阿里云
DataV数据可视化

最佳实践

文档版本：20190819

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
	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。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能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，恢复业务所需时间约10分钟。
	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。	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。	单击 **确定**。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	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。	`swich` *`{stand | slave}`*

目录

1 在DataV中展示日志服务数据

本文档为您介绍如何使用DataV完成日志服务数据的展示，实现实时业务监控。

概述

本案例的整体步骤如下。

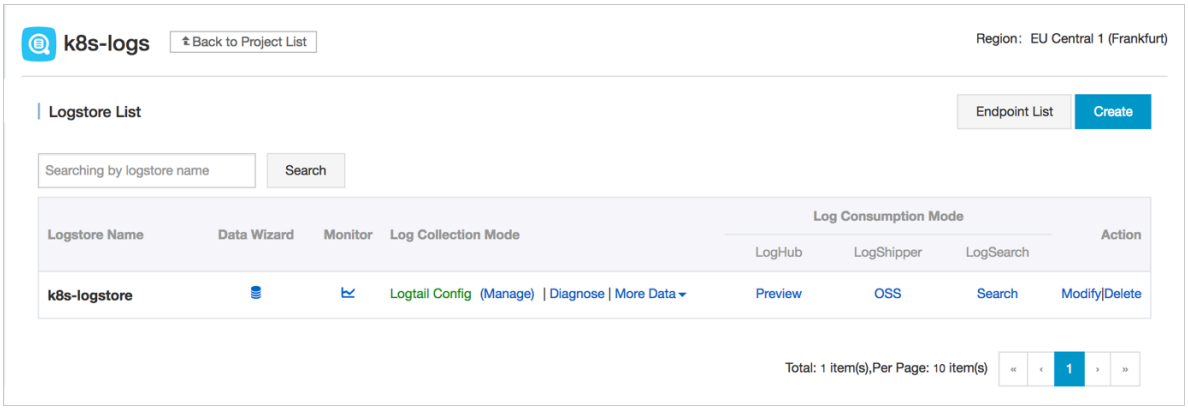
1. 准备工作。
2. 配置日志服务。
3. 配置DataV。

准备工作

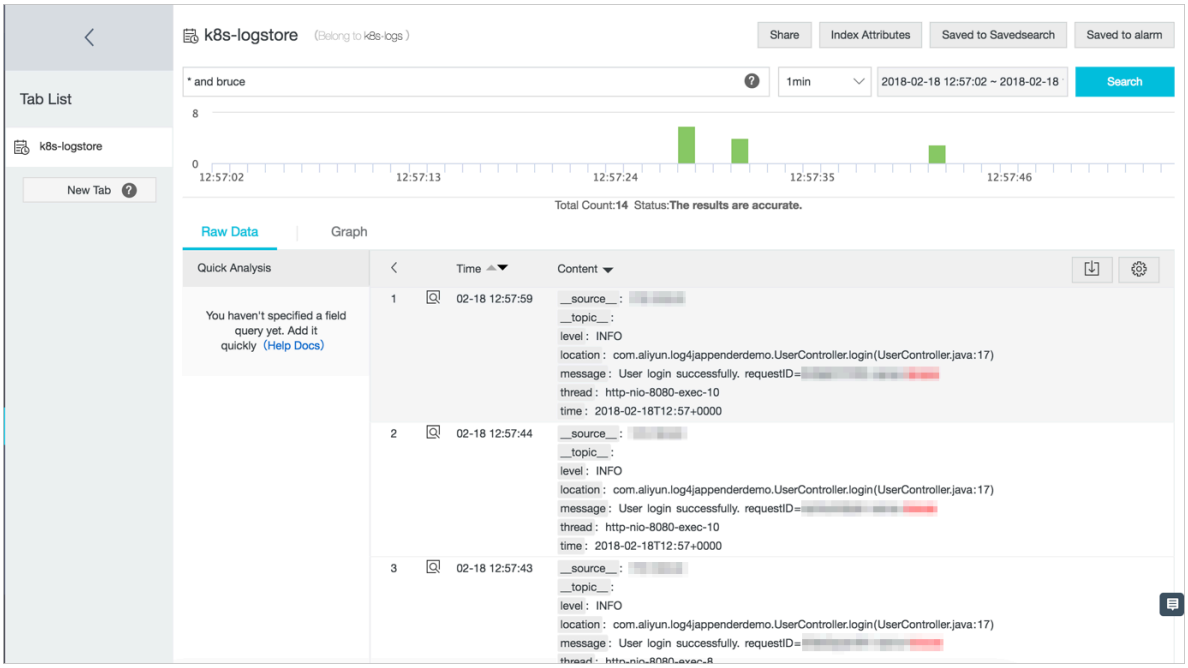
- 完成为Kubernetes和日志服务配置Log4JAppender中的步骤，并且服务运行正常。
- 购买DataV企业版。

配置日志服务

1. 登录日志服务控制台，查看Logstore列表。



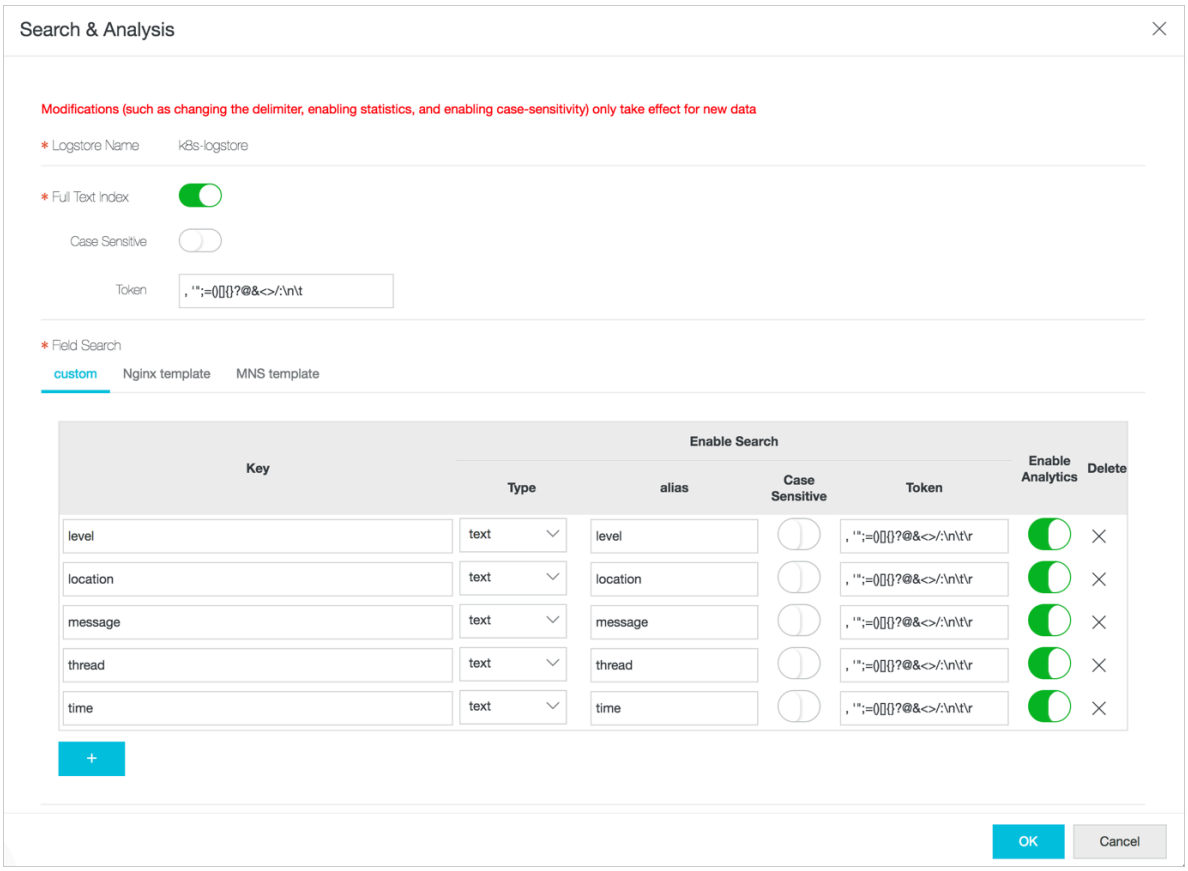
2. 单击列表中的查询，出现如下界面。



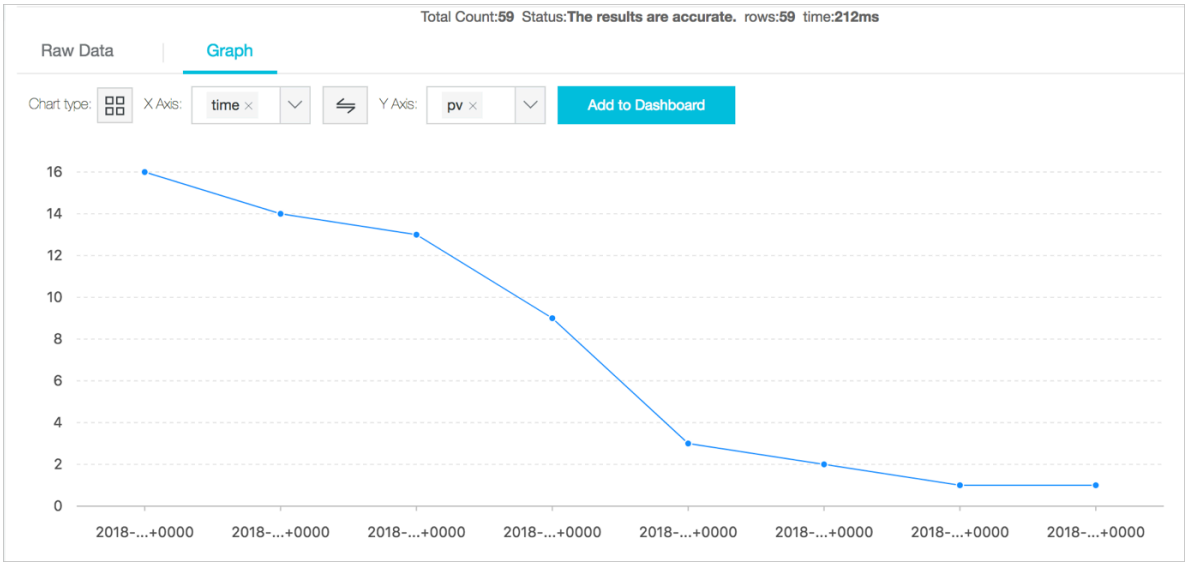
3. 为所有必填字段创建索引。

   单击页面右上角菜单栏的查询分析属性，选择设置，为每个项目创建索引。

4. 在查询分析页面验证数据。



5. 数据导入成功后，单击左侧菜单栏的仪表板，切换至图形页面（下图中的X轴为时间）。



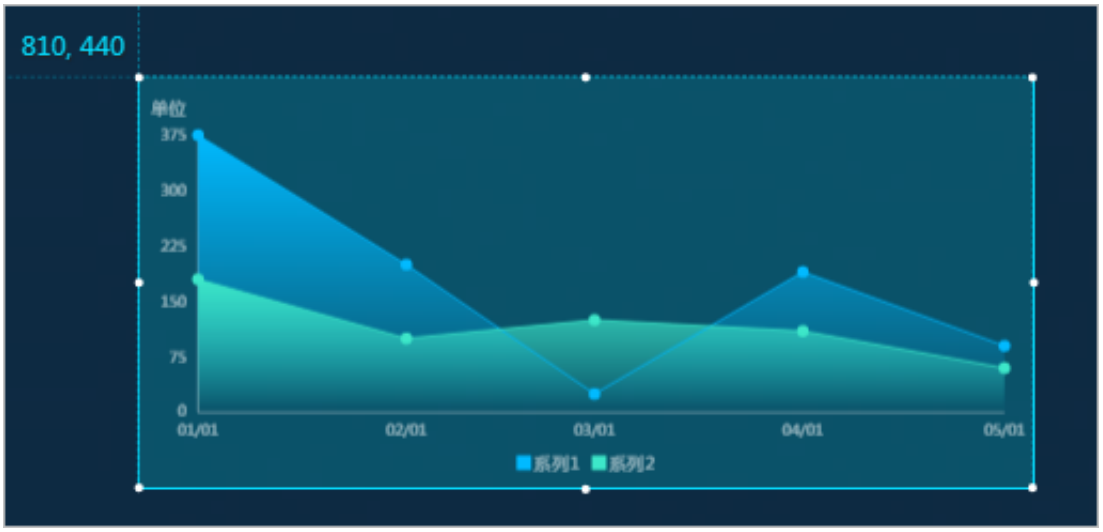
配置DataV

1. 登录DataV控制台。

2. 单击创建项目，选择一个空白模板并输入项目名称，单击创建。

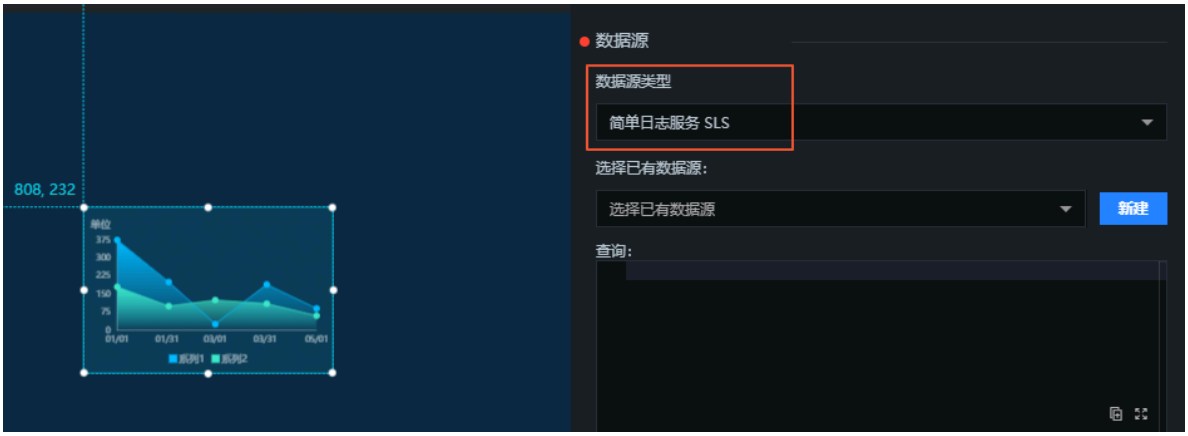
3. 在画布中添加一个组件。

该组件默认显示静态数据集。



4. 单击该组件，在右侧的数据面板中，单击配置数据源。

5. 在设置数据源页面中，选择数据源类型为简单日志服务 SLS。



6. 单击新建，在添加数据对话框中，填入数据源相关信息，单击确定。

参数	说明
数据源名称	数据源的显示名称，您可以自由命名。
AppKey	拥有目标SLS访问权限的账号的AccessKey ID。
AppSecret	拥有目标SLS访问权限的账号的AccessKey Secret。
EndPoint	填写SLS服务的EndPoint。请参考日志#unique_8文档，根据您SLS服务的网络类型和所在区域进行填写。例如VPC网络下，上海区域的EndPoint填写为https://cn-shanghai-intranet.log.aliyuncs.com。 说明: Endpoint前需要添加http://或者https://。

7. 选择添加完成的数据源，并在数据查询框中输入如下示例脚本进行查询。

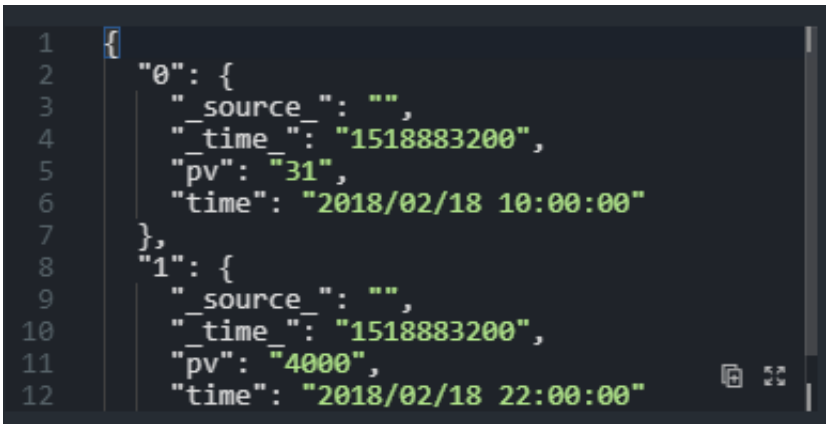
```
{
```

```
"projectName": "k8s-logs",
"logStoreName": "k8s-logstore",
"topic": "",
"from": "1518883200",
"to": "1518969600",
"query": "* | select count(1) as pv, date_format(from_unixtime(
__time__ - __time__%3600) ,'%Y/%m/%d %H:%i:%s')   as time group by
time  order by time limit 1000" ,
"line": 100,
"offset": 0
}
```

说明:

from和to是时间戳，可以用来检查查询中的原始数据。

8. 单击预览数据源返回结果，结果如下图所示。

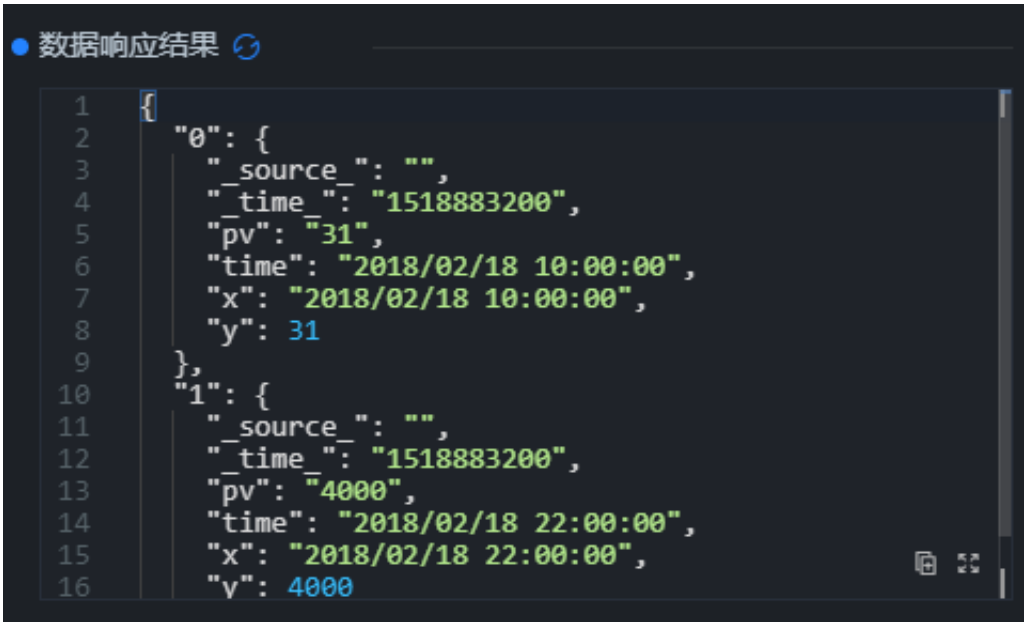


9. 勾选使用过滤器，添加如下的过滤器，确保pv为整数，单击确定。

```
return Object.keys(data).map((key) => {
 let d= data[key];
 d["pv"] = parseInt(d["pv"]);
 return d;
}
```

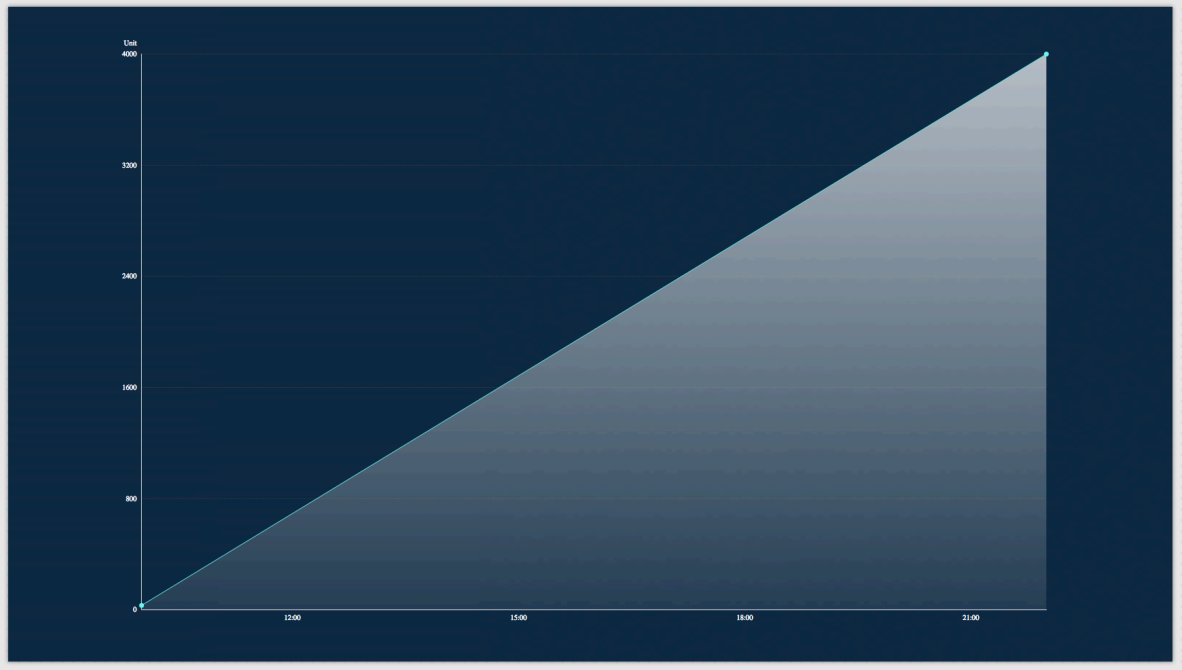
```
)
```

10.设置坐标轴并验证是否正确。



11.单击预览。

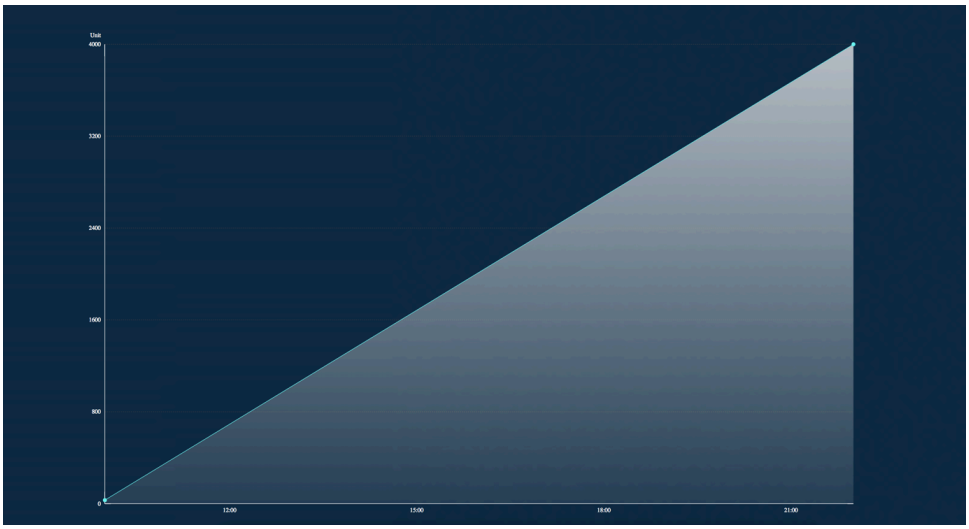
您可以看到x和y使用正确的数据类型，并且pv为整数。



12.单击界面右上角的发布，发布该大屏项目，使得您可以公开访问该大屏。

查看结果

本案例的发布结果如下图所示。



您已在阿里云上成功配置DataV和Log Service，并使用Log Service通过自定义DataV可视化大屏实现了实时监控。

参考文档

有关日志服务和容器服务的更多信息，请参见：

- 日志服务

- 容器服务

2 使用DataV查看春节前后空气质量的全国分布变化

2.1 教程概述

本章节为您介绍使用DataV大屏，展示春节前后空气质量的全国分布变化的方法。

本案例的整体操作步骤如下。

1. 准备工作。
   a. 了解相关功能。
   b. #unique_12。
   c. #unique_13。
   d. #unique_14。
2. 制作大屏。
   a. #unique_15。
   b. #unique_16。
   c. #unique_17。
3. #unique_18。

