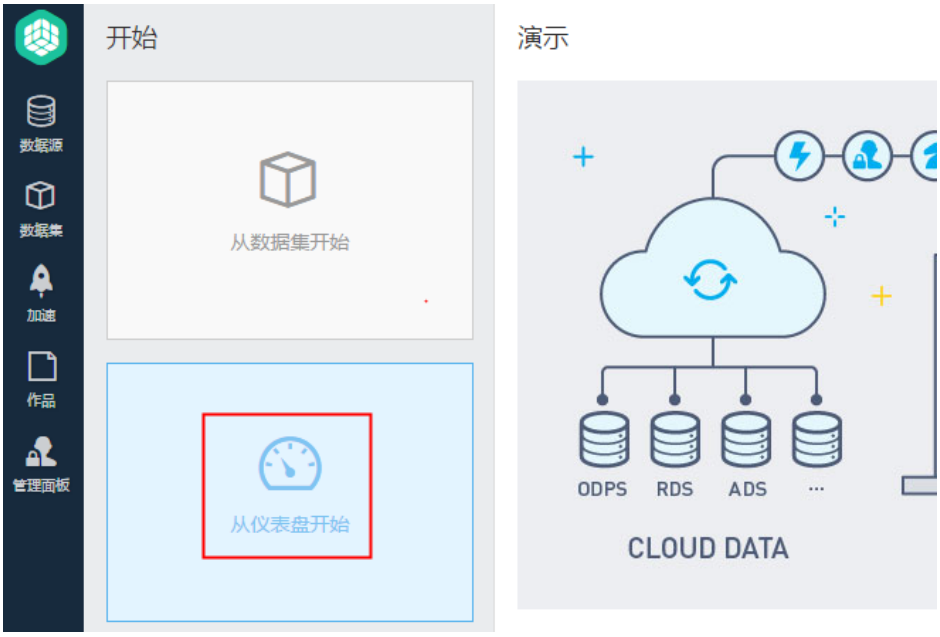
点击数据源，添加表adm_user_measures和表adm_refer_info为数据集。



添加好后如下图：

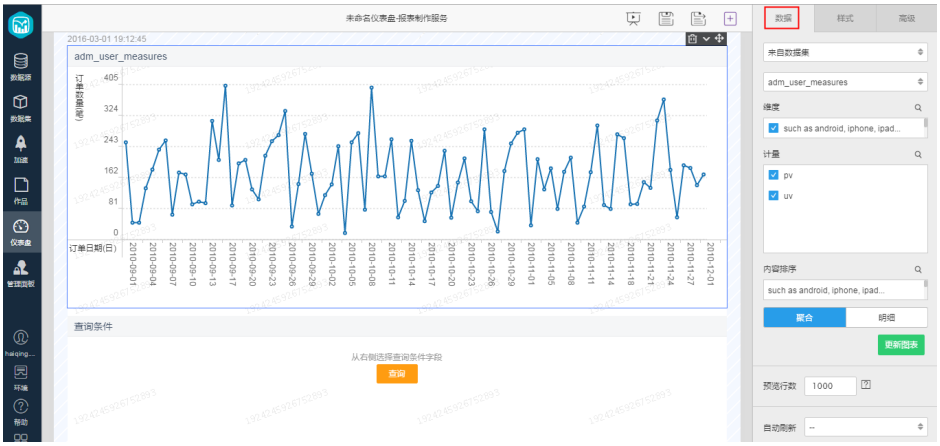


创建 coolshell网站PV/UV统计图：回到数据分析服务首页，点击从仪表盘开始>PC空白页，



选择 线图图表和查询条件控件，拖拽到工作区。

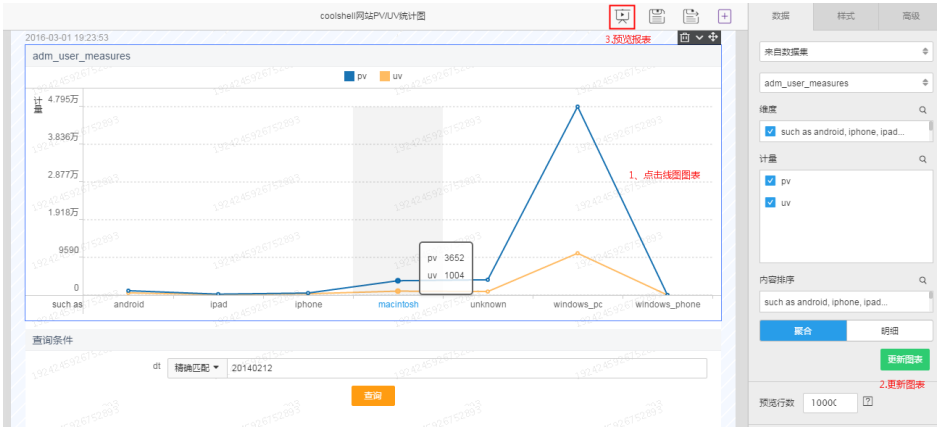
点击线图图表，右侧数据参数配置，数据源选来自数据集，选择表adm_user_measures，选择好维度和计量如图，其他配置保持默认（本实验只做简单报表制作）。



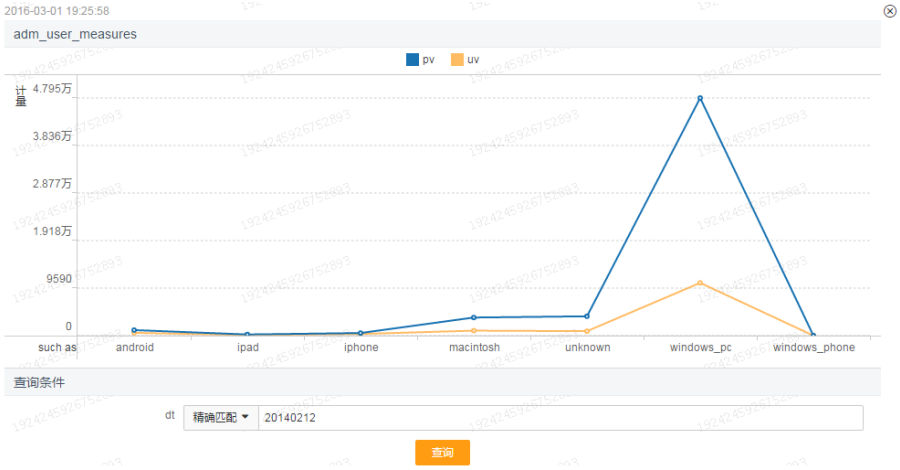
点击查询条件，右侧数据条件“请选择作用范围”选择adm_user_measures弹出的字段勾选 dt 分区字段，其他默认。

查询条件中，dt需要精确匹配，本实验中我们的表分区为20140212。

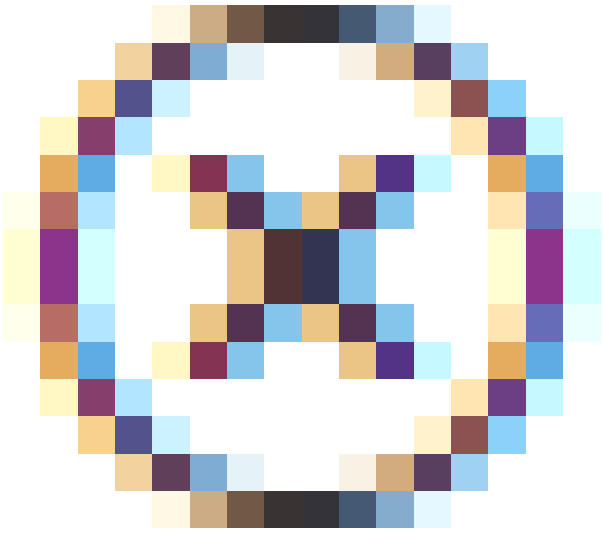
点击线图图表，右侧点击更新图表，预览报表。



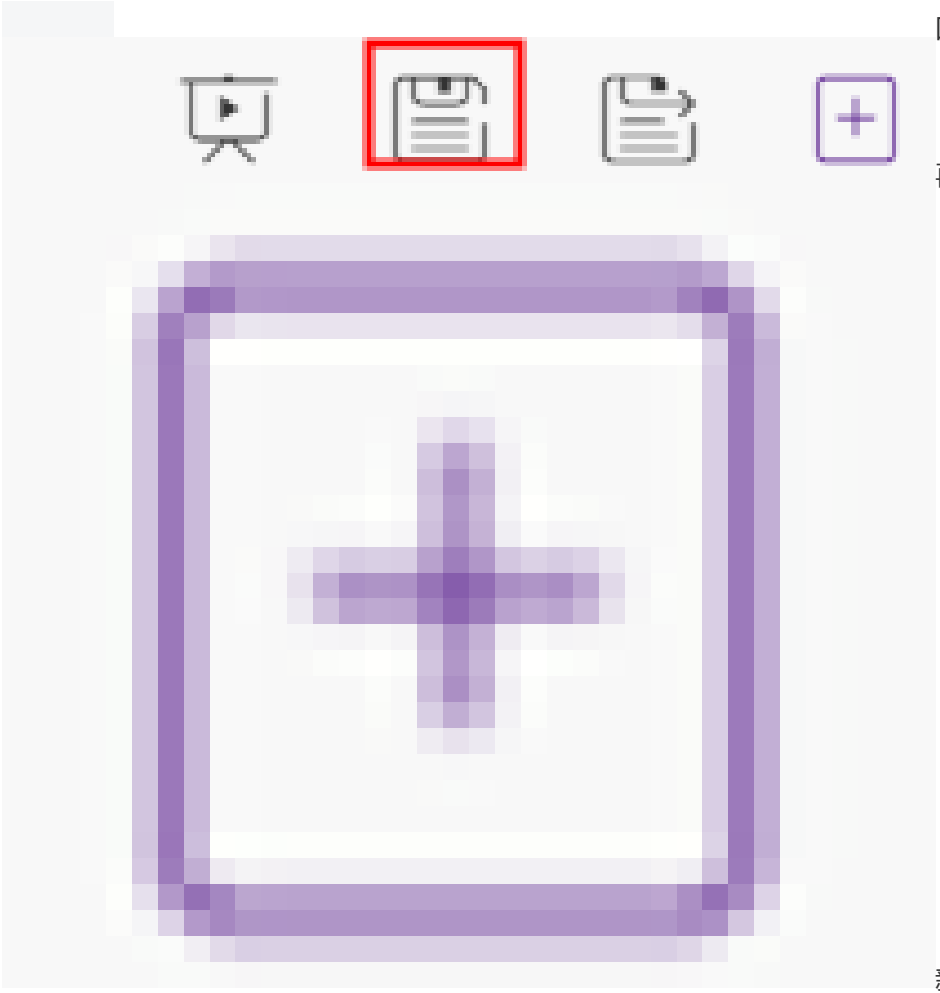
预览结果如下图



可以尝试dt输入其他分区值，如不存在的分区 20140101，看是否符合预期（没有数据），若符合点击预览页面右上角关闭按钮



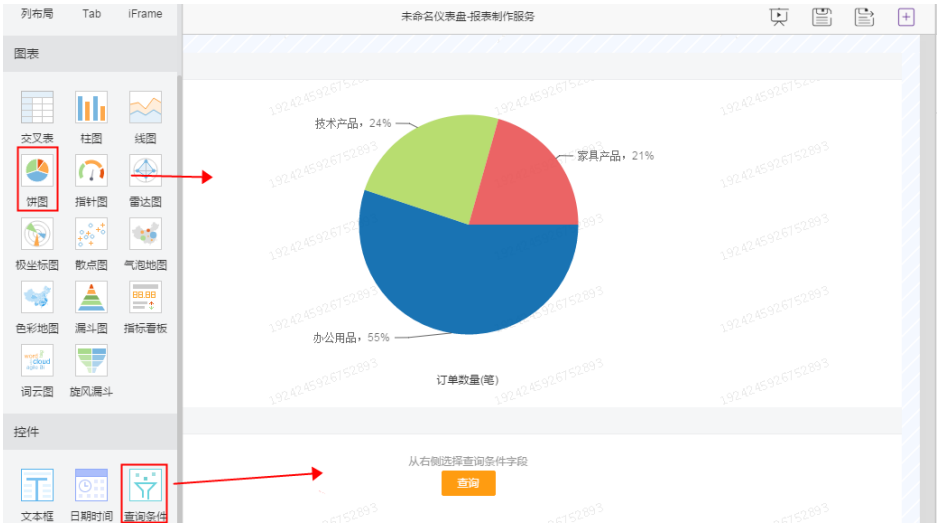
回到编辑页，保存仪表盘



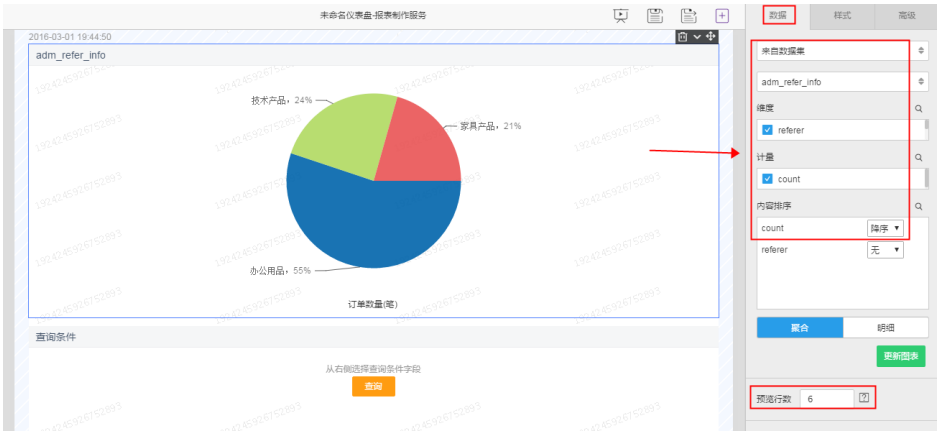
再点击新建按钮

新建空白仪表盘，选择饼图

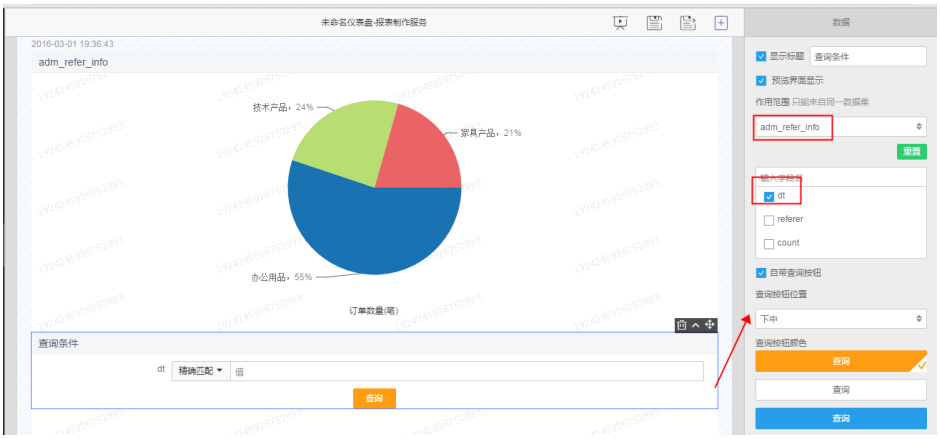
图表和查询条件控件拖拽到工作区：



单击饼图图表，右侧配置数据参数。数据源选择来自数据集，选择adm_refer_info数据集，维度选择referer，计量选择count，内容排序count选择降序，预览行数为6（此处我们只看top6），其他默认。



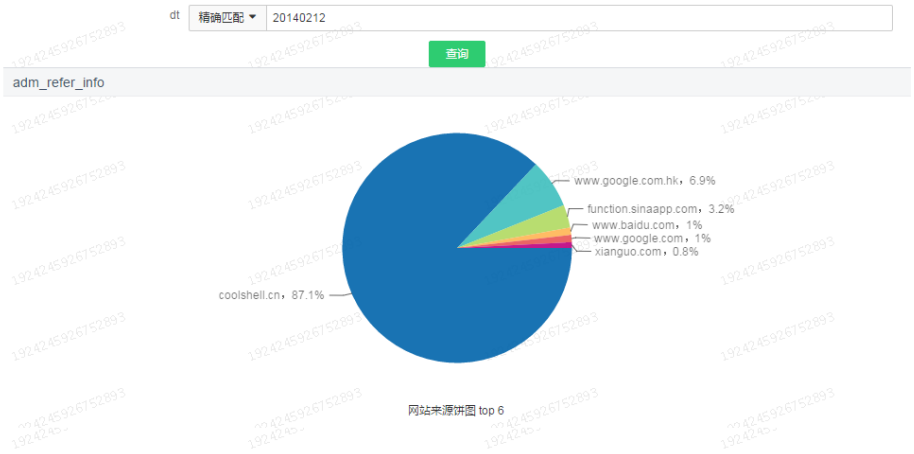
单击查询条件控件，右侧进行配置，作用范围选择adm_refer_info，弹出的字段选择分区字段 dt，其他为默认。



