2.1 数据预处理

这一步将会使用PAI中的四个组件：

“源/目标”--“读数据表”：从MaxCompute读取数据表至PAI；

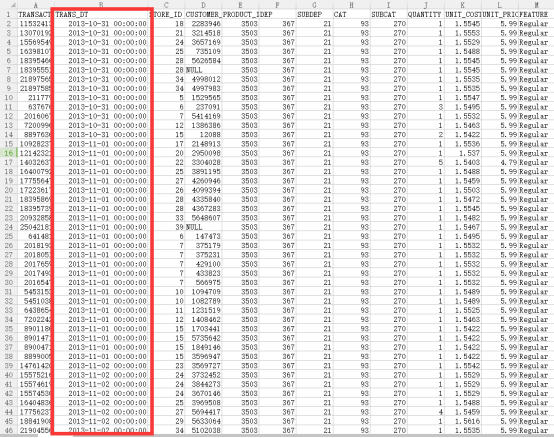
“工具”--“SQL脚本”：可通过SQL脚本编辑器编写SQL语句；

“数据预处理”--“拆分”：可以对数据按比例或按阀值进行拆分；

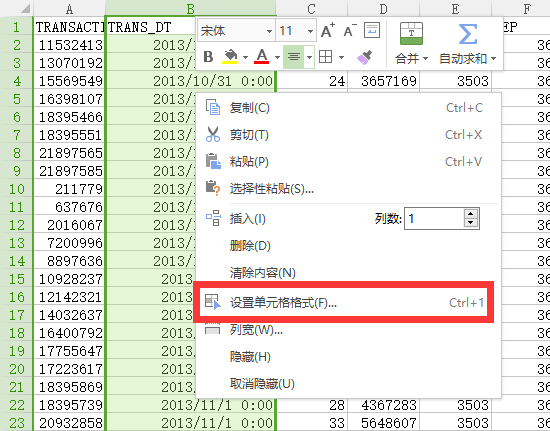
“数据预处理”--“缺失值填充”：可以对数据的缺失值进行均值、中位数或按特定规则进行填充。