2.2 准备工作

2.2.1 了解相关功能

本文档为您介绍在执行本案例的操作前，需要了解的相关功能。

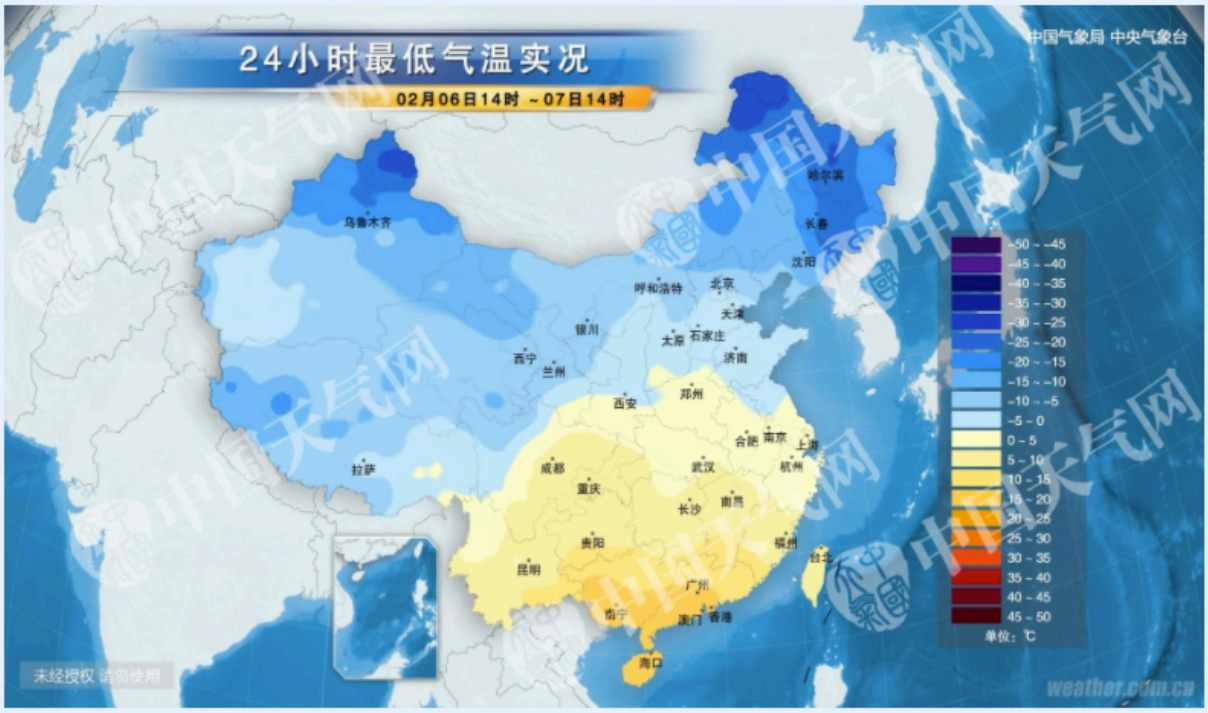
制作大屏时，您可能需要用到以下几种功能。

- 空间插值。
- 等值面组件。
- 时间轴组件。

空间插值

空间插值常用于将离散点的测量数据转换为连续的数据曲面，以便与其它空间现象的分布模式进行比较。

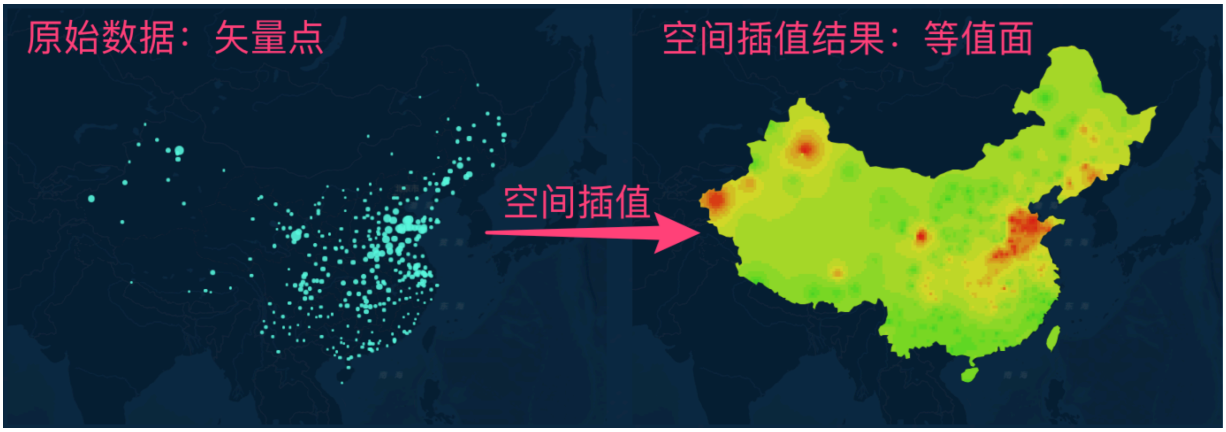
通过空间差值，您可以根据已知的监测站点监测出的数据，去推算其他任意空间位置的数据。再根据数值处在的不同区间范围，去映射对应的颜色，就可以得到一张典型的等温面图，如下图所示。



【图形解读】

气温体现了空气的冷热程度。地面气象观测中测定的是离地面1.5m高度处的气温。同时为了避免辐射误差，测量仪器为放置在避光且通风的百叶箱内。

如果用DataV来制作一张等温面图，就可以很清楚地看到，空间插值就是根据离散的已知点去插值出连续的面数据，如下图所示。



等值面组件

DataV提供了一个轻分析的等值面地图组件，帮助您将已知的矢量点数据制作成栅格区域图。您可以使用等值面地图组件，实时插值出全国的空气质量图，如下图所示。

时间轴组件

通过时间轴组件，您可以查看一段时间内的空气质量变化。

时间轴组件支持回调ID。您可以通过配置时间轴的回调ID，实现组件联动。即当时间轴的时间发生变化时，其他组件的数据也会自动更新。当填写了正确的回调ID后，系统会在每次时间变化的时候重新触发一次数据请求，并自动在其它组件所对应的API接口的参数列表中加上当前的回调ID，以及其对应的值。示例如下：

- 初始接口地址：`http://127.0.0.1:8888/aqi`。
- 回调触发后：`http://127.0.0.1:8888/aqi?date=2017012722`。

以上示例的回调ID为`date,2017012722`。

> 说明：
> 回调ID也支持SQL语句，您需要在SQL语句中，使用`:`加上回调ID名称来使用回调ID。
>
> - 初始SQL：`select :date as value;`。
> - 回调触发后：`select '2017022722' as value;`。

2.2.2 获取数据

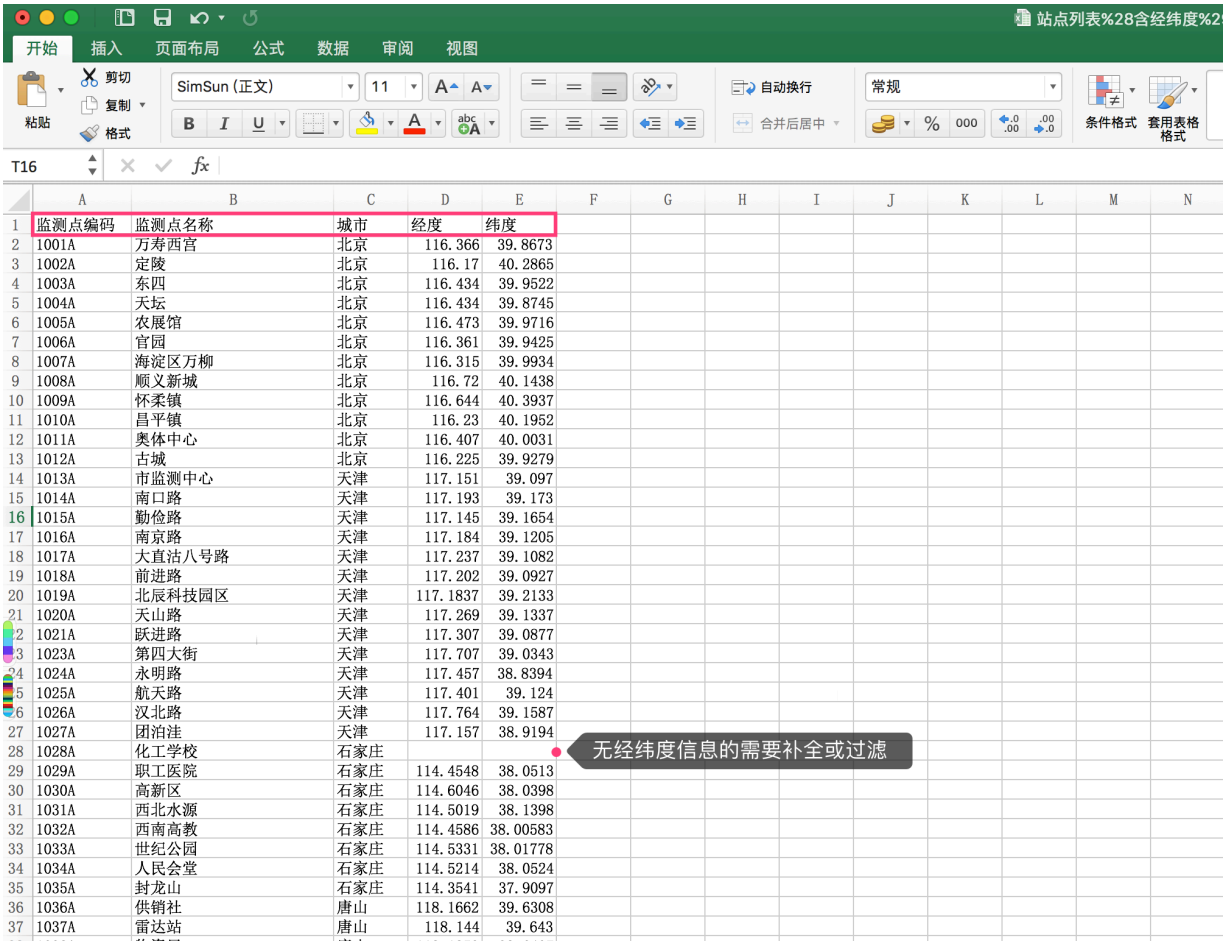
数据是可视化的原材料，本文档为您介绍获取春节期间全国的空气质量数据的方法。

您可以从全国空气质量历史数据上面下载需要的数据。

> 说明：
> 推荐下载格式为.csv格式的文件。

本示例中，采用了2017年1月1日至2017年2月2日，全国1497个监测点的数据。

下载完成后，打开文件，查看是否有需要补全或者需要过滤的数据。



	A	B	C	D	E
1	监测点编码	监测点名称	城市	经度	纬度
2	1001A	万寿西宫	北京	116.366	39.8673
3	1002A	定陵	北京	116.17	40.2865
4	1003A	东四	北京	116.434	39.9522
5	1004A	天坛	北京	116.434	39.8745
6	1005A	农展馆	北京	116.473	39.9716
7	1006A	官园	北京	116.361	39.9425
8	1007A	海淀区万柳	北京	116.315	39.9934
9	1008A	顺义新城	北京	116.72	40.1438
10	1009A	怀柔镇	北京	116.644	40.3937
11	1010A	昌平镇	北京	116.23	40.1952
12	1011A	奥体中心	北京	116.407	40.0031
13	1012A	古城	北京	116.225	39.9279
14	1013A	市监测中心	天津	117.151	39.097
15	1014A	南口路	天津	117.193	39.173
16	1015A	勤俭路	天津	117.145	39.1654
17	1016A	南京路	天津	117.184	39.1205
18	1017A	大直沽八号路	天津	117.237	39.1082
19	1018A	前进路	天津	117.202	39.0927
20	1019A	北辰科技园区	天津	117.1837	39.2133
21	1020A	天山路	天津	117.269	39.1337
22	1021A	跃进路	天津	117.307	39.0877
23	1023A	第四大街	天津	117.707	39.0343
24	1024A	永明路	天津	117.457	38.8394
25	1025A	航天路	天津	117.401	39.124
26	1026A	汉北路	天津	117.764	39.1587
27	1027A	团泊洼	天津	117.157	38.9194
28	1028A	化工学校	石家庄		
29	1029A	职工医院	石家庄	114.4548	38.0513
30	1030A	高新区	石家庄	114.6046	38.0398
31	1031A	西北水源	石家庄	114.5019	38.1398
32	1032A	西南高教	石家庄	114.4586	38.00583
33	1033A	世纪公园	石家庄	114.5331	38.01778
34	1034A	人民会堂	石家庄	114.5214	38.0524
35	1035A	封龙山	石家庄	114.3541	37.9097
36	1036A	供销社	唐山	118.1662	39.6308
37	1037A	雷达站	唐山	118.144	39.643

2.2.3 处理数据

本文档为您介绍如何将CSV格式的文件处理成JSON格式的数据。

等值面组件需要的数据格式如下图所示。您需要对数据做进一步的加工处理，让其更符合DataV的数据规范。

- 裁剪面：研究区域的边界数据。这里是全国区域，是一个GeoJSON格式的数据。

  GeoJSON是一种地理交换格式，如需了解更多关于GeoJSON的内容，请参见GeoJSON标准。
- 插值点数据：示例数据是一个包含经度、纬度、值的数组，对应的需求为监测站点的经纬度和对应的某个指标的值。

> **说明:**
> 如果仅做一天的某个时段的等值面图，例如2017年1月20日的中午12点关于空气质量指数（AQI）的指标图，那么您需要明确当天这个时段，每个监测站点的位置（经纬度信息）和对应的AQI值。

通过以下步骤处理数据。

1. 使用以下的Node.js脚本处理全国监测站点的CSV格式文件。

   > **说明:**
   > 获取全国监测站点的CSV格式文件，请参见获取数据。

```
var csv = require("fast-csv");
var fs = require('fs');
var map = {};
csv
.fromPath("./站点列表(含经纬度)-新-1497个.csv", { headers: true,
objectMode: true })
.on("data", function (data) {
 map[data['code']] = data;
})
.on("end", function () {
 fs.writeFile('./站点列表经纬度映射.json', JSON.stringify(map));
```

```
 console.log("done");
});
```

得到监测站点编号为key，站点信息为value的字典。

```
{
 "1001A": {
     "code": "1001A",
     "name": "万寿西宫",
     "city": "北京",
     "lng": "116.366",
     "lat": "39.8673"
 },
 "1002A": {
     "code": "1002A",
     "name": "定陵",
     "city": "北京",
     "lng": "116.17",
     "lat": "40.2865"
 },
 "1003A": {
     "code": "1003A",
     "name": "东四",
     "city": "北京",
     "lng": "116.434",
     "lat": "39.9522"
 },
 ...
}
```

2. 处理2017年1月20日的全国1497个监测点数据。

使用如下脚本，处理当天24小时每个监测站点各个空气质量指标的信息。将这些信息提取出来，并根据前面获取的站点列表经纬度映射表，给站点加上经纬度信息。

```
var fs = require('fs');
var csv = require("fast-csv");
var mapdata = require('./站点列表经纬度映射.json');
var file = './站点_20170101-20170202/china_sites_20170120.csv';
var filename = file.replace(/^.*[\\\/]/, '').split('.')[0].split('_
')[2];
var datas = {};
csv
.fromPath(file, { headers: true, objectMode: true })
.on("data", function (data) {
 if (data.type === 'AQI') {
   datas[data.hour] = [];
   for (var key in data) {
     if (mapdata[key]) {
       datas[data.hour].push({
         name: mapdata[key].name,
         value: +data[key],
         code: mapdata[key].code,
         city: mapdata[key].city,
         lng: +mapdata[key].lng,
         lat: +mapdata[key].lat
       })
     }
   }
 }
})
```

```
.on("end", function () {
 fs.writeFile('./data/' + filename + '.json', JSON.stringify(datas
));
 console.log("done");
});
```

将每天的时间段作为key，每个时间段所对应的所有监测站点的AQI和位置等信息的数组，作为对应的value值。这样就可以方便地获取当天每个时间段的数据，并应用到等值面组件中。

```
{
"0": [{ "name": "万寿西宫", "value": 18, "code": "1001A", "city": "北
京", "lng": 116.366, "lat": 39.8673 }, { "name": "定陵", "value":
25, "code": "1002A", "city": "北京", "lng": 116.17, "lat": 40.2865
 }, ...],
"1": [{ "name": "万寿西宫", "value": 28, "code": "1001A", "city": "北
京", "lng": 116.366, "lat": 39.8673 }, { "name": "定陵", "value":
65, "code": "1002A", "city": "北京", "lng": 116.17, "lat": 40.2865
 }, ...],
"2": [{ "name": "万寿西宫", "value": 88, "code": "1001A", "city": "北
京", "lng": 116.366, "lat": 39.8673 }, { "name": "定陵", "value":
95, "code": "1002A", "city": "北京", "lng": 116.17, "lat": 40.2865
 }, ...]
 ...
}
```

2.2.4 处理接口

根据时间轴的特性，如果您需要时间轴变化的同时，等值面的数据也发生变化，那么可以开发一个接口或者数据库，能根据时间参数来获取不同时间段的全国各个监测站点的数据。本文档为您介绍如何使用Node.js完成接口的开发和发布（您也可以使用其他的开发语言，原理相同）。

推荐您将上一步已经处理完成的数据包装成一个简单的接口，并在DataV地图组件的数据面板配置API数据源，调用此接口，来完成这个需求。

接口信息如下：

- 请求地址：/aqi
- 请求方式：GET
- 请求参数：
  - 参数名称：date。
  - 参数类型：string，示例2017012722，时间格式为YYYYmmDDHH。

1. 安装Node.js（包括npm）并使用npm install <module_name>命令安装依赖模块。
2. 处理下载的所有数据，Node.js提供了一个glob模块可以对文件夹下的所有数据进行批量处理。

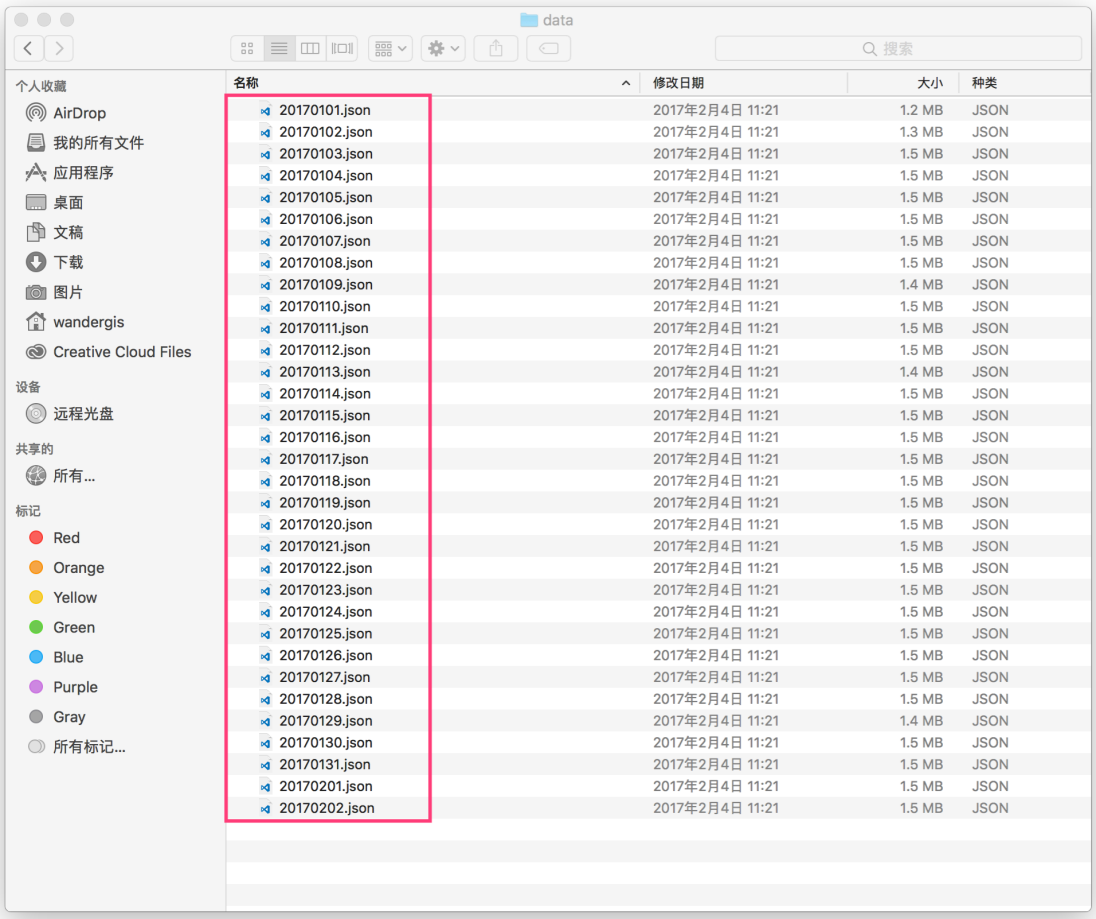
   将如下示例程序保存为js脚本文件，并在Node.js环境中运行。

```
var fs = require('fs');
var csv = require("fast-csv");
var glob = require('glob');
var mapdata = require('./站点列表经纬度映射.json');
```

```
glob("./站点_20170101-20170202/*.csv", function (err, files) {
files.forEach(function (file) {
 var filename = file.replace(/^.*[\\\/]/, '').split('.')[0].split('_
')[2];
 var datas = {};
 csv
   .fromPath(file, { headers: true, objectMode: true })
   .on("data", function (data) {
     if (data.type === 'AQI') {
       datas[data.hour] = [];
       for (var key in data) {
         if (mapdata[key]) {
           datas[data.hour].push({
             name: mapdata[key].name,
             value: +data[key],
             code: mapdata[key].code,
             city: mapdata[key].city,
             lng: +mapdata[key].lng,
             lat: +mapdata[key].lat
           })
         }
       }
     }
   })
   .on("end", function () {
     fs.writeFile('./data/' + filename + '.json', JSON.stringify(
datas));
     console.log("done");
   });
});
```

```
});
```

运行结果如下。



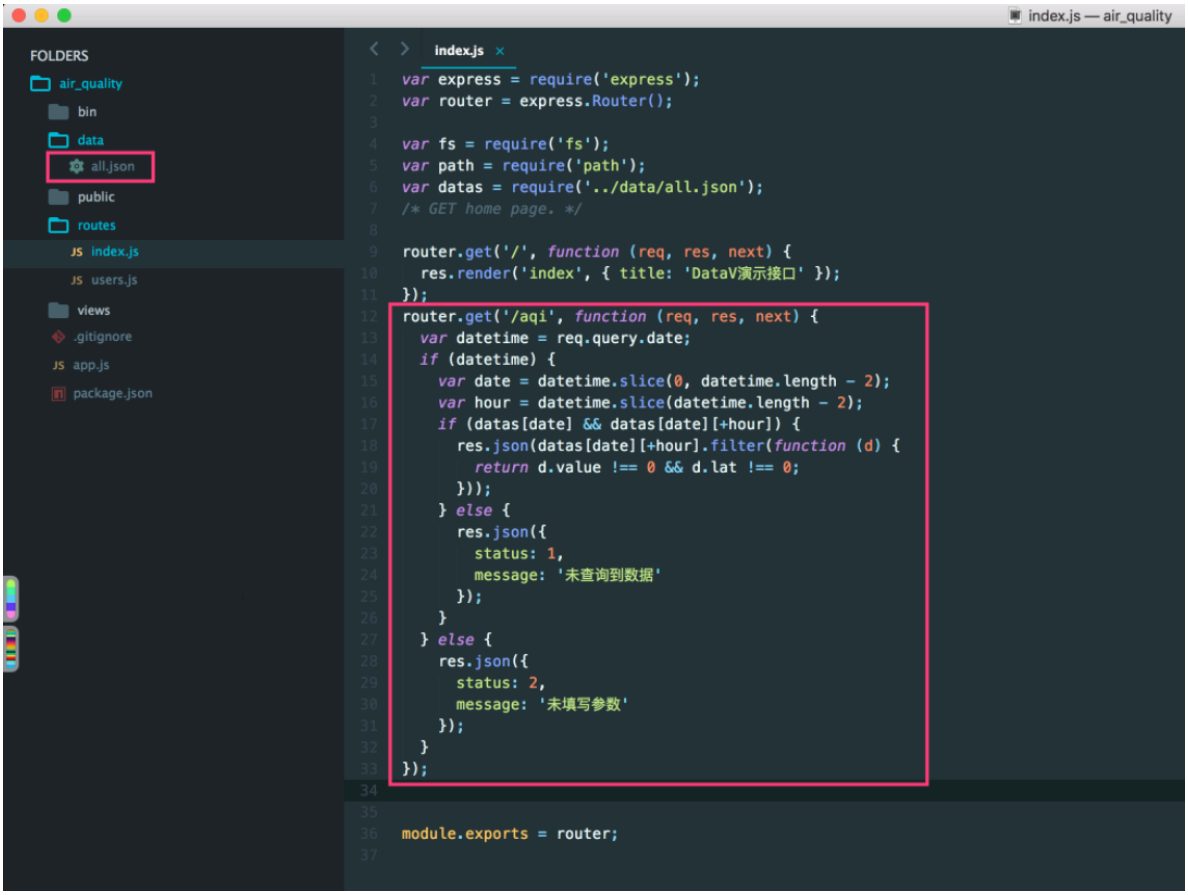
3. 使用glob模块对数据进行一次整合。

将如下示例程序保存为js脚本文件，并在Node.js环境中运行。

```
//以下方式不适用大批量的数据。
//将文件名也就是日期作为key，对应的内容作为值，得到一个all.json整合文件。
var fs = require('fs');
var csv = require("fast-csv");
var glob = require('glob');
glob("./data/*.json", function (err, files) {
var datas = {};
files.forEach(function (file) {
 var filename = file.replace(/^.*[\\\/]/, '').split('.')[0];
 datas[filename] = require(file);
});
fs.writeFile('./data/all.json', JSON.stringify(datas));
console.log('done');
```