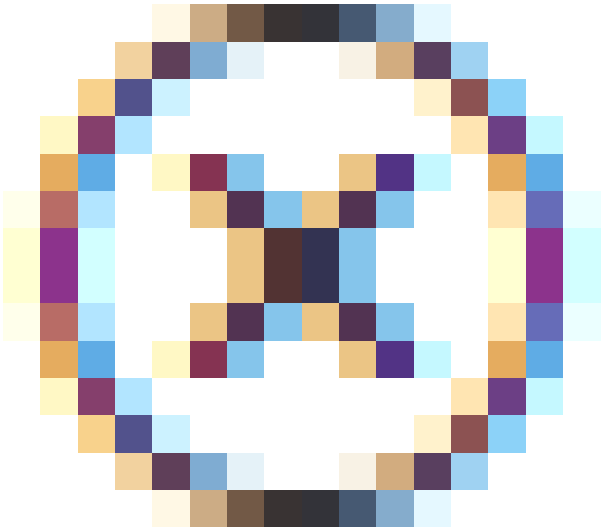
查询条件中，dt需要精确匹配，本实验中我们的表分区为20140212。

点击饼图图表，右侧点击更新图表，预览报表

预览结果如下图：



可以尝试dt输入不同分区看是否符合预期。 若符合点击预览页面右上角关闭按钮



回到编辑页，保存仪表盘


。

到此本实验两个需求报表就制作完成,后续节点每天自动调度表生成新分区，可以直接到BI报表里换新分区即可查看报表图。

附件：示例代码

```
CREATE TABLE IF NOT EXISTS ods_log_tracker(
    ip STRING COMMENT 'client ip address',
    user STRING,
    time DATETIME,
    request STRING COMMENT 'HTTP request type + requested path without args + HTTP protocol version',
    status BIGINT COMMENT 'HTTP reponse code from server',
    size BIGINT,
    referer STRING,
    agent STRING)
COMMENT 'Log from coolshell.cn'
PARTITIONED BY(dt STRING);
```

dw_log_parser建表语句

```
CREATE TABLE IF NOT EXISTS dw_log_parser(
    ip STRING COMMENT 'client ip address',
    user STRING,
    time DATETIME,
    method STRING COMMENT 'HTTP request type, such as GET POST...',
    url STRING,
    protocol STRING,
    status BIGINT COMMENT 'HTTP reponse code from server',
    size BIGINT,
    referer STRING,
    agent STRING)
PARTITIONED BY(dt STRING);
```

dw_log_detail建表语句

```
CREATE TABLE IF NOT EXISTS dw_log_detail(
    ip STRING COMMENT 'client ip address',
```

```
    time DATETIME,
    method STRING COMMENT 'HTTP request type, such as GET POST...',
    url STRING,
    protocol STRING,
    status BIGINT COMMENT 'HTTP reponse code from server',
    size BIGINT,
    referer STRING COMMENT 'referer domain',
    agent STRING,
    device STRING COMMENT 'android|iphone|ipad...',
    identity STRING  COMMENT 'identify: user, crawler, feed')
PARTITIONED BY(dt STRING);
```

dim_user_info建表语句

```
CREATE TABLE IF NOT EXISTS dim_user_info(
    uid STRING COMMENT 'unique user id',
    ip STRING COMMENT 'client ip address',
    device STRING,
    protocol STRING,
    identity STRING  COMMENT 'user, crawler, feed',
    agent STRING)
PARTITIONED BY(dt STRING);
```

dw_log_fact建表语句

```
CREATE TABLE IF NOT EXISTS dw_log_fact(
    uid STRING COMMENT 'unique user id',
    time DATETIME,
    method STRING COMMENT 'HTTP request type, such as GET POST...',
    url STRING,
    status BIGINT COMMENT 'HTTP reponse code from server',
    size BIGINT,
    referer STRING)
PARTITIONED BY(dt STRING);
```

adm_user_measures建表语句

```
CREATE TABLE IF NOT EXISTS adm_user_measures(
    device STRING COMMENT 'such as android, iphone, ipad...',
    pv BIGINT,
    uv BIGINT)
PARTITIONED BY(dt STRING);
adm_refer_info_ddl
CREATE TABLE adm_refer_info(
    referer STRING,
    count BIGINT)
PARTITIONED BY(dt STRING);
```

dw_log_parser节点代码

```
INSERT OVERWRITE TABLE dw_log_parser PARTITION (dt=${bdp.system.bizdate})
SELECT ip
    , user
    , time
    , regexp_substr(request, '(^[^ ]+ )') AS method
    , regexp_extract(request, '^[^ ]+ (.*) [^ ]+$') AS url
    , regexp_substr(request, '([^ ]+$)') AS protocol
    , status
    , size
    , referer
    , agent
FROM ods_log_tracker
WHERE dt = ${bdp.system.bizdate};
```

dw_log_detail节点代码

```
INSERT OVERWRITE TABLE dw_log_detail PARTITION (dt=${bdp.system.bizdate})
SELECT ip
    , time
    , method
    , url
    , protocol
    , status
    , size
    , regexp_extract(referer, '^[^/]+://([^/]+){1}') AS referer
    , agent
    , CASE
        WHEN TOLOWER(agent) RLIKE 'android' THEN 'android'
        WHEN TOLOWER(agent) RLIKE 'iphone' THEN 'iphone'
        WHEN TOLOWER(agent) RLIKE 'ipad' THEN 'ipad'
        WHEN TOLOWER(agent) RLIKE 'macintosh' THEN 'macintosh'
        WHEN TOLOWER(agent) RLIKE 'windows phone' THEN 'windows_phone'
        WHEN TOLOWER(agent) RLIKE 'windows' THEN 'windows_pc'
        ELSE 'unknown'
    END AS device
    , CASE
        WHEN TOLOWER(agent) RLIKE '(bot|spider|crawler|slurp)' THEN 'crawler'
        WHEN TOLOWER(agent) RLIKE 'feed'
        OR url RLIKE 'feed' THEN 'feed'
        WHEN TOLOWER(agent) NOT RLIKE '(bot|spider|crawler|feed|slurp)'
        AND agent RLIKE '^[Mozilla|Opera]'
        AND url NOT RLIKE 'feed' THEN 'user'
        ELSE 'unknown'
    END AS identity
FROM dw_log_parser
WHERE url NOT RLIKE '^[/]+wp-'
    AND dt =${bdp.system.bizdate};
```

dim_user_info节点代码

```
INSERT OVERWRITE TABLE dim_user_info PARTITION (dt=${bdp.system.bizdate})
SELECT md5(concat(t1.ip, t1.device, t1.protocol, t1.identity, t1.agent))
    , t1.ip
    , t1.device
    , t1.protocol
    , t1.identity
    , t1.agent
FROM (
    SELECT ip
        , protocol
        , agent
        , device
        , identity
    FROM dw_log_detail
    WHERE dt = ${bdp.system.bizdate}
    GROUP BY ip,
        protocol,
        agent,
        device,
        identity
) t1;
```

dw_log_fact节点代码

```
INSERT OVERWRITE TABLE dw_log_fact PARTITION (dt=${bdp.system.bizdate})
SELECT u.uid
    , d.time
    , d.method
    , d.url
    , d.status
    , d.size
    , d.referer
FROM dw_log_detail d
JOIN dim_user_info u
ON (d.ip = u.ip
    AND d.protocol = u.protocol
    AND d.agent = u.agent) and d.dt = ${bdp.system.bizdate}    AND u.dt =${bdp.system.bizdate};
```

adm_user_measures节点代码

```
INSERT OVERWRITE TABLE adm_user_measures PARTITION (dt='${bdp.system.bizdate}')
SELECT u.device
    , COUNT(*) AS pv
    , COUNT(DISTINCT u.uid) AS uv
FROM dw_log_fact f
JOIN dim_user_info u
ON f.uid = u.uid
    AND u.identity = 'user'
```

```
    AND f.dt = '${bdp.system.bizdate}'
    AND u.dt = '${bdp.system.bizdate}'
GROUP BY u.device;
```

adm_refer_info节点代码

```
INSERT OVERWRITE TABLE adm_refer_info PARTITION (dt='${bdp.system.bizdate}')
SELECT referer
    , COUNT(*) AS cnt
FROM dw_log_fact
WHERE LENGTH(referer) > 1
    AND dt = '${bdp.system.bizdate}'
GROUP BY referer;
```

FTP日志数据上传

本文将以 FTP 数据源为例，说明如何利用数据集成功能将 FTP 数据源中的日志数据上传到 DataWorks（数据工场，原大数据开发套件）中。

操作步骤

新增数据源

进入项目空间后，导航至 **数据集成 > 数据源** 页面，单击右上角的 **新增数据源**。

在新增数据源页面填写相关信息，选择数据源类型为 ftp，配置如下：

FTP 数据源配置信息如下：

数据源名称：ftp_workshop_log

数据源描述：ftp日志文件同步

数据源类型：ftp

网络类型：经典网络

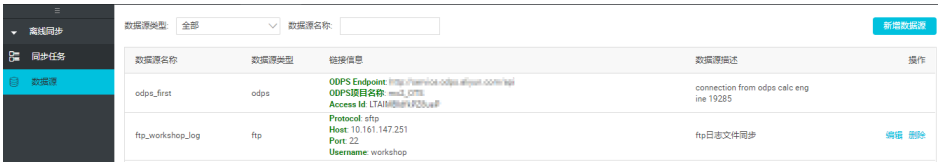
Protocol：sftp

Host：10.80.177.33

Port：22

用户名/密码：workshop/workshop

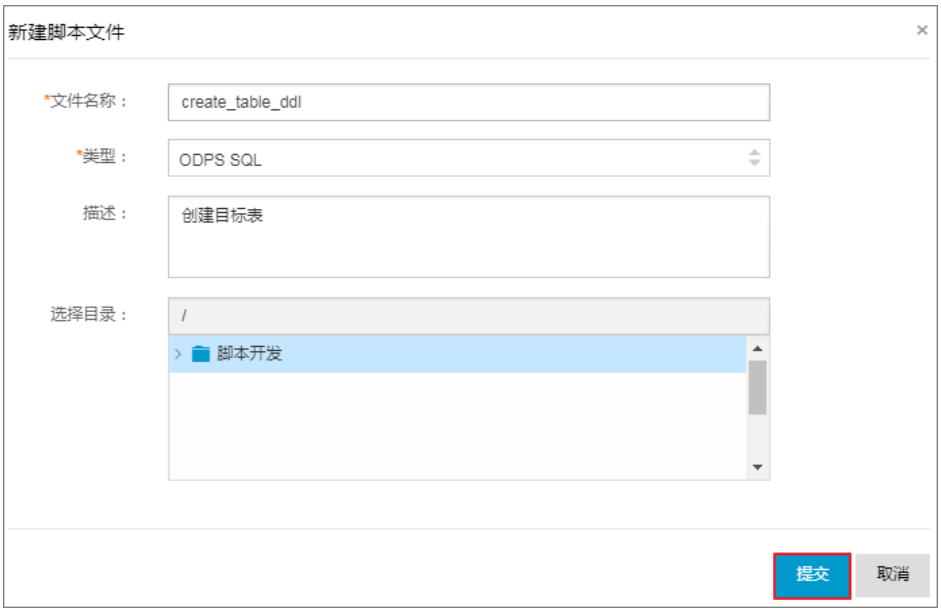
单击 **测试连通性**，如果测试成功，单击 **确定**，即成功新增数据源。



创建目标表

单击顶部导航栏中的 **数据开发**，进入数据开发首页后单击 **新建 > 新建脚本文件** 或 **新建脚本**。

配置新建脚本文件弹出框中的相关信息，填写文件名称，选择类型为 **ODPS SQL** 后，单击**提交**。如下图所示：

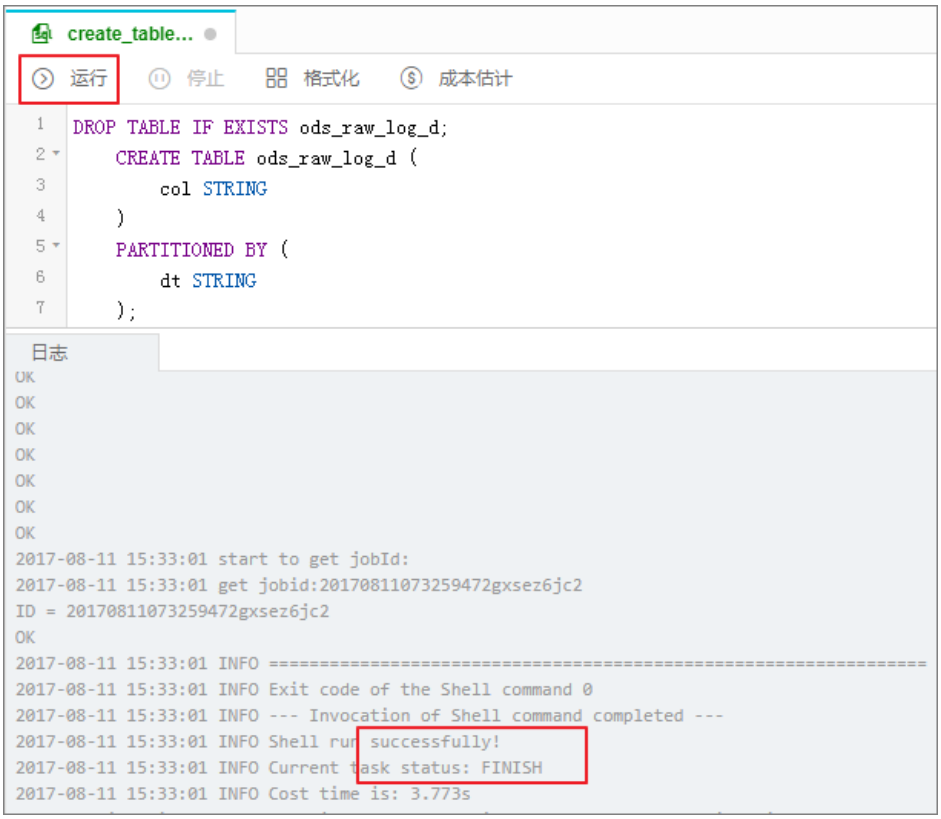


输入创建 FTP 日志对应目标表的语句，如下所示：

```
 DROP TABLE IF EXISTS ods_raw_log_d;
CREATE TABLE ods_raw_log_d (
col STRING
)
```

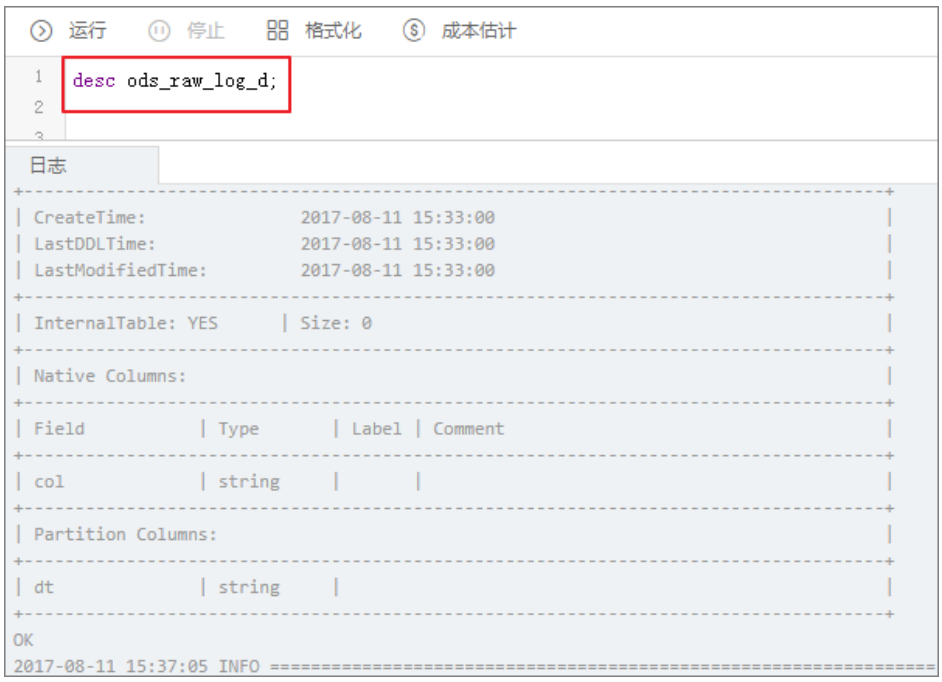
```
PARTITIONED BY (
dt STRING
);
```

单击 **运行**，直至日志信息返回成功表示目标表创建成功。



注意：

可以使用 desc 语法来确认创建表是否成功。

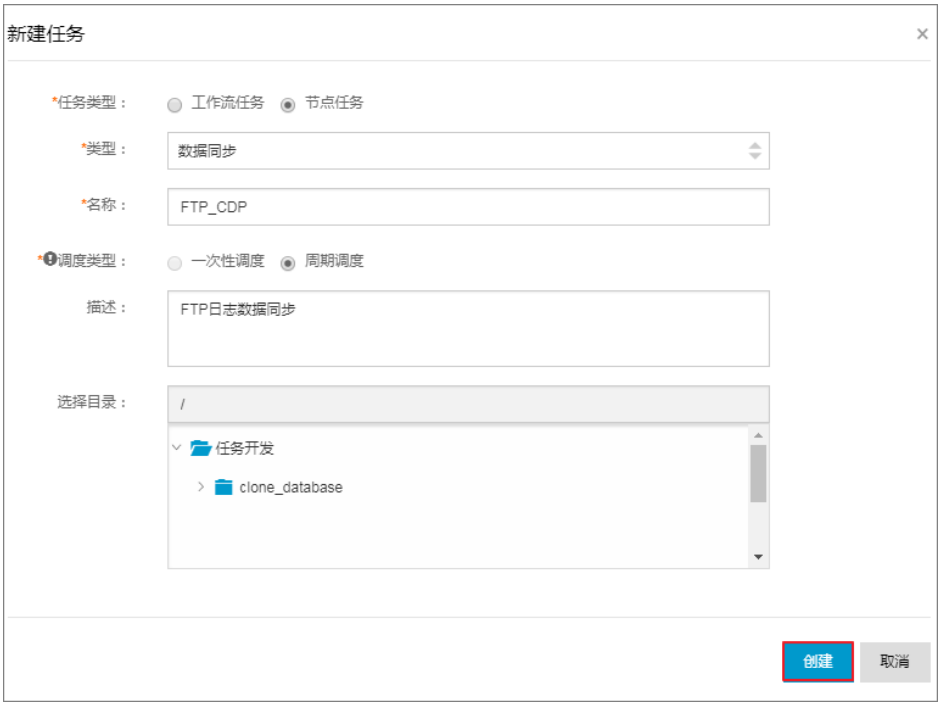


单击 **保存**，保存输入的 SQL 建表语句。

新建数据同步任务

单击 **新建** 并选择 **新建任务**。

配置新建的节点任务，单击 **创建**。配置项如下图所示：



配置数据同步任务

进入节点配置页面，选择来源。如下图所示：

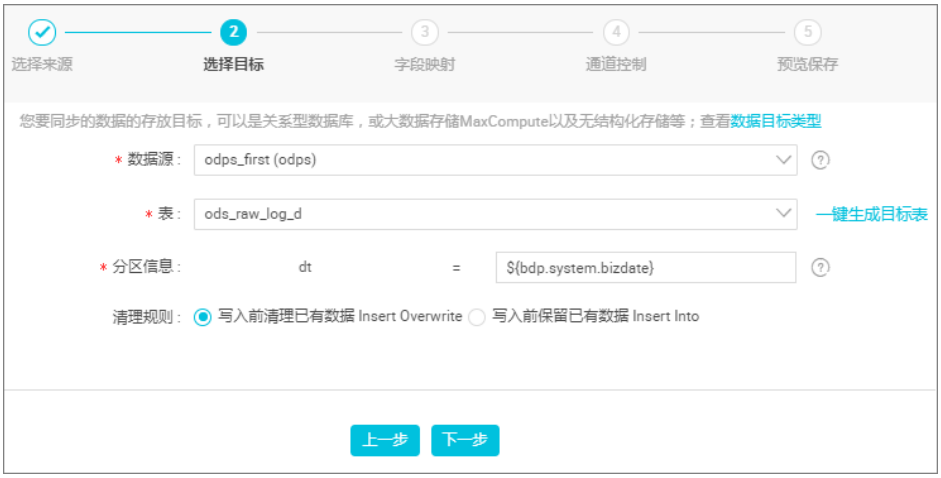
数据来源配置项说明：

数据源：选择已创建好的 ftp 数据源。

文件路径：/home/workshop/user_log.txt

列分隔符：|

单击 **下一步**，选择目标。如下图所示：



数据目标配置项说明：

数据源：数据存放目标源选择 odps_first。

表：数据存放目标表选择 ods_raw_log_d。

分区信息：${bdp.system.bizdate}。

清理规则：写入前清理已有数据。

单击 **下一步**，连接要同步的数据，配置字段映射。如下图所示：

单击 **下一步**，配置通道控制，作业速率上限为 10MB/s。如下图所示：

单击 **下一步**，进入预览保存页面中预览上述的配置情况，也可以进行修改，确认无误后，单击 **保存**。

提交数据同步任务

单击 **提交**，提交已经配置的数据同步任务。

在 **提交新版本** 弹出框中单击 **确认提交**，即可将数据同步任务提交到调度系统中。

测试运行数据同步任务

单击工具栏中的 **测试运行**。

在 **周期运行任务** 弹出框中单击 **确定**。

在 **测试运行** 弹出框中，实例名称和业务日期都保持默认，单击**运行**。

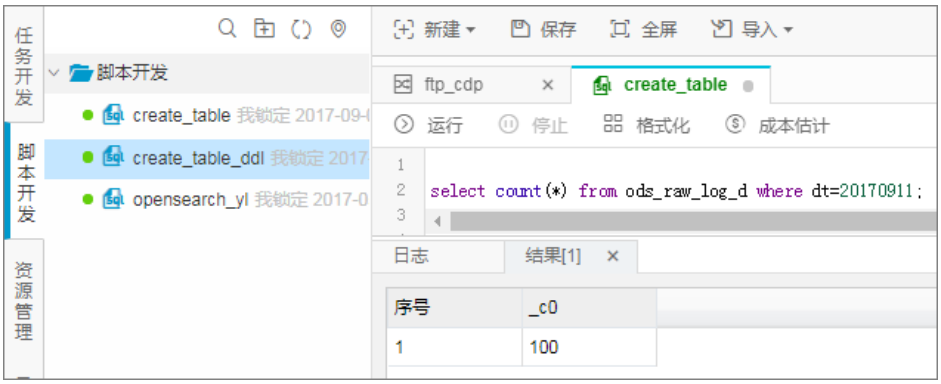
在 **工作流任务测试运行** 弹出框中单击 **前往运维中心**。

在运维中心即可查看实例运行状态，如下图所示：

确认数据是否成功导入 MaxCompute

返回到 create_table_ddl 脚本文件中。

编写并执行 SQL 语句查看导入 ods_raw_log_d 的记录数。



SQL 语句如下，其中分区键需要更新为业务日期，如测试运行任务的日期为 20170712，那么业务日期为 20170711。

```
---查看是否成功写入MaxCompute
select count(*) from ods_raw_log_d where dt=业务日期;
```