**1.将csv文件上传至MaxCompute**

从附件中下载到本地的csv文件如下所示：

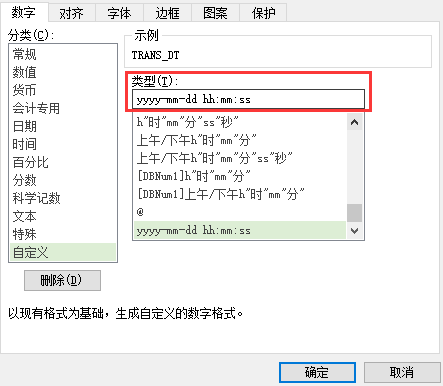


**【注】**TRANS\_DT列应为上图的格式。如果是其他格式，则先鼠标左键单击上方的B选中该列，右键单击“设置单元格格式”。



在数字类型选择自定义，在类型处输入‘yyyy-mm-dd hh:mm:ss’。然后点确定然后关闭窗口。

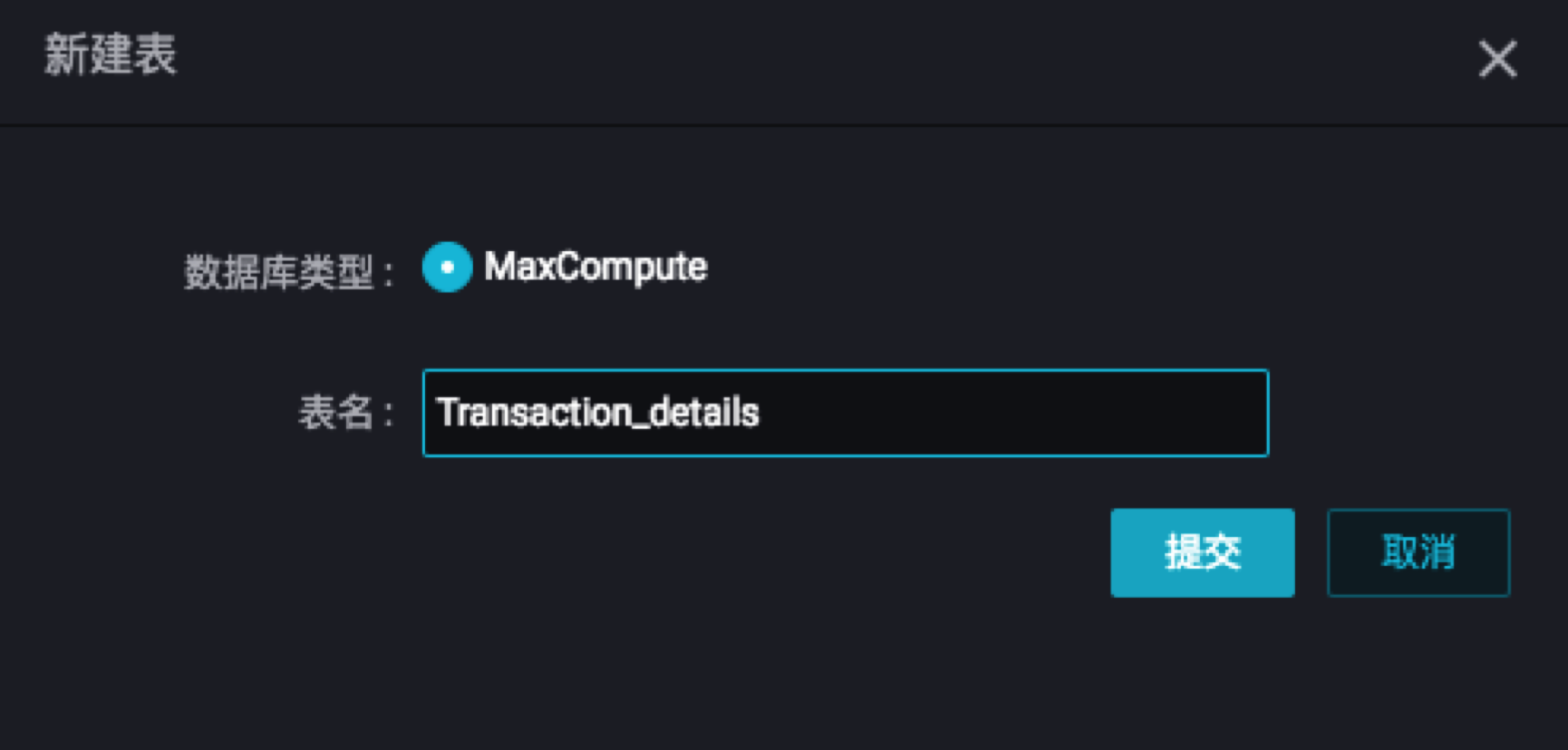
这个格式是Data IDE 可以识别的DateTime格式。



进入DataStudio（原数据开发）页面，选择新建 > 表。



填写新建表对话框中的表名。



单击提交。进入新建表页面，选择DDL模式，在DDL模式对话框中输入建表语句，单击生成表结构。

**CREATE** **TABLE** transaction\_details

(

TRANSACTION\_ID **string**,

TRANS\_DT datetime,

STORE\_ID **string**,

CUSTOMER\_ID **string**,

PRODUCT\_ID **string**,

DEP **string**,

SUBDEP **string**,

CAT **string**,

SUBCAT **string**,

QUANTITY int,

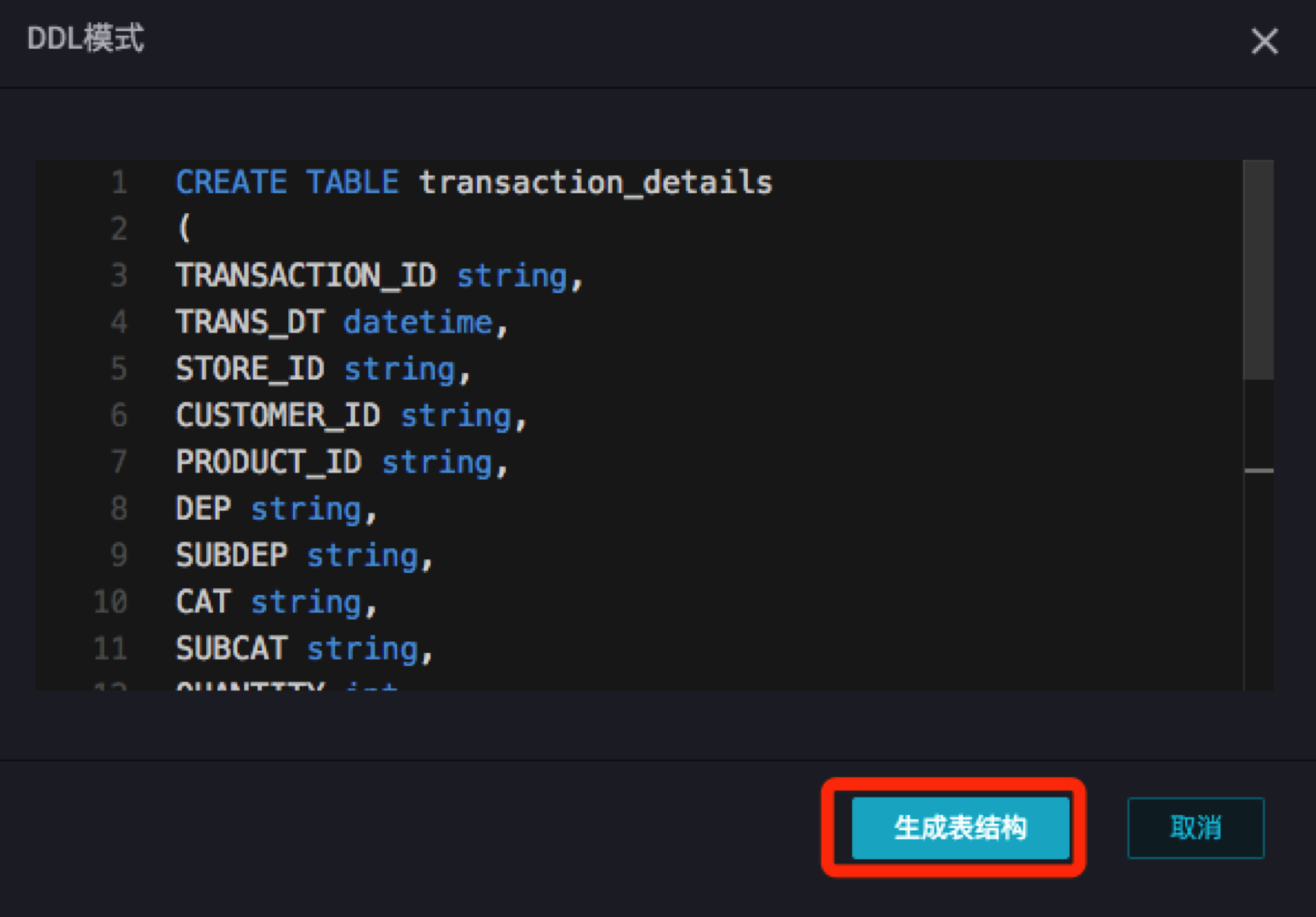
UNIT\_COST **double**,

UNIT\_PRICE **double**,

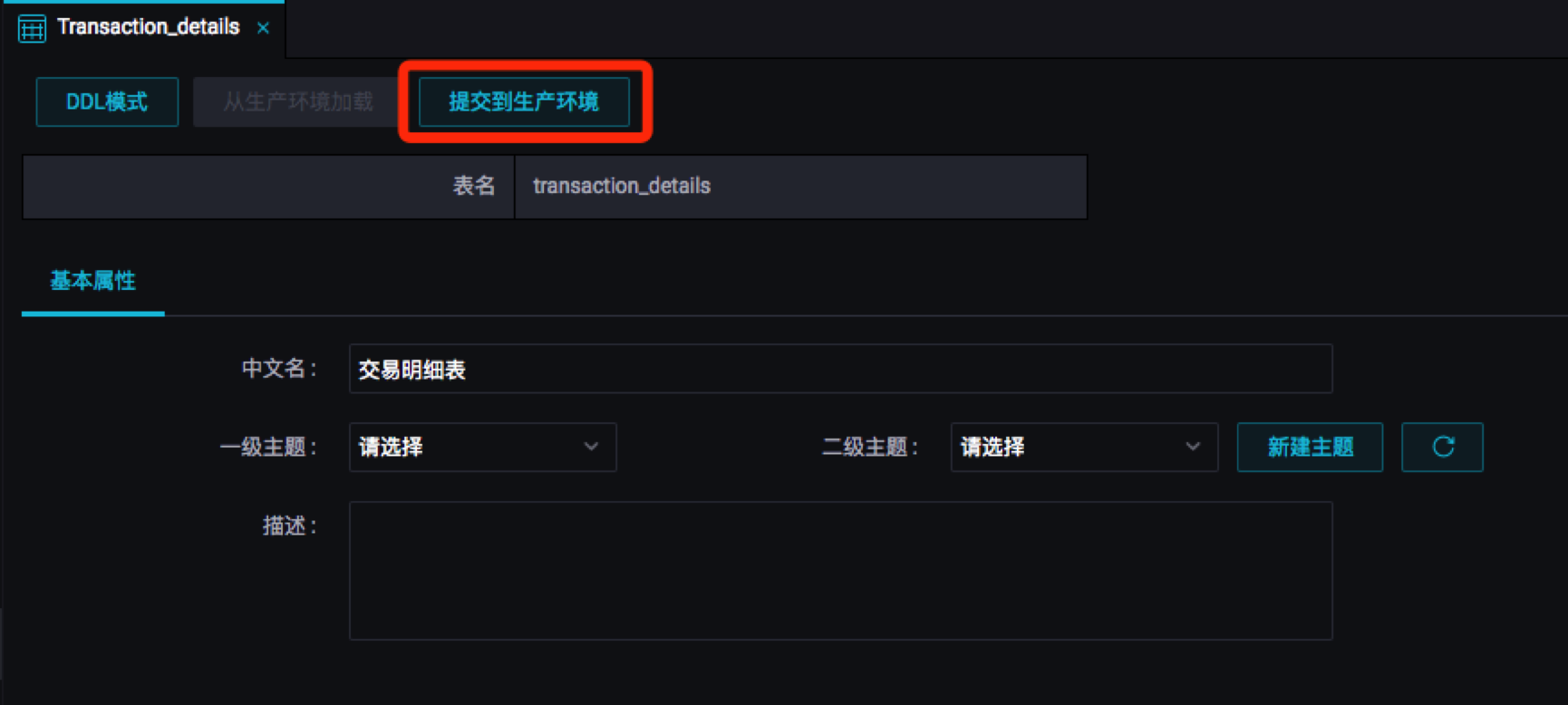
FEATURE **string**

);