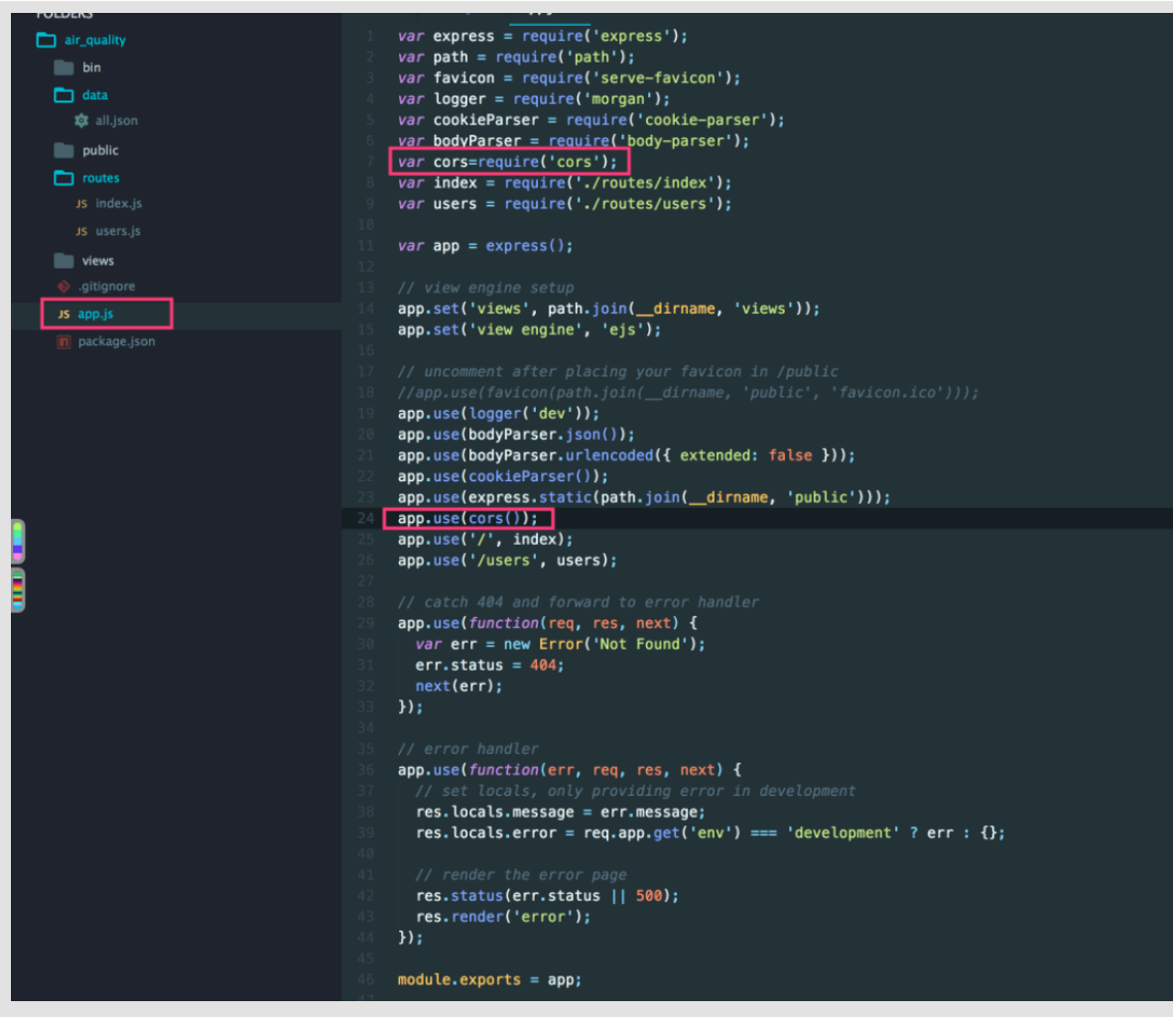
```
});
```

4. 在Node.js环境下，使用Node.js的express框架初始化一个express项目，并按照上面的接口需求增加一个简单的接口，示例程序如下所示。

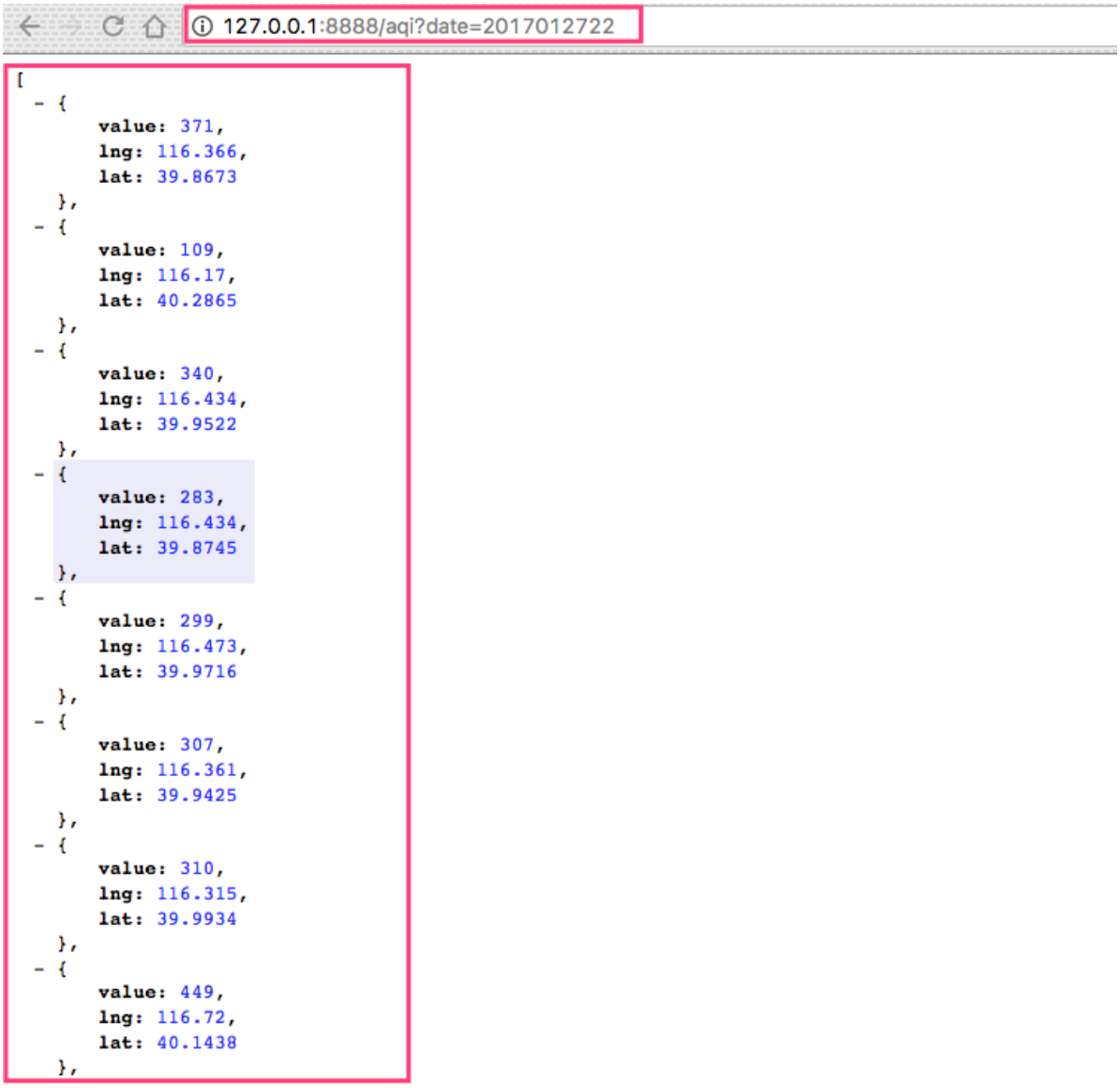


说明:

为了避免跨域请求的问题，您可以在*app.js*文件中增加cors模块。


air_quality
bin
data
all.json
public
routes
index.js
users.js
views
.gitignore
app.js
package.json
var express = require('express');
var path = require('path');
var favicon = require('serve-favicon');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');
var cors=require('cors');
var index = require('./routes/index');
var users = require('./routes/users');
var app = express();
// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');
// uncomment after placing your favicon in /public
//app.use(favicon(path.join(__dirname, 'public', 'favicon.ico')));
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));
app.use(cors());
app.use('/', index);
app.use('/users', users);
// catch 404 and forward to error handler
app.use(function(req, res, next) {
var err = new Error('Not Found');
err.status = 404;
next(err);
});
// error handler
app.use(function(err, req, res, next) {
// set locals, only providing error in development
res.locals.message = err.message;
res.locals.error = req.app.get('env') === 'development' ? err : {};
// render the error page
res.status(err.status || 500);
res.render('error');
});
module.exports = app;

5. 接口处理完成后，在Node.js环境下使用npm start命令测试接口，测试成功的截图如下所示。



2.3 制作大屏

2.3.1 创建大屏项目

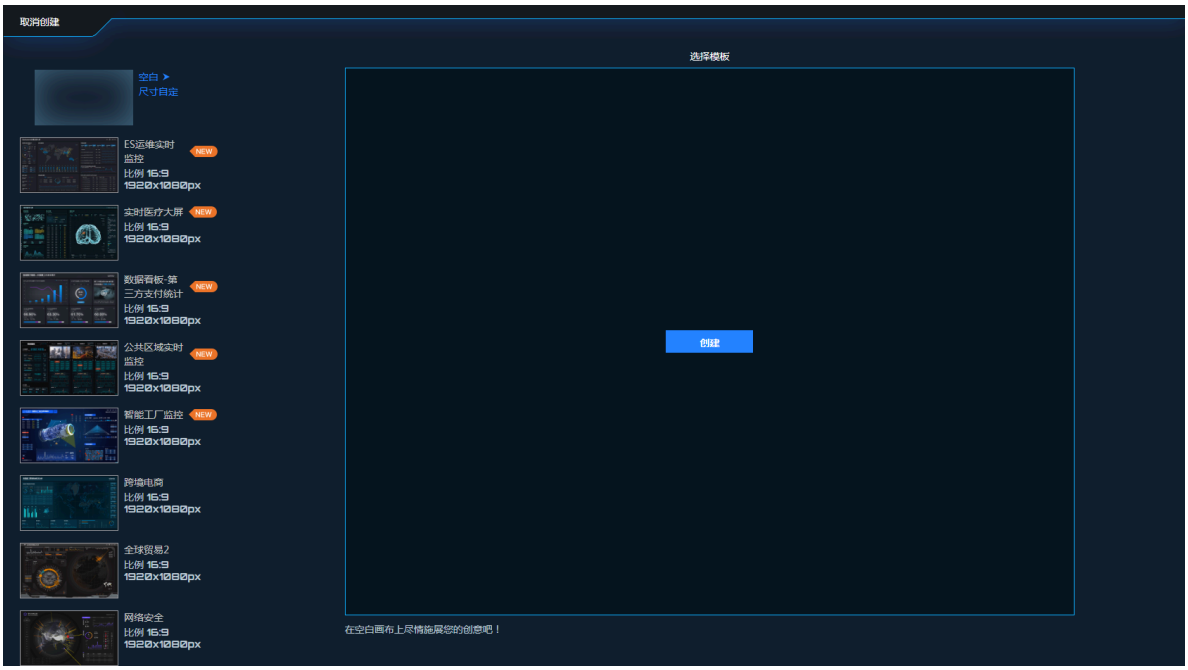
本文档为您介绍创建DataV大屏项目的方法。

说明:

本示例中使用的数据源是本地API文件，因此您不需要在DataV中添加数据源，直接在可视化项目的组件中调用API即可。如果您使用的是其他数据源，在创建可视化应用之前，需要先添加数据源。

1. 登录DataV控制台。

2. 单击我的可视化 > 新建可视化。
3. 选择空白模板，单击创建。



4. 在创建数据大屏对话框中，输入数据大屏名称，单击创建。

   大屏创建成功后会跳转到大屏编辑器页面。

2.3.2 添加组件

本文档为您介绍在DataV大屏项目中添加组件的方法。

添加地图和子组件

1. 在应用编辑器页面，单击地图 > 基础平面地图。

2. 在右侧的配置面板中，删除除底图层以外的子组件。

3. 添加等值面层子组件。

单击子组件管理左侧的+号，选择等值面层，单击添加子组件，完成子组件添加。

4. 单击全局设置，展开全局设置面板，调整地图的大小。

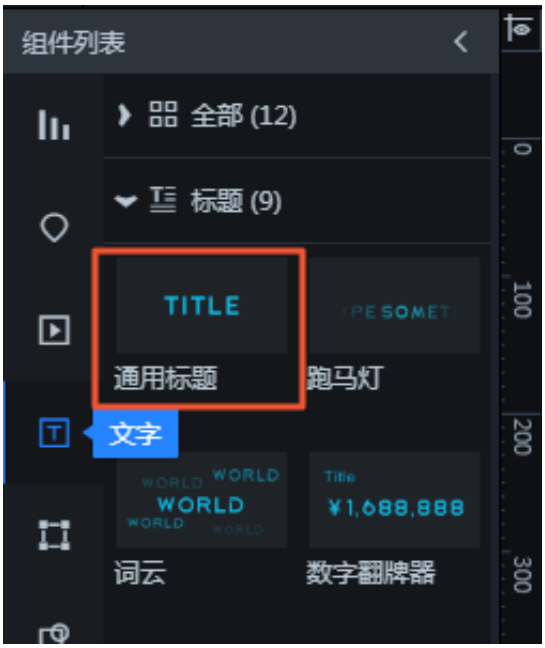
   您可以拖拽滑块或者手动输入数值来调整地图的大小和显示范围。

添加时间轴

单击交互 > 时间轴，在地图上添加一个时间轴组件。

添加地图标题

单击文字 > 标题 > 通用标题，为地图添加一个标题。



调整组件的图层和位置

组件添加完成后，按照以下说明调整组件之间的图层和位置关系。

1. 在大屏编辑器页面左侧的图层面板中，调整图层的顺序、选择图层、修改图层标题等。

2. 在大屏编辑器页面右侧的配置面板中调整组件的尺寸、位置等参数。

您也可以选中某一个组件，在画板上通过拖拽调整组件在画板上的位置。

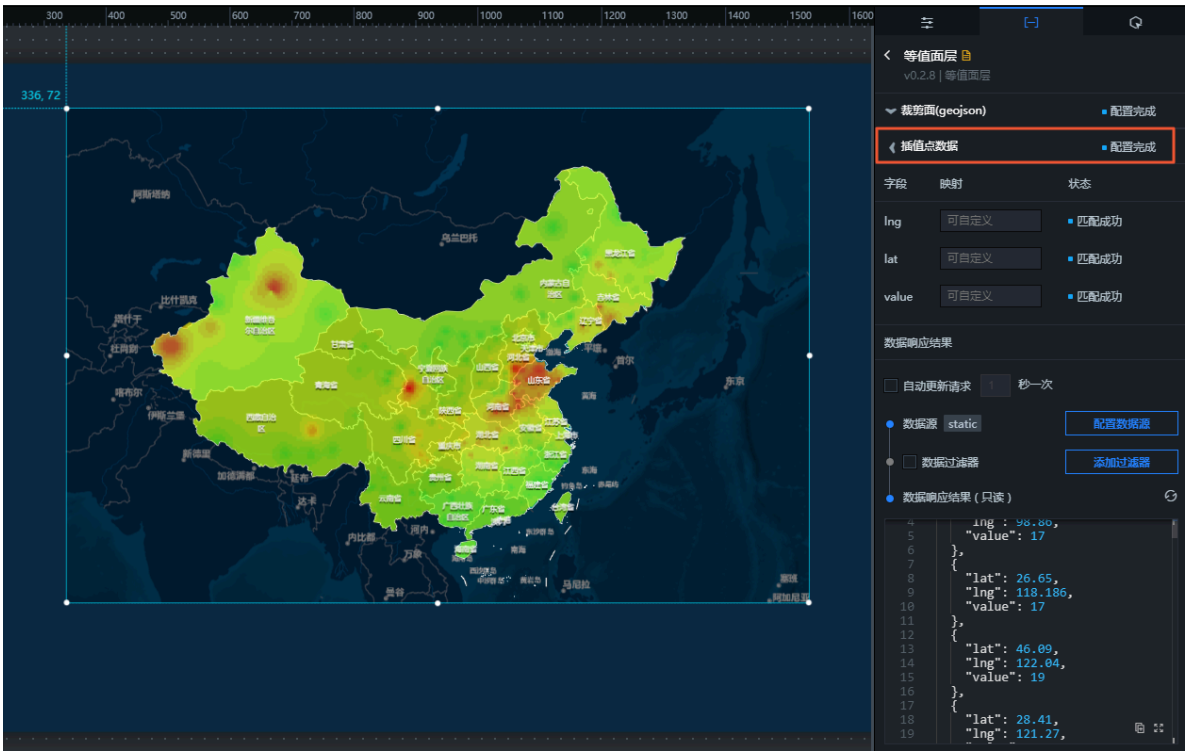
2.3.3 添加数据

本文档为您介绍在DataV大屏项目中，为组件添加数据的方法。

添加地图数据

1. 在大屏编辑器页面，单击基础平面地图组件。

2. 在右侧的配置面板中，单击数据。
3. 在数据面板中，单击等值面层。
4. 在等值面层的数据面板中，单击插值点数据。



> **说明:**
> 等值面层的数据包括了裁剪面数据和插值点数据。由于本示例的数据区域是全国范围，因此裁剪面的数据可以保持不变。您也可以根据需要修改裁剪面的数据。

5. 在插值点数据的配置页面，单击配置数据源。
6. 在设置数据源页面，按照以下说明配置数据源。
   - 数据源类型：选择API。由于处理接口章节已经写好了对应的API，也已经测试了数据获取，所以修改等值面层组件的插值点的数据源类型为API。
   - URL：填写#unique_29中测试的地址（本文测试`http://127.0.0.1:8888/aqi?date=2017012722`）。
7. 单击预览数据源返回结果，查看数据返回结果。
8. 单击数据响应结果右侧的刷新图标，查看数据响应结果。

   数据响应成功后，组件将在画布上展示对应的数据样式。

9. 退出设置数据源页面，按照以下说明，设置等值面层组件的配置样式。

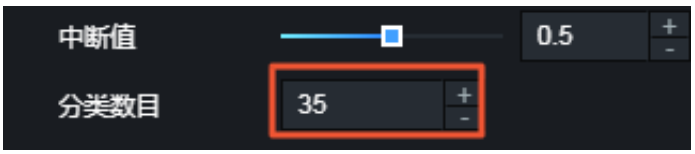
   a. 在等值面层的配置页面，单击右侧面板的配置。

   b. 在配置面板中，设置像元大小，推荐设置为3。

> **注意:**
> 像元大小越大，插值越快，精度越低。

   c. 设置渲染方式，推荐设置为线性渲染。

   d. 设置分类数目，推荐设置为35。



添加时间轴数据

1. 在大屏编辑器页面，单击时间轴组件。
2. 在右侧的配置面板中，单击数据。
3. 在数据配置面板中，单击配置数据源。
4. 在设置数据源页面，选择数据源类型为静态数据。

5. 参考时间轴组件的默认数据，创建您需要的数据，并替换示例静态数据。

   例如，选择2017年1月22日到2017年2月2日，每天22点作为时间轴数据。

```
[
{
 "name": "2017年1月22日22时",
 "date": 2017012222,
 "value": 2017012222
},
{
 "name": "2017年1月23日22时",
 "date": 2017012322,
 "value": 2017012322
},
{
 "name": "2017年1月24日22时",
 "date": 2017012422,
 "value": 2017012422
},
{
 "name": "2017年1月25日22时",
 "date": 2017012522,
 "value": 2017012522
},
{
 "name": "2017年1月26日22时",
 "date": 2017012622,
 "value": 2017012622
},
{
 "name": "2017年1月27日22时",
 "date": 2017012722,
 "value": 2017012722
},
{
 "name": "2017年1月28日22时",
 "date": 2017012822,
 "value": 2017012822
},
{
 "name": "2017年1月29日22时",
 "date": 2017012922,
 "value": 2017012922
},
{
 "name": "2017年1月30日22时",
 "date": 2017013022,
 "value": 2017013022
},
{
 "name": "2017年1月31日22时",
 "date": 2017013122,
 "value": 2017013122
},
{
 "name": "2017年2月1日22时",
 "date": 2017020122,
 "value": 2017020122
},
{
 "name": "2017年2月2日22时",
 "date": 2017020222,
```

```
 "value": 2017020222
}
]
```

- name：时间轴的轴点显示的内容。
- date：作为回调ID选项使用。
- value：对应的时间。

数据响应成功后，组件将在画布上展示对应的数据样式。

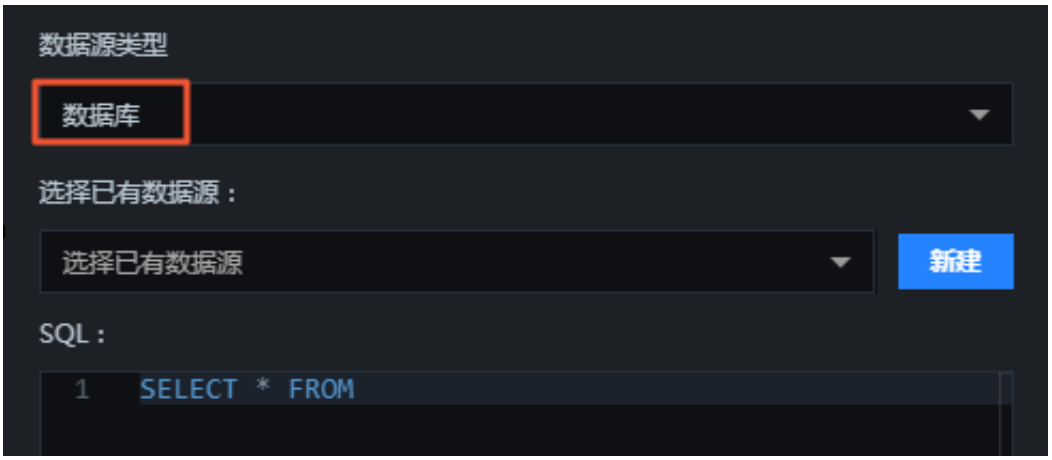
6. 退出设置数据源页面，在时间轴组件的配置面板中，单击事件节点，设置数据格式为%Y%m%d%H。