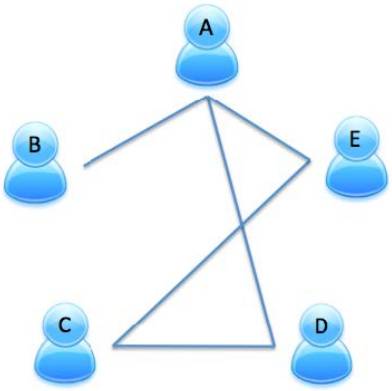
通过MR实现好友推荐

社交网络是现如今影响力巨大的信息平台，社交网站中，您可以通过 **可能感兴趣的人** 途径增加交友方式。**可能**

感兴趣的人 也称作 **好友推荐**， 它主要是通过查找两个非好友之间的共同好友情况来实现的。本文将通过一个示例，简单介绍如何通过 MapReduce 的方式实现好友推荐功能。

实验介绍

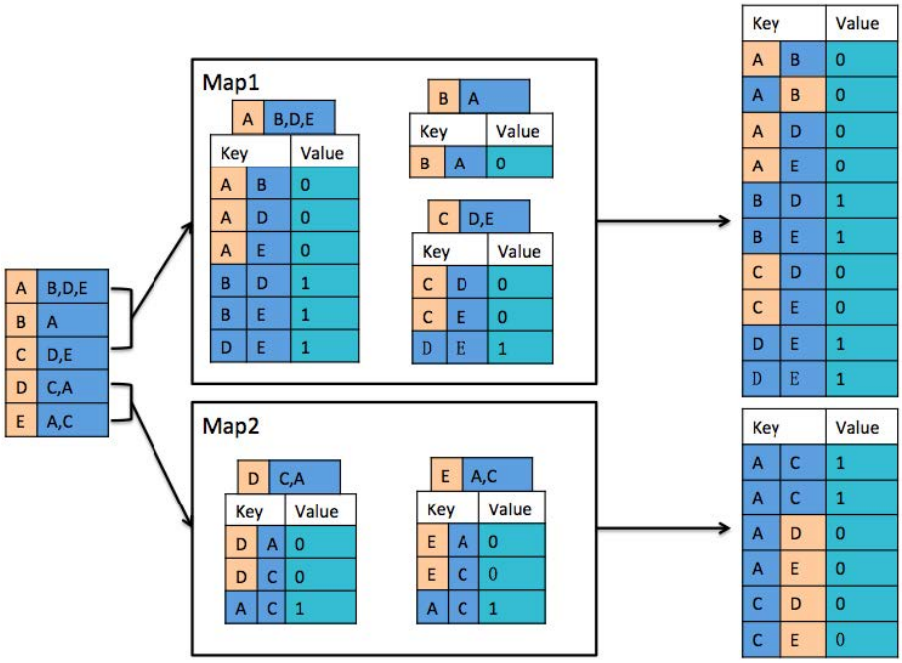
A，B，C，D，E 五个人的好友关系如下图所示，其中实线表示互为好友关系。那么，如何获取两个不是好友的两个人之间的好友数，并以此为参考，向用户推荐陌生人呢？



User	Friend
A	B,D,E
B	A
C	D,E
D	C,A
E	A,C

主要通过以下几个步骤实现：

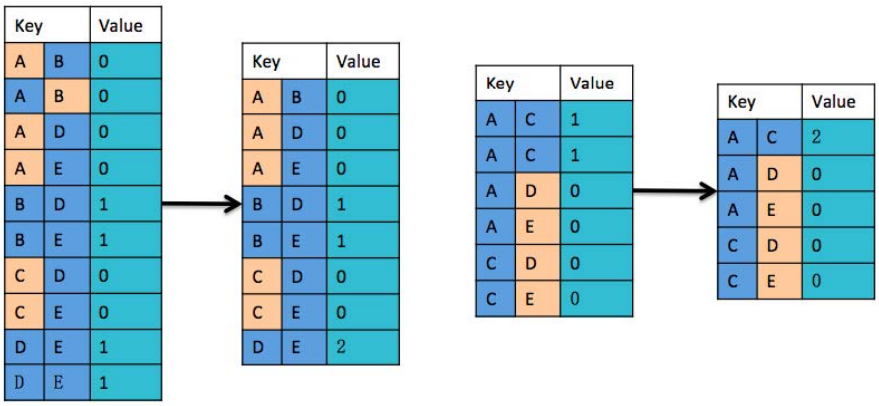
将好友关系分配到两个 Map 进行处理，其中每个 Map 包含 3 条好友关系。对每一条好友关系进行拆分，若 Key 中的两个人为朋友，则记录 value 值为0，否则 value 值为 1。将拆分的结果进行排序，其中（A B）和（B A）作为同一个 key（A B）。



分别对两个 Map 处理的记录进行初步合并，若两个记录的 Key 值相同且每条记录的 Value 都不为 0，则 Value 值加 1。

注意：

在 Combine 阶段，必须保留 Value 为 0 的记录，否则，在 Reduce 阶段，获取的结果会出错。

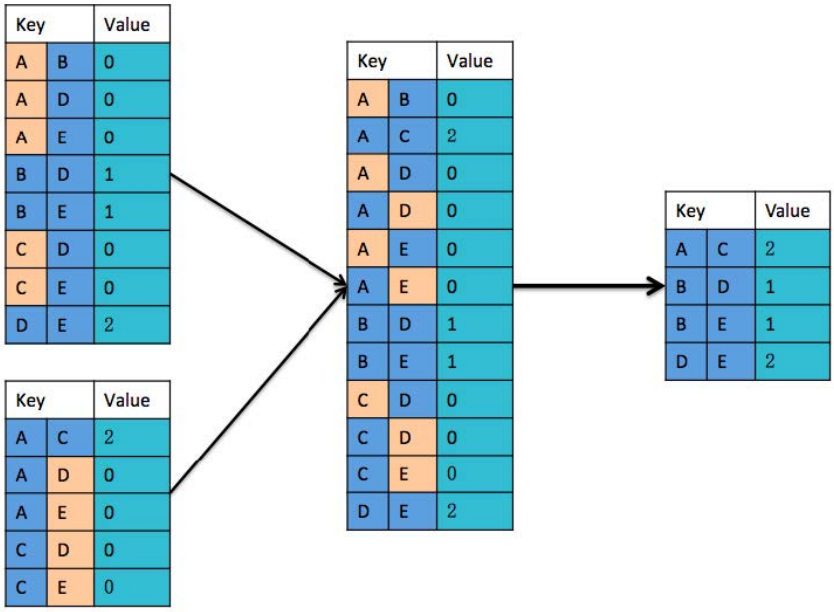


通过 Reduce 方式，合并两个 Map 处理的 Combine 结果。

若两个记录的 Key 值相同且每条记录的 Value 都不为 0，则 Value 值加 1。

将 Value 值为 0 的记录删除。

获取不为好友的两个用户之间的公共好友数：Key 为两个不为好友的用户，Value 是两个不是好友的用户之间的共同好友数。社交网站或者 APP 可以根据这个数值对不是好友的两个用户进行推荐。



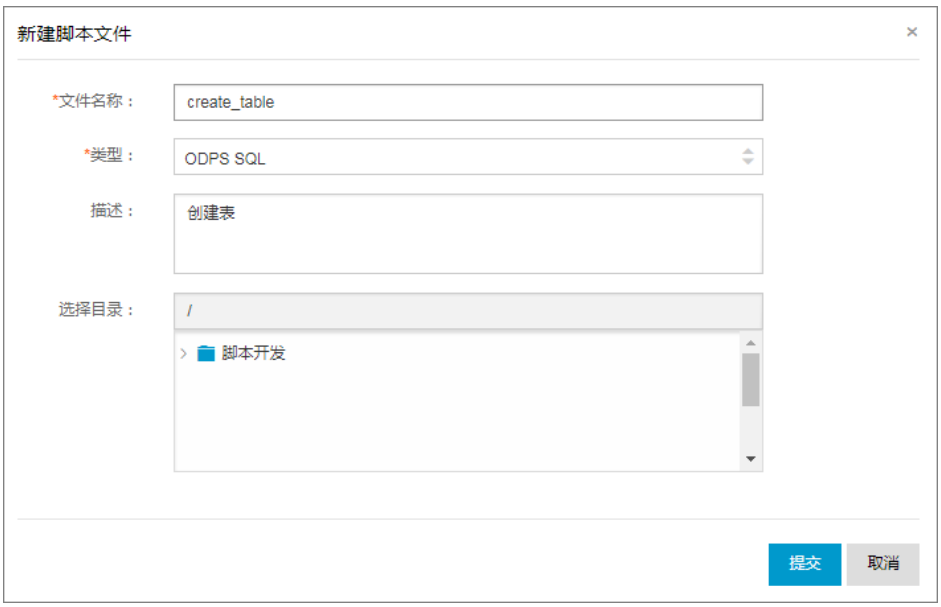
操作步骤

新建数据表

登录 DataWorks 管理控制台，单击相应项目空间后的 **进入工作区**。

单击顶部导航栏中的 **数据开发**，进入数据开发首页后单击 **新建** > **新建脚本文件** 或 **新建脚本**。

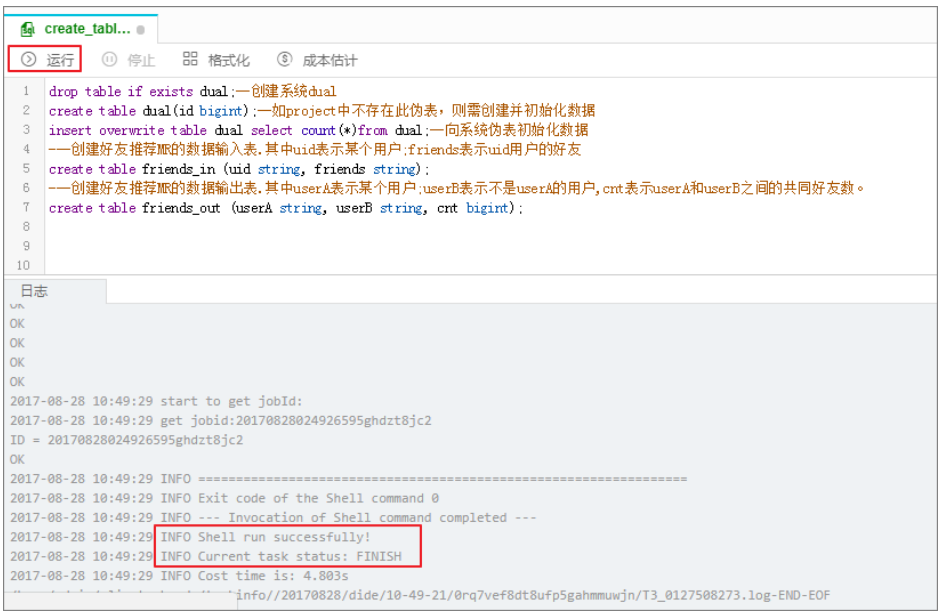
配置新建脚本文件弹出框中的相关信息，填写文件名称，选择类型为 **ODPS SQL** 后，单击**提交**。如下图所示：



输入建表语句，如下所示：

```
drop table if exists dual;--创建系统dual
create table dual(id bigint);--如project中不存在此伪表，则需创建并初始化数据
insert overwrite table dual select count(*)from dual;--向系统伪表初始化数据
---创建好友推荐MR的数据输入表.其中uid表示某个用户;friends表示uid用户的好友
create table friends_in (uid string, friends string);
---创建好友推荐MR的数据输出表.其中userA表示某个用户;userB表示不是userA的用户,cnt表示userA和userB之间的共同好友数。
create table friends_out (userA string, userB string, cnt bigint);
```

单击 **运行**，直至日志信息返回成功表示目标表创建成功。



单击 **保存**，保存输入的 SQL 建表语句。

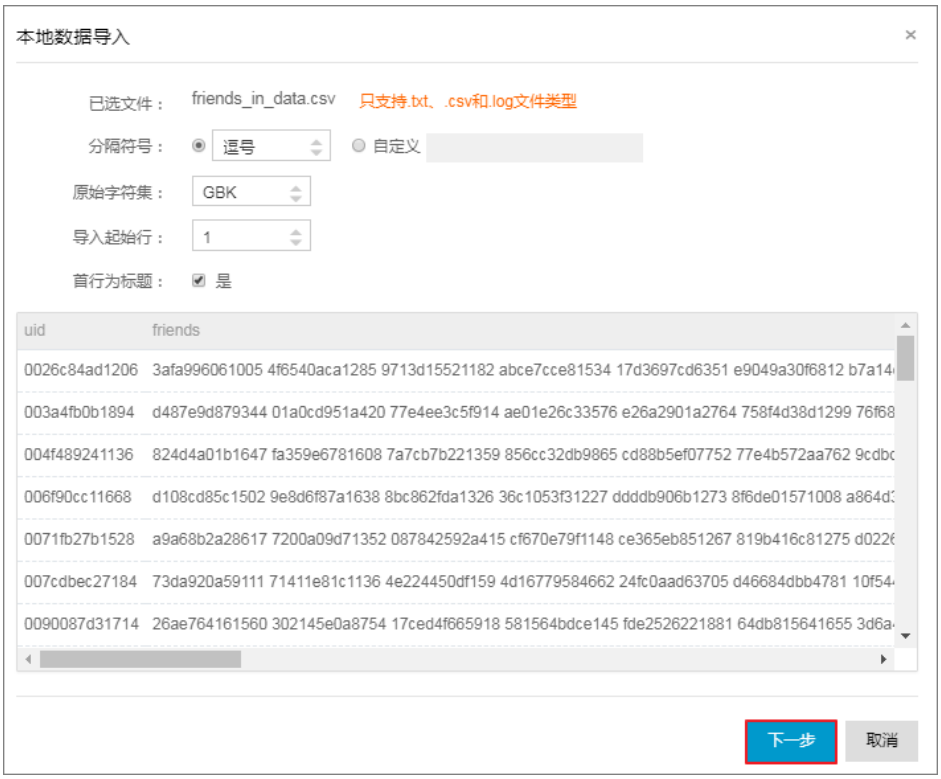
导入本地数据

单击顶部功能栏中的 **导入** > **导入本地数据**，打开本地保存的文件 friends_in_data.csv（**点此下载**）。

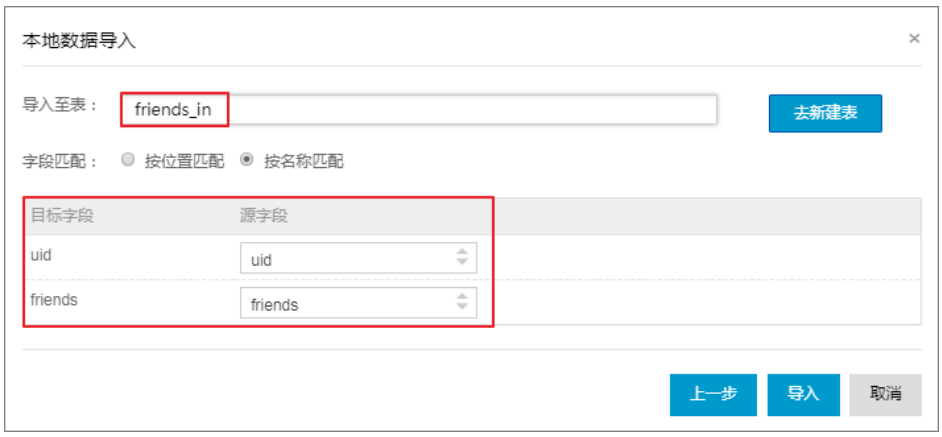
所有配置均设为默认，并查看导入的数据。完成后，单击 **下一步**。

注意：

在真实的工作环境中，数据必须以txt或csv的文件类型导入。

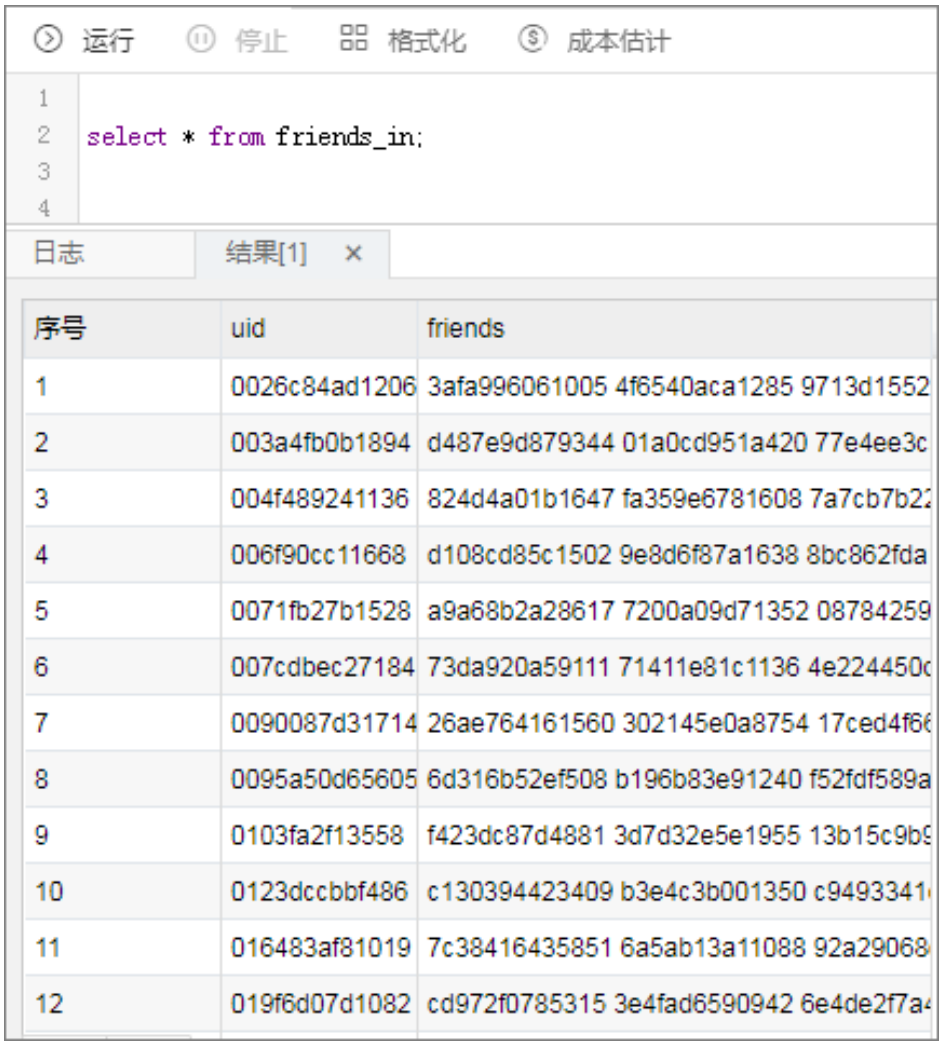


在本地数据导入页面的 **导入至表** 中，输入 friends_in，即将本次实验的测试数据，导入到好友推荐的输入表 friends_in 中，确定 **目标字段** 与 **源字段** 匹配。完成后单击 **导入**。