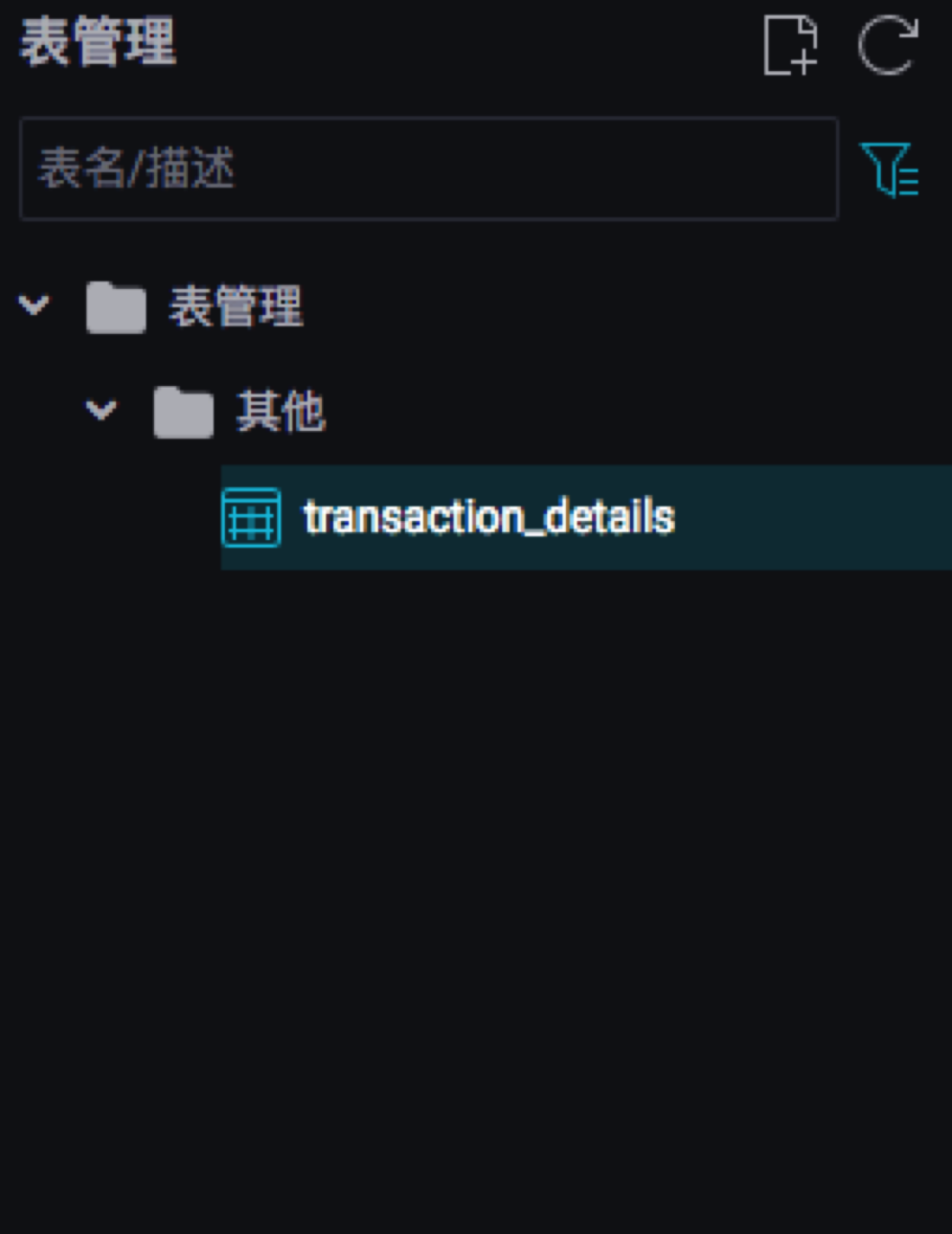
|  |
| --- |
| CREATE TABLE transaction\_details  (  TRANSACTION\_ID string,  TRANS\_DT datetime,  STORE\_ID string,  CUSTOMER\_ID string,  PRODUCT\_ID string,  DEP string,  SUBDEP string,  CAT string,  SUBCAT string,  QUANTITY int,  UNIT\_COST double,  UNIT\_PRICE double,  FEATURE string  ); |