7. 单击交互，设置回调ID的值为*data*。

添加地图标题数据

1. 在大屏编辑器页面，单击标题组件。

2. 在右侧的配置面板中，单击数据。
3. 在标题组件的数据配置页面，单击配置数据源。
4. 在设置数据源页面，选择数据源类型为数据库。



5. 在选择已有数据源列表中，选择一个数据库。

> 说明:
> 如果没有可选的数据库，您可以单击新建，按照系统提示，新建一个数据库。详情请参见配置数据源。

6. 在SQL编辑区域，输入以下SQL脚本。

```
select to_char(to_timestamp(:date,'YYYYMMDDHH24'),'YYYY年mm月DD日HH24时')||'空气质量' as value;
```

`:date`：在实际浏览时会传入回调ID对应的值。

7. 单击预览数据源返回结果，查看数据返回结果。
8. 单击数据响应结果右侧的刷新图标，查看数据响应结果。

   数据响应成功后，组件将在画布上展示对应的数据样式。

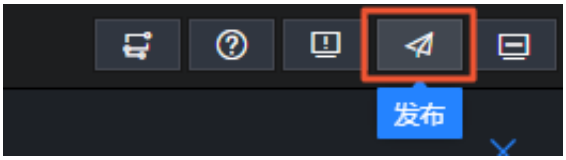
2.4 预览并发布大屏

组件的样式和数据都配置完成后，您可以预览并发布大屏，实现大屏的在线播放和演示。本文档为您介绍预览并发布可视化大屏的方法。

1. 单击大屏页面右上角的预览图标，预览可视化大屏。

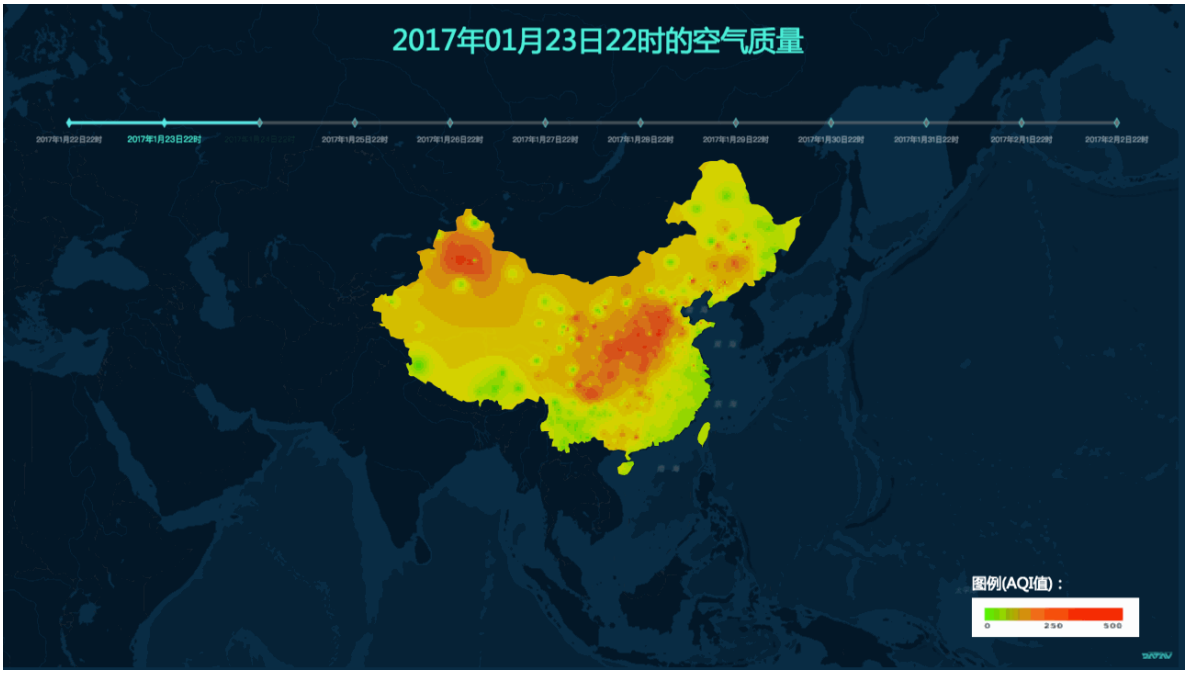
   预览成功后，可按照以下步骤发布可视化大屏。

2. 单击大屏页面右上角的发布图标。



3. 打开发布分享开关，单击分享链接右侧的复制。

4. 打开浏览器，将复制的链接粘贴到导航栏中，即可在线观看发布成功的大屏。



3 DataV大屏展示IoT设备数据案例教程

3.1 教程概述

本文档通过一个案例，演示了如何基于阿里云产品和服务实现设备数据在大屏上展示。

本案例的原理如下：

1. 在设备端模拟两个字段，通过MQTT协议向阿里云物联网平台设备（高级版）发送数据。
2. 物联网平台接收到数据后通过规则引擎转发至DataHub。
3. 在DataHub中通过DataConnector将数据同步到RDS for MySQL数据库中。
4. 使用DataV将RDS for MySQL中的数据展示在大屏上。

> 说明：
> 物联网平台转发至DataHub，是因为DataHub可以将数据同步至MaxCompute，为后续数据计算做准备。

本案例的整体步骤如下：

1. #unique_34。
2. #unique_35。
3. #unique_36。
4. #unique_37。
5. #unique_38。
6. #unique_39。

3.2 创建RDS for MySQL数据库表

本文档为您介绍创建RDS for MySQL数据库表的具体步骤。

操作步骤

1. 登录阿里云云数据库RDS控制台。

2. 单击创建实例，创建RDS for MySQL实例。

3. 单击RDS for MySQL实例链接，进入基本信息页面。

> **说明:**
> 您需要记录RDS for MySQL实例的内网和外网地址，后面在DataV中创建数据源时会用到。

4. 单击左侧菜单栏的账号管理，创建账号。
5. 单击左侧菜单栏的数据库管理，创建数据库。
6. 单击左侧菜单栏的数据安全性，参考#unique_41，添加数据库白名单。

   根据您数据库所在的网络类型，将DataV的白名单添加到RDS for MySQL数据库中，详情请参见#unique_42。
7. 单击左侧菜单栏的基本信息，获取RDS实例的基本信息。

> **说明:**
> 此信息需要填入DataHub的DataConnector中，用于同步数据到RDS for MySQL。

8. 单击基本信息页面右上角的登录数据库，输入账号和密码，进入DMS系统。
9. 创建表mytable，包含两个字段，如下图所示。

3.3 创建DataHub项目

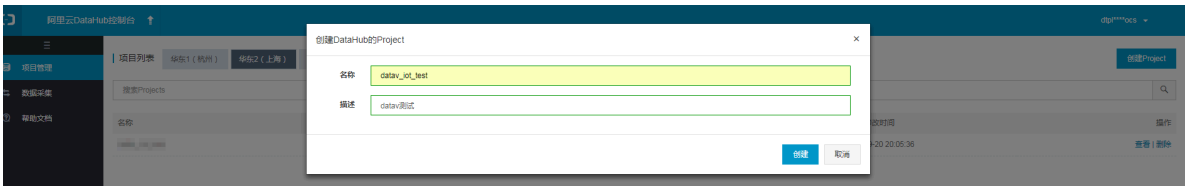
阿里云流数据处理平台DataHub是流式数据（Streaming Data）的处理平台，提供对流式数据的发布（Publish），订阅（Subscribe）和分发功能，让您可以轻松构建基于流式数据的分析和应用。

前提条件

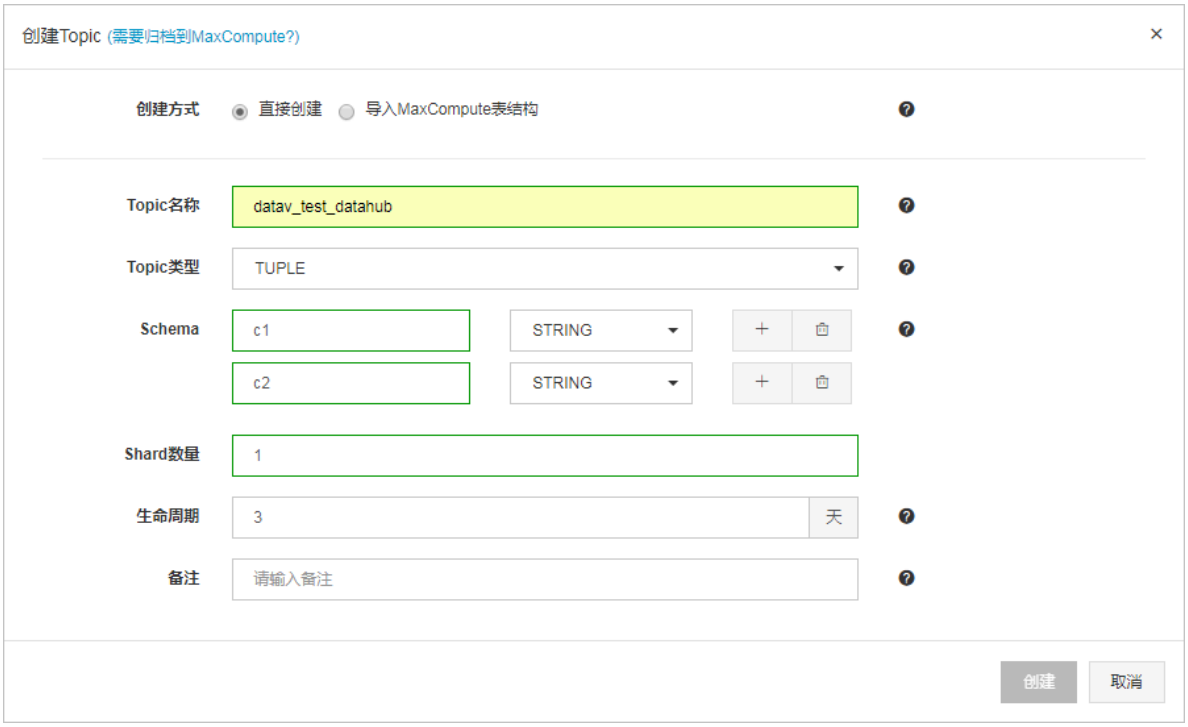
您已经完成了RDS for MySQL数据库的创建。

操作步骤

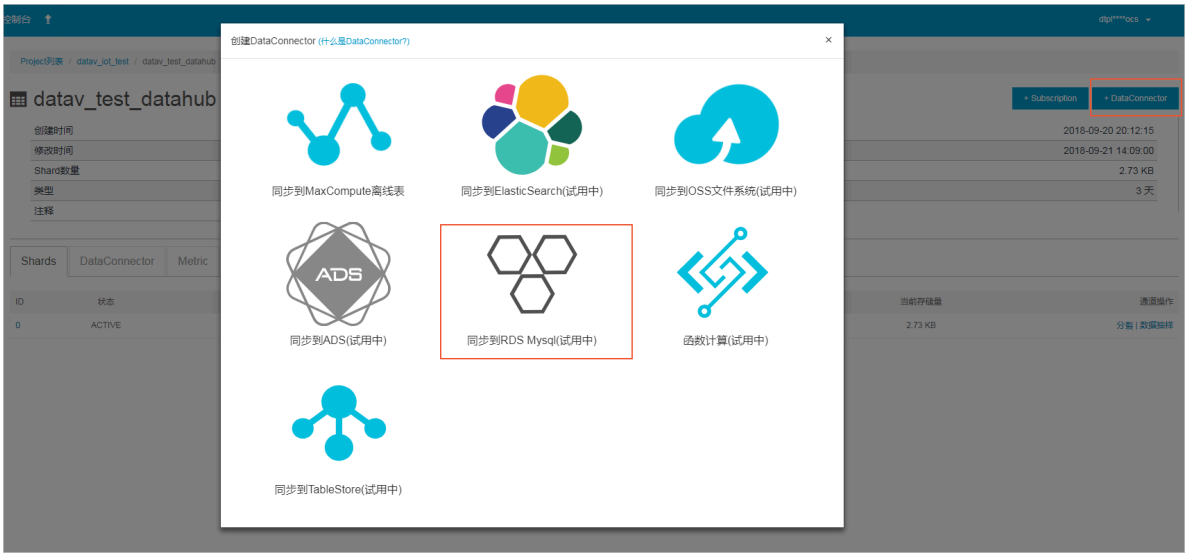
1. 登录阿里云DataHub控制台。
2. 在项目管理中，选择华东2，单击创建Project。



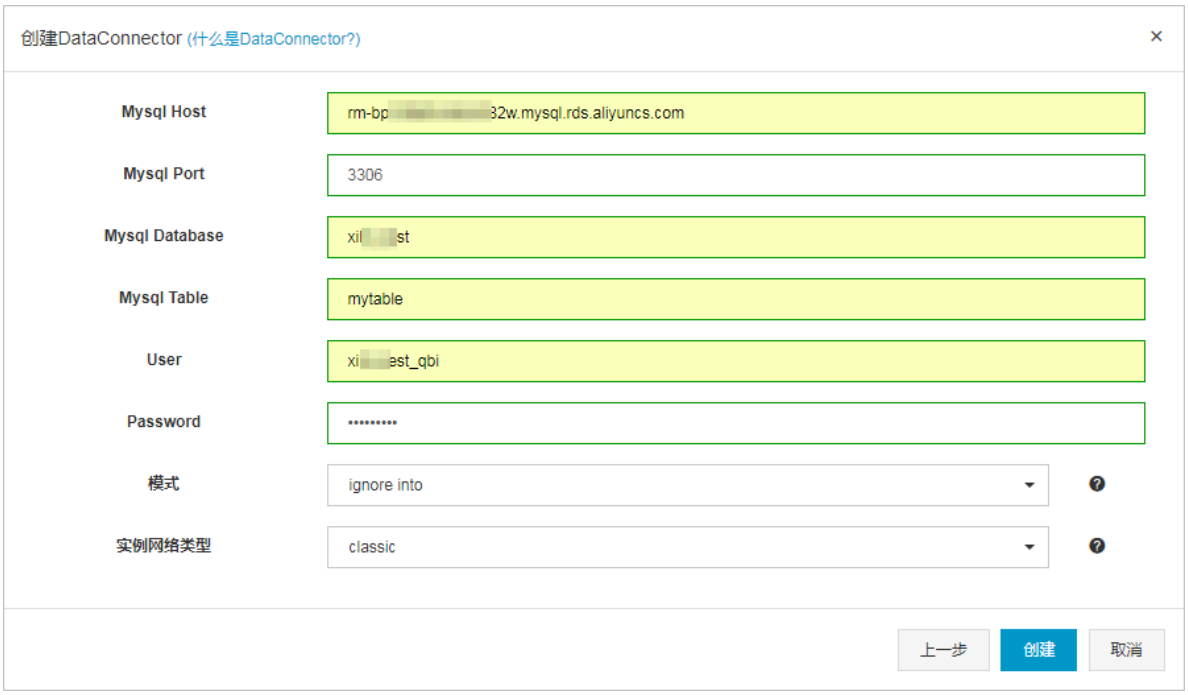
3. 选择项目右侧的查看，单击创建Topic。
4. 在创建Topic页面，输入Topic名称及Schema，单击创建，如下图所示。



5. 选择Topic右侧的查看，单击页面右上角的+ DataConnector，选择同步到RDS Mysql。



6. 在创建DataConnector页面，输入RDS for MySQL数据库的相关信息，如下图所示。



参数	说明
Mysql Host	RDS数据库的内网地址。
Mysql Port	RDS数据库的端口号，一般为3306。
Mysql Database	RDS数据库的名称。
Mysql Table	RDS数据库表的名称。
User	登录RDS数据库的账号。

参数	说明
Password	登录RDS数据库的密码。

您可以在云数据库RDS控制台中，单击RDS实例链接，获取以上数据库信息。

3.4 配置物联网平台设备

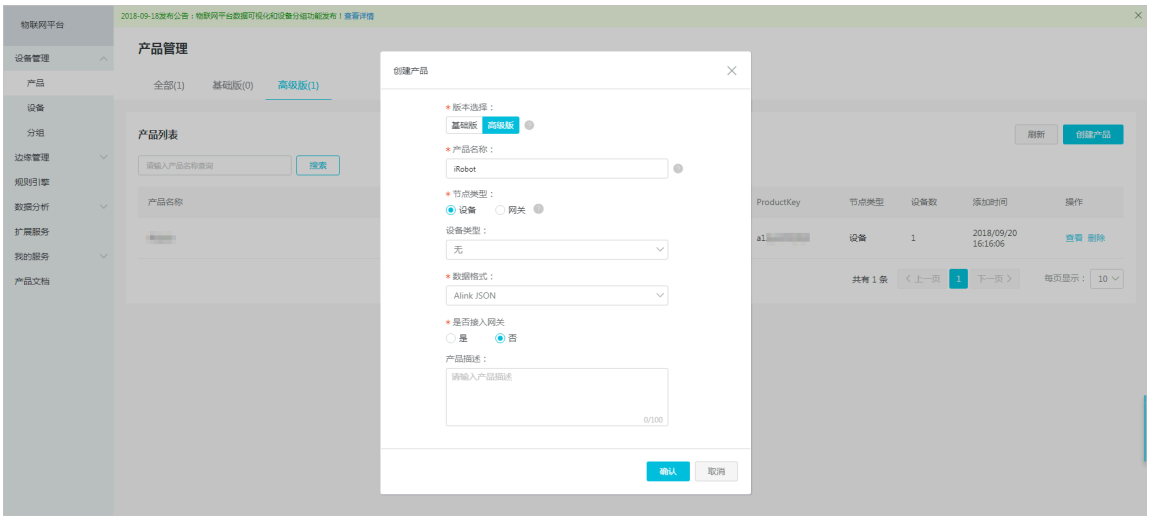
物联网平台主要负责获取设备端数据，并通过规则引擎将数据转发至Table Store、DataHub、RDS、Message Service、Message Queue、HiTSDB、FC以及另外一个Topic。

前提条件

您已经完成了DataHub项目的创建。

操作步骤

1. 创建产品。
   a) 登录阿里云物联网平台控制台。
   b) 选择左侧菜单栏的设备管理 > 产品。
   c) 在产品管理中，选择高级版，单击创建产品。
   d) 输入产品名称，其它选项保持默认，单击确认，完成产品的创建。