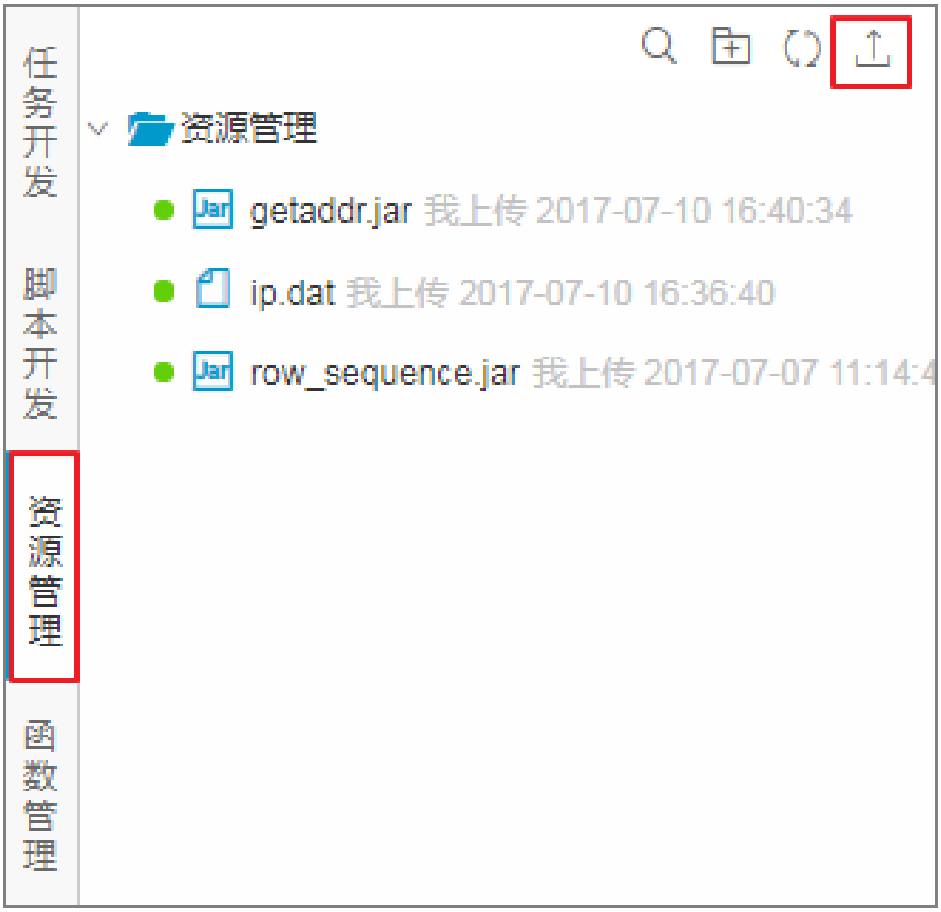
由于数据量较大，请等待1-2分钟。

数据导入完成后，可输入语句进行查询、确认。如下图所示：

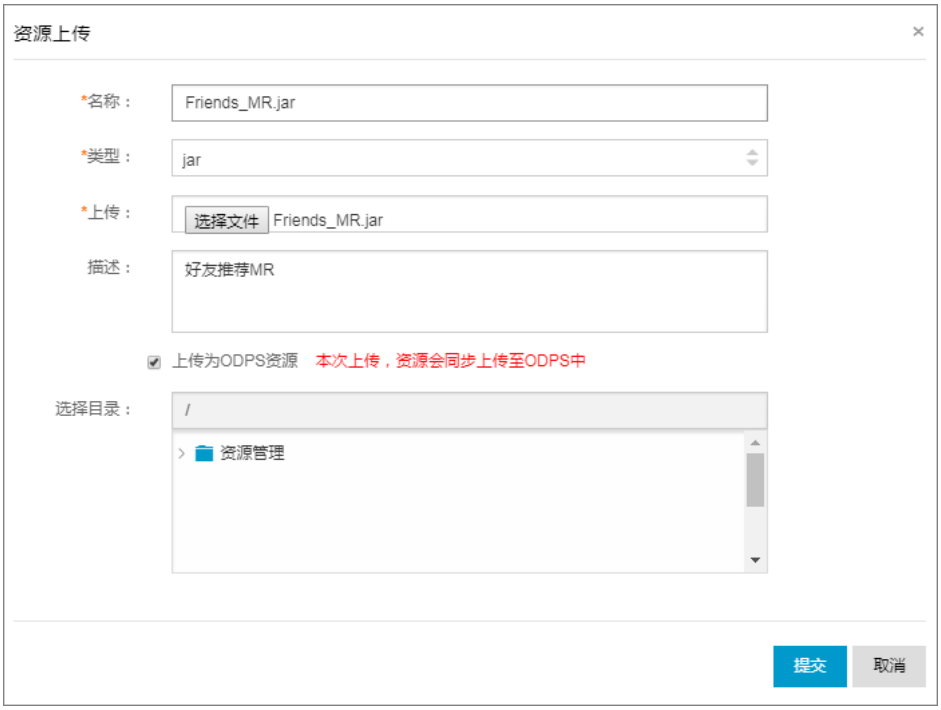


添加 MR 资源

单击左侧导航栏中的 **资源管理**，单击列表右上角的 **上传资源**。



配置资源上传弹出框中的信息，选择需要上传的文件 Friends_MR。如下图所示：



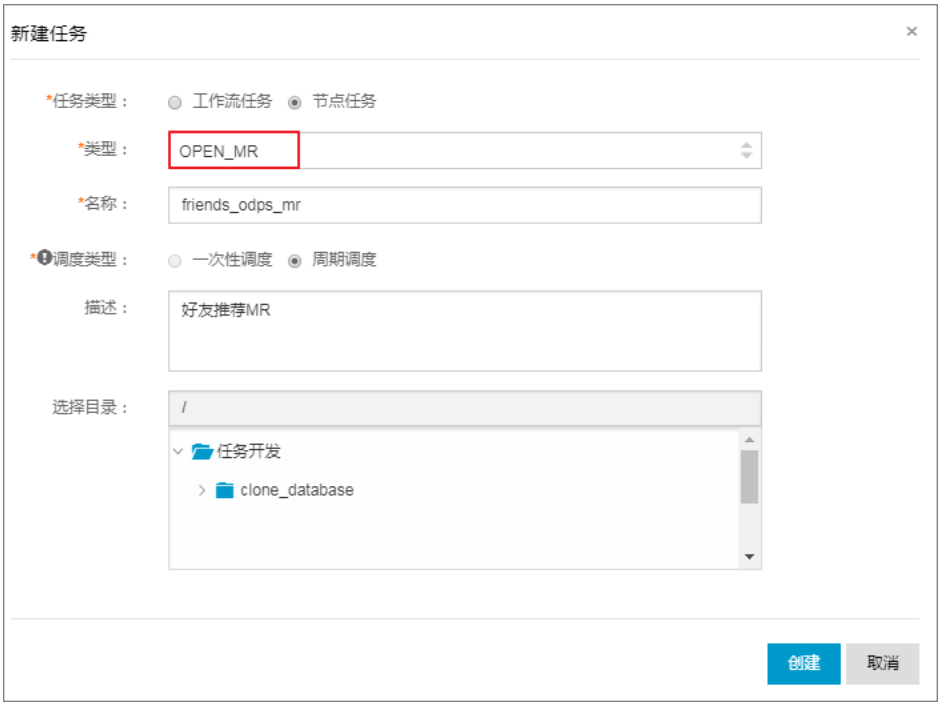
单击 **提交**。

在左侧导航栏的 **资源管理** 下，即可看到上传成功的 Jar 包 friends_mr.jar。

测试并验证好友推荐

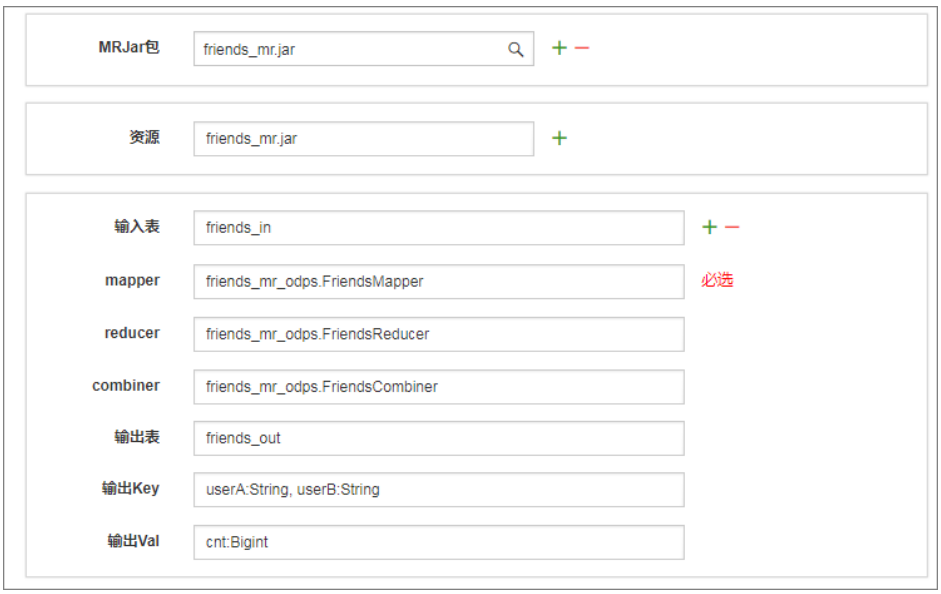
单击顶部导航栏中的 **新建** > **新建任务**，开始创建本次实验的 MR 任务。

在弹出的对话框中，选择新建任务的 **任务类型** 为 **节点任务**，配置如下图所示：



单击 **创建**。

在任务页面中输入各配置信息，如下图所示：



配置项说明：

- MRJar 包：单击文本框，选择 friends_mr.jar。

  资源：默认设置为 friends_mr.jar。

  输入表：输入 friends_in。

  mapper：输入 friends_mr_odps.FriendsMapper，此为 Jar 包中 Mapper 的 class 全名。

  reducer：输入 friends_mr_odps.FriendsReducer，此为 Jar 包中 Reducer 的 class 全名。

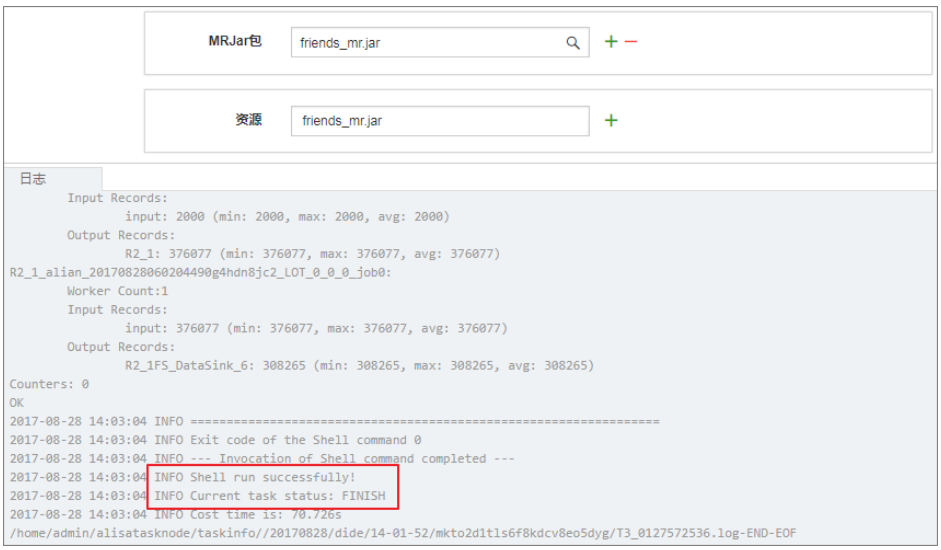
  combiner：输入 friends_mr_odps.FriendsCombiner，此为 Jar 包中 Combiner 的 class 全名。

  输出表：输入 friends_out。

  输出 Key：输入 userA:String，userB:String。

  输出 Val：输入 cnt:Bigint。

保存 并 **运行** 配置的 OPEN MR 任务，可在底部的 **日志** 中，查看运行状态和运行结果。如下图所示：



在脚本文件中输入如下的 SQL 命令，并单击 **运行**，查询共同好友超过 2 个的数据信息。

```
SELECT * FROM friends_out WHERE cnt>2 order by cnt desc limit 100;
```

日志 结果[1]

序号	usera	userb	cnt
1	0a46b354f4538	7955ee2e82985	5
2	9aa5c6a21c794	a0a407dca1360	5
3	072698a972386	777830fd25726	5
4	3f1c568b25585	a441354dfc773	4
5	2847433fb8376	fbf8a5facb295	4
6	cf1d008ac1921	cfb1bd78af546	4
7	13d57cdbce661	af48ca8831531	4
8	a9a68b2a28617	cf670e79f1148	4
9	581564bdce145	85f2f762b1221	4
10	36104d9311265	3d24c66cf1512	4
11	57b3be71a1525	6a5ab13a11088	4
12	0569c5b231912	31683e78ed838	4
13	0ee2ec1dd5638	ca29d06d81882	4
14	281f4a52ee598	937858c768216	4

统计分析网站数据

示例说明

示例背景

本示例主要介绍如何通过数加 MaxCompute + DataWorks 两个产品实现简单的网站数据统计分析。

您通过本示例可快速上手 MaxCompute 进行大数据开发，简单了解在 MaxCompute 做大数据 ETL 的过程，同时了解一些 MaxCompute SQL 和常用数据库 SQL 的基本区别。

适用人群

MaxCompute 初学者，特别是无大数据开发基础但有数据库使用基础者。

示例介绍

房产网上经常会看到一些排行榜，如最近 30 日签约的楼盘排行、签约金额的楼盘排行等，本示例将简单介绍通过对二手房产数据信息表（house_basic_info）的统计分析，得出每个城市二手房均价 Top 5 的楼盘，并且给出该楼盘所在城区，最后让这些数据能够在房产网上呈现。

需求分析

核心目标

统计分析出每个城市二手房均价 Top 5 的楼盘，并且给出该楼盘所在城区，即（城市、楼盘、均价、排名和所在城区）。

数据现状

> 信息表中，每个楼盘可能有多条记录，多个均价信息，本示例只针对整个楼盘的均价求平均。
>
> 信息表中，house_region 中包含城区、街道地址信息，需要拆分出城区信息。

每天数据都有变化，每个数据日期的数据都是全量数据。

操作步骤

步骤1：准备数据

步骤2：配置 RDS 数据源

步骤3：配置数据同步任务

步骤4：执行数据导入任务

步骤5：数据统计分析

步骤6：数据回流

数据回流是指：将结果表回流到网站业务系统，以便网站直接调用数据进行前端显示。

总结

通过后续示例中对数据统计分析的实现，您可以了解到以下内容：

DataWorks（数据工场，原大数据开发套件）是架构在 MaxCompute 的 web 工具，提供界面操作以及数据集成和任务调度功能，而 MaxCompute 提供计算和存储服务。

MaxCompute SQL 作业提交后会有几十秒到数分钟不等的排队调度，所以适合处理跑批作业，一次作业批量处理海量数据，不适合直接对接需要每秒处理几千至数万笔事务的前台业务系统。

MaxCompute SQL 采用的是类似于 SQL 的语法，可以看作是标准 SQL 的子集，但不能因此简单的把 MaxCompute 等价成一个数据库，它在很多方面并不具备数据库的特征，如事务、主键约束、索引等都不支持，更多差异请参见 与其他 SQL 的语法差异。

DataWorks（数据工场）中的数据同步可以实现跨 region 的 RDS 与 MaxCompute 的数据互传，无需特殊处理。

更多的高级功能组件（MapReduce、Graph 等），请参见 MaxCompute 相关文档。

步骤1：数据准备

本示例中的数据为二手房网产品数据信息表 house_basic_info，存储于 RDS-MySQL（区域：阿里云华南 1 可

用区 A，网络为专有网络），表数据每天全量更新。

注意：

您可以通过 数加平台公开数据集-二手房产数据集 直接使用 **二手房网产品数据信息表** ，不过数据量可能与本示例呈现的不完全一致。

数据说明如下：

字段	字段类型	字段说明
house_id	varchar	房产 ID
house_city	varchar	房产所在城市
house_total_price	Double	房产总价
house_unit_price	Double	房产均价
house_type	varchar	房产类型
house_floor	varchar	房产楼层
house_direction	varchar	房产方向
house_deckoration	varchar	房产装修
house_area	Double	房产面积
house_community_name	varchar	房产所在小区
house_region	varchar	房产所在地区
proj_name	varchar	楼盘名称
proj_addr	varchar	项目地址
period	int	产权年限
property	varchar	物业公司
greening_rate	varchar	绿化率
property_costs	varchar	物业费用
datetime	varchar	数据日期

数据样例（英文逗号分隔）：

```
000404705c6add1dc08e54ba10720698,beijing,8000000,72717,3室1厅,低楼层/共24层,南,平层/精装,137,玺萌丽苑,丰台草桥 三至四环,null,null,null,null,null,null,20170605
```

RDS-MySQL 上 house_basic_info 表的建表语句，如下所示：

```
CREATE TABLE `house_basic_info` (
`house_id` varchar(1024) NOT NULL COMMENT '房产 ID',
```

```
`house_city` varchar(1024) NULL COMMENT '房产所在城市',
`house_total_price` double NULL COMMENT '房产总价',
`house_unit_price` double NULL COMMENT '房产均价',
`house_type` varchar(1024) NULL COMMENT '房产类型',
`house_floor` varchar(1024) NULL COMMENT '房产楼层',
`house_direction` varchar(1024) NULL COMMENT '房产方向',
`house_deckoration` varchar(512) NULL COMMENT '房产装修',
`house_area` double NULL COMMENT '房产面积',
`house_community_name` varchar(1024) NULL COMMENT '房产所在小区',
`house_region` varchar(1024) NULL COMMENT '房产所在地区',
`proj_name` varchar(1024) NULL,
`proj_addr` varchar(1024) NULL,
`period` int(11) NULL,
`property` varchar(1024) NULL,
`greening_rate` varchar(1024) NULL,
`property_costs` varchar(1024) NULL,
`datetime` varchar(512) NULL COMMENT '数据日期'
) ENGINE=InnoDB
DEFAULT CHARACTER SET=utf8 COLLATE=utf8_general_ci
COMMENT='二手房网产品数据信息表';
```

后续步骤

现在，您已经对实验所需的数据做了一定的准备和了解，您可以继续学习下一个教程。在该教程中您将学习如何配置实验所需的 RDS 数据源。详情请参见 **配置 RDS 数据源。**

步骤2：配置RDS数据源

前提条件

因 RDS 数据安全的限制，DataWorks（数据工场，原大数据开发套件）的数据同步任务要与 RDS 数据库进行连通，必须将执行数据同步任务的机器 IP 添加到 RDS 的白名单中，详情请参见 **IP 白名单**，您也可通过配置数据源界面中的 IP 查看入口进行查看。

操作步骤

以开发者身份进入 DataWorks **管理控制台**，单击对应项目操作栏中的 **进入工作区。**

单击顶部菜单栏中的 **数据集成**，导航至 **数据源** 页面。

单击 **新增数据源。**

在新建数据源弹出框中，选择数据源类型为 RDS > MySQL。

选择以 RDS 实例形配置该 MySQL 数据源。

查看 RDS > MySQL 中的实例 ID，如下图所示：

单击 **测试连通性。**

测试连通性通过后，单击 **确定。**

注意：

本示例中 RDS 实例所在区域为华南 1，网络类型为专有网络，通过 DataWorks 进行数据同步时，属于跨 region 走专有网络方式导数据。

DataWorks 的数据集成针对 RDS 通过反向代理自动检测使得网络能够互通，无需其他特殊处理即可保证数据同步正常连通。

后续步骤

现在，您已经学习了如何配置 RDS 数据源，您可以继续学习下一个教程。在该教程中您将学习如何通过创建同步任务来把 RDS 数据导入到 MaxCompute 中。详情请参见 **配置数据同步任务**。