表结构生成后，输入表的中文名，并单击提交到开发环境。



创建成功后，您可在左侧导航栏表管理中输入表名进行搜索，查看表信息。



接下来将本地csv数据上传至transaction\_details。

DataWorks支持以下操作：

将保存在本地的文本文件中的数据上传到工作空间的表中。

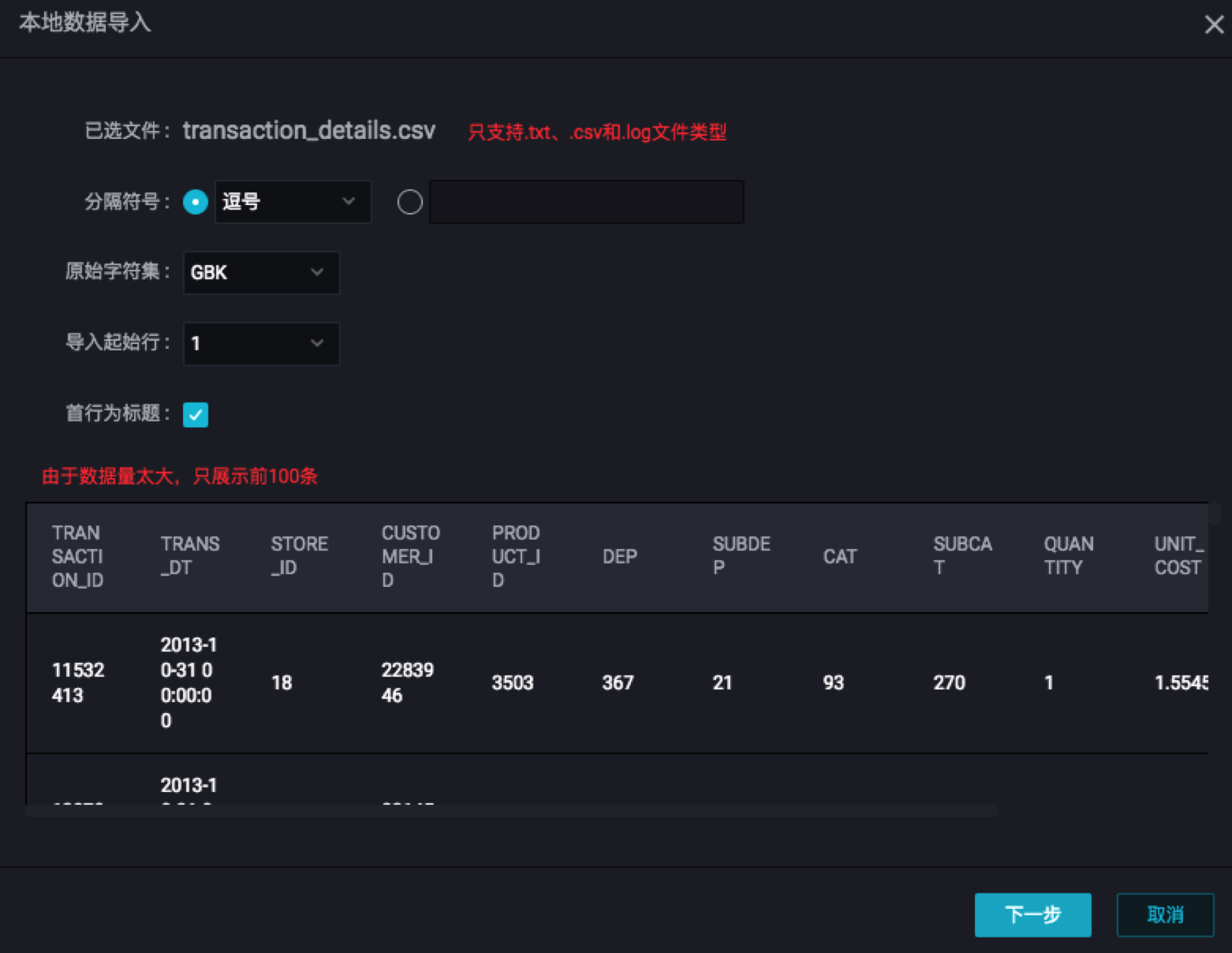
通过数据集成模块将业务数据从多个不同的数据源导入到工作空间。

导入csv文件的操作如下：

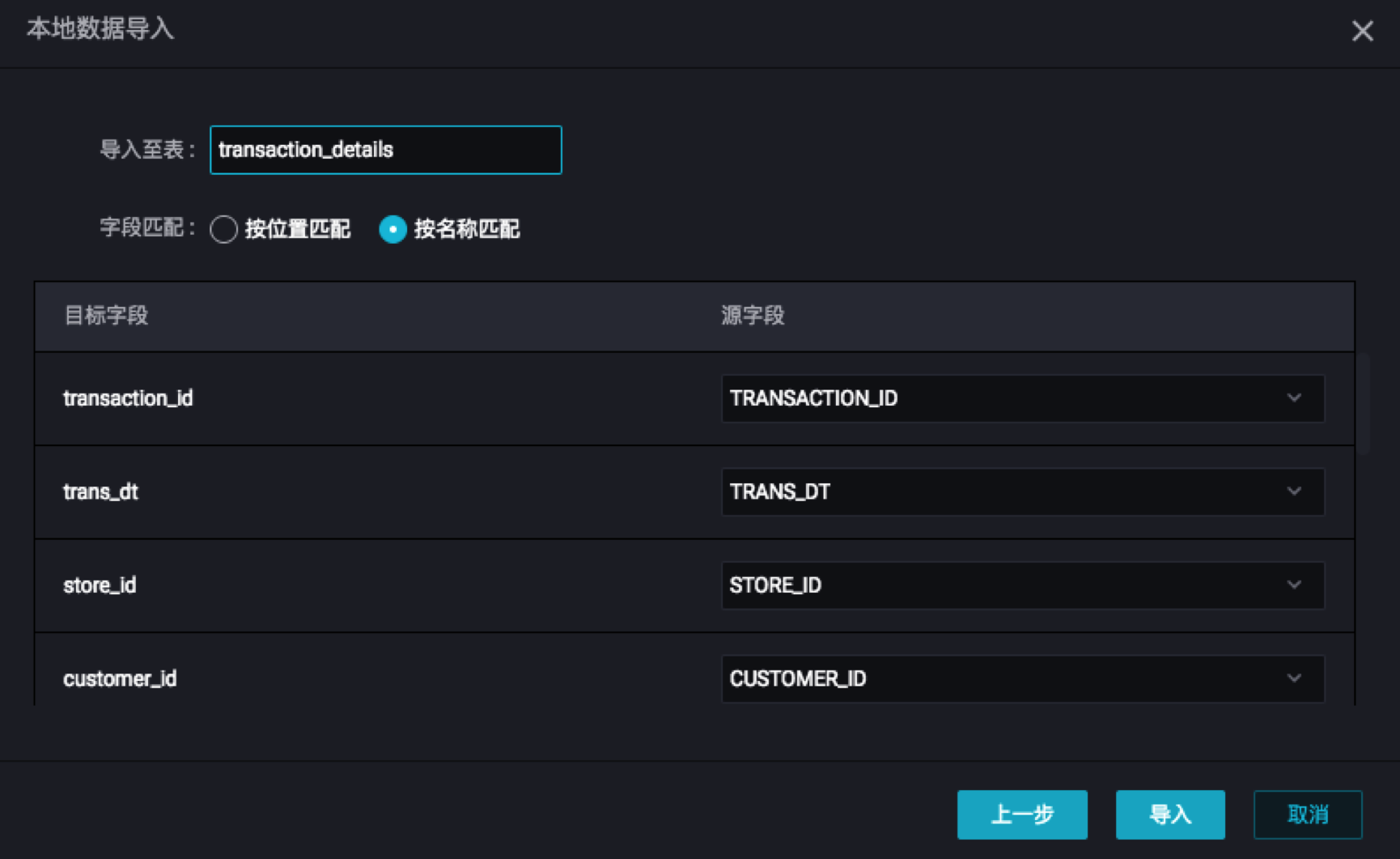
单击导入，选择导入本地数据。