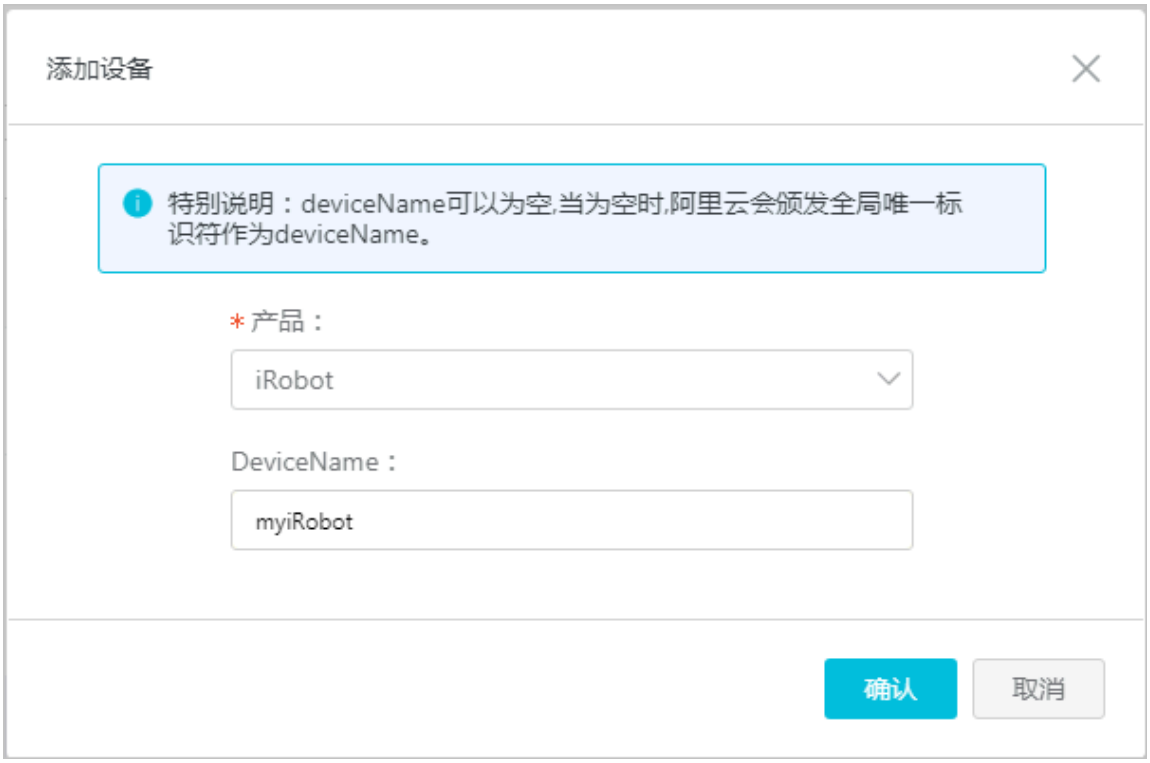


2. 添加设备。

a) 选择左侧菜单栏的设备管理 > 设备。

b) 在设备管理页面，单击添加设备。

c) 选择第一步中创建的产品，并输入设备名称，完成设备的添加。

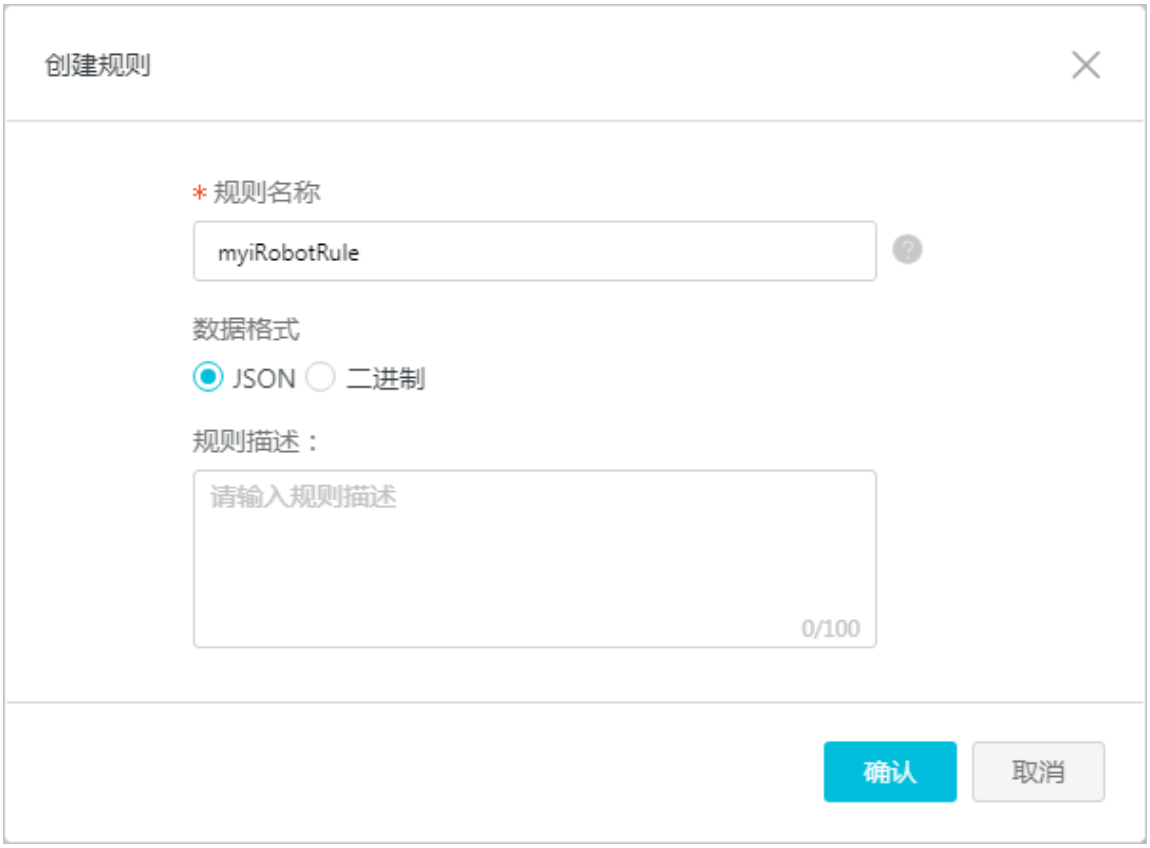


3. 创建并配置规则引擎。

a) 选择左侧菜单栏的规则引擎。

b) 在规则引擎页面，单击创建规则。

c) 输入规则名称，单击确认，完成规则的创建。



d) 规则创建完成后，单击该规则右侧的管理。

e) 在处理数据模块，单击编辑，跳出编写SQL对话框。

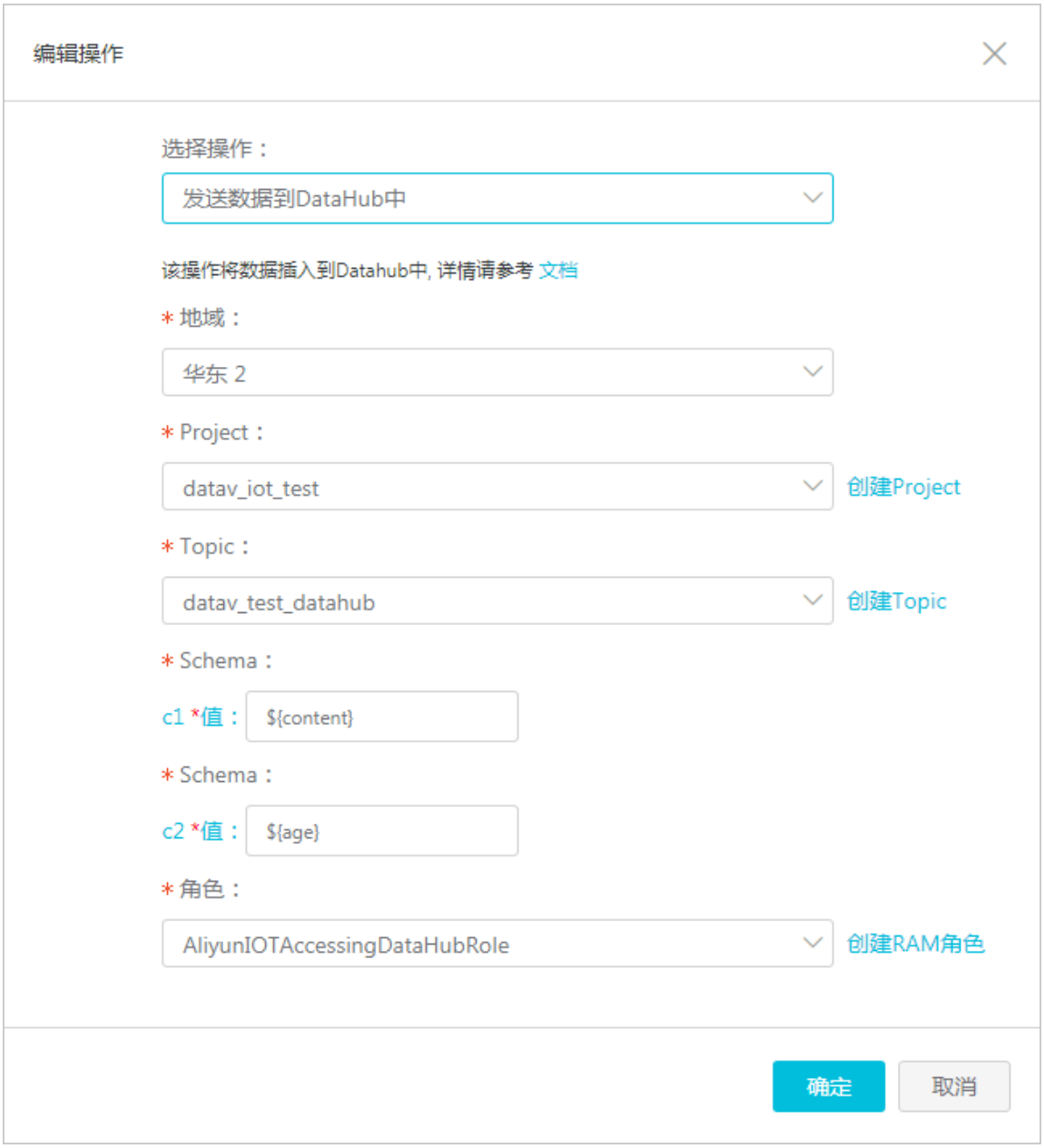
f) 输入字段和Topic信息，单击确认，完成SQL编写。

系统自动生成规则查询语句：

```
SELECT content,age FROM "/a1ZwJ6HOfcR/myiRobot/user/update" WHERE
```

g) **在转发数据模块，单击添加操作。**

h) **在添加操作对话框中，输入Project（DataHub的项目名称）、Topic（DataHub项目的Topic名称）及Schema信息，如下图所示。**



i) 单击确定，完成数据转发配置。

4. 启动规则引擎。单击左侧菜单栏的规则引擎，回到规则引擎页面。单击已经创建规则右侧的启动。

3.5 运行MQTT客户端

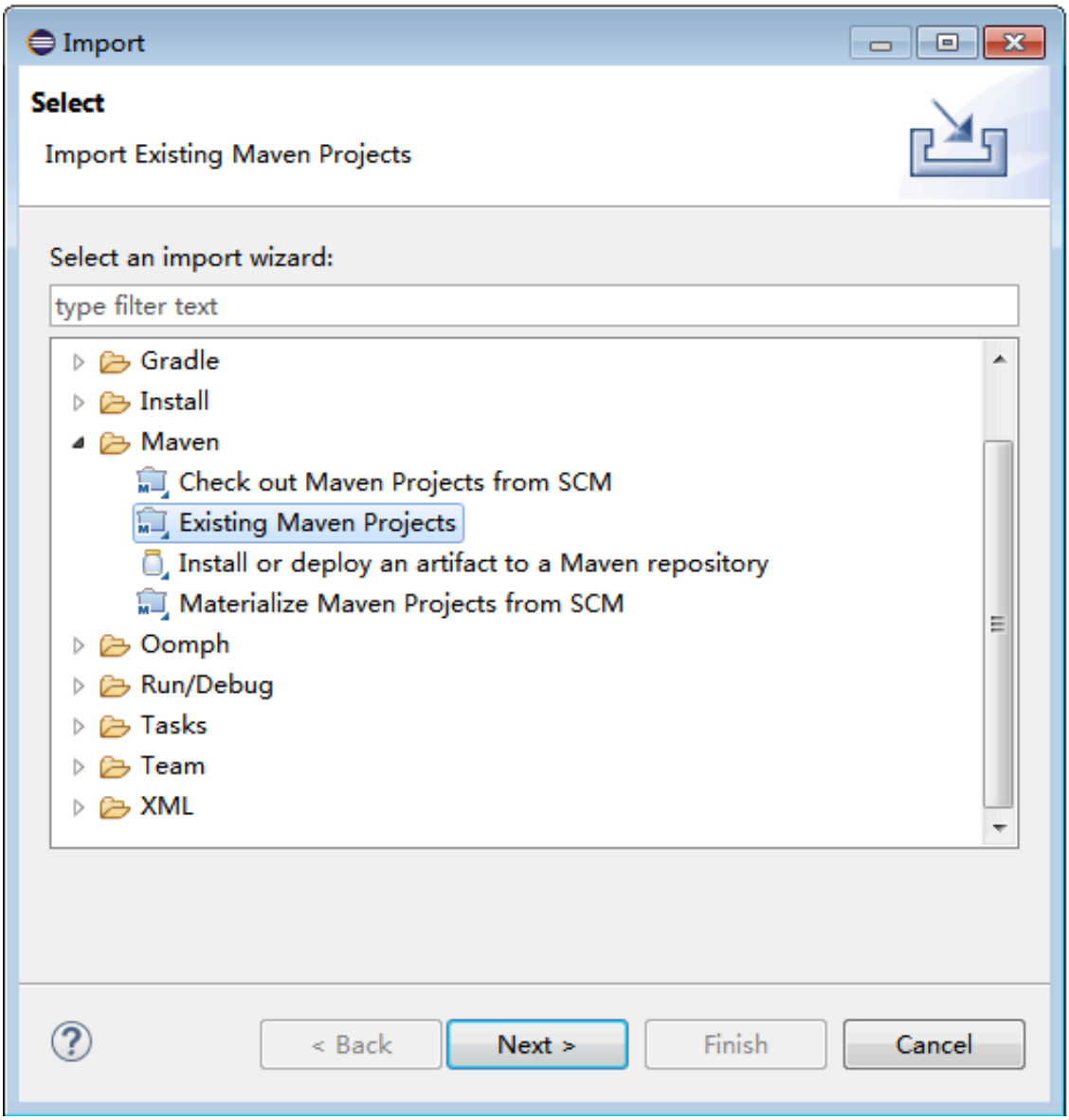
MQTT是基于TCP/IP协议栈构建的异步通信消息协议，是一种轻量级的发布/订阅信息传输协议。本文档为您介绍通过阿里云物联网平台提供的设备端Java SDK，运行MQTT客户端的方法。

前提条件

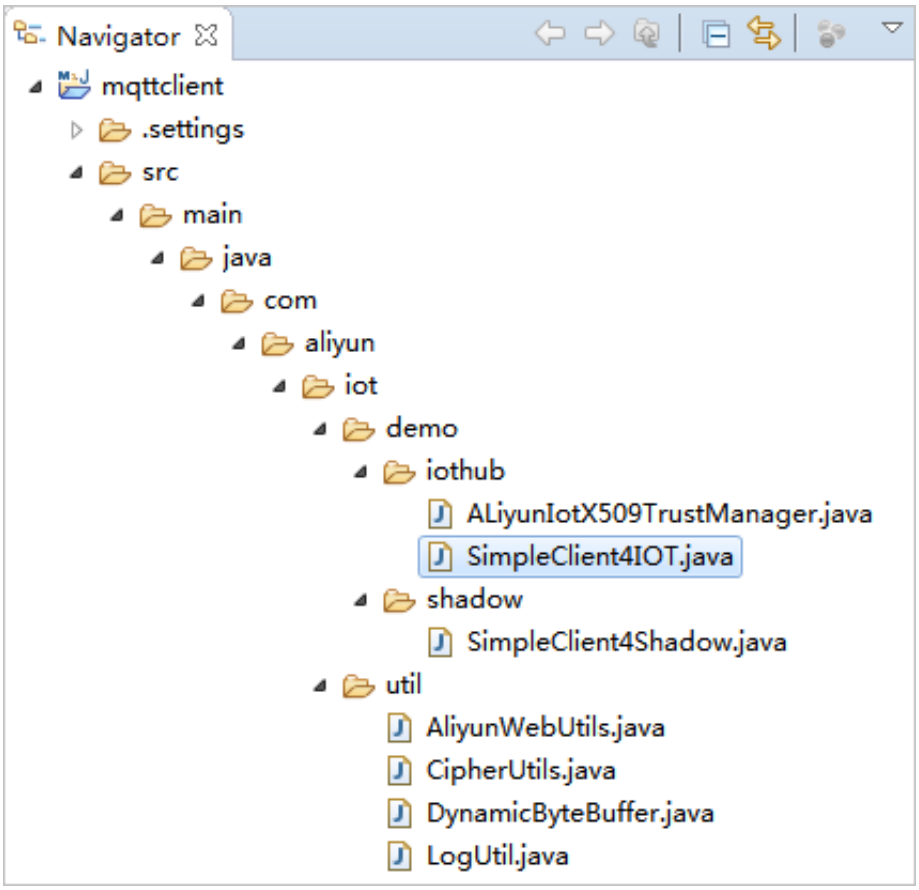
- 下载Eclipse软件，并配置好Java环境。
- 此Demo为maven工程，请先安装maven。

操作步骤

1. 参考#unique_47，下载iotx-sdk-mqtt-java并解压。
2. 打开Eclipse应用，选择file > import > Existing Maven Projects，导入上一步解压后的Java SDK文件。



3. 双击打开*SimpleClient4IOT.java*文件。



4. 配置设备三元组和topic。修改*deviceName*、*productKey*、*secret*、*subTopic*、*pubTopic*变量，如下图所示。

```
import java.net.InetAddress;

/**
 * IoT套件JAVA版设备接入demo
 */
public class SimpleClient4IOT {

    /******这里是客户端需要的参数*******/
    public static String deviceName = "myiRobot";
    public static String productKey = "a1      cR";
    public static String secret = "wrx                    WDH6";

    //用于测试的topic
    private static String subTopic = "/" + productKey + "/" + deviceName + "/user/get";
    private static String pubTopic = "/" + productKey + "/" + deviceName + "/user/update";

    public static void main(String... strings) throws Exception {
        //客户端设备自己的一个标识，建议是MAC或SN，不能为空，32字符内
        String clientId = InetAddress.getLocalHost().getHostAddress();

        //设备认证
        Map<String, String> params = new HashMap<String, String>();
        params.put("productKey", productKey); //这个是对应用户在控制台注册的 设备productkey
        params.put("deviceName", deviceName); //这个是对应用户在控制台注册的 设备name
        params.put("clientId", clientId);
        String t = System.currentTimeMillis() + "";
        params.put("timestamp", t);
```

说明:

进入阿里云物联网控制台，单击物联网平台的设备，选择查看，获取以上信息。*subTopic*和*pubTopic*与上图保持一致即可。

```
private static String subTopic = "/" + productKey + "/" +
deviceName + "/user/get";
```

```
private static String pubTopic = "/" + productKey + "/" +
deviceName + "/user/update";
```

5. 修改content，如下图所示。

```
        @Override
        public void deliveryComplete(IMqttDeliveryToken token) {
            //如果是QoS0的消息，token.resp是没有回复的
            LogUtil.print("消息发送成功！ " + ((token == null || token.getResponse() == null) ? "null"
                : token.getResponse().getKey()));
        }
    });
    LogUtil.print("连接成功:---");

    //这里测试发送一条消息
    String content1 = "{'content':'shoen1','age':'20'}";
    String content2 = "{'content':'shoen2','age':'21'}";
    String content3 = "{'content':'shoen3','age':'22'}";

    MqttMessage message1 = new MqttMessage(content1.getBytes("utf-8"));
    MqttMessage message2 = new MqttMessage(content2.getBytes("utf-8"));
    MqttMessage message3 = new MqttMessage(content3.getBytes("utf-8"));

    message1.setQos(0);
    message2.setQos(0);
    message3.setQos(0);

    //System.out.println(System.currentTimeMillis() + "消息发布:---");
    sampleClient.publish(pubTopic, message1);
    sampleClient.publish(pubTopic, message2);
    sampleClient.publish(pubTopic, message3);

    LogUtil.print(pubTopic);
```

```
String content1 = "{'content':'shoen1','age':'20'}";
String content2 = "{'content':'shoen2','age':'21'}";
String content3 = "{'content':'shoen3','age':'22'}";


MqttMessage message1 = new MqttMessage(content1.getBytes("utf-8"));
MqttMessage message2 = new MqttMessage(content2.getBytes("utf-8"));
MqttMessage message3 = new MqttMessage(content3.getBytes("utf-8"));

message1.setQos(0);
message2.setQos(0);
message3.setQos(0);

//System.out.println(System.currentTimeMillis() + "消息发布:---");
sampleClient.publish(pubTopic, message1);
sampleClient.publish(pubTopic, message2);
sampleClient.publish(pubTopic, message3);
```

6. 单击运行。

运行成功后，返回如下信息。

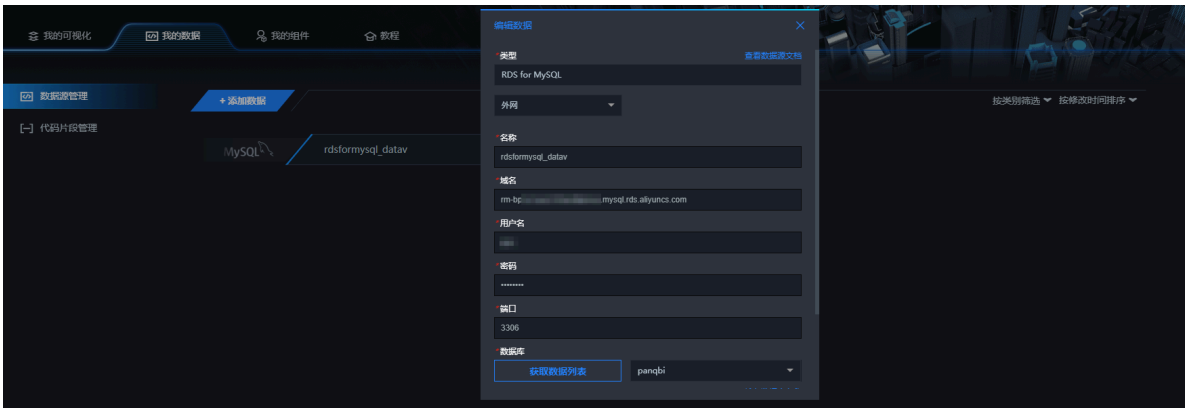
```
2018-09-25 11:21:02.183 - [SimpleClient4IOT.java] - connectMqtt(89):30.[redacted].76|securemode=2,signmethod=hm
2018-09-25 11:21:02.398 - [SimpleClient4IOT.java] - connectMqtt(112):连接成功:---
2018-09-25 11:21:02.401 - [SimpleClient4IOT.java] - deliveryComplete(108):消息发送成功！ null
2018-09-25 11:21:02.401 - [SimpleClient4IOT.java] - deliveryComplete(108):消息发送成功！ null
2018-09-25 11:21:02.401 - [SimpleClient4IOT.java] - connectMqtt(132):/a1[redacted]cR/myiRobot/user/update
2018-09-25 11:21:02.401 - [SimpleClient4IOT.java] - deliveryComplete(108):消息发送成功！ null
```

3.6 配置DataV数据源

本文档为您介绍在DataV中使用RDS for MySQL数据源的方法。

操作步骤

1. 登录Datav控制台。
2. 选择我的数据 > 添加数据。
3. 填写您已经创建完成的数据库的相关信息，单击确定。



> 说明：
>
> - 进入云数据库RDS控制台，单击RDS for MySQL实例链接，进入实例的基本信息页面，获取以上信息。
> - 如果您的网络类型为内网，则对应的域名为RDS for MySQL实例的内网地址。
> - 如果您的网络类型为外网，则对应的域名为RDS for MySQL实例的外网地址。

4. 单击我的可视化 > 新建可视化，选择云资源监控模板，单击创建。
5. 在创建数据大屏弹出框中，输入数据大屏名称，单击创建。

6. 配置组件样式。

a) 选择运行耗时最长任务下的轮播列表组件，单击画布右侧的配置图标（  ）。

b) 在组件配置页面，单击自定义列。

c) 单击标签1，设置列字段名为c1，列显示名为content。

d) 单击标签2，设置列字段名为c2，列显示名为age。

7. 配置组件数据。

a) 单击画布右侧的数据图标（ ）。

b) 在组件的数据配置面板中，单击配置数据源。

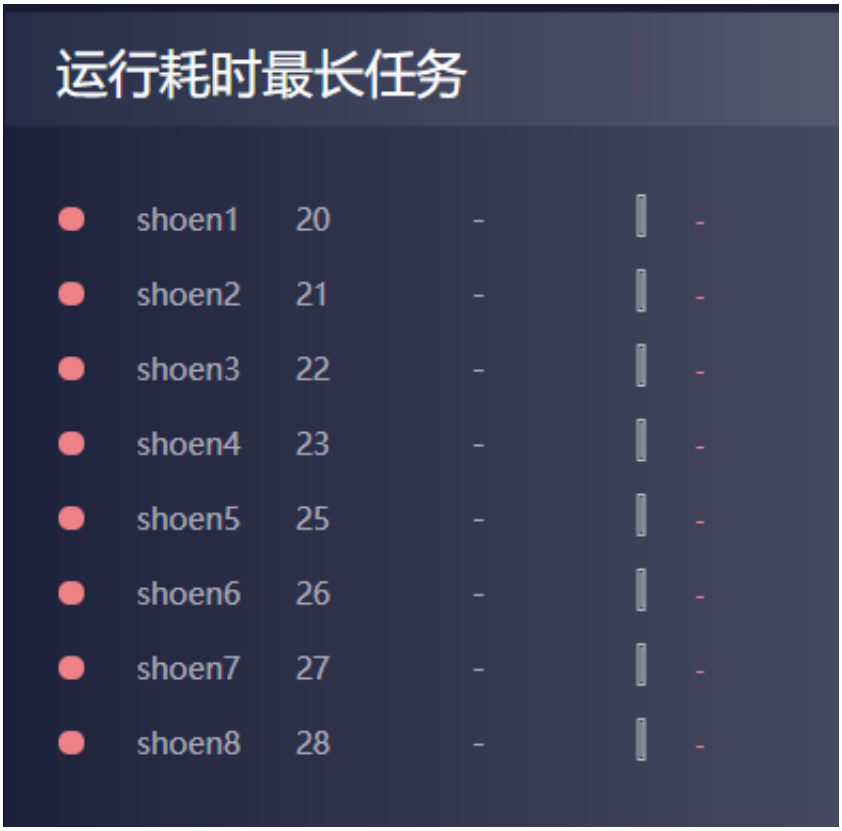
c) 在设置数据源页面中，选择数据源类型为数据库，已有数据源为您前面步骤中创建的数据源。

d) 在SQL输入区域输入以下SQL语句。

```
SELECT c1,c2 FROM mytable
```

e) 单击预览数据源返回结果，查看数据源返回结果。

最终组件显示效果如下图所示。



3.7 查看结果

本文档为您展示此案例的最终运行结果，包括DataHub的topic状态、RDS for MySQL的数据同步结果以及最终大屏的展示效果。

操作步骤

1. **进入阿里云DataHub控制台，查看topic状态，如下图所示。**

2. 进入云数据库RDS控制台，查看mytable的数据，如下图所示。

3. 进入DataV控制台，组件显示效果，如下图所示。

4 DataV大屏展示实时计算数据案例教程

4.1 教程概述

本案例使用阿里云DTS将您的增量数据同步到DataHub上，再通过阿里云实时计算Flink订阅DataHub的数据进行实时计算，并将结果插入到RDS数据库中，最终通过DataV大屏进行展示。

> **说明:**
> 本案例仅提供具体的操作指导，业务场景和数据与实际情况会有一定偏差。具体业务场景可参考电商之订单与销量统计。

本案例的整体步骤如下：

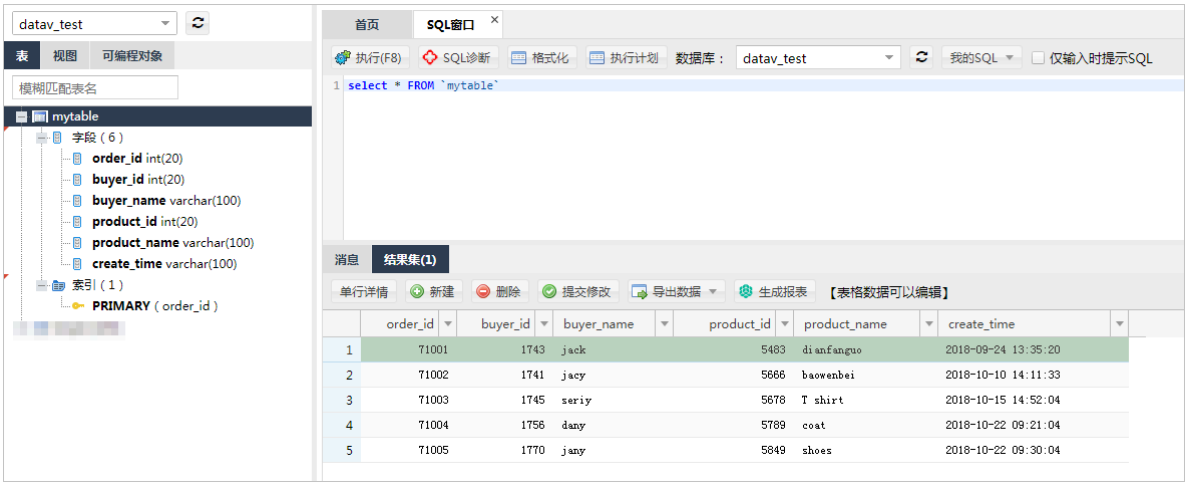
1. 准备工作：完成阿里云RDS实例和流计算项目的创建。
2. 数据采集：通过阿里云DTS的数据同步功能，将RDS中的数据同步到DataHub中，完成数据采集。
3. 数据订阅：通过阿里云实时计算Flink，订阅DataHub数据进行实时计算，插入到RDS实例的目标表中。
4. 数据展示：通过DataV大屏，对数据进行处理并展示。