步骤3：配置数据同步任务

根据前文的操作，您已经成功 **配置 RDS 数据源**，本文将为您介绍如何配置数据同步任务，以将 RDS 数据源中的数据同步至 MaxCompute 中。

操作步骤

以开发者身份进入 DataWorks **管理控制台**，单击对应项目操作栏中的 **进入工作区**。

单击顶部菜单栏中的 **数据集成**，导航至 **数据同步** 页面。

单击 **向导模式**，新建一个同步任务。

选择来源。

选择 mysql 数据源及源头表 hw_test，然后单击 **下一步**，如下图所示：

表每天全量更新，每次统计数据时，只需统计数据日期为昨天完整一天数据。因此数据过滤时，每天自动调度取 datatime 为昨天的日期，可以使用系统参数 **${bdp.system.bizdate}** 代替，使任务每天

调度执行自动替换字段值，系统参数详情请参见 **系统调度参数**。

选择目标。

本示例是将数据导入到 MaxCompute 项目中，所以目标选择默认的数据源 odps_first(odps)，此时并未创建目标表，所以需要单击 **快速建表** 来创建目标表。更多建表方式请参见 **创建表**。

快速建表弹框中显示系统自动根据源表结构生成的对应 MaxCompute 建表语句：

```
CREATE TABLE IF NOT EXISTS  your_table_name  (
house_id STRING COMMENT '*',
house_city STRING COMMENT '*',
house_total_price DOUBLE COMMENT '*',
house_unit_price DOUBLE COMMENT '*',
house_type STRING COMMENT '*',
house_floor STRING COMMENT '*',
house_direction STRING COMMENT '*',
house_deckoration STRING COMMENT '*',
house_area DOUBLE COMMENT '*',
house_community_name STRING COMMENT '*',
house_region STRING COMMENT '*',
proj_name STRING COMMENT '*',
proj_addr STRING COMMENT '*',
period BIGINT COMMENT '*',
property STRING COMMENT '*',
greening_rate STRING COMMENT '*',
property_costs STRING COMMENT '*',
datetime STRING COMMENT '*'
)
COMMENT '*'
PARTITIONED BY (pt STRING);
```

注意：

自动生成的代码中，表名需要修改成真正的目标表表名，可以与源表表名一致，即 house_basic_info。

自动生成的代码中，源表中 varchar 类型会对应 string 类型，int 类型会对应 bigint

类型。MaxCompute 目前 只支持 6 种数据类型，与常用数据库数据类型有所差异。

自动生成的代码中，字段不能指定默认值、不能指定是否非空默认都是可空、不能指定长度默认每个字段长度上限为 8M。

自动生成的代码会创建分区表，且分区名称为 pt。MySQL 数据库中没有分区概念，MaxCompute 的分区概念与 Hadoop 分区概念类似，详情请参见 分区 。本示例中的目标表可以保留分区设置，以时间作为分区。

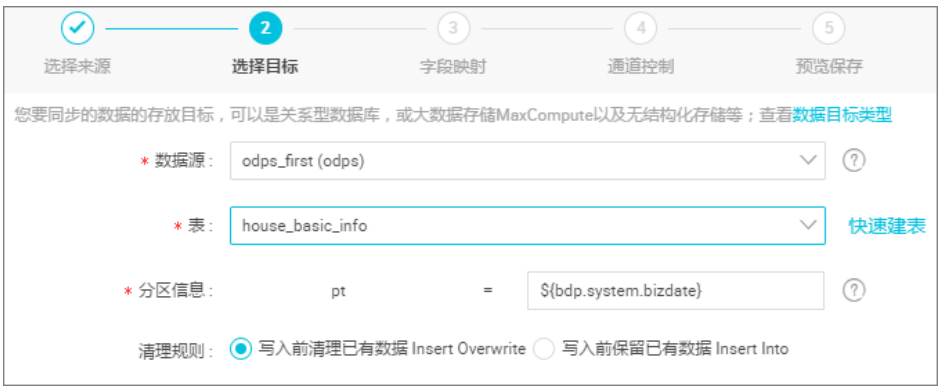
既然已经有时间分区，那么源表的 datetime 字段可以不需要同步到目标表，表也可以不需要创建该字段。

常用数据库 SQL 与 MaxCompute SQL 的更多差异请参见 与主流 SQL 的差异。

综上所述，修改弹出框中的建表语句，并单击 **提交**。MaxCompute 建表语句如下所示：

```
CREATE TABLE IF NOT EXISTS  house_basic_info  (
house_id STRING COMMENT '*',
house_city STRING COMMENT '*',
house_total_price DOUBLE COMMENT '*',
house_unit_price DOUBLE COMMENT '*',
house_type STRING COMMENT '*',
house_floor STRING COMMENT '*',
house_direction STRING COMMENT '*',
house_deckoration STRING COMMENT '*',
house_area DOUBLE COMMENT '*',
house_community_name STRING COMMENT '*',
house_region STRING COMMENT '*',
proj_name STRING COMMENT '*',
proj_addr STRING COMMENT '*',
period BIGINT COMMENT '*',
property STRING COMMENT '*',
greening_rate STRING COMMENT '*',
property_costs STRING COMMENT '*'
)
COMMENT '*'
PARTITIONED BY (pt STRING);
```

配置目标如下：



分区值保留默认的 ${bdp.system.bizdate}，与来源表的过滤条件取的 datetime 数据日期对应，表示该分区存放的数据为源表中 datetime=${bdp.system.bizdate} 的数据。

清理规则保留默认选项，写入前清理已有数据，若是分区表，则只清理当前分区中的数据（若有）。

字段映射。

直接保留默认设置即可。源表和目标表字段名都一致会自动对应好，源表 datetime 字段无对应目标字段且不用同步，因此无需做任何处理。

通道控制。

本示例中都保留默认设置即可，通道控制各项配置的详细说明请参见 数据同步通道控制参数设置。

保存 并 **提交**。

保存任务时，您可以创建单独的目录进行存放，本示例直接用目标表名称作为任务名称。

提交任务主要是将任务提交到调度系统，使得任务可以按照调度配置进行自动运行。本示例调度配置保留默认配置，调度周期为 **天** 调度。

后续步骤

现在，您已经学习了如何配置数据同步任务，您可以继续学习下一个教程。在该教程中您将学习如何执行数据同步任务，将 RDS 中的数据成功导入 MaxCompute 中。详情请参见 执行数据导入任务。

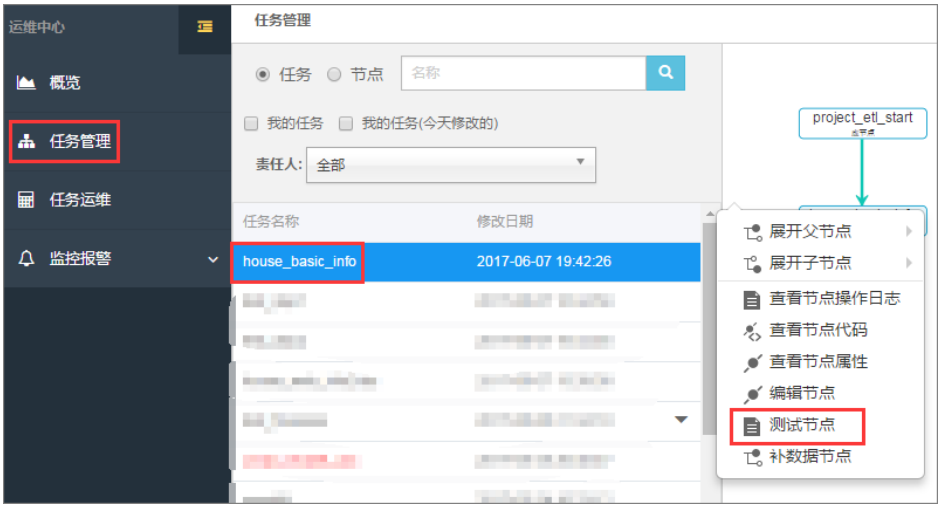
步骤4：执行数据导入任务

根据前文的操作，您已成功 **配置数据同步任务**，本文将为您介绍如何执行数据同步任务，将 RDS 中的数据成功导入 MaxCompute 中。

操作步骤

进入 **运维中心** > **任务管理** 页面。

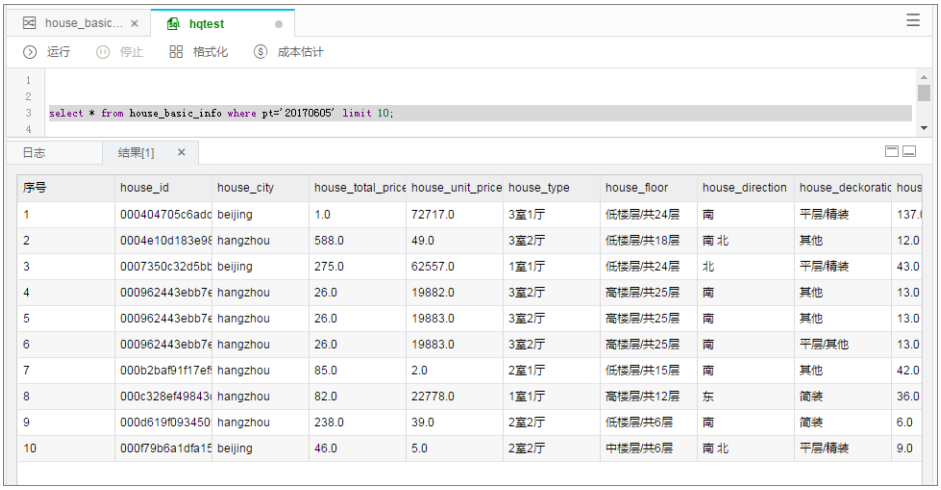
打开任务 house_basic_info，在任务视图上右键单击任务名，选择 **测试节点**。



根据跳转页面的提示，单击 **确认** 和 **运行**。

等待任务执行成功后，进入 DataWorks > **数据开发** 页面，创建一个脚本文件。

执行 select 语句，查看表 house_basic_info 数据是否同步成功。如下图所示：



后续步骤

现在，您已经学习了如何执行数据同步任务，并验证是否同步成功，您可以继续学习下一个教程。在该教程中您将学习如何通过 MaxCompute SQL、MR 等对数据进行加工处理。详情请参见 **数据统计分析**。

步骤5：数据统计分析

通过 **前文** 的操作，您已经成功将 RDS 数据源中的数据同步至 MaxCompute 表中，本文将为您介绍如何通过 MaxCompute SQL、MR 等对数据进行统计分析。

操作步骤

创建目标表

本示例的核心目标为：统计分析出每个城市二手房均价 Top 5 的楼盘，并且给出该楼盘所在城区，即（城市、楼盘、均价、排名和所在城区）。所以要首先创建目标表。

以项目管理员身份进入 **大数据开发套件管理控制台**，单击 **项目列表** 下对应项目操作栏中的 **进入工作区**。

进入顶部菜单栏中的 **数据开发** 页面，单击 **新建**，选择 **新建表**。如下图所示：

在新建表页面，输入如下建表语句，单击 **确认**。

```
CREATE TABLE IF NOT EXISTS house_unit_price_top5 (
house_city STRING,
house_community_name STRING,
```

```
house_unit_price_all DOUBLE,
area STRING,
tops BIGINT
)
PARTITIONED BY (
pt STRING
);
```

创建任务进行数据统计分析

进入顶部菜单栏中的 **数据开发** 页面，单击 **新建**，选择 **新建任务**。如下图所示：

新建 ODPS_SQL 节点任务，如下图所示：

编辑 SQL 代码

进入 ODPS_SQL 节点任务页面后，编辑如下 SQL 代码：

```
--产出每个城市每个楼盘的均价临时表
--分区值是对应数据导入任务配置的分区值，保证每天运行都是取当天导入的最新分区。
DROP TABLE IF EXISTS t_house_unit_price_info;
CREATE TABLE IF NOT EXISTS t_house_unit_price_info
AS
SELECT house_city,
```

```
house_community_name,
AVG(house_unit_price) AS house_unit_price_all
FROM house_basic_info
WHERE pt = '${bdp.system.bizdate}'
GROUP BY house_city,
house_community_name;

--拆分house_region字段只取城区名称输出字段为area，并存储到一个临时表。
--分区值是对应数据导入任务配置的分区值，保证每天运行都是取当天导入的最新分区。
DROP TABLE IF EXISTS t_house_area;
CREATE TABLE IF NOT EXISTS t_house_area
AS
SELECT distinct house_city,
house_community_name,
split_part(house_region, ' ', 1) AS area
FROM house_basic_info
WHERE pt = '${bdp.system.bizdate}';

--产出最终目标表：每天每个城市二手房均价top 5的楼盘并且给出该楼盘所在城区。
--分区值是对应数据导入任务配置的分区值，保证每天运行产出的日期分区值与源表数据日期一致。
INSERT OVERWRITE TABLE house_unit_price_top5 PARTITION (pt='${bdp.system.bizdate}')
SELECT a.house_city,
a.house_community_name,
a.house_unit_price_all,
b.area,
a.tops
FROM (
SELECT house_city
house_community_name,
house_unit_price_all,
ROW_NUMBER() OVER (PARTITION BY house_city ORDER BY house_unit_price_all DESC) AS tops
FROM t_house_unit_price_info
) a
JOIN t_house_area b
ON a.house_city = b.house_city
AND a.house_community_name = b.house_community_name
AND a.tops < 6;
```

注意：

MaxCompoute SQL 语法类似于常用 SQL 语法，可以看作是标准 SQL 的子集，但 MaxCompute 在很多方面并不具备常用数据库的特征，如事务、主键约束、索引等都不支持，因而 SQL 也有一定的差异。

在将数据导入目标表时，已经简单介绍了一些 DDL 语法的差异，针对此处的 DML 语句，简单补充以下内容：

产出每个城市每个楼盘的均价临时表 的整个语句只需要修改 where 条件中的 pt 条件，即可直接在 MySQL 上执行。

拆分 house_region 字段 语句中 split_part() 函数是 MaxCompute 内置的字符串函数，可以直接在 SQL 中使用，对应 MySQL 上 substring_index() 或其他。

产出目标表语句中，ROW_NUMBER() 是 MaxCompute 内置的窗口函数，在本示i例中主要作用于计算排行，可在 SQL 中直接使用，MySQL 上没有可直接对应的函数。

产出目标表语句中，insert overwrite（或 insert into） 后要加 table 关键字，MySQL 或 Oracle 不需要 table 关键字。

MaxCompute SQL 和常用 SQl 的更多差异请参见 与其他 SQL 的差异。

调度配置和参数配置

编辑好代码后，单击工具栏中的执行按钮执行 SQL 语句，对其进行探查。确定无误后进行调度配置。主要包括调度属性和依赖属性：

调度属性：由于每天调度一次，直接保留默认配置即可。

依赖属性：由于本任务处理的数据来源是数据导入任务 house_basic_info 产出大数据，为了保证本任务执行时，数据导入已经完成，需要将导入任务设置为本任务的上游任务（即父任务）。

注意：

由于本任务中只用到系统参数 ${bdp.system.bizdate}，这个参数在系统调度任务时会自动替换，所以无需再进行参数的其他配置。详情请参见 **系统参数说明**。

保存并提交

单击工具栏中的 **保存** 和 **提交** 按钮，将任务提交到调度系统。

单击工作区右上角 **前往运维** 按钮，即可到运维中心查看工作流状态。

执行任务

与执行数据导入任务的操作类似。执行成功后可以在 **数据开发** 模块的 SQL 脚本中查看目标表数据。如下图所示：

```
SELECT house_city, house_community_name, house_unit_price_all, area, tops
FROM dataplus_private_test_4.house_unit_price_top5 WHERE pt = '20170605' ORDER BY house_city, topsLIMIT 100;
```

日志 结果[1]

序号	house_city	house_communit	house_unit_price	area	tops
1	beijing	首开瑾堤公馆	113526.0	丰台	1
2	beijing	志新村	92562.0	海淀	2
3	beijing	御景春天	88372.0	丰台	3
4	beijing	二里庄小区	87969.0	海淀	4
5	beijing	靛厂路6号院	84450.57142857	丰台	5
6	hangzhou	林语别墅	83307.5	西湖	1
7	hangzhou	东方润园	77510.0	江干	2
8	hangzhou	宝石山下二弄	67961.0	西湖	3
9	hangzhou	马塍路35号	65682.33333333	西湖	4
10	hangzhou	文二路25号	64396.0	西湖	5

到目前为止，目标表已经正常产出。但是 MaxCompute SQL 在执行时会有一定的等待调度时间，适合做大数据批处理，网站前端读取数据就不适合直接读 MaxCompute 的数据，所以接下来需要把目标表回流到网站业务库。

后续步骤

现在，您已经学习了如何通过 MaxCompute SQL 对数据进行加工处理，并产出最终目标表，您可以继续学习下一个教程。在该教程中您将学习如何把目标表回流到网站业务库。详情请参见 **数据回流**。

步骤6：数据回流

数据回流与数据导入一样，需要配置数据同步任务，不过回流任务来源是 MaxCompute 的表，目标库是业务库，即示例中的 RDS-MySQL 的 house_web_master 数据库。

操作步骤

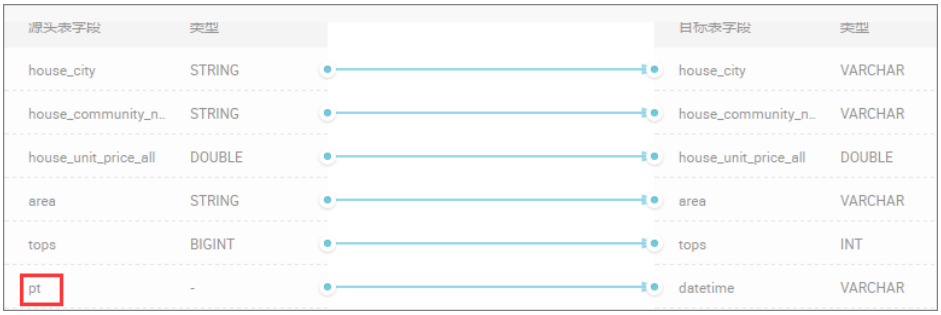
在 RDS > MySQL 中创建好对应的表，若需要保留每天的数据，可以加一个字段保存日期信息。

进入 **DataWorks > 数据集成** 页面配置数据源，详情请参见 配置 RDS 数据源。

创建并配置数据同步任务。

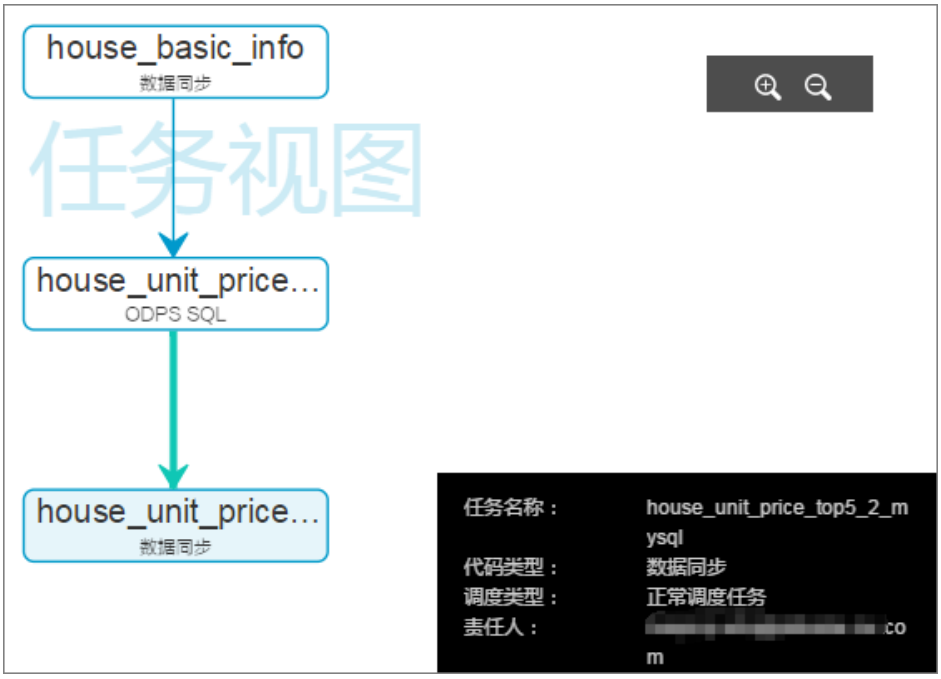
假设命名为 house_unit_price_top5_2_mysql，将 MaxCompute 表中的数据同步至 RDS > MySql 中。其中的两项配置如下：