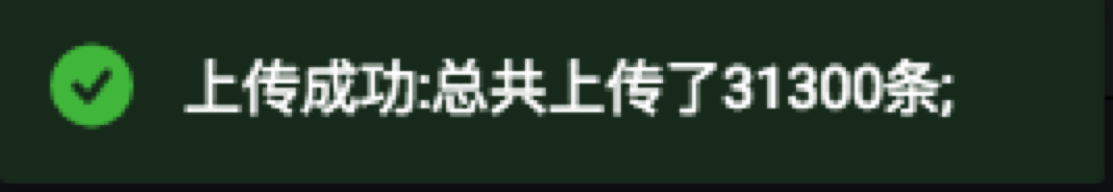
选择本地数据文件，配置导入信息，单击下一步。



至少输入2个字母搜索表名，选择需导入数据的表transaction\_details，单击导入。

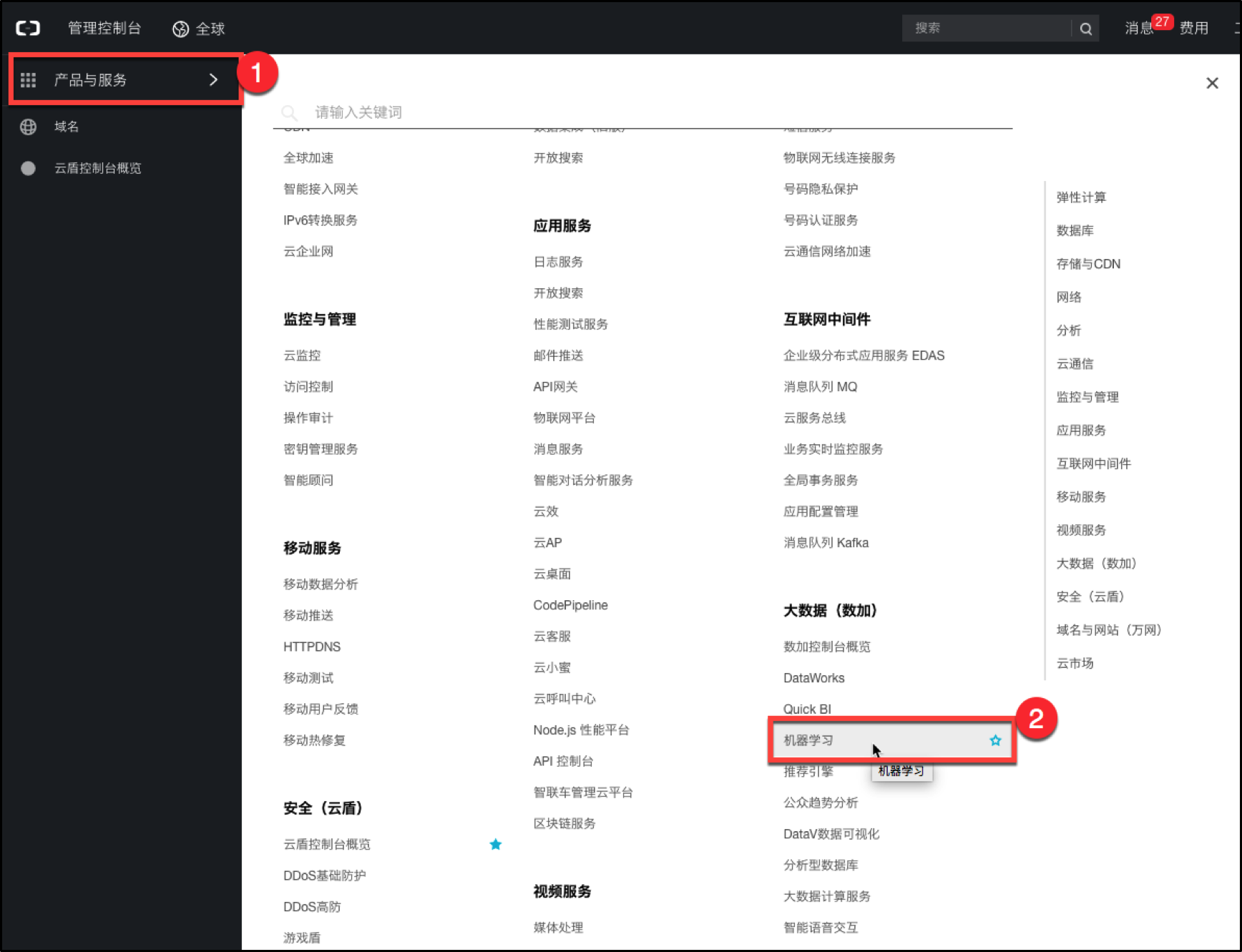


随后界面右上角会显示导入成功。



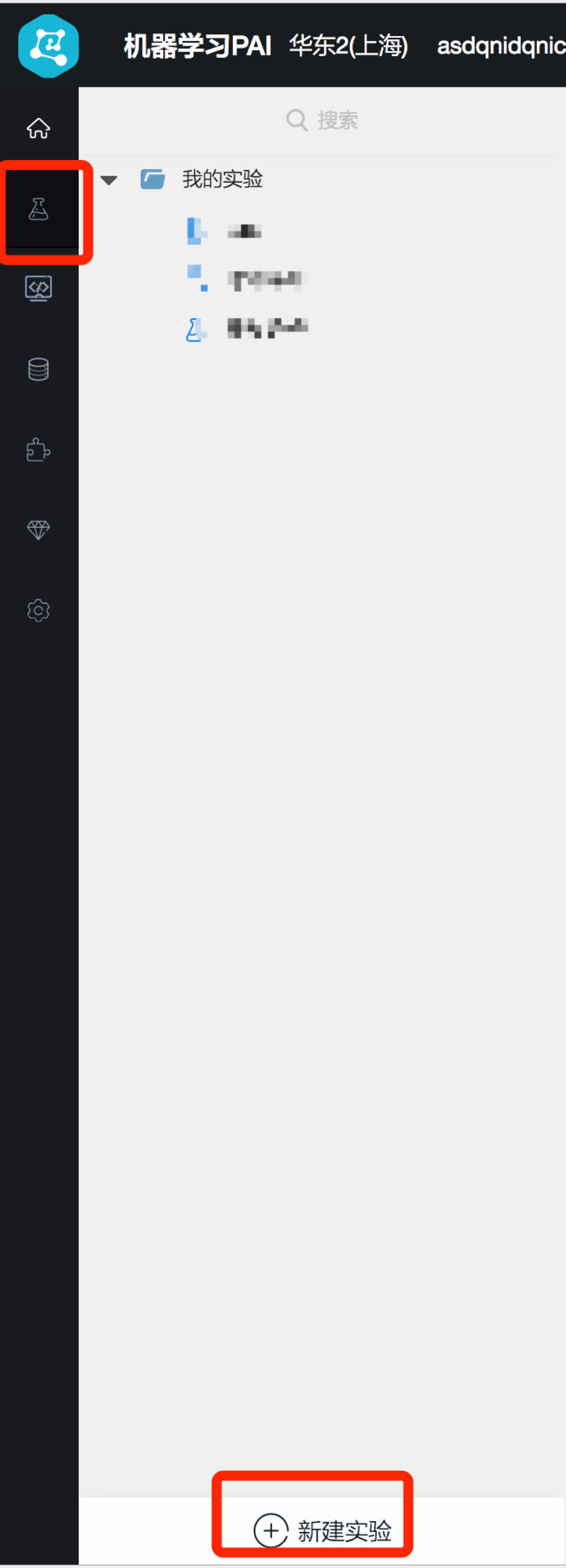
**2.导入实验数据**

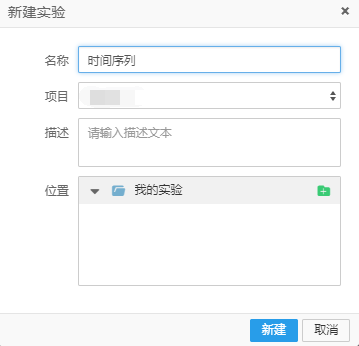
接下来的实验还需要进入机器学习平台，可以左键单击大数据(数加)下的机器学习，进入机器学习的控制台。