4.2 准备工作

在开始本案例前，您需要首先完成以下工作。

1. 在云数据库RDS控制台上，创建RDS for MySQL实例，并新建数据库和表。

   本案例使用的表结构如下图所示，其中已经插入了几条数据。



2. 购买实时计算服务，并创建项目。

   本案例购买了一个共享模式的Flink服务，区域为华东2，计算资源为4 CU。并创建了一个名称为doc_flink的项目。

4.3 通过DTS采集数据

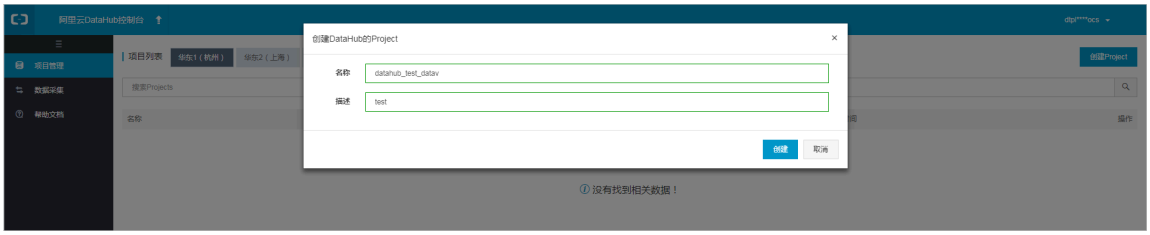
通过阿里数据传输中的数据同步（DTS），将RDS的数据实时传输至DataHub中。

1. 创建DataHub项目。

   a. 登录DataHub控制台。

   b. 在项目管理中，选择区域，本案例选择华东1区。

   c. 单击创建Project，输入名称和描述，创建一个DataHub项目。本案例的项目名称为datahub_test_datav。



2. 配置DTS数据同步作业。

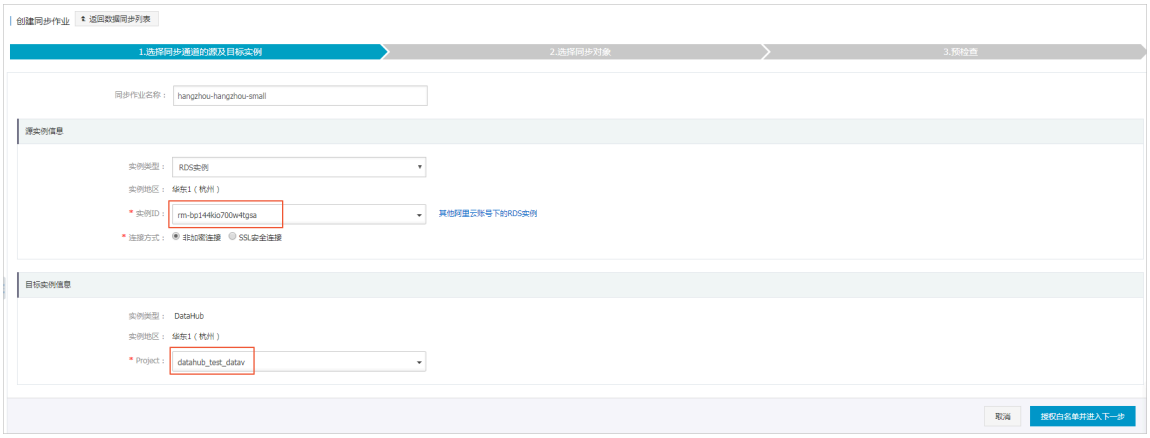
   a. 登录DTS控制台。

   b. 选择数据同步 > 创建同步作业。

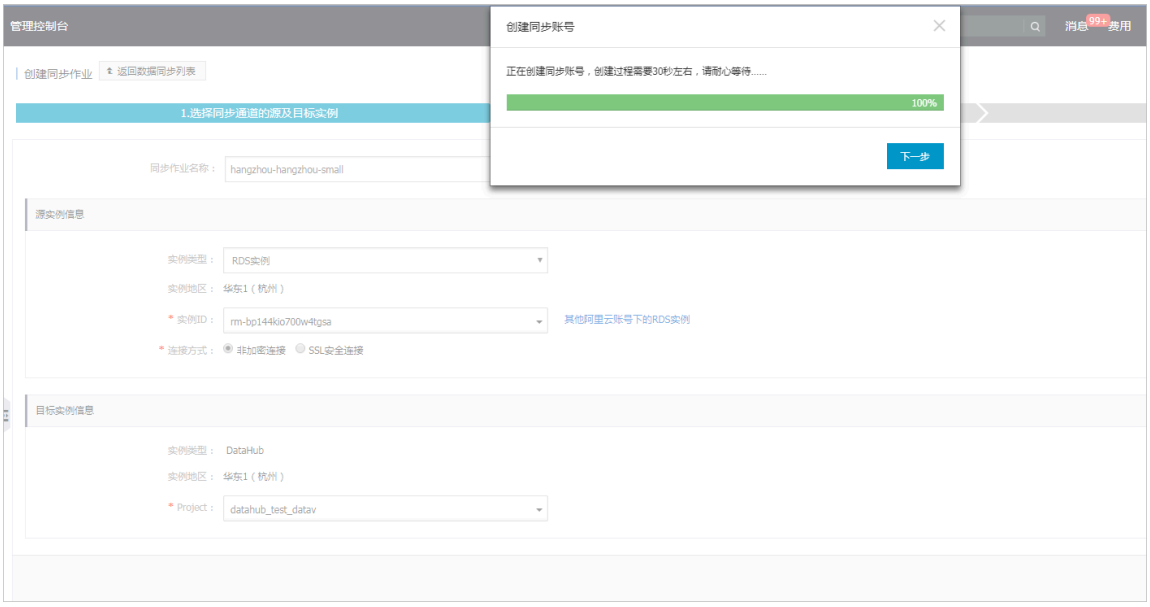
   c. 选择同步作业的基本配置。本案例的配置如下图所示，其中源实例选择MySQL，源实例地域选择华东1区，目标实例选择DataHub。

   d. 返回数据同步页面，单击实例右侧的配置同步链路。

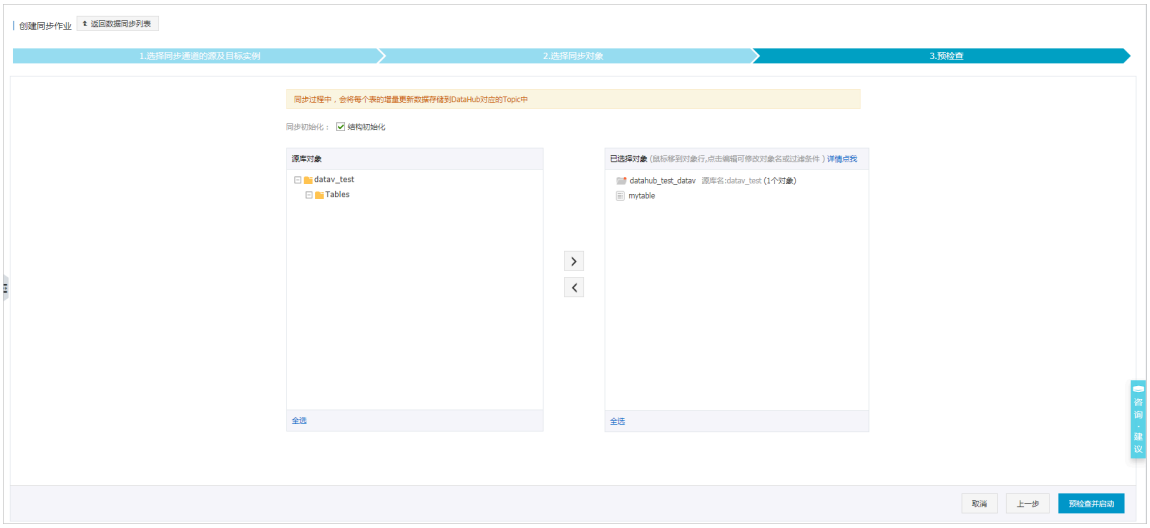
   e. 选择同步通道的源及目标实例，完成后单击授权白名单并进入下一步。



系统会自动为您创建同步账号，创建过程需要30秒左右，请耐心等待。当进度条显示为100%时，单击下一步。



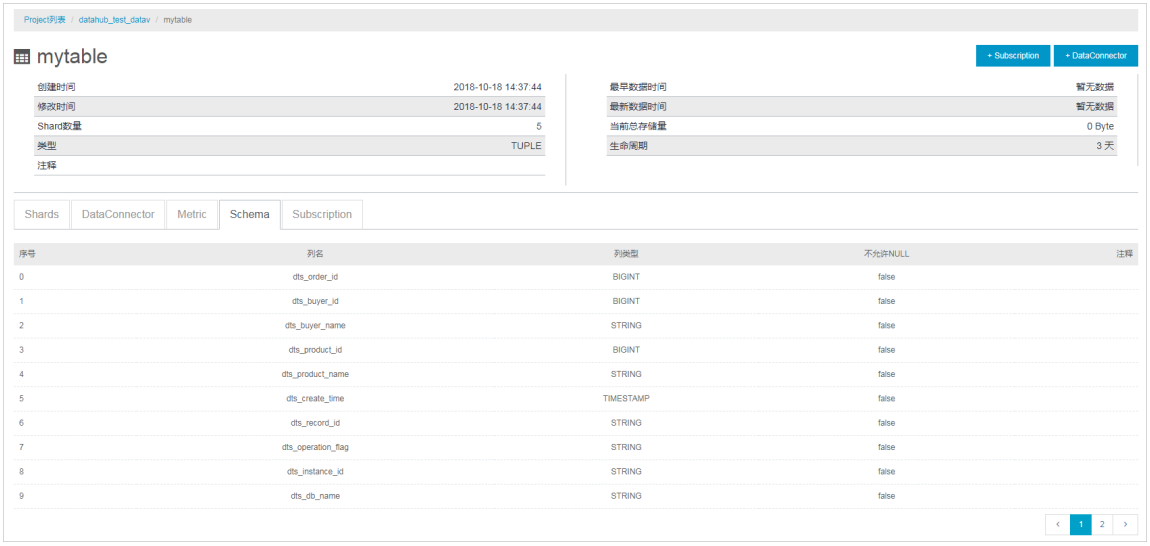
f. **选择同步对象，单击>图标按钮，此时需要同步的对象会出现在已选择对象列表中。本案例的同步对象为datav_test数据库中的mytable表，如下图所示。**



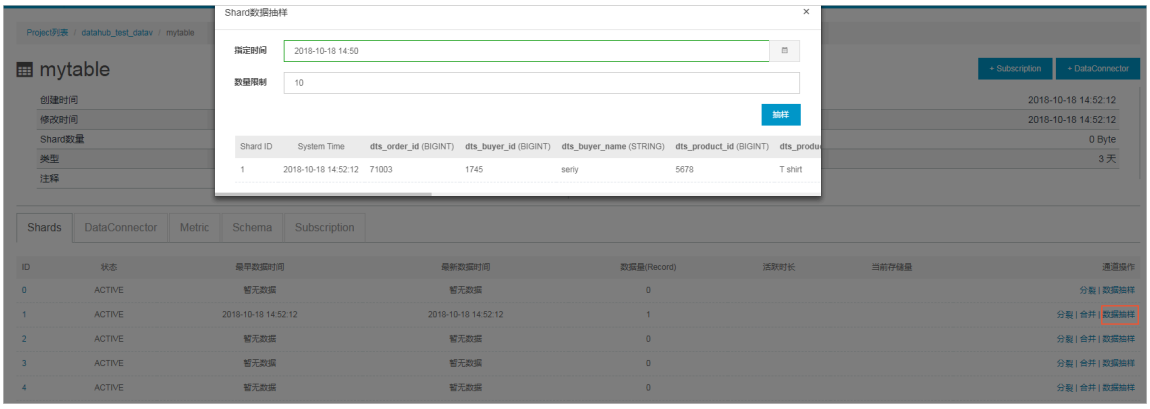
g. 单击预检查并启动，预检查成功后，系统会自动跳转回数据同步页面。正常情况下，可以看到实例的状态为初始化中或同步中，且延时为0毫秒。

3. 查看数据采集结果。

a. 回到DataHub控制台，查看通过DTS创建的数据结构是否有缺失。



b. 在RDS数据库中插入一条数据，单击数据抽样，查看增量数据同步结果。



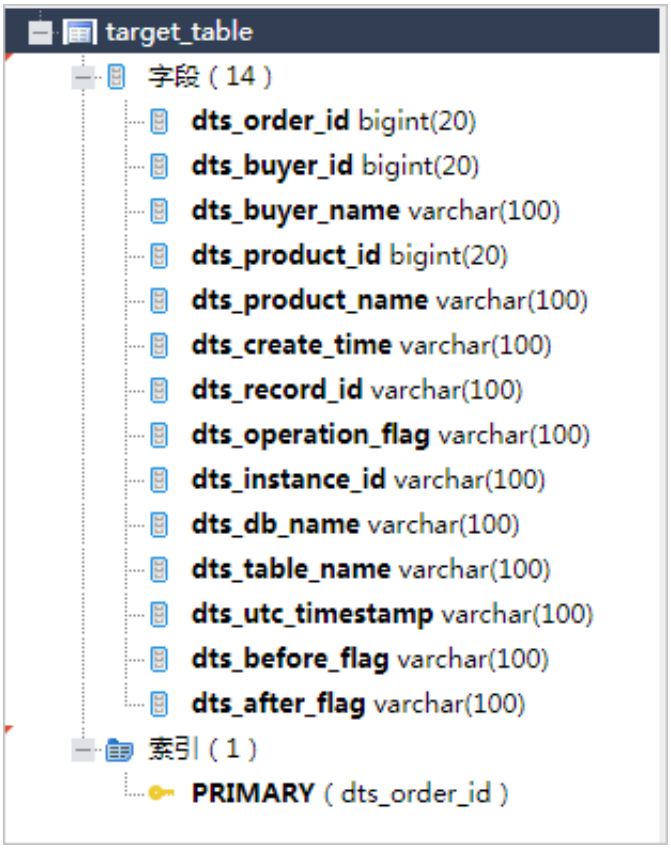
注意:

- DataHub同步的是增量数据，您的数据库中必须插入、删除或更新一条或多条数据后，才会同步到DataHub中。本案例采用手动插入数据的方法，但在实际情况中，都是将用户的购买行为，通过程序的方式动态同步到数据库中的。
- 进行数据抽样时，指定的时间必须在您插入数据之前。
- 参考创建 MySQL 到 MaxCompute 数据实时同步作业的同步原理章节，了解DataHub中各字段的含义。

4.4 通过实时计算订阅数据

订阅DataHub数据进行实时计算，插入到RDS实例的目标表中。

1. 在RDS数据库中，按照源表结构创建相应的表。

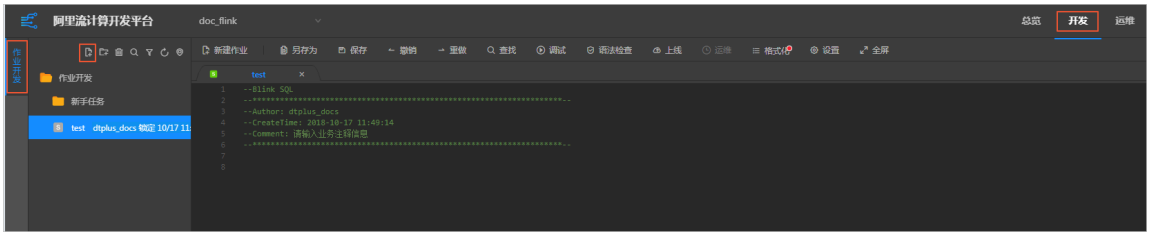


2. 参考数据存储配置白名单，配置RDS数据库白名单。

   由于本案例的RDS数据库位于华东1区，而流计算项目位于华东2区，所以需要在RDS数据库中配置流计算的白名单。

3. 创建流计算开发作业，将DataHub中的数据实时同步到RDS的目标表中。

   a. 进入阿里云流计算开发平台，单击开发 > 作业开发 > 新建作业，创建一个名称为test的开发作业。

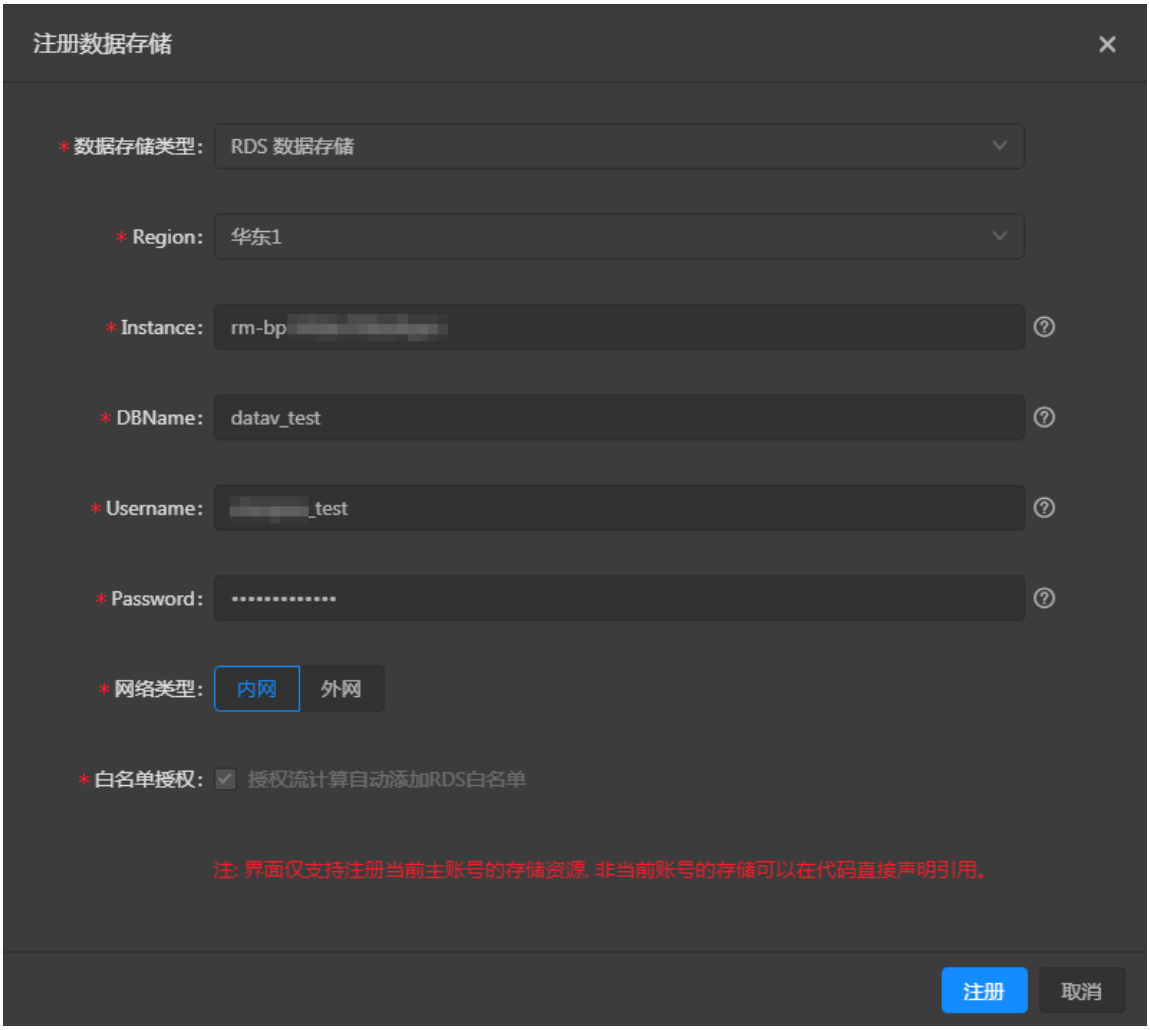


   b. 单击左侧菜单栏的数据存储，右键单击DataHub 数据存储，选择注册数据存储，填入相关信息，单击注册。

参数	说明
数据存储类型	DataHub数据存储。
EndPoint	通过DataHub访问域名获取。

参数	说明
Project	DataHub的项目名称，可在DataHub控制台中获取。

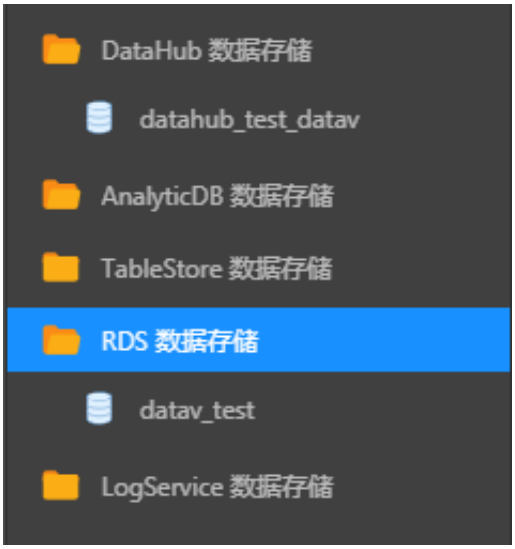
c. 使用同样的方式注册RDS数据存储，参数说明如下。



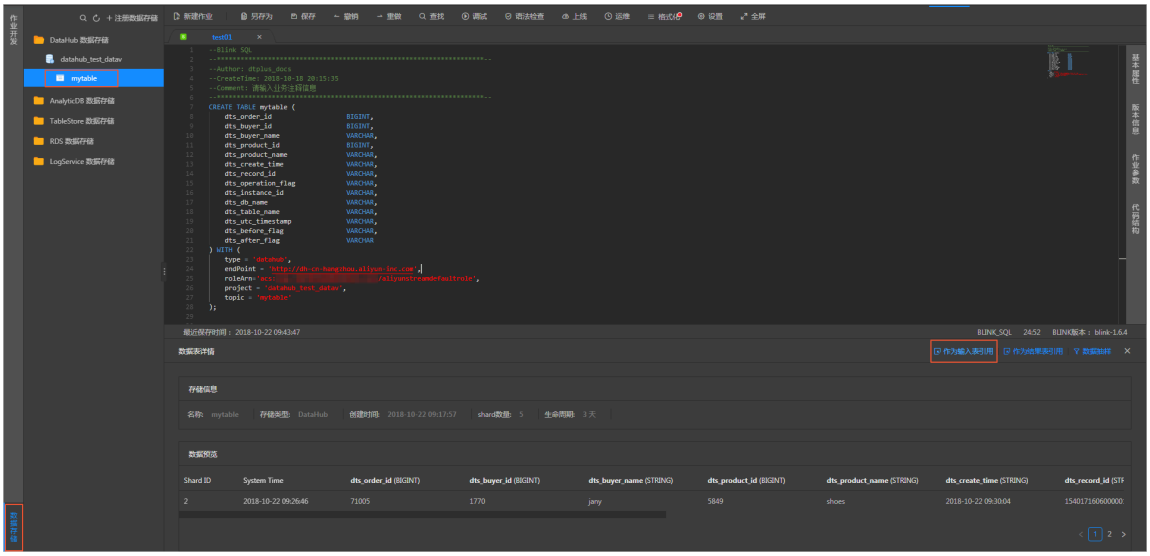
参数	说明
Instance	RDS的实例ID，可在RDS实例的基本信息页面获取。
DBName	数据库名称，可在RDS实例的数据库管理页面获取。
Username	数据库绑定的账号名称，可通过RDS实例的数据库管理页面获取。