字段配置：如果想直接把源表的分区字段同步到 MySQL 的日期信息字段，如下图所示：



依赖属性中，为了保证每次回流都是最新的数据，将数据加工任务 house_unit_price_top5 设置为父任务。如下图所示：

保存并提交任务后，在运维管理可以看到工作流状态：



执行回流任务，具体操作可参见 **执行数据导入任务。**

执行成功后，即可到 RDS > MySQL 上查看表数据是否正常导入。

完整数据开发

示例说明

示例背景

本示例主要介绍如何使用 DataWorks（数据工场，原大数据开发套件）完成一个完整的 MaxCompute SQl 工作流。您可以根据本示例，了解一个完整的工作流开发过程，包括：创建 MaxCompute 表、数据导入 MaxCompute 表、创建工作流、创建节点、测试运行工作流等过程。

示例介绍

本示例主要通过准备 **用户 > 品牌特征** 表为后期天猫品牌推荐模型做铺垫。在天猫，每天都会有数千万的用户通过品牌发现自己喜欢的商品，品牌推荐是链接商家和消费者的重要纽带。

本示例通过分析用户前三个月的的品牌购买情况以及最近3天、7天的用户偏好特征，得出下个月 **用户 > 品牌特征**，为后续品牌个性化推荐模型做准备。

步骤1：数据准备

本示例假设 **用户 > 品牌信息（源数据表）** 存储在业务方的 RDS 上，进而利用 DataWorks（数据工场，原大数据开发套件）进行数据同步、数据加工等操作，来详细阐述常见开发流程 **数据产生 > 数据收集和存储 > 数据分析和计算。**

源数据 请参见 **附件**，数据说明如下：

字段	字段说明	提取说明
user_id	用户标识	
brand_id	品牌ID	
type	用户对品牌的行为类型	点击：0；购买：1；收藏：2；加入购物车：3
visit_datetime	行为时间	格式：年月日（yyyymmdd）

该数据主要记录 20150415-20150815 四个月的用户行为信息，本示例将以该数据作为源数据进行分析，产出目标表。

本示例实现过程中，涉及到的 MaxCompute 表说明如下：

序号	表名	说明

1	s_user_brand_demo	用户-品牌行为信息源表
2	b_cvr_demo	品牌转化率表，前3个月品牌的购买用户数/点击数
3	ub_action_demo	用户偏好表，统计用户最近7天和最近3天的行为次数
4	ub_features_demo	用户-品牌所有特征表

经分析，源数据 visit_datetime 字段刚好是年月日，为了提高后续查询速度，源表 s_user_brand_demo 建为分区表，以字段 visit_datetime 为分区。

用户数据每天都不断新增变化，本示例的表，都以年月日作为分区表。

后续步骤

现在，您已经对实验所需的数据做了一定的准备和了解，您可以继续学习下一个教程。在该教程中您将学习如何配置实验所需的 RDS 数据源。详情请参见 配置 RDS 数据源。

步骤2：配置RDS数据源

由前文可知，原始数据在 RDS 上，那么需要把 RDS 数据导入到 MaxCompute 中。本示例通过数据同步任务来完成数据的导入，而数据同步任务必须事先创建好数据源。

前提条件

创建 RDS 数据源必须要清楚 RDS 的相关信息（可在 RDS 的基本信息页面中获取），此处数据源配置成通过 RDS 实例形式连接，所以需要提前知道的 RDS 信息包括：RDS 实例 ID，RDS 实例购买者 ID，数据库名，用户名和密码。

操作步骤

以项目管理员身份进入 **DataWorks 管理控制台**，单击对应项目操作栏中的 **进入工作区**。

单击顶部菜单栏中的 **数据集成**，导航至 **数据源** 页面。

单击 **新增数据源**。

在新建数据源弹出框中，选择数据源类型为 RDS > MySQL。

选择以 RDS 实例形配置该 MySQL 数据源。

查看 RDS > MySQL 中的实例 ID，如下图所示：

单击 **测试连通性。**

测试连通性通过后，单击 **确定。**

> **注意：**
>
> 若测试连通性失败，请参见 RDS 数据源测试连通性不通。

后续步骤

现在，您已经学习了如何配置 RDS 数据源，您可以继续学习下一个教程。在该教程中您将学习如何准备相关的

MaxCompute 表。详情请参见 创建 MaxCompute 表。

步骤3：创建MaxCompute表

操作步骤

以新建 s_user_brand_demo 数据表为例，具体操作如下：

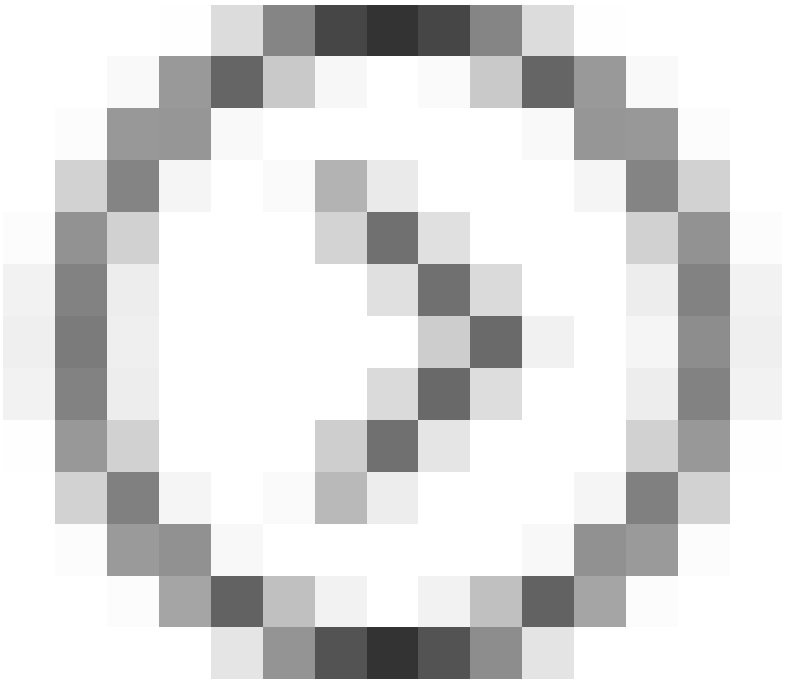
以开发者身份进入 DataWorks 管理控制台，单击对应项目操作栏中的 **进入工作区**。

创建脚本文件。

单击顶部菜单栏中的 **数据开发**，导航至 **新建 > 新建脚本**。

编辑建表语句。

```
CREATE TABLE IF NOT EXISTS s_user_brand_demo (
user_id STRING COMMENT '用户标识',
brand_id STRING COMMENT '品牌ID',
type STRING COMMENT '用户对品牌的行为类型，点击：0，购买：1，收藏：2，加入购物车：3'
)
PARTITIONED BY (
dt STRING
)
LIFECYCLE 150;
```



单击运行按钮 运行建表语句。

语句运行成功，则建表成功。

注意：

您可以执行 desc tablename;，查看表是否真正创建成功。

建表语句

您可根据上述步骤，完成其他表的创建。需要创建的表和对应的建表语句，如下所示：

b_cvr_demo（品牌转化率表）

```
--品牌转化率表，品牌的购买用户数/点击数
CREATE TABLE IF NOT EXISTS b_cvr_demo (
brand_id STRING,
```

```
cvr DOUBLE
)
PARTITIONED BY (
dt STRING
)
LIFECYCLE 7;
```

ub_action_demo（用户偏好表）

```
--用户偏好表，这里统计用户最近 7 天和最近 3 天的行为次数
CREATE TABLE IF NOT EXISTS ub_action_demo (
user_id STRING,
brand_id STRING,
buy_cnt BIGINT,
click_d7 BIGINT,
collect_d7 BIGINT,
shopping_cart_d7 BIGINT,
click_d3 BIGINT,
collect_d3 BIGINT,
shopping_cart_d3 BIGINT
)
PARTITIONED BY (
dt STRING
)
LIFECYCLE 7;
```

ub_features_demo（用户-品牌所有特征表）

```
--品牌-用户所有特征表
CREATE TABLE IF NOT EXISTS ub_features_demo (
user_id STRING,
brand_id STRING,
buy_cnt BIGINT,
click_d7 BIGINT,
collect_d7 BIGINT,
shopping_cart_d7 BIGINT,
click_d3 BIGINT,
collect_d3 BIGINT,
shopping_cart_d3 BIGINT,
cvr DOUBLE
)
PARTITIONED BY (
dt STRING
)
LIFECYCLE 7;
```

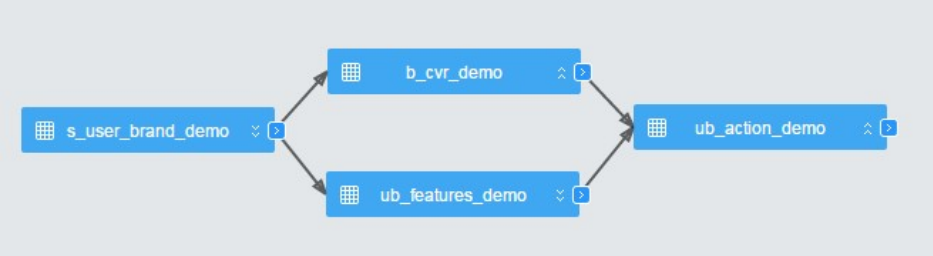
后续步骤

现在，您已经学习了如何创建 MaxCompute 表，您可以继续学习下一个教程。在该教程中您将学习如何创建

工作流来对项目空间的数据进行进一步的计算与分析。详情请参见 创建工作流。

步骤4：创建工作流

本示例中，数据的分析流程如下图所示：



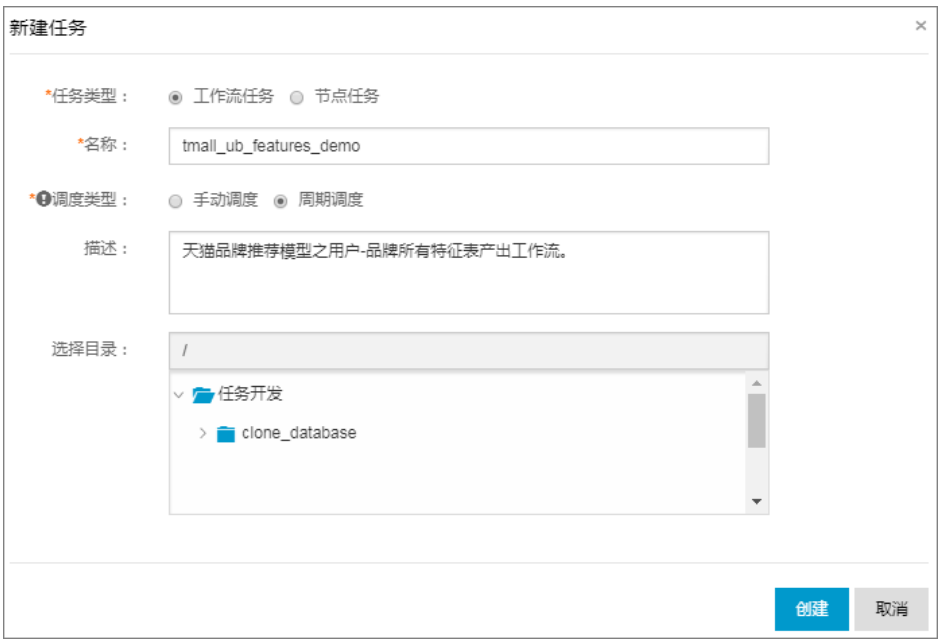
源表经过加工成为两个中间表，最后通过两个中间表加工得出目标表，一个工作流即可完成。同时，在 **数据准备** 中分析得出需要创建日分区表，也就是每日一分区。因此工作流需配置为周期性天调度。

操作步骤

以开发者身份进入 DataIDE 管理控制台，单击对应项目操作栏中的 进入工作区。

单击顶部导航栏中的 **数据开发**，导航至 **新建 > 新建任务**。

填写弹出框中的各配置项，指定任务类型为 **工作流任务**。如下图所示：



单击 **创建**。

进入工作流页面后，单击右侧导航栏的 **调度配置** 进行配置。

基本属性无需修改。

调度属性保留默认配置。

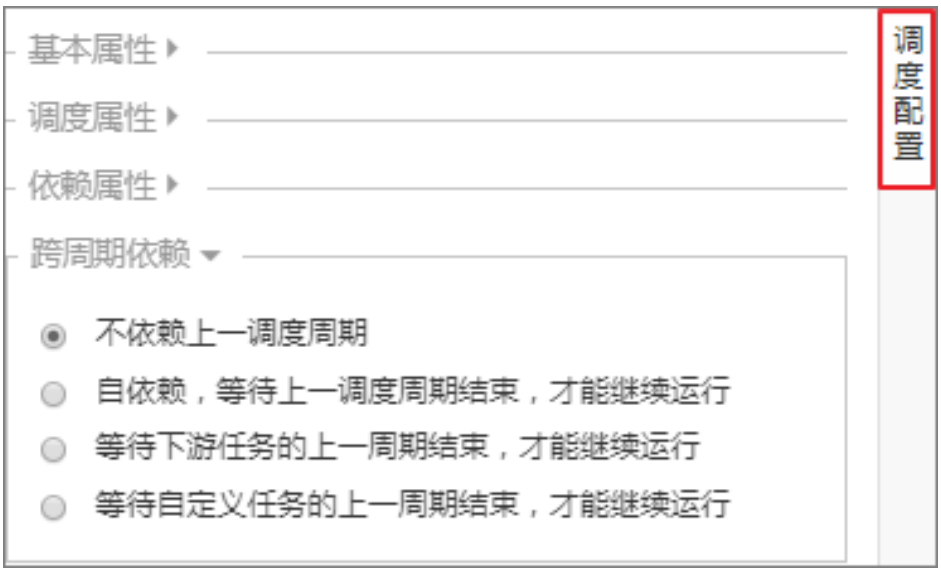
因为工作流需要周期调度，且目前没有预设下线时间，因此所有配置项保留默认。

调度周期为天，具体时间为 0 点整，即每日 0 点调度服务开始调度当天示例时，即可开始调度此工作流。

依赖属性保留默认配置。

因为源头数据导入后，打算直接在本工作流中配置任务，没有必须依赖的上游工作流，所以此配置保持不变。

跨周期依赖可根据自己的需求进行相应的配置。



后续步骤

现在，您已经学习了如何创建工作流，您可以继续学习下一个教程。在该教程中您将学习如何通过创建同步任务来把数据导入到 MaxCompute 中。详情请参见 配置数据导入任务。

步骤5：配置数据导入任务

原始数据在 RDS 数据库上，若想通过 MaxCompute 对数据进行加工、分析，需要先把数据导入到 MaxCompute 中。前文中已成功 配置 RDS 数据源 和 创建 MaxCompute 表，接下来即可开始创建数据导入任务。

操作步骤

打开创建的工作流（tmall_ub_features_demo），将数据同步节点组件拖拽至画布中。

名称：s_user_brand_demo。

描述：RDS 上同步数据到表 s_user_brand_demo。

双击该节点或右键查看节点内容进入任务配置界面。

选择来源。

源头默认为单表，选择前面添加的数据源，和对应的原始数据表。

选择目标。

目标选择本项目对应的 MaxCompute project，所以数据源为 odps_frist，目标表为 s_user_brand_demo 表。

字段映射。

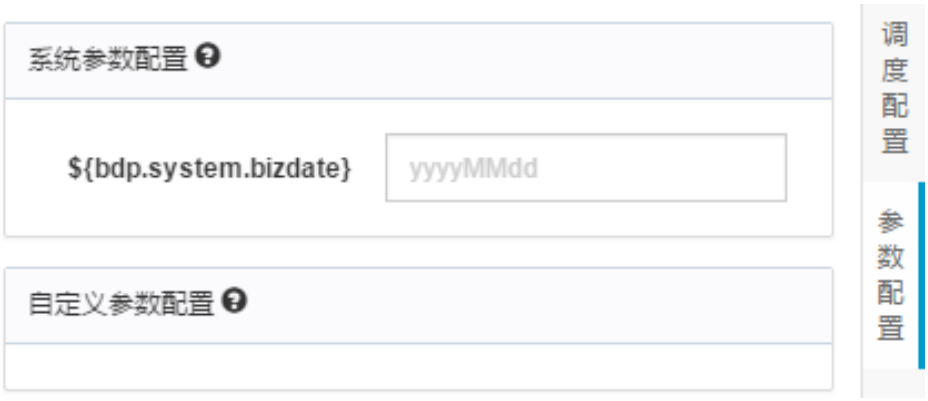
选择要抽取的列，并映射到目标表字段。

选好源和目标表之后，列会先自动按照字段名对应匹配，匹配不到的目标字段留空，默认显示所有源表字段，数据同步任务执行的时候就按该字段配置顺序一一对应读写。

通道控制。

完成以上配置后，单击 **保存**。

配置节点参数。



由于 ${bdp.system.bizdate} 为系统参数，因此参数配置中无需赋值。

单击 **保存**。

后续步骤

现在，您已经学习了如何配置数据同步任务，您可以继续学习下一个教程。在该教程中您将学习如何配置 SQL 任务，产出结果表。详情请参见 配置 SQL 任务产出特征表。

步骤6：配置SQL任务产出特征表

本示例为了更形象的说明工作流配置，一个 MaxCompute SQL 节点产出一个表。经分析需要创建 3 个 MaxCompute SQL 节点。

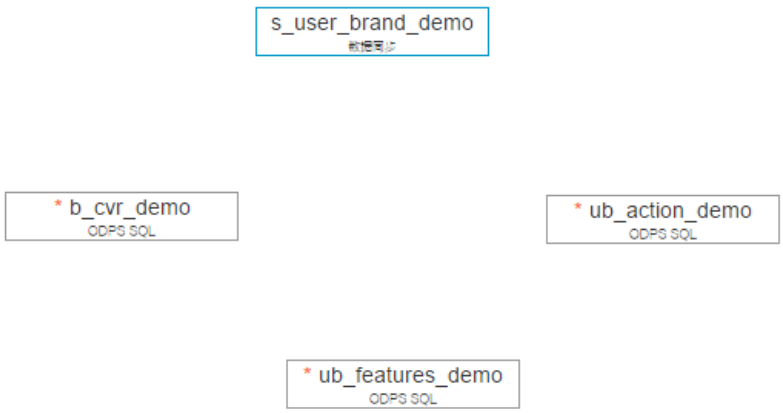
操作步骤

工作流（tmall_ub_features_demo）设计器的节点组件中向画布拖拽 3 个 MaxCompute SQL 节点组件，进行创建。

节点名称分别为：b_cvr_demo、ub_action_demo、ub_features_demo。

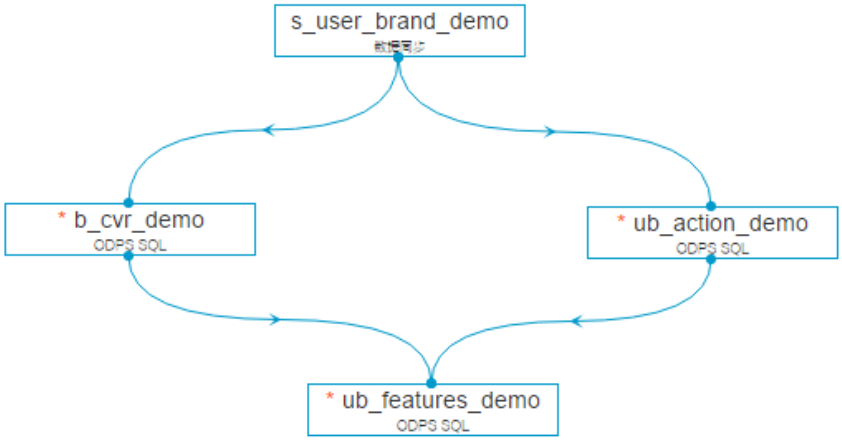
描述：对应上面的节点名称分别为：产出品牌转化率表、产出用户偏好表、产出用户-品牌所有特征表。