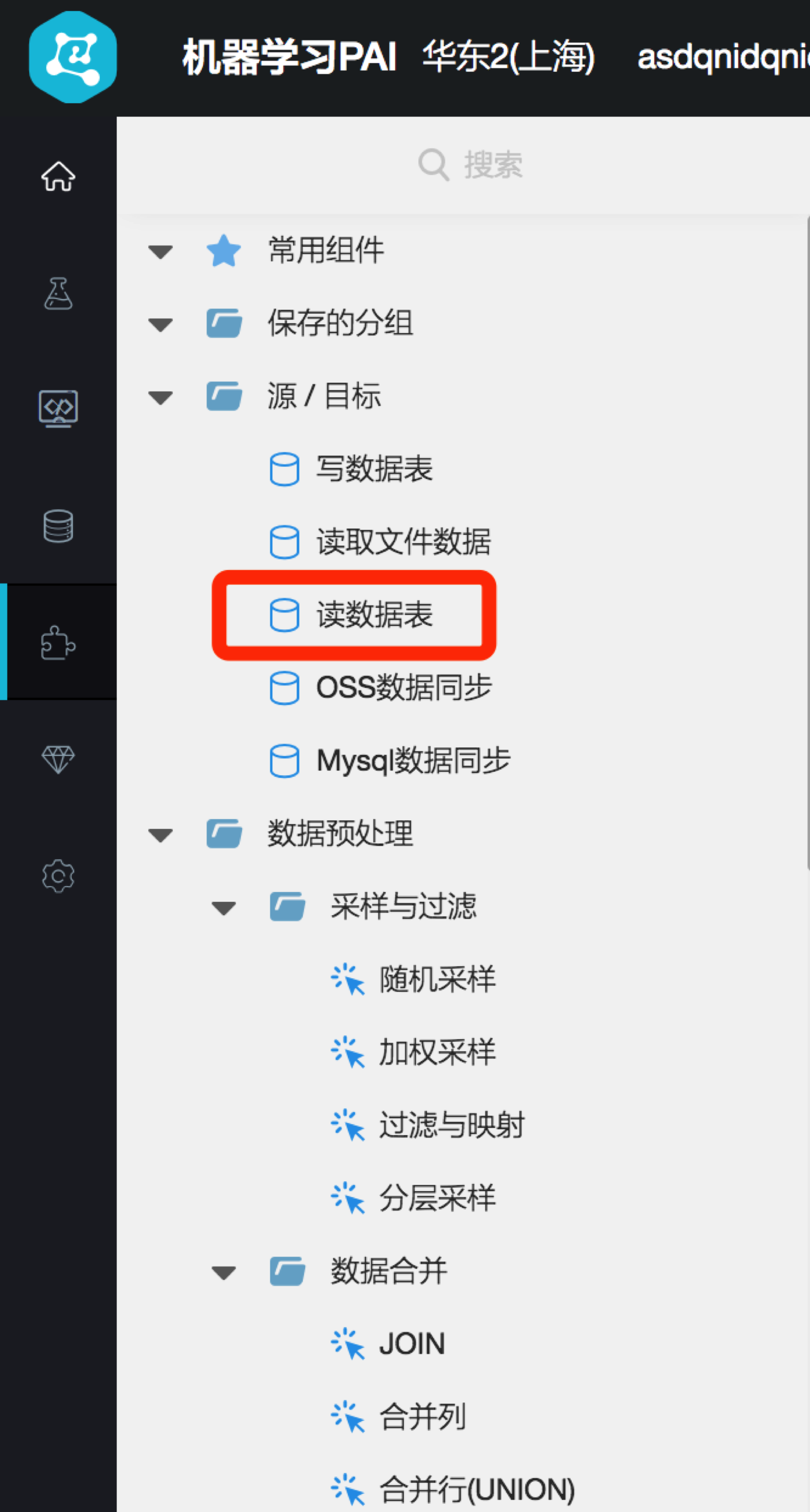
进入机器学习平台之后，左键单击实验图标--“新建空白实验”，输入任意名称。



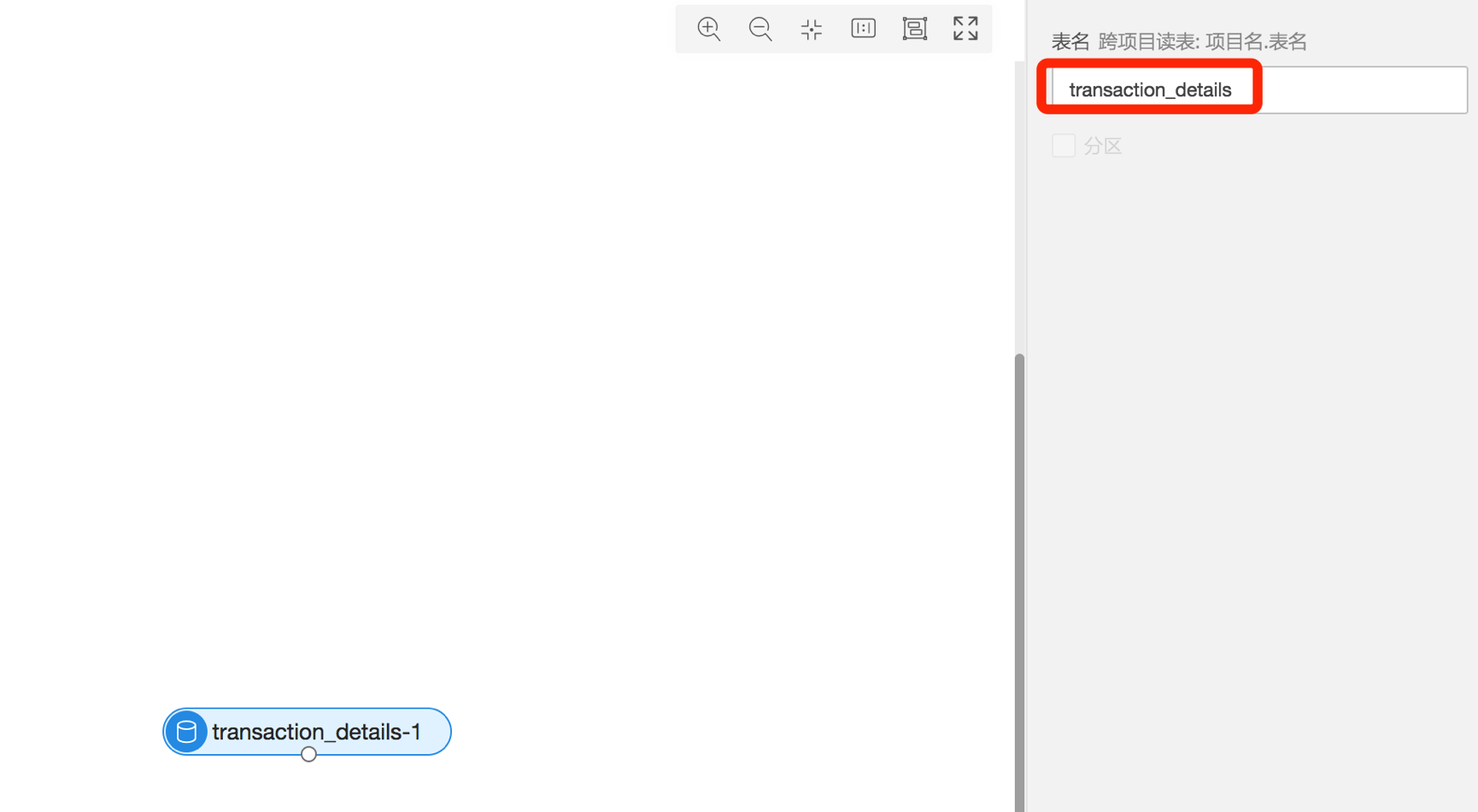


首先，我们先导入实验需要用到的数据表：transaction\_details。

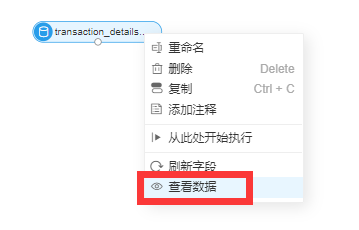
在PAI机器学习平台左侧左键单击“组件”--“源/目标”--“读数据表”，将其拖至工作区

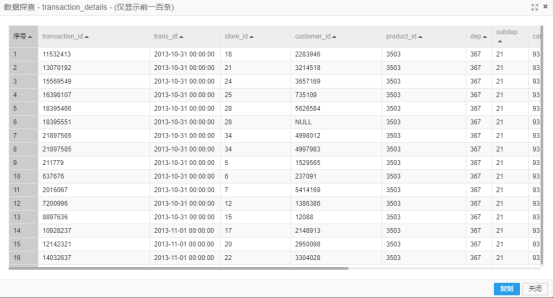


左键单击“读数据表”，左键单击“transaction\_details”，搜索’transaction\_details’并按回车键。



在工作区中鼠标右键点击刚刚拖进来的表，左键单击“查看数据”。





本次实验所用到的数据一共有13列，各变量的含义如下表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 变量 | 含义 | 变量 | 含义 |
| TRANSACTION\_ID | 交易ID | CAT | 分类 |
| TRANS\_DT | 交易日期 | SUBCAT | 子分类 |
| STORE\_ID | 商店ID | QUANTITY | 销量 |
| CUSTOMER\_ID | 顾客ID | UNIT\_COST | 单位成本 |
| PRODUCT\_ID | 产品ID | UNIT\_PRICE | 单价 |
| DEP | 部门 | FEATURE | 销售类型 |
| SUBDEP | 子部门 |  |  |