此时看到工作流设计器上有 4 个 节点：1 个同步任务，3 个 MaxCompute SQL 任务，如下图所示：



配置节点依赖。

根据前面的数据分析，中间表数据来自同步任务产出的源表，最终特征表数据来自两个中间表，因此，节点的依赖关系如下图所示：



编辑 MaxCompute SQL 节点代码，与参数配置（内置调度时间参数说明请参见 **数据开发手册 > 系统调度参数**）。

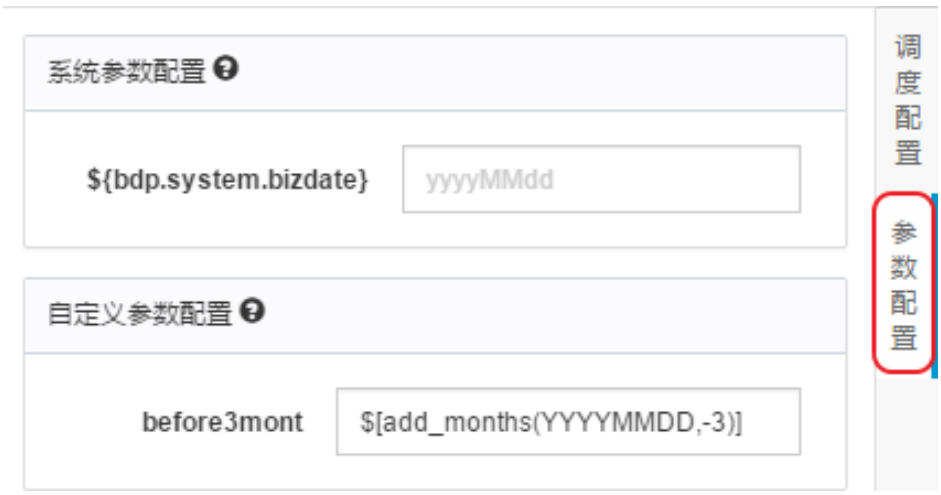
分别双击 SQL 节点进入代码编辑页面进行代码编辑，代码如下：

节点 b_cvr_demo 代码与参数配置

```
--产出品牌转化率表,前3个月品牌的购买用户数/点击数
INSERT OVERWRITE TABLE b_cvr_demo PARTITION (dt=${bdp.system.bizdate})
SELECT brand_id
, CASE
WHEN click_cnt > 0 THEN buy_cnt / click_cnt
ELSE 0
END AS cvr
FROM (
SELECT brand_id
, COUNT(DISTINCT CASE
WHEN type = '1' THEN user_id
ELSE NULL
END) AS buy_cnt
, COUNT(DISTINCT CASE
WHEN type = '0' THEN user_id
ELSE NULL
END) AS click_cnt
FROM s_user_brand_demo
WHERE dt >= ${before3mont}
GROUP BY brand_id
) t1;
```

产出表分区表达式与前面数据同步任务分区表达式一致，每次运行读源表分区为前三个月分区，源表分区过滤条件 **dt>=${before3mont}**

参数配置如下图：



${bdp.system.bizdate} 变量在调度的时候会自动替换成业务日期，所以不需要赋值。

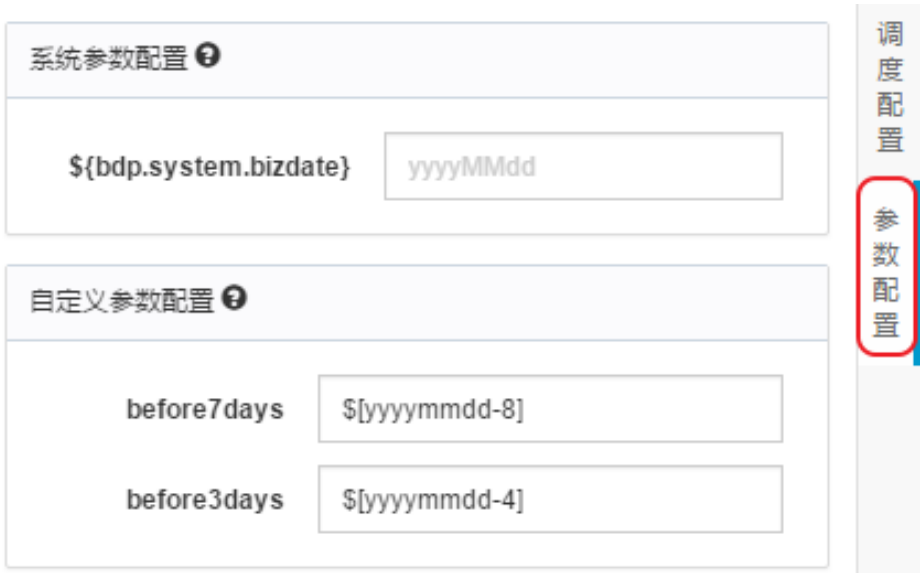
${before3mont} 自定义变量需要在此赋值，因为是取前 3 个月的数据，所以可以去当前节点实例定时时间减 3 个月，即 $[add_months(YYYYMMDD,-3)]。

节点 ub_action_demo 代码与参数配置

```
--产出用户偏好表,这里统计用户最近7天和最近3天的行为次数
INSERT OVERWRITE TABLE ub_action_demo PARTITION (dt=${bdp.system.bizdate})
SELECT user_id
, brand_id
, SUM(CASE
WHEN type = '1' THEN 1
ELSE 0
END) AS buy_cnt
, SUM(CASE
WHEN type = '0'
AND dt > '${before7days}' THEN 1
ELSE 0
END) AS click_d7
, SUM(CASE
WHEN type = '2'
AND dt > '${before7days}' THEN 1
ELSE 0
END) AS collect_d7
, SUM(CASE
WHEN type = '3'
AND dt > '${before7days}' THEN 1
ELSE 0
END) AS shopping_cart_d7
, SUM(CASE
WHEN type = '0'
AND dt > '${before3days}' THEN 1
ELSE 0
END) AS click_d3
```

```
, SUM(CASE
WHEN type = '2'
AND dt > '${before3days}' THEN 1
ELSE 0
END) AS collect_d3
, SUM(CASE
WHEN type = '3'
AND dt > '${before3days}' THEN 1
ELSE 0
END) AS shopping_cart_d3
FROM s_user_brand_demo
WHERE dt >= ${before7days}
and dt <= ${bdp.system.bizdate}
GROUP BY user_id,
brand_id;
```

参数配置如下图：



${bdp.system.bizdate} 变量在调度的时候会自动替换成业务日期，所以不需要赋值。

${before7days} 和 ${before3days} 需要的是业务日期的前7天和前3天，所以可以用调度时间参数 $[yyyymmdd-8] 和 $[yyyymmdd-4]，即当前节点实例定时时间年月日减 8 天/减 4 天。

节点ub_features_demo代码与参数配置

```
INSERT OVERWRITE TABLE ub_features_demo PARTITION (dt=${bdp.system.bizdate})
SELECT t1.user_id
, t1.brand_id
, t1.buy_cnt
, t1.click_d7
, t1.collect_d7
```

```
, t1.shopping_cart_d7
, t1.click_d3
, t1.collect_d3
, t1.shopping_cart_d3
, t2.cvr
FROM ub_action_demo t1
LEFT OUTER JOIN b_cvr_demo t2
ON t1.brand_id = t2.brand_id
AND t1.dt = ${bdp.system.bizdate}
AND t2.dt = ${bdp.system.bizdate};
```

${bdp.system.bizdate}变量在调度的时候会自动替换成业务日期，所以不需要赋值,代码中没用到其他自定义变量名，所以不需要配置参数。

返回工作流设置器页面，单击 **保存**。

至此，已完成本示例的工作流配置，但要想让工作流根据配置每天自动调度，还需要把工作流提交到调度系统，即在工作流设计页面（整体视图页面）单击 **提交**，在变更节点列表中选择所有节点并单击 **确定提交**，提交成功则成功的把工作流提交到调度服务。

后续步骤

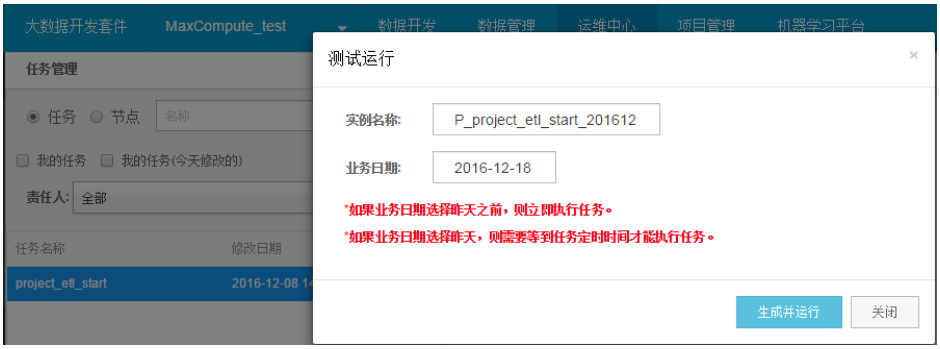
现在，您已经学习了如何通过 MaxCompute SQL 对数据进行加工处理，并产出最终目标表，您可以继续学习下一个教程。在该教程中您将学习如何测试工作流。详情请参见 测试任务。

测试任务

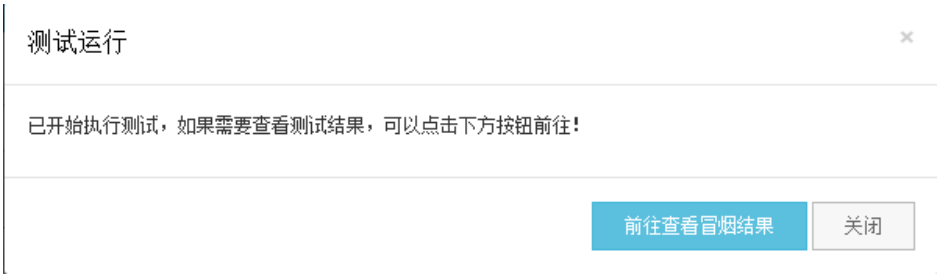
工作流提交后，即可对整个工作流手动在调度上试运行一次，目的是看运行过程中代码、调度配置是否符合预期。（注意测试运行是真正的在跑任务，结果是真实的结果。）具体操作步骤如下：

方式一：

步骤1：紧接上个章节，在整体视图页面右击节点，点击“测试节点”，在弹出的业务日期选择框里选择业务日期，点击“生成并运行”。



步骤2：前往运维中心查看工作流测试情况。



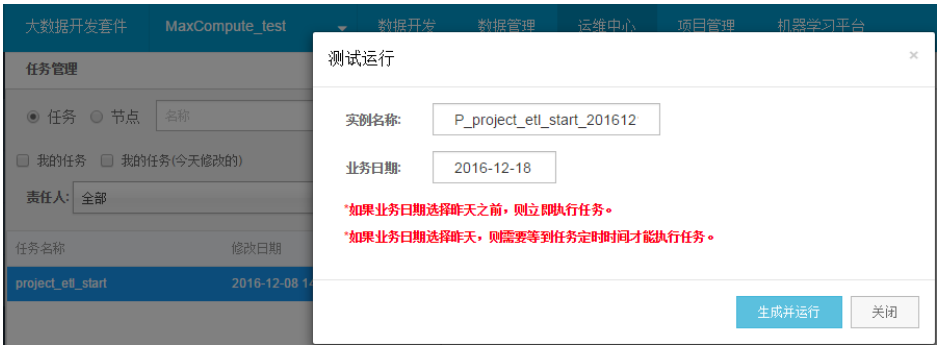
点击“前往查看冒烟结果”会直接跳到运维中心>>任务运维>>测试页面中且定位到具体的工作流测试实例。

步骤3：查看任务具体运行日志。不管是执行成功还是失败，都应该去查看一次每个节点的运行日志，如查看真正运行的代码是什么，查看生产的节点实例的SKYNET_BIZDATE是否就是选择的测试业务日期，查看使用的调度参数是否正常替换等。

方式二：

步骤1：进入运维中心>>任务管理，搜索到本任务，选择后，在右边的DAG图里对工作流右键->测试节点。

步骤2：选择需要测试的业务日期，点击“生成并运行”。



点击后会直接跳到测试模块，且定位到具体的工作流测试实例。

步骤3：查看任务具体运行日志。不管是执行成功还是失败，都应该去查看一次每个节点的运行日志，如查看真正运行的代码是什么，查看生产的节点实例的SKYNET_BIZDATE是否就是选择的测试业务日期，查看使用的调度参数是否正常替换等。

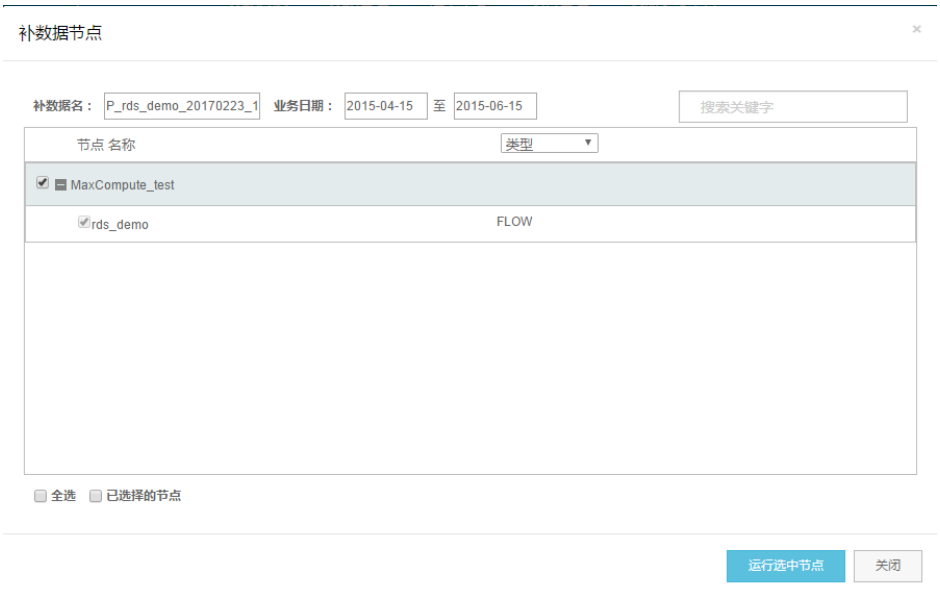
【注意】：由于本示例准备的数据仅仅是2015/04/15—2015/08/15 四个月的数据，而测试的时候只挑了1天的数据来测试，那么除了同步任务能看出具体结果外，3个sql类型节点由于是要加工前3个月或者前

7/3天的数据，只跑一天的数据对于sql任务来说只能测试查看代码和调度配置是否正常而已，无法看最终结果表的结果是否符合预期，所以一般情况下工作流配置好后，我们会通过补数据方式把前面已有的数据都导入ODPS表中并进行加工处理，补数据操作请看后面小章节。

补数据

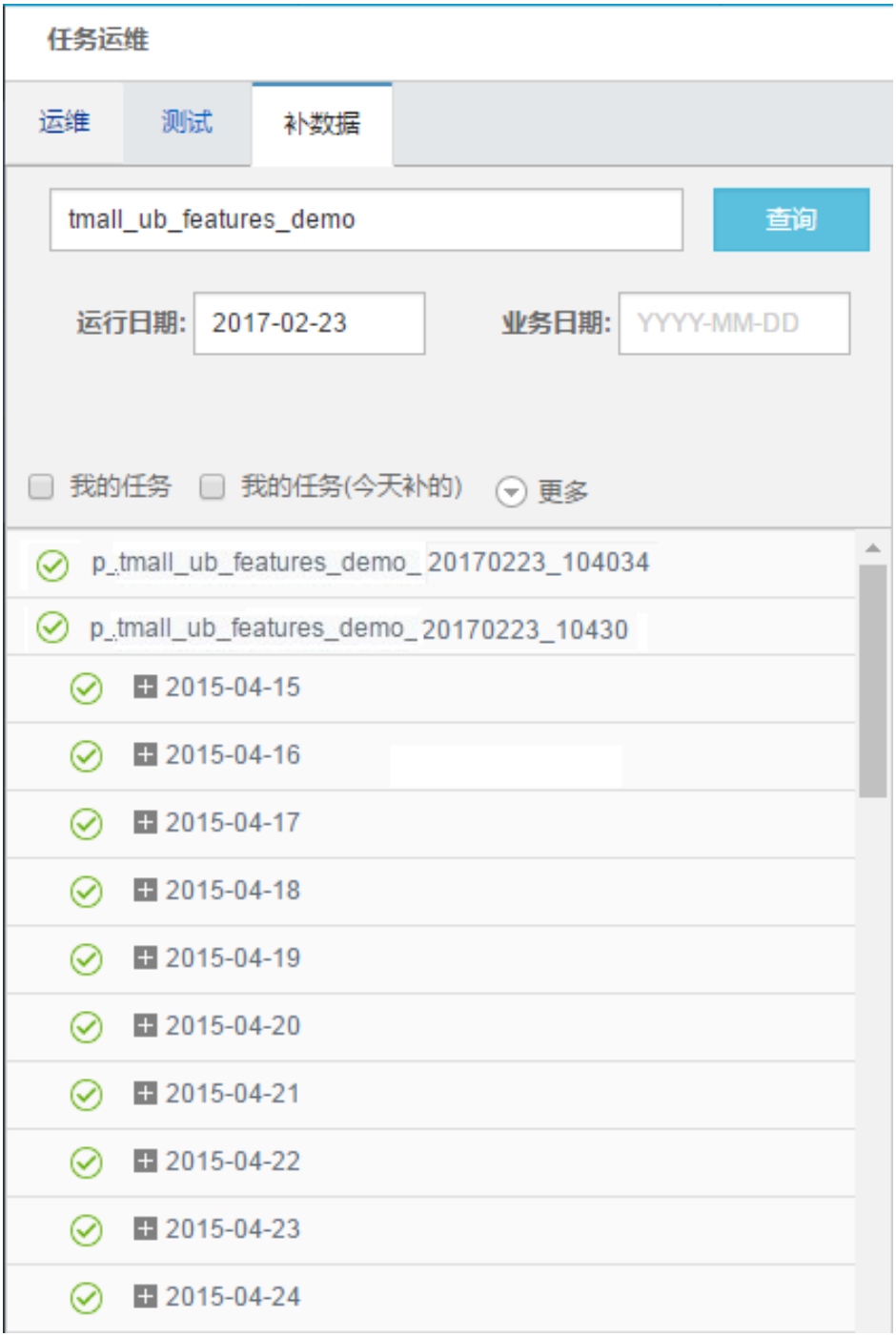
补数据可以对工作流/节点操作执行发生在过去的一段时间的调度。如本示例中数据是2015/04/15—2015/08/15 四个月的时间，补数据的时候可以分1次或2次操作把这段时间的工作流实例全部生成好。具体操作如下：