由于本次实验仅针对一个产品，所以仅需要用到如下字段：

TRANS\_DT、PRODUCT\_ID、QUANTITY、UNIT\_PRICE、FEATURE。

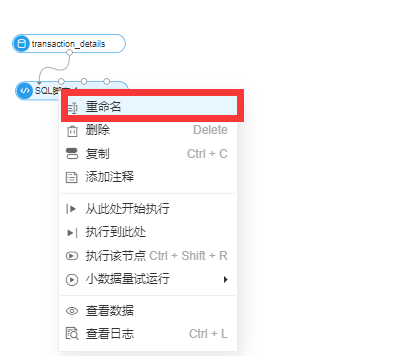
**3.聚合原始数据**

本次实验我们选取一个月为一个周期对产品的销量进行预测。

首先，我们需要对原始数据进行聚合，将交易数据整理为按月聚合的表。

**【注】**其实是按四周28天聚合，一年13个月，方便起见写成按月聚合。后面不再赘述。

在工作区左侧左键单击“工具”--“SQL脚本”，将其拖拽至工作区，右键单击重命名为“按月聚合”。



左键单击“按月聚合”，并在工作区右侧“按月聚合”中输入如下代码。随后，左键单击“transaction\_details”下方的圆口，按住鼠标，将其拖至“按月聚合”最左侧的圆口。

select

monthno,trans,custs,sales,unit\_cost,unit\_price,quantity

,flyer/quantity as flyer

,markdown/quantity as markdown

,other\_events/quantity as other\_events

,regular/quantity as regular

from

(select product\_id

, monthno

, count(distinct transaction\_id) as trans

, count(distinct customer\_id) as custs

, sum(unit\_price \* quantity) as sales

, avg(UNIT\_COST) as unit\_cost

, avg(UNIT\_price) as unit\_price

, sum(quantity) as quantity

, sum(case when feature = 'Flyer' then quantity else 0 end) as flyer

, sum(case when feature = 'Markdown' then quantity else 0 end) as markdown

, sum(case when feature = 'Other\_Events' then quantity else 0 end) as other\_events

, sum(case when feature = 'Regular' then quantity else 0 end) as regular

from (select ceil(DATEDIFF(TRANS\_DT, '2013-10-31 00:00:00', "dd")/28) as monthno

, transaction\_id, product\_id, customer\_id, unit\_price, UNIT\_COST, quantity, feature

from ${t1}

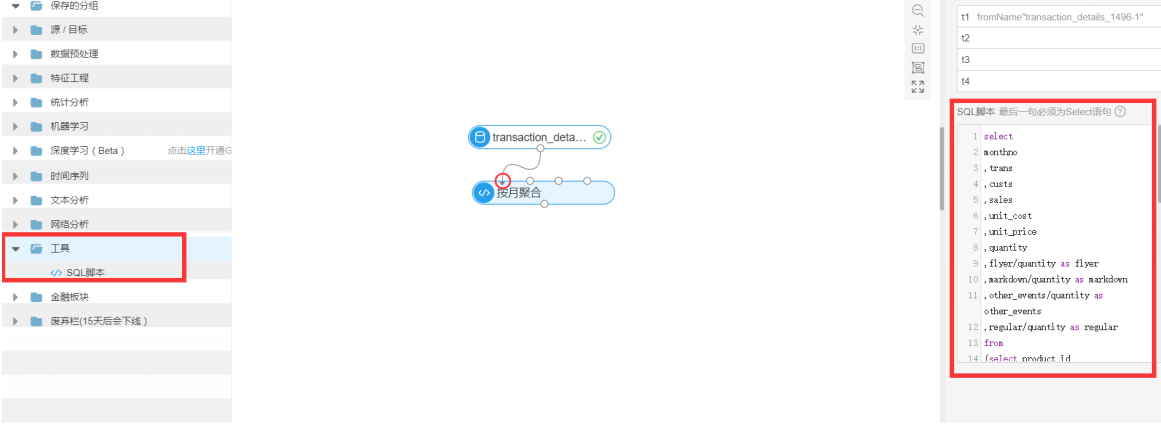
WHERE quantity > 0

and trans\_dt > '2013-10-31 00:00:00'

and product\_id = '3503'

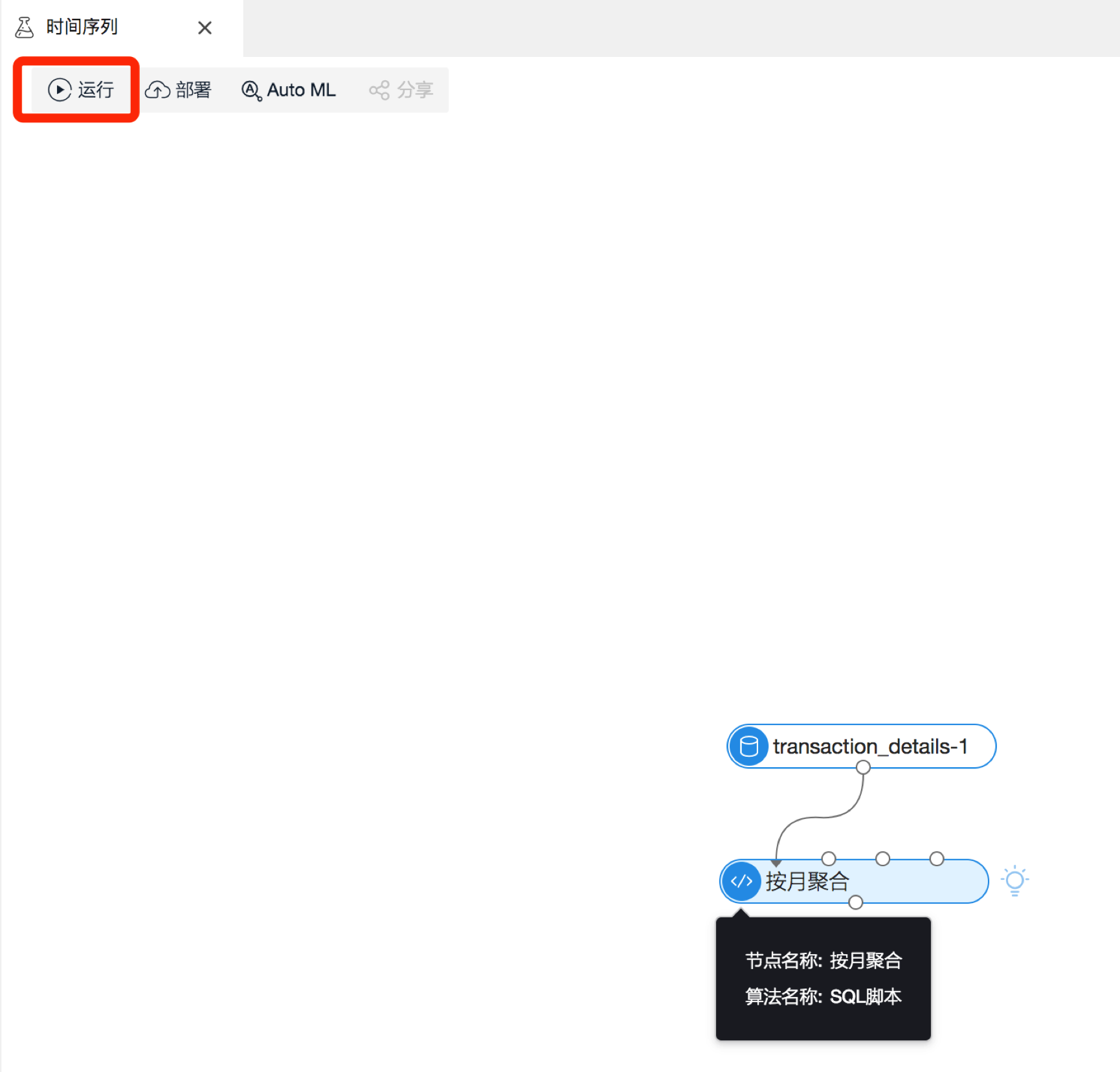
) a

group by product\_id, monthno) b

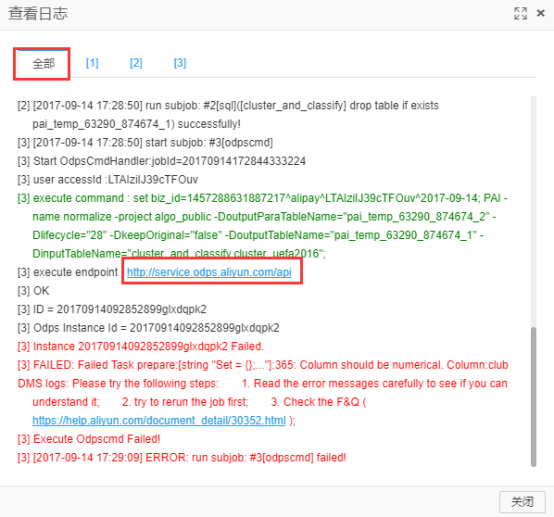


这段代码根据原始数据的交易日期生成了交易日期属于第几个月（选取2013-10-31至2013-11-27为第一个月），以月为单位聚合。新增销售类型的变量“flyer”、“markdown”、“other\_events”、“regular”。对于销售类型：flyer（海报）、markdown（降价）、regular（常规）、other（其他），计算了每个月销售数据中各种销售类型所占的百分比。

左键单击左上角的“运行”：



如果运行失败，右键单击该节点，左键单击“查看日志”。



左键单击“全部”标签下的最后一段链接即可跳转到日志页面。

**【注】**日志有所差异的话是正常的，日志的详细说明文档参见

https://help.aliyun.com/document\_detail/27987.html?spm=5176.product27797.6.747.ky2HKF

再次拖入一个“SQL脚本”，重命名为“计算促销幅度%”，将其第一个输入端口连接至“按月聚合”，并输入如下代码：

select \*

,(avg\_price - unit\_price)/avg\_price as pcnt\_price

from

(select \*

,avg(unit\_price) over(partition by 1) as avg\_price

from ${t1}) a

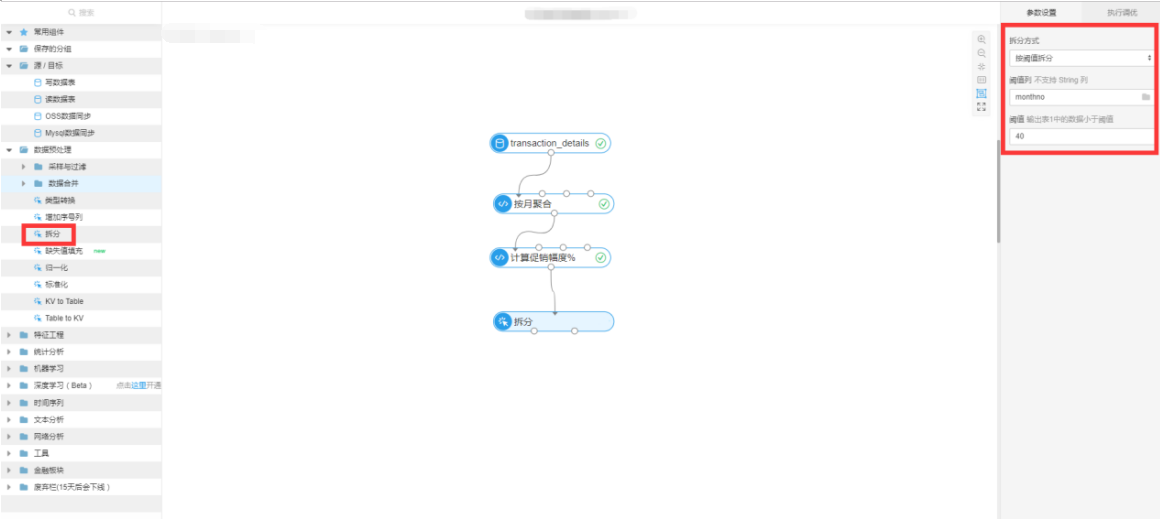
这两段代码筛选了id为'3503'的产品，计算了产品的平均价格，并定义产品每月的促销幅度%=（平均价格-产品价格）/平均价格。

鼠标右键单击“计算促销幅度%”，左键单击“执行到此处”：



**4.选取训练集**

从左侧组件找到“数据预处理”--“拆分”，将其拖入工作区，连接至“计算促销幅度%”。左键单击“拆分”，在工作区右侧“参数设置”中，将“拆分方式”改为“按阀值拆分”，“阀值列”选择“monthno”，“阀值”设为“40”。



随后右键单击“拆分”，左键单击“执行该节点”。

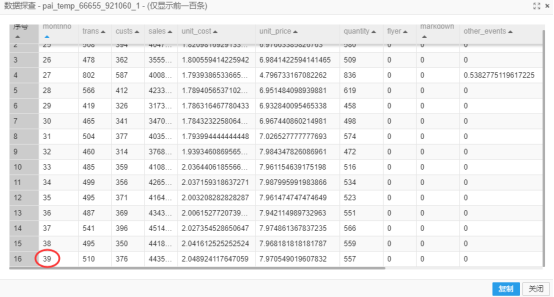
这个组件将实验原始数据取前39个月的数据为训练集（从“拆分”组件左侧端口输出）。

取1个月为28天时，一年取13个月，共13×28=364天。因为计算时间序列移动平均需要第一年的均值做下一年第一个月的移动平均，因此训练集实际上是第14个月到第39个月，共两年的数据。

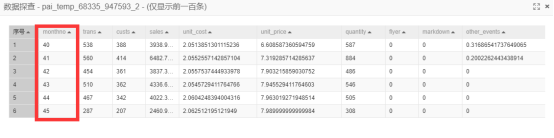
右键单击“拆分”，左键单击“查看数据”--“查看输出桩1”



左键单击monthno下方的小三角，使其按monthno顺序排序。拉到最下方，可以看到数据只到第39个月，说明拆分成功，得到了训练集.。



再次右键单击“拆分”，左键单击“查看数据”--“查看输出桩2”

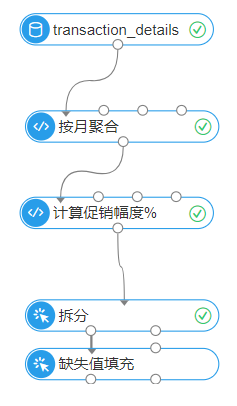


后面会对这部分数据再次拆分以选取测试集。

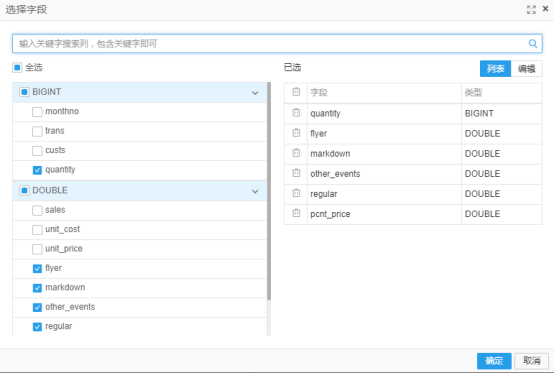
**5.缺失值填充**

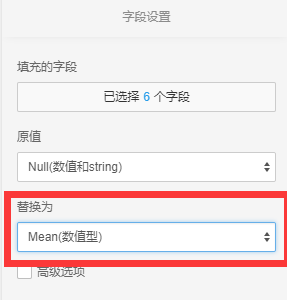
由于销售数据是以月为单位进行划分的，如果选定的商品在某月的销量为0，可能会导致时间序列中出现了空缺值。对于存在缺失值的部分字段，可以使用缺失值填充组件进行自动填充。

在工作区左侧找到“数据预处理”--“缺失值填充”，将其拖至工作区，连接至“拆分”。



鼠标左键单击“缺失值填充”，在右侧“字段设置”--“填充的字段”中选择quantity、flyer、markdown、other\_events、regular、pcnt\_pirce这6个字段。将“替换为”设置为“Mean（数值型）”。





右键单击“缺失值填充”，左键单击“执行该节点”。