进入运维中心>>任务管理>>工作流管理>>定义页面，搜索到本工作流，选择后，在右边的DAG图里对工作流右键->补数据任务。



这里分开两次操作，先补2015/04/15—2015/06/15业务日期，点击“运行选择工作流”后会生产补数据实例

，页面跳到“补数据”模块，并定位到具体的工作流。



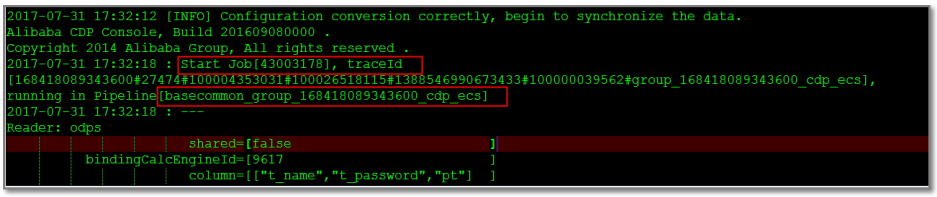
生产好补数据实例后接下来就是等待运行结果并查看具体运行情况。

同步日志排查

简介

数据集成，是阿里巴巴对外提供的稳定高效、弹性伸缩的数据同步平台。 致力于提供复杂网络环境下、丰富的异构数据源之间数据高速稳定的数据移动及同步能力。丰富的数据源支持:文本存储(FTP/SFTP/OSS/多媒体文件 等)、数据库(RDS/DRDS/MySQL/PostgreSQL 等)、NoSQL(Memcache/Redis/MongoDB/HBase 等)、大数据(MaxCompute/ AnalyticDB/HDFS 等) 、MPP数据库（HybridDB for MySQL等）。正因为数据集成兼容了复杂网络环境下，多种数据库之间共通，所以在使用的时候，难免会遇到出错的情况，那么下面我们来解析一下数据集成的日志组成。

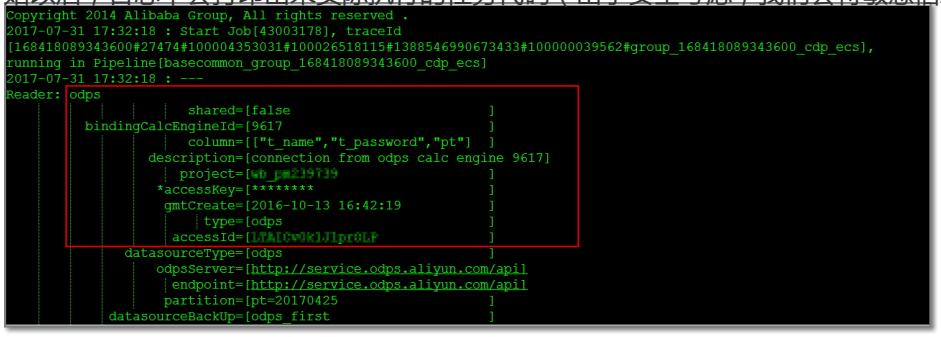
任务是从哪里开始的



如图所示：“**Start Job**” 表示开始这个任务；Start Job 下面会有段日志 “**running in Pipeline[XXXXX]**” 主要是用来区分任务是跑在什么机器上，如果XXXXX中含有 “ basecommon_group_XXXX ” 等字样，说明是跑在公共资源组的机器上，如果不包含 “ basecommon_group_XXXX ” 的字样，说明在您的自定义资源组上运行。如何查看具体执行任务的机器名，请参考 “**任务运行情况**这节的介绍” 。

实际运行的任务代码

在任务开始以后，日志中会打印出来实际执行的任务代码（出于安全考虑，我们会将敏感信息以*号表示），如图所示：



图示这个任务的实际代码样例如下：

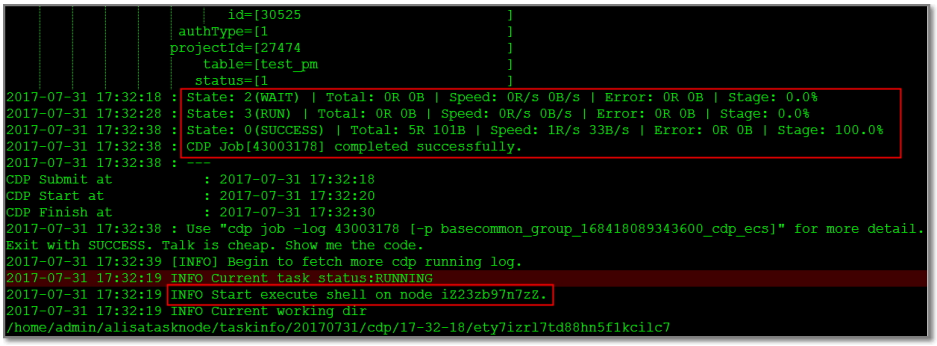
```
Reader: odps
shared=[false ]
bindingCalcEngineId=[9617 ]
column=[["t_name","t_password","pt"] ]
description=[connection from odps calc engine 9617]
project=[XXXXXXXXX ]
*accessKey=[******* ]
gmtCreate=[2016-10-13 16:42:19 ]
```

```
type=[odps ]
accessId=[XXXXXXXXX ]
datasourceType=[odps ]
odpsServer=[http://service.odps.aliyun.com/api]
endpoint=[http://service.odps.aliyun.com/api]
partition=[pt=20170425 ]
datasourceBackUp=[odps_first ]
name=[odps_first ]
tenantId=[168418089343600 ]
subType=[ ]
id=[30525 ]
authType=[1 ]
projectId=[27474 ]
table=[t_name ]
status=[1 ]
Writer: odps
shared=[false ]
bindingCalcEngineId=[9617 ]
column=[["id","name","pt"] ]
description=[connection from odps calc engine 9617]
project=[XXXXXXXXX ]
*accessKey=[******** ]
gmtCreate=[2016-10-13 16:42:19 ]
type=[odps ]
accessId=[XXXXXXXXX ]
datasourceType=[odps ]
odpsServer=[http://service.odps.aliyun.com/api]
endpoint=[http://service.odps.aliyun.com/api]
partition=[ ]
truncate=[true ]
datasourceBackUp=[odps_first ]
name=[odps_first ]
tenantId=[XXXXXXXXX ]
subType=[ ]
id=[30525 ]
authType=[1 ]
projectId=[27474 ]
table=[test_pm ]
status=[1 ]
```

这是一个典型的 Maxcompute（原ODPS）数据源同步到 Maxcompute 数据源的任务代码，关于这段任务代码的解析，请参考MaxCompute Reader和 MaxCompute Writer

任务运行情况

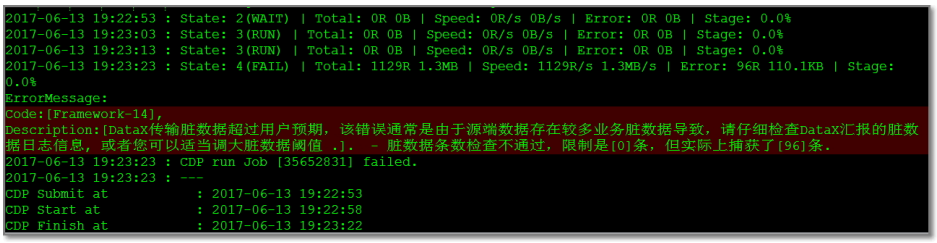
上面我们介绍了实际运行的任务代码，在实际运行的任务代码下，会打印出来该任务的运行情况，如图所示：



图中用红框标记出来的内容，记录了这个任务何时开始运行，何时运行结束。当：“State: 2(WAIT)” State状态为2的时候，还在等待任务运行；
当：“State: 3(RUN)” State 状态为3的时候，表示任务正在运行；
当：“State: 0(SUCCESS)” State 状态为0的时候，表示任务已经成功运行完毕；

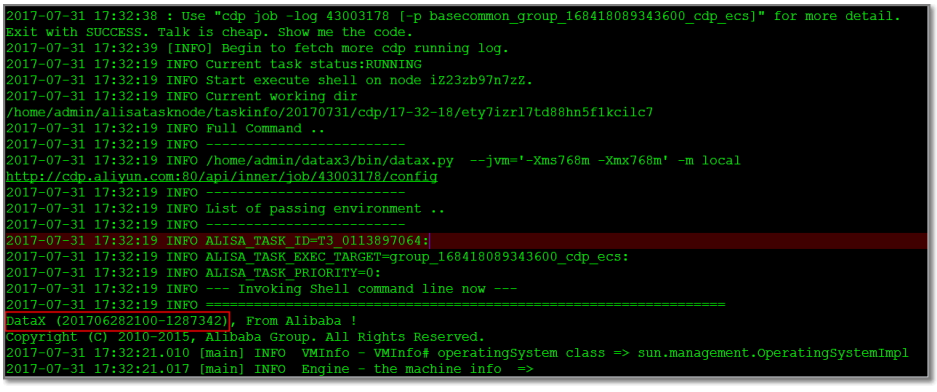
> 注意：在任务运行完毕的记录下面，有“INFO Start execute shell on node XXXXXXX” 这段话表示，该任务实际运行在 XXXXXXX 这台机器上。

排错小助手：当有脏数据的时候，无法将数据写入进去，日志就会报如下错误：



详细运行日志

其实数据同步的任务日志，到上节为止，就结束了，下面的一长串日志，是DataX的详细执行日志（数据集成功能是针对阿里巴巴开源项目DataX做了一层封装），如图所示：

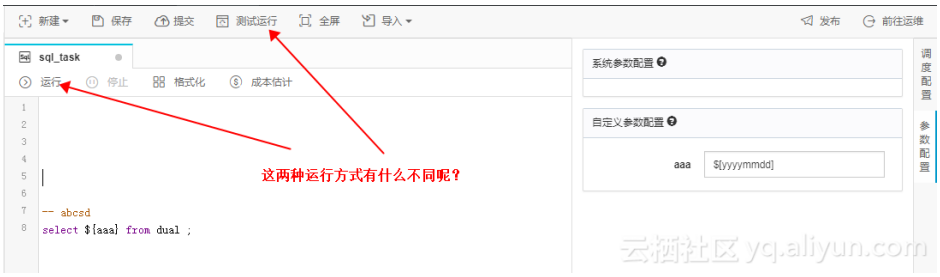


很多同学运行数据同步任务会报错，可以参考如下文档先进行错误排查：常见报错。

若常见报错无法解决您的问题，请带上完整的日志提工单反馈给我们。

解析Dataworks中的运行和测试运行的区别

有很多用户在使用Dataworks的数据开发中运行SQL和在数据集成中运行同步任务时，都会有一个疑惑。我在页面上运行和测试运行有什么区别呢？为什么我明明配置了系统参数，在代码中运行时，却没有自动解析，而提醒我去填写系统变量的临时值？



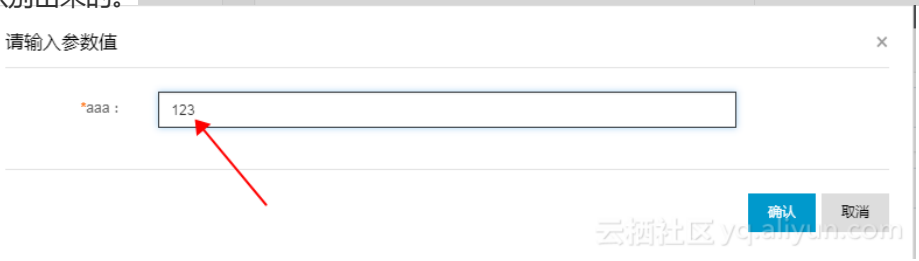
下面我就给大家讲讲这两者的主要区别。

页面上的运行

页面上的运行是不会经过调度系统的，直接将任务下发到底层去执行。所以在使用了调度参数后，运行时，是需要指定调度参数解析出来的值的。页面上触发的运行是不会生成实例的，所以也就没有办法去指定运行任务的机器，只能下发到Dataworks的默认资源组上去执行。

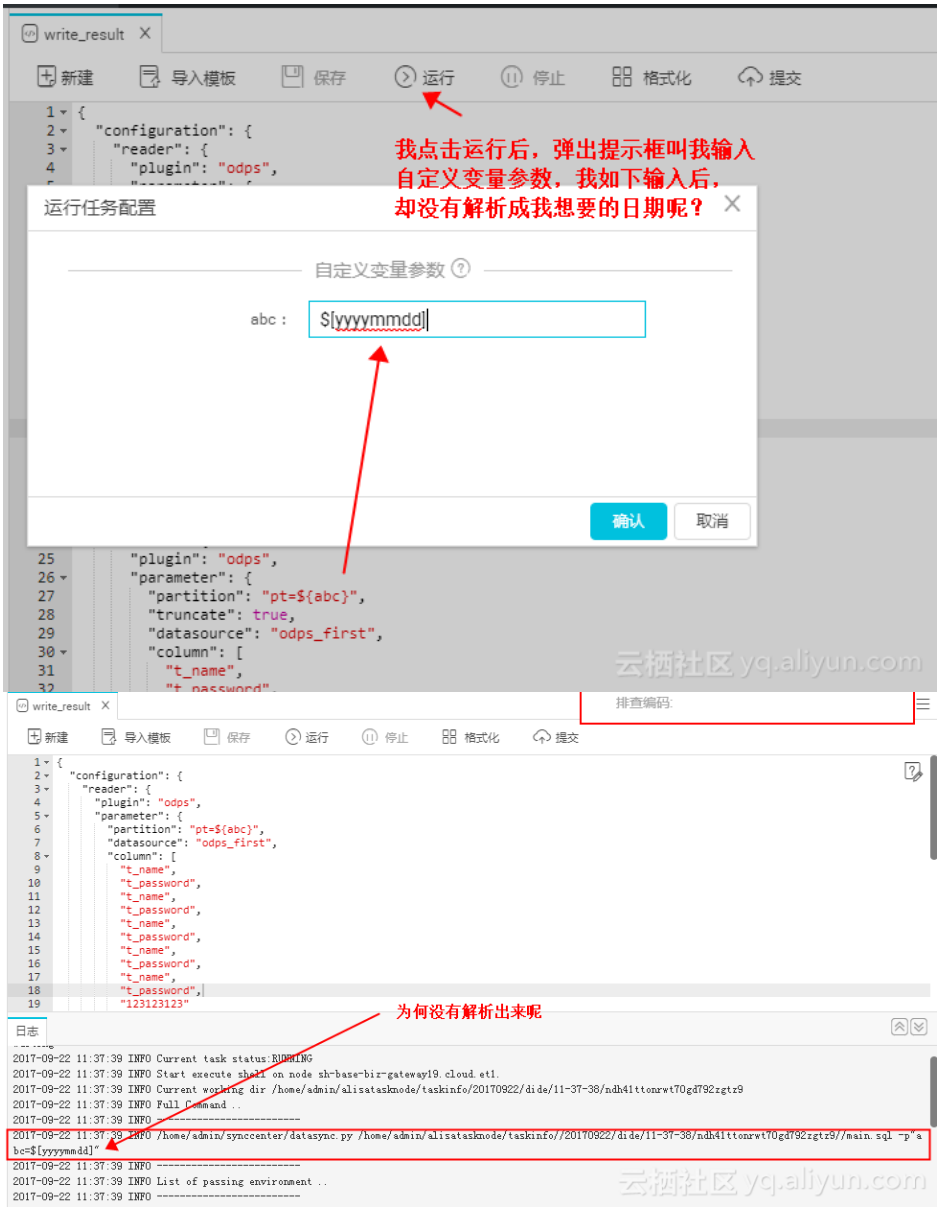
数据开发在页面上运行时如何给自定义参数赋值

在数据开发中，创建了SQL节点任务时，在SQL中使用了自定义参数。点击页面上的运行，会弹出一个提示框，在这个提示框里一定要填一个具体的值，而不要填${yyyymmdd}这种，不然在代码中${yyyymmdd}是不会识别出来的。



数据集成在页面上运行时如何给自定义参数赋值

在数据集成中，创建脚本模式的任务时。在脚本中使用了自定义参数，保存后，点击页面上的运行，提示我需要给自定义参数赋值。我填了一个值以后，却没有解析出来呢？



原因是因为：系统参数和自定义系统参数，是调度系统的参数，只有通过调度系统后，才会解析出来。而我们

点击的运行，是没有经过调度系统的，所以提示你输入的自定义变量参数 是需要填一个具体的值才行，这样在执行任务的时候，才会直接替换掉。

测试运行

测试运行会通过调度系统，去生成实例的，所以在使用了调度参数后，运行时，调度参数就会自动解析出来了，而且可以指定实例运行所在的资源组。

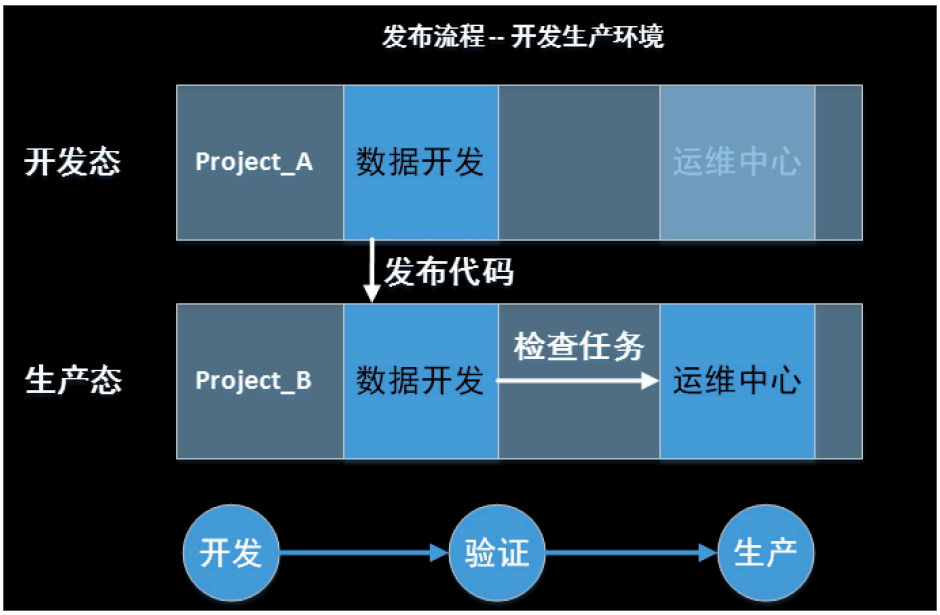
安全的数据开发模式

实验背景

因为开发角色拥有删除表的权限，有用户质疑如果让其直接操作生产环境的表，会导致数据不安全。本文将为您介绍如何保证生产环境的数据安全。

解决方案

解决数据安全问题的解决方案的整体流程，如下图所示：



操作步骤

前期准备

创建两个项目，一个作为开发项目，一个作为生产项目，比如：Project_A 和 Project_B。

进入 Project_A 的 **项目管理** 页面，指定此项目发布到 Project_B 下。指定后，这两个项目便具有了关联关系，可以通过发布功能，将任务发布。

在项目管理中 Maxcompute 配置下，配置使用个人账号访问 Maxcompute 资源。如下图所示：

代码编辑

在 Project_A 中进行编辑代码，配置任务等操作。

将编辑好的代码和任务，通过 **发布** 功能，发布到 Project_B 中。

查询生产项目下的数据

如果需要操作生产项目下的表，可以进入 **数据管理** 页面，申请生产项目表的权限，这样开发角色便可在 Project_A 项目中通过 Project_B.table 的方式来查询生产项目中表的数据（申请的只有查询表权限，没有 drop 表权限）。

在 **数据管理** 页面，您不仅可以申请表的权限，还可以申请资源以及函数的权限。

注意：

从 Project_A 发布到 Project_B 后，有一些项目级别的配置是不会发布过去的，比如说数据源、表、资源、函数等，都需要在 Project_B中重新建立。