参数	说明
Password	创建数据库时设置的密码。

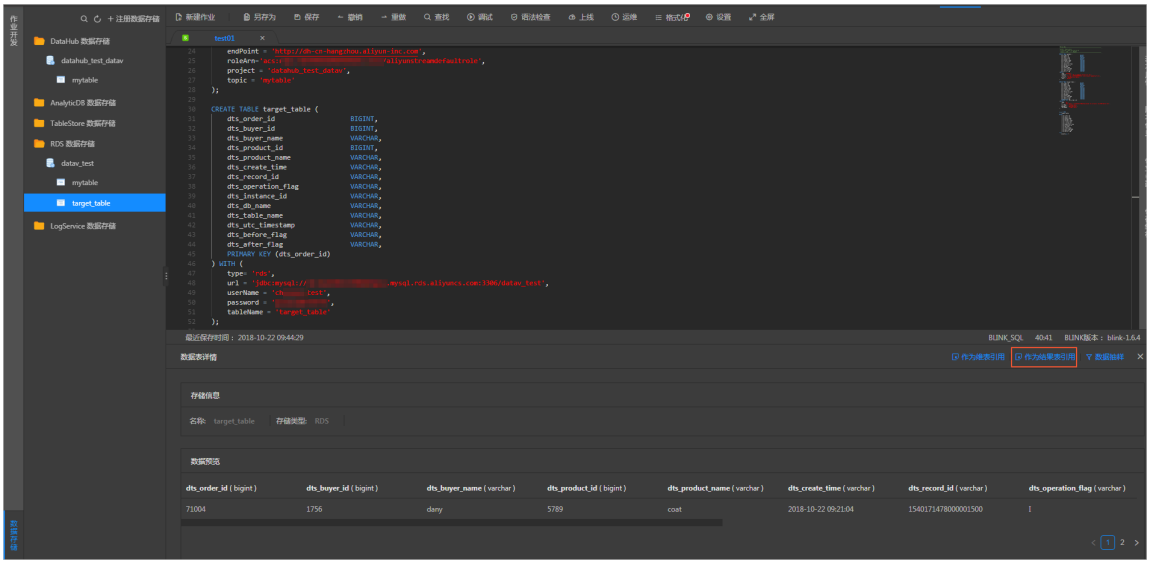
注册完成后，系统显示如下。



d. 依次双击DataHub 数据存储 > 项目名称（datahub_test_datav） > 表名（mytable），选择右侧的作为输入表引用，在开发作业中引用数据源。



e. 依次双击RDS 数据存储 > 数据库名称（datav_test） > 表名（target_table），选择右侧的作为结果表引用，在开发作业中引用目标表。



f. 通过INSERT INTO语句，将实时计算后的源表数据插入目标表中。

```
INSERT INTO
    target_table
SELECT
    t.dts_order_id,
    t.dts_buyer_id,
    t.dts_buyer_name,
    t.dts_product_id,
    t.dts_product_name,
    t.dts_create_time,
    t.dts_record_id,
    t.dts_operation_flag,
    t.dts_instance_id,
    t.dts_db_name,
    t.dts_table_name,
    t.dts_utc_timestamp,
    t.dts_before_flag,
    t.dts_after_flag
FROM
    mytable as t;
```

说明:

如果数据格式不匹配，需要进行相应的数据格式转换工作，例如使用from_unixtime函数等。

示例代码如下：

```
INSERT INTO
target_table
SELECT
t.dts_order_id,
t.dts_buyer_id,
t.dts_buyer_name,
t.dts_product_id,
t.dts_product_name,
```

```
t.dts_create_time,
t.dts_record_id,
t.dts_operation_flag,
t.dts_instance_id,
t.dts_db_name,
t.dts_table_name,
t.dts_utc_timestamp,
t.dts_before_flag,
t.dts_after_flag
FROM
mytable as t;
```

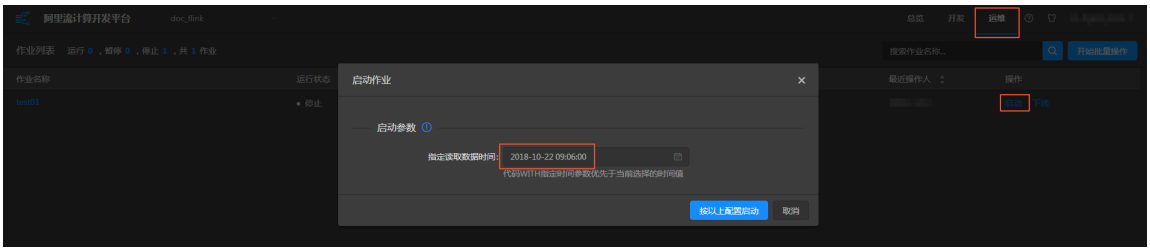
g. 单击上线，选择默认资源配置并进行上线前检查，检查无误后显示如下页面。

h. 单击下一步，填入注释，单击上线。

i. 作业上线成功后，选择控制台上方的运维，单击作业右侧的启动。

j. 在启动作业弹出框中，选择读取数据时间，并单击按以上配置启动。

启动成功后，可查看作业状态，按照需要停止或重启作业，并查看业务延迟。



注意:

选择的读取数据时间必须在数据同步到DataHub之前，否则可能造成数据丢失，影响查询结果。

4.5 通过DataV展示数据

通过DataV的SQL语句功能，处理数据，并将结果展示在大屏上。

1. 添加DataV数据源。
   a. 参考#unique_60，根据您数据库的网络类型，将DataV的白名单添加到您的RDS数据库中。
   b. 登录DataV控制台。
   c. 选择我的数据 > 添加数据。
   d. 填写RDS实例的相关信息，单击确定。

说明:

- 进入云数据库RDS控制台，单击RDS for MySQL实例链接，进入实例的基本信息页面，获取以上信息。
- 如果您的网络类型为内网，则对应的域名为RDS MySQL实例的内网地址。
- 如果您的网络类型为外网，则对应的域名为RDS MySQL实例的外网地址。

2. 创建大屏项目。

a. 选择我的可视化 > 新建可视化。

b. 选择全球贸易模板（本案例以此为例），单击创建。

c. 在创建数据大屏弹出框中，输入数据大屏名称，单击创建。

3. 配置数据源。

a. 在大屏编辑器页面，选择单值百分比饼图组件（本案例以此为例），单击画布右侧的数据图标。

b. 在数据面板中，单击配置数据源。

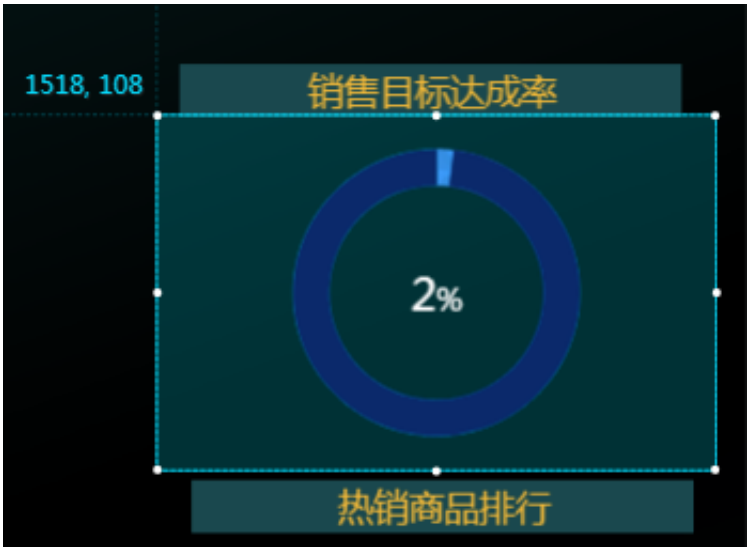
c. 在设置数据源面板中，选择数据源类型为数据库，已有数据源为第一步中创建的数据源。

d. 在SQL脚本编辑区域，输入以下SQL语句。

```
SELECT count(*)/100 as value FROM target_table where dts_operat
ion_flag='I'
```

e. 单击数据响应结果右侧的刷新图标。

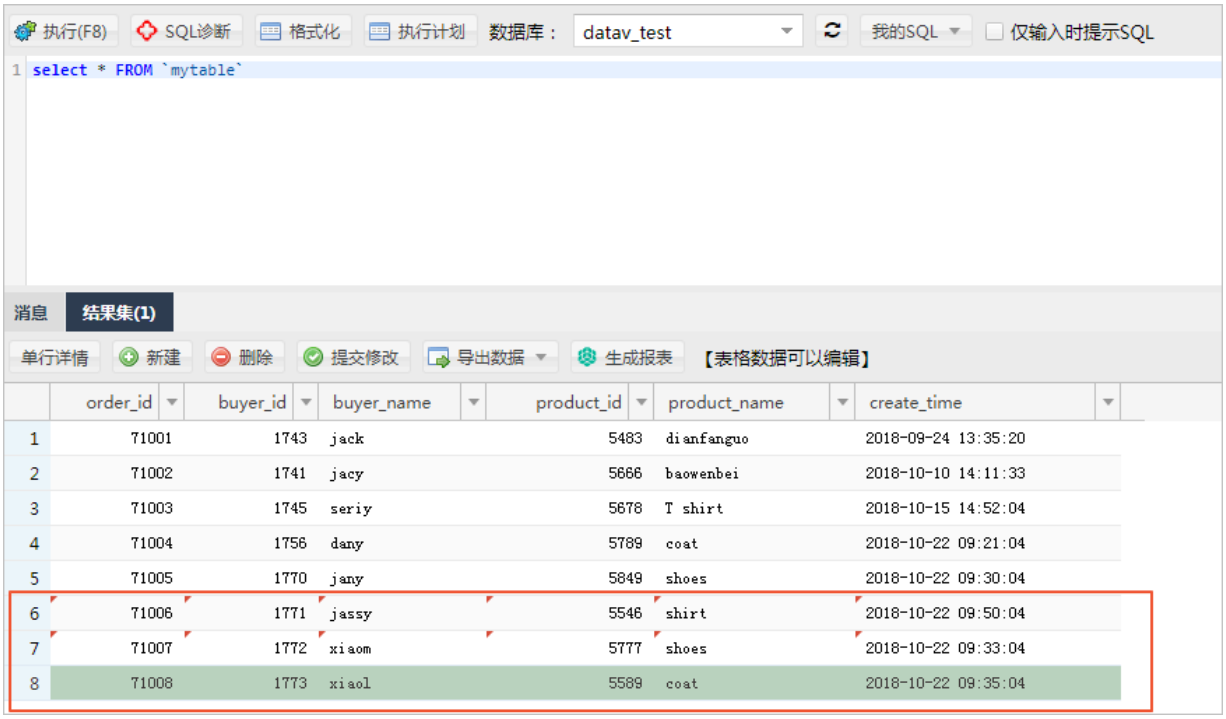
数据响应成功后的结果如下。



4.6 查看结果

本文档为您介绍验证数据同步结果的方法。

在源数据表（mytable）中插入三行数据，验证数据同步结果。



可以看到，在源表中插入三条数据后，销售目标达成率由原来的2%变成了5%。

使用同样的方式，配置其他组件的数据。配置完成后，参考发布可视化应用发布大屏，进行实时计算结果数据的在线展示。

4.7 常见问题

本文档根据实践，介绍了本案例中比较常见的两个问题及解决方法。

1. 流计算中注册RDS数据存储失败。

   可能原因：您的RDS数据库与流计算项目不在同一区域。

   解决方法：需要手动将流计算服务的白名单添加到RDS白名单中，详情请参见#unique_64。

2. 源表数据无法同步到目标表中。

   可能原因：源表的数据类型与目标表不一致，在进行类型转换时出错。例如源表中的`create_time`字段为DATETIME类型，通过DTS同步到DataHub中时，自动转换成TIMESTAMP类型，而目标表为DATETIME类型。此时需要通过SQL函数将TIMESTAMP类型的数据转换成DATETIME类型，在转换的过程中会出现各种预料不到的错误。

   解决方法：目标表中最好不要使用TIMESTAMP或DATETIME类型的字段，使用VARCHAR类型代替。

参考文档：【流数据与大屏DataV】如何使用DTS，Datahub，StreamCompute，RDS及DataV搭建流数据大屏。

5 DataV调用DataWorks数据服务API展示数据成果

5.1 教程概述

本文为您介绍当需要通过DataV展示海量数据的分析结果时，如何使用DataWorks的数据服务开发数据API，并快速在DataV中调用API，最终将来自MaxCompute的数据成果展示在DataV大屏中，数据开发到数据服务再到数据分析展现一气呵成。

注意:

本案例仅提供具体的操作指导，业务场景和数据与实际情况会有一定偏差。

DataWorks数据服务与DataV进行无缝打通后，就不再需要使用DataV中的API数据源去填写一个URL调用API，而是直接新建一个DataWorks数据服务作为数据源，就可以选用数据服务中的API。无需每个API都设置AppKey和AppSecret认证信息，并且支持通过表单填写API参数，使用起来十分方便可靠。

本案例以在DataV中展示成交金额增长速度为例，为您介绍在DataV中配置DataWorks数据服务API的方法，整体步骤如下。

1. 准备工作。
2. 使用DataWorks的数据服务功能生成数据API。
3. 在DataV中调用数据服务API。

参考文档

- 一分钟零代码生成API，DataWorks数据服务上手指南。
- DataWorks数据服务帮助文档。
- MaxCompute Lightning帮助文档。
- DataV帮助文档。

5.2 注意事项

本文提供一些教程中的注意事项供您参考。

- DataWorks数据服务向导模式生成API，只支持单表简单条件查询。脚本模式支持用户编写查询SQL语句，支持多表关联查询、函数以及复杂条件，您可以根据自己的需求灵活选择。

- Lightning采用PostgreSQL的语法，因此在编写SQL时，需要注意：
  - 要使用PostgreSQL函数，而不是MaxCompute的UDF。
  - 目前Lightning仅支持`max_pt`这个MaxCompute UDF，可用于获取当前最新分区。
  - 连接字符串时使用||。
- Lightning目前只支持秒级查询，并且查询的MaxCompute不宜过大（控制在GB级），尽量将分区作为请求参数，避免扫描过多分区，否则查询速度会比较慢。
- 如果您要求毫秒级API查询，则建议采用关系型数据库、NoSQL数据库或AnalyticDB作为数据源。
- DataV组件要求的数据格式是个数组，数据服务生成的API返回结果是个带有错误码的完整JSON，因此要使用过滤器对API结果进行处理。

  您可以在DataV中添加过滤器，也可以直接在数据服务配置API时添加过滤器。一般来说，对于未分页查询的API，直接返回`data`数组即可；对于分页查询的API直接返回`data.rows`数组。
- 如果您需要在DataV的折线图或柱状图中添加多个系列，DataV一般要求每个系列的数据是一个对象，并通过s字段来区分系列，此时要注意使用数据过滤器进行格式转换。

5.3 准备工作

本文为您介绍在使用DataV调用DataWorks数据服务API展示数据成果前，需要完成的准备工作。

1. 创建DataWorks工作空间，选择region（本案例选择华东1），勾选MaxCompute和数据开发

2. 购买DataV产品企业版。
3. 连接交互式分析（Lightning）服务。

   由于本案例使用的是Lightning数据源，故需要首先连接MaxCompute Lightning服务。
4. 开通API网关服务。

   根据界面提示，开通API网关服务，以确保您能够使用DataWorks数据服务的API功能。
5. 准备好本案例的数据表并上传数据。

   本案例使用Lightning数据源，故首先需要在MaxCompute控制台中创建表，并上传数据。

   - 本案例使用的建表DDL语句如下。

```
CREATE TABLE IF NOT EXISTS `demo_trade_amount` (
`id` bigint COMMENT 'id',
`date1` string COMMENT '成交日期',
`amount` string COMMENT '成交额'
) ;
```

   - 本案例最终的表结构和部分数据如下图所示。

	A	B	C
1	id	date1	amount
2	1	2019-01-01	2000
3	2	2019-01-02	2011
4	3	2019-01-03	2060
5	4	2019-01-04	2105
6	5	2019-01-05	2500
7	6	2019-01-06	2070
8	7	2019-01-07	2060
9	8	2019-01-08	2700
10	9	2019-01-09	2040
11	10	2019-01-10	2002
12	11	2019-01-11	2010
13	12	2019-01-12	2000
14	13	2019-01-13	2080
15	14	2019-01-14	2060
16	15	2019-01-15	2300

5.4 使用DataWorks数据服务生成API

本文为您介绍如何通过DataWorks的数据服务，生成并发布数据API，用于在DataV中进行调用并展示。

前提条件

在开始本案例前，您需要首先完成准备工作。

背景信息

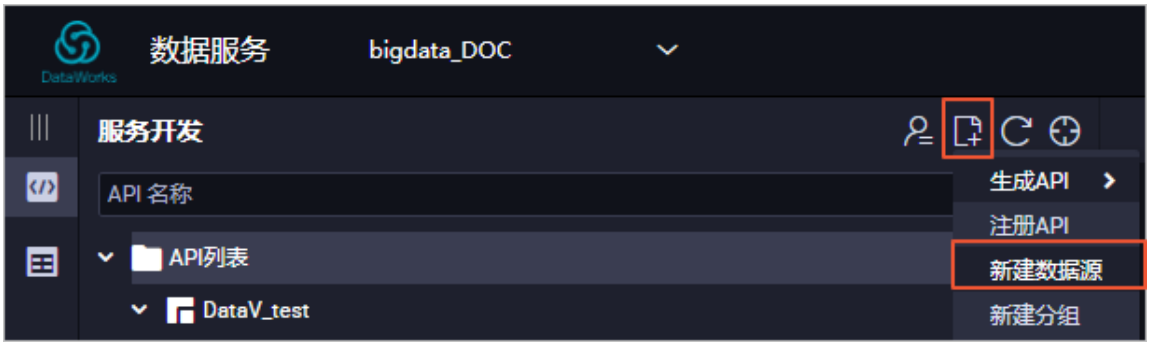
> **注意:**
> 按照本文档操作后，可以获取到您数据API的AppCode、AppKey和AppSecret，请妥善保管，谨防泄露。

通过DataWorks数据服务生成API主要包含以下三个步骤。

1. 创建数据源：新建MaxCompute Lightning数据源。
2. 配置API：创建好数据源后，在数据服务页面，以向导模式生成并配置API。
3. 发布API：API配置完成并测试成功后，就可以进行发布，提供给DataV调用。

操作步骤

1. 创建数据源。
   a) 进入DataWorks的数据服务控制台，单击新建 > 新建数据源。

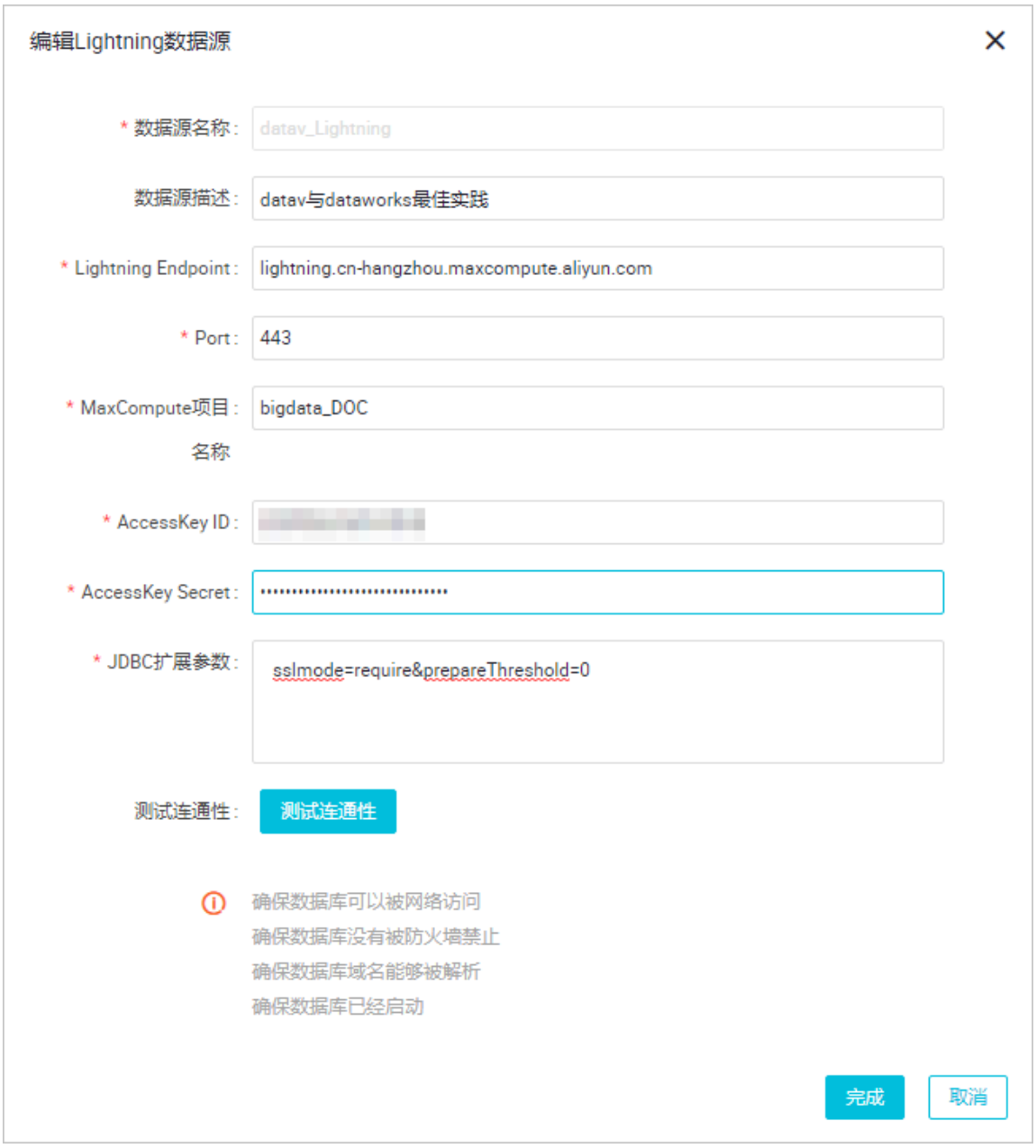


   b) 在新开的数据集成页面，单击数据源 > 新增数据源。
   c) 在新增数据源弹出框中，单击大数据存储模块下的Lightning。
   d) 在新增Lightning数据源弹出框中，输入数据源名称、Lightning Endpoint等相关信息，完成后单击测试连通性，连通性测试通过后即可完成数据源的创建。

> **说明:**

参考配置 JDBC 连接获取Lightning的连接信息，如Lightning Endpoint、Port等。

本案例的配置如下图所示。

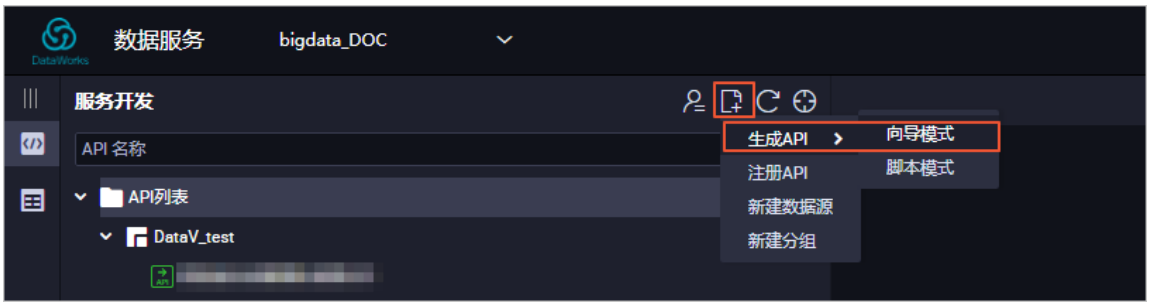


注意：

JDBC扩展参数中的sslmode=require&prepareThreshold=0是必须的，不可删除，否则会无法连接。

2. 新建API。

a) 在DataWorks的数据服务页面，单击新建 > 生成API > 向导模式。



说明:

本案例以向导模式为例生成API，您也可以使用脚本模式。

b) 在生成API 弹出框中，输入相关信息，单击确认。

本案例的配置信息如下图所示。

c) 在API配置页面配置API。

A. 选择表。

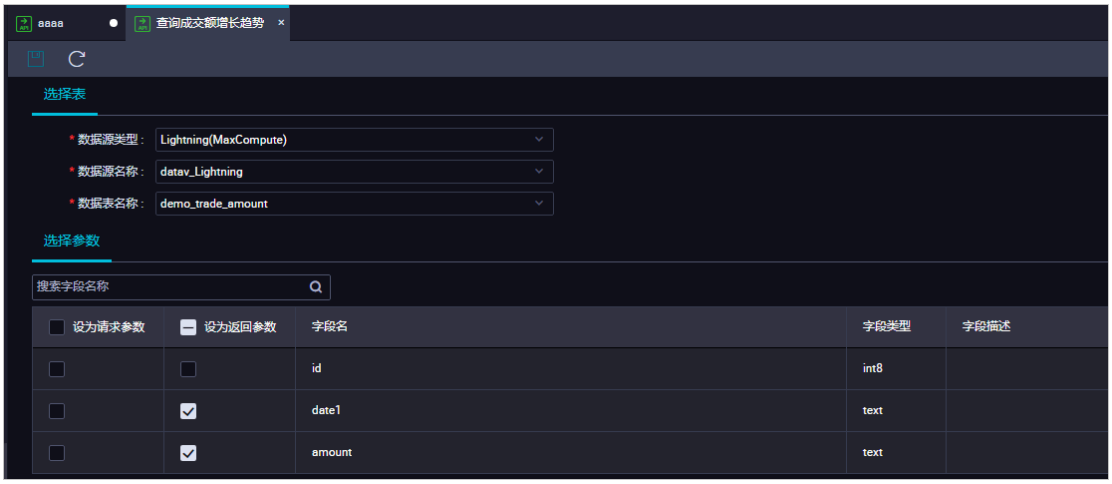
- 数据源类型为Lightning(MaxCompute)。
- 数据源名称为您之前步骤新建的数据源。
- 数据表名称为您已经准备的数据表。

B. 选择参数。

选择好表之后，会自动展示表的字段列表。勾选您要作为API请求参数的字段和作为返回参数的字段。

本案例是为了查询成交金额趋势，因此要返回所有数据，即将日期和成交金额都作为返回参数，不设请求参数。

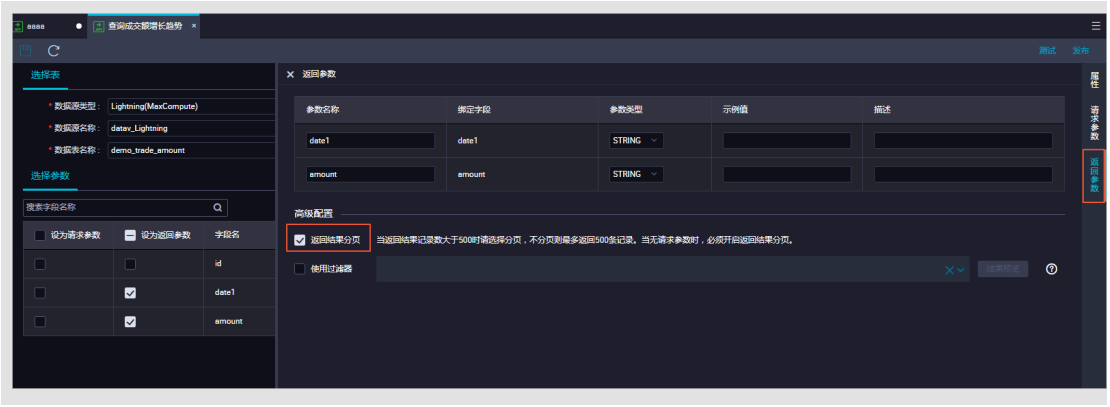
本案例的最终配置如下图所示。



C. 单击页面右侧的返回参数，设置参数描述信息。

注意：

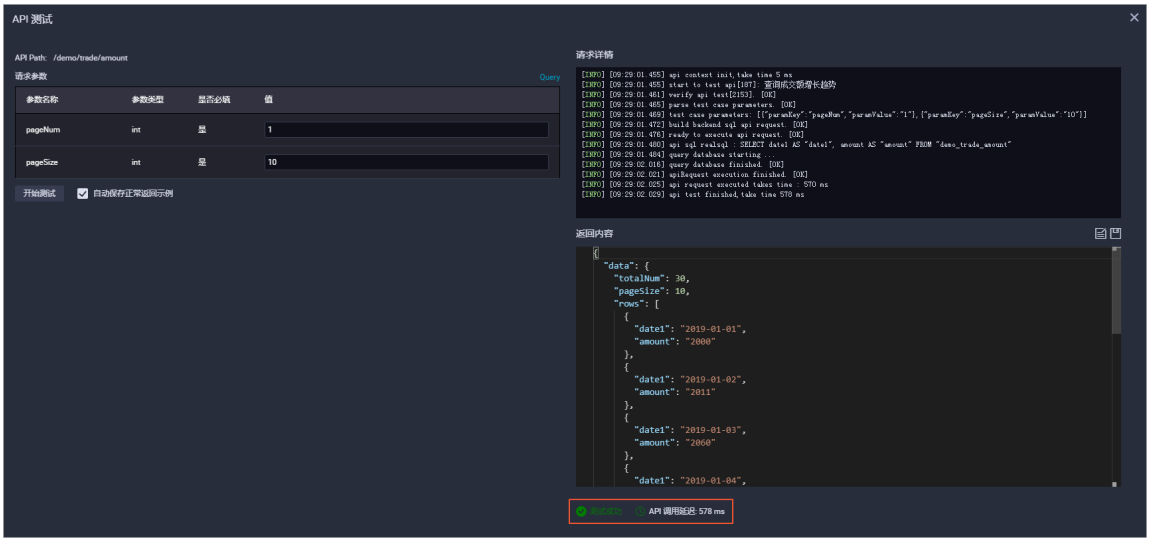
如果不设置请求参数，则需要勾选返回结果分页，进行分页查询，以避免单次查询返回数据量过大影响性能。



d) API测试。

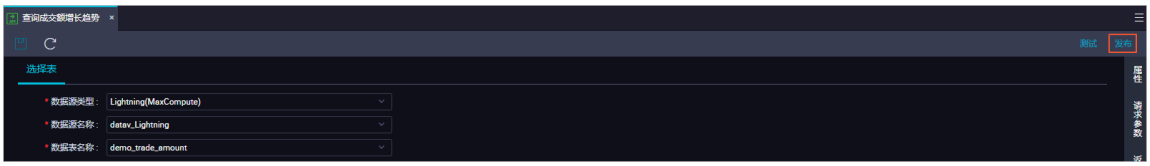
单击页面工具栏最右侧的测试，填写API请求参数（由于打开了分页查询开关，系统会自动添加两个分页参数），单击开始测试。

测试成功后，系统返回请求数据，并显示测试成功和 API调用延迟时间，如下图所示，可以看到通过Lightning查询MaxCompute表只花费了不到1秒，比直接通过MaxCompute SQL查询快了几十上百倍。

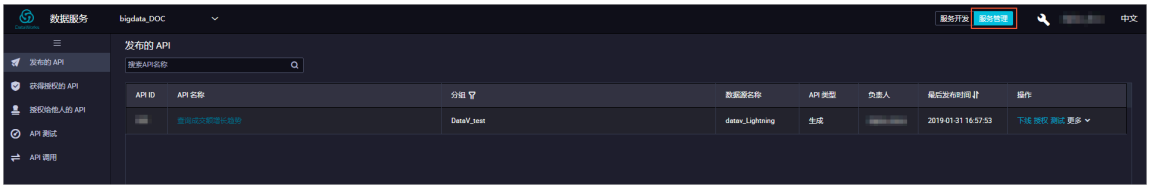


3. 发布API。

a) 单击页面工具栏最右侧的发布，即可进行发布。



b) 发布成功后，单击页面右上角的服务管理，再单击API名称，查看API详情。



c) 单击左侧导航栏的API调用，获取AppCode（简单身份认证）和AppKey、AppSecret（加密签名身份认证），在调用API时需要进行认证。

注意:

请妥善保管您的AppCode、AppKey和AppSecret，谨防泄露。

发布成功后，系统显示发布成功。

5.5 使用DataV大屏展示数据返回结果

本文为您介绍如何在DataV中调用DataWorks的数据服务API，并将数据返回结果展示在DataV大屏中。

前提条件

在开始本案例前，您需要首先完成准备工作，并已经生成了数据服务API。

背景信息

警告:

您必须使用HTTP协议进入DataV控制台，否则会导致API数据响应失败。

通过DataV调用数据服务API主要包含以下二个步骤。

1. 添加DataWorks数据源：在DataV中添加DataWorks数据源。
2. 在大屏中调用数据服务API：在组件的数据配置中，配置API数据源参数。

操作步骤

1. 添加DataWorks数据源。
   a) 使用HTTP协议进入DataV控制台，选择我的数据 > 添加数据。
   b) 单击类型下拉箭头，选择数据类型为DataWorks数据服务。
   c) 在添加数据对话框中填写DataWorks数据服务项目信息，本案例的配置信息如下图所示。

参数	说明
自定义数据源名称	数据源的显示名称，可以自由命名。
项目	DataWorks项目（工作空间）。
Region	需要与您的DataWorks项目在同一Region下。
AppKey/AppSecret	参考使用DataWorks数据服务生成API中的第三步，获取AppKey和AppSecret。

2. 在大屏中调用数据服务API。

a) 在DataV控制台中，单击我的可视化 > 新建可视化。

b) 选择一个模板，单击创建。

c) 在创建数据大屏对话框中，输入数据大屏名称，单击创建。

本案例选择企业实时销售数据模板。模板中的组件自带了静态数据，本案例以模板中的基本折线图组件为例，将组件数据配置为查询成交金额增长趋势的API。

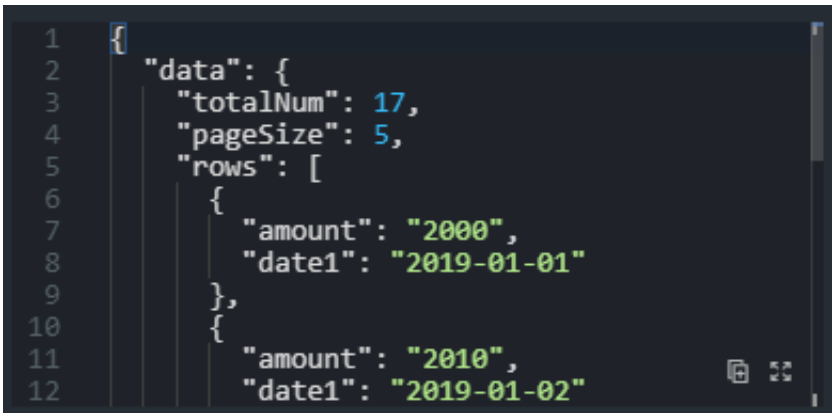
d) 单击模板中的基本折线图组件，在右侧的数据配置面板中，单击配置数据源。

e) 在设置数据源页面中，选择数据源类型为DataWorks数据服务，已有数据源为您第一步中添加的数据源，API为您已经发布的API。

说明:

本案例将上图中的pageSize设置为5，表示查询5天的数据。

f) 单击预览数据源返回结果，查看API的查询结果，如下图所示。



g) 返回组件的数据面板，在字段映射关系中，将x字段映射为date1（日期为x轴），y字段映射为amount（成交额为y轴）。

说明:

配置完成后，可以看到，当前x和y无法匹配到字段。这是由于DataV对数据格式有一定要求，不能识别结构较深的字段。因此需要添加一个数据过滤器，过滤掉不必要的字段，在本案例中直接返回rows数组即可。

h) 在组件的数据面板中，勾选数据过滤器，单击添加过滤器。

i) 在设置数据源页面中，单击添加过滤器右侧的+。

j) 在过滤器代码编辑框中输入`return data.data.rows;`，单击测试（需要开启过滤器调试功能）。测试成功后，单击保存 > 完成。

数据过滤器支持使用JavaScript代码对数据结果进行二次过滤和处理，过滤器的data参数为API返回结果的JSON对象。本案例中，您只需要返回API结果中的rows数组，故需要输入代码`return data.data.rows;`即可，过滤完成后可以看到数据匹配成功。

说明:

此时折线图并没有正确展示，由于API返回的日期格式与组件默认的格式不一样，因此您还需要设置x轴的轴标签样式。

k) 切换到组件的配置面板，设置x轴轴标签的数据种类为类目型。

组件的样式和数据都配置完成后，可以看到组件使用API数据，且显示正常。

后续步骤

按照以上方法，可以配置其他组件的数据和样式，完成大屏制作。

5.6 发布大屏

本文档为您介绍预览和发布可视化大屏的方法。

按照以上方法，配置其他组件的数据和样式，完成大屏制作。大屏制作完成后，可进行#unique_84和#unique_85，最终得到一个您满意的可视化大屏。


企业实时销售数据
2019-07-31 11:26:31
销售额分省排名
Sales ranking points province
销售总额
232,425,710元
同比增长
+150%
一季度销售进度
Quarterly sales progress
01
83%
一季度销售额
1321万
同比增长
+150%
城市销售额排名
Sales ranking
02
销售额增速最快
北京 +20.60%
销售员业绩排名
Sales performance ranking
03
销售额同比增速最快
员工A +20.60%
品类销售额排名
Category sales rankings
04
品类销售额占比
Types of sales
05
18%
TYPE_1
客户行业占比
Customer industry distribution
06
18%
TYPE_1
XXX类商品
10.3亿
XXX类商品
9.8亿
新增客户数
107个
同比增长
+150%
销售额增速
企业logo图片
企业经营指标1 12425万元
企业经营指标2 14354万元
企业经营指标3 12425万元
企业经营指标4 14354万元
企业经营指标5 12425万元
企业经营指标6 14354万元

6 使用DataV节点编程搭建交互式学区地图大屏教程

6.1 教程概述

本教程为您介绍使用DataV的节点编程功能，搭建学区房可视化交互大屏的方法。

> **注意:**
> 本教程中的数据仅供参考，无实际意义，旨在帮助您了解较为复杂的交互式大屏的配置方法。具体数据需要根据您的实际需求进行配置。

具体操作步骤如下。

1. 准备工作。

   准备交互需求。

2. 配置学区房节点编程交互。

   a. 创建学区地图大屏。

      使用学区地图模板创建大屏。

   b. #unique_90。

      当单击Tab列表的选项时，切换展示小学和初中的统计信息（包括学校数量、区位占比等）和地理位置信息。

   c. 配置学区地图单选框交互。

      当单击单选框的选项时，切换展示不同类型学校的散点层数据，以及Tab列表和单选框的双重触发判断。

   d. 配置学区地图区域热力层交互。

      当鼠标划过地图的区域热力层子组件时，切换展示当前区域对应的学校数据。

   e. 配置学区地图轮播列表交互。

      当单击轮播列表组件时，在地图上切换展示对应学校的位置信息和数据。

3. 查看大屏效果。

   将交互效果应用到大屏上，并进行预览发布。

6.2 准备工作

本文档为您介绍在配置学区房交互大屏之前需要完成的准备工作。

准备交互需求。

本案例中的交互需求如下。

- 当单击Tab列表选项卡时，可以切换小学和初中这两个场景。
- 当单击单选框的选项时，可以切换全选、公办和民办这三个场景。
- 当用户鼠标滑过学校地图的区域热力层时：
  - 大屏左下角切换展示当前学校的基本信息，包括名称、地址和属性。
  - 大屏右上角切换展示该区域内学区房的成交量排行榜信息和房价趋势图。
- 当单击用来展示学校排名的轮播列表组件内某一行时，在地图组件上定位该学校位置，并通过散点层子组件返回该学校的位置信息。

6.3 配置学区地图节点编程交互

6.3.1 创建学区地图大屏

本文档为您介绍创建学区地图大屏的方法。

操作步骤

1. 登录DataV控制台。
2. 单击我的可视化 > 新建可视化。

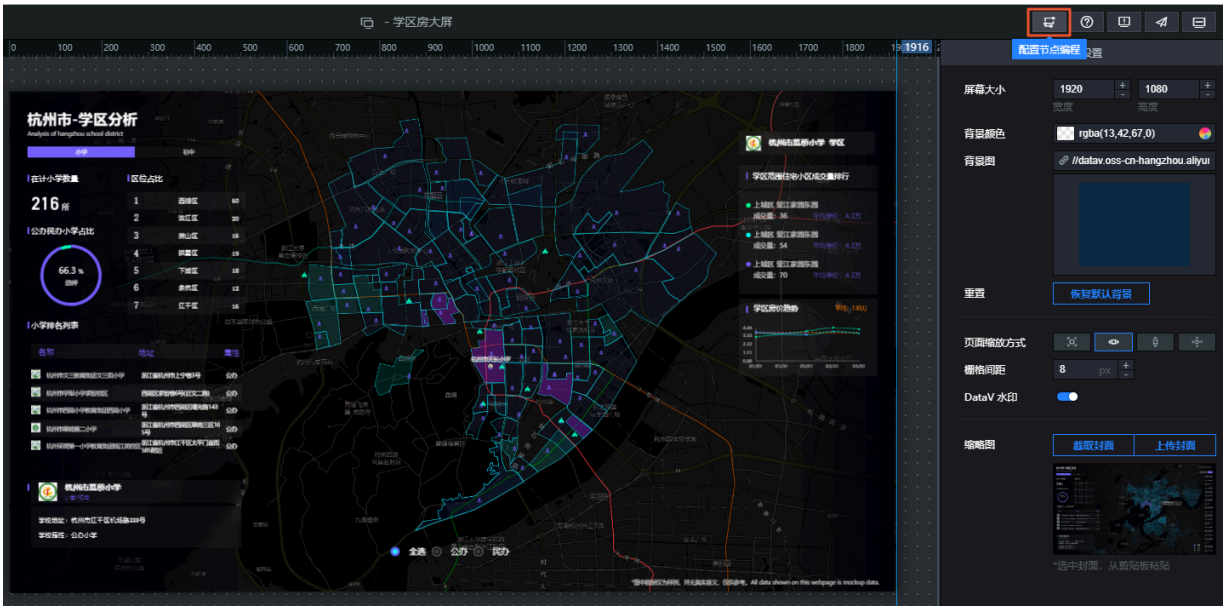
3. 选择学区地图模板，单击创建。

4. 在创建数据大屏对话框中，输入数据大屏名称，单击创建。

   大屏创建成功后会跳转到大屏编辑器页面。您可以在大屏编辑器中查看或编辑组件，完成可视化应用的开发。

后续步骤

学区房大屏创建成功后，单击大屏右上角的配置节点编程图标，进入节点编程页面，配置大屏交互，详情请参见#unique_90。



6.3.2 配置学区地图Tab列表交互

本文档为您介绍通过单击Tab列表选项，切换展示小学和初中的统计信息（包括学校数量、区位占比等）和地理位置信息的方法。

背景信息

本文档包括以下两个场景的配置。

- Tab列表切换小学和初中的统计信息。
- Tab列表切换小学和初中的地理位置信息。

Tab列表切换小学和初中的统计信息

需要实现的交互：当单击Tab列表的选项时，大屏中切换小学和初中2类学校的统计信息。

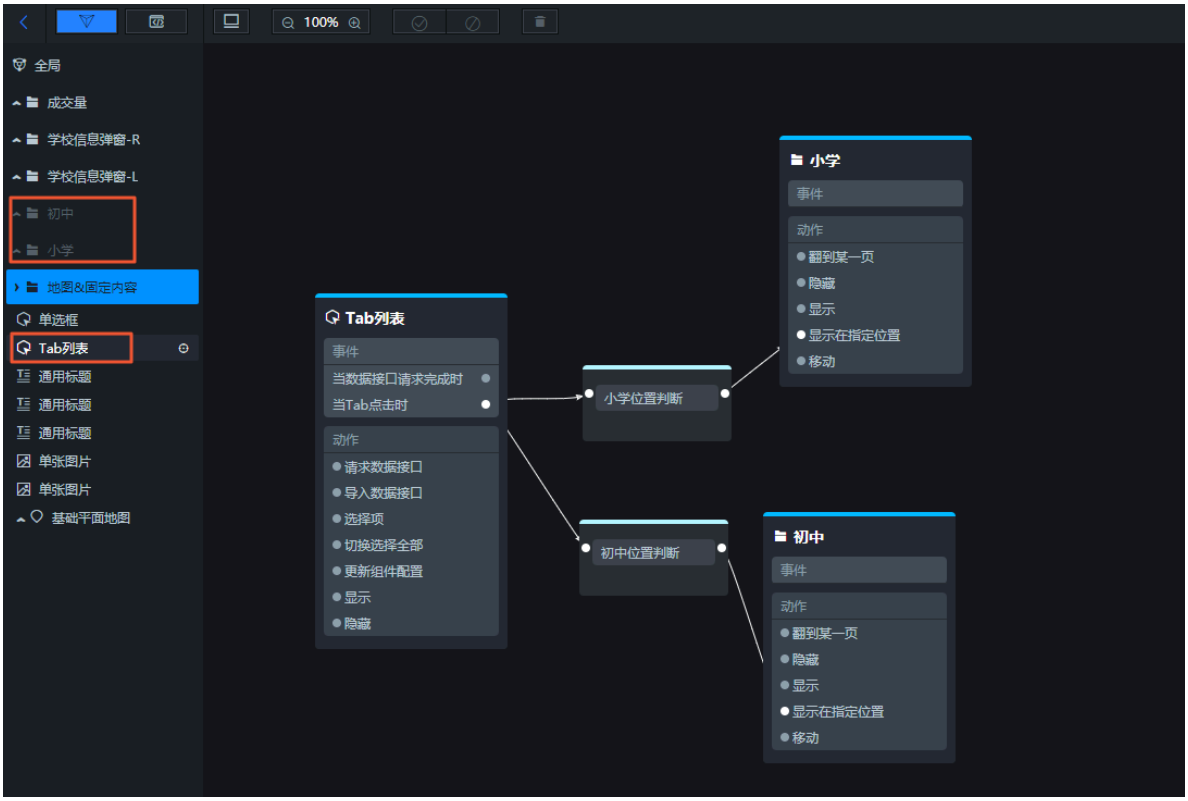
可实现方案：

- Tab列表控制成组组件的显隐样式，详情请参见#unique_97。
- 控制成组组件的位置显示，本文以此为例，具体操作步骤如下。

1. 在节点编程配置页面，将左侧地图&固定内容中的Tab列表、小学组和初中组节点拖至画布中。
2. 按照以下说明进行连线。

   将Tab列表的当Tab点击时事件分别与小学组和初中组的显示在指定位置动作连线。
3. 在连线中添加两个转换器，分别命名为小学位置判断和初中位置判断。

   具体操作方法请参见#unique_98。转换器添加完成的效果如下图所示。



4. 配置转换器。

a) 右键单击名称为小学位置判断的转换器，选择编辑。

b) 在转换器设置对话框的转换器脚本编辑区域，输入转换条件，完成后单击保存。

当前转换器的示例代码如下。

```
if (data.id == 1) {      //小学位置判断。
  return {
    "x": -16,
    "y": 160             //x,y轴数据代表目标位置处。
  };
} else {
  return {
    "x": -16,
    "y": 1080          //这个数据下的位置将被被移除画布范围。
  };
}
```

c) 使用同样的方式配置初中位置判断转换器，示例代码如下。

```
if (data.id == 2) {      //初中位置判断。
  return {
    "x": -16,
    "y": 160             //x,y轴数据代表目标位置处。
  };
} else {
  return {
    "x": -16,
    "y": 1080          //这个数据下的位置将被被移除画布范围。
  };
}
```

Tab列表切换小学和初中的地理位置信息

需要实现的交互：当单击Tab列表的选项时，大屏中的基础平面地图组件切换小学和初中2个区域热力层样式，并显示对应的散点。

实现方案：使用Tab列表和触发器控制地图子组件的显隐样式，具体操作步骤如下。您也可以通过转换器实现，详情请参见#unique_97。

1. 在节点编程配置页面，将左侧地图&固定内容 > 基础平面地图中的公办小学、公办初中、公办小学点、公办初中点、民办小学和民办初中六个节点拖至画布中。

   说明:

   Tab列表节点在上一步中已经添加至画布中，不需要重复添加，可在下文的步骤中继续使用。

2. 按照以下说明进行节点连线，并添加触发器。

   a) 将Tab列表中的当Tab点击时事件与公办小学的显示动作连线。

   b) 在上述连线中添加触发器，命名为id=1：小学。

      详细操作方法请参见#unique_98。

   c) 将Tab列表中的当Tab点击时事件与公办初中的显示动作连线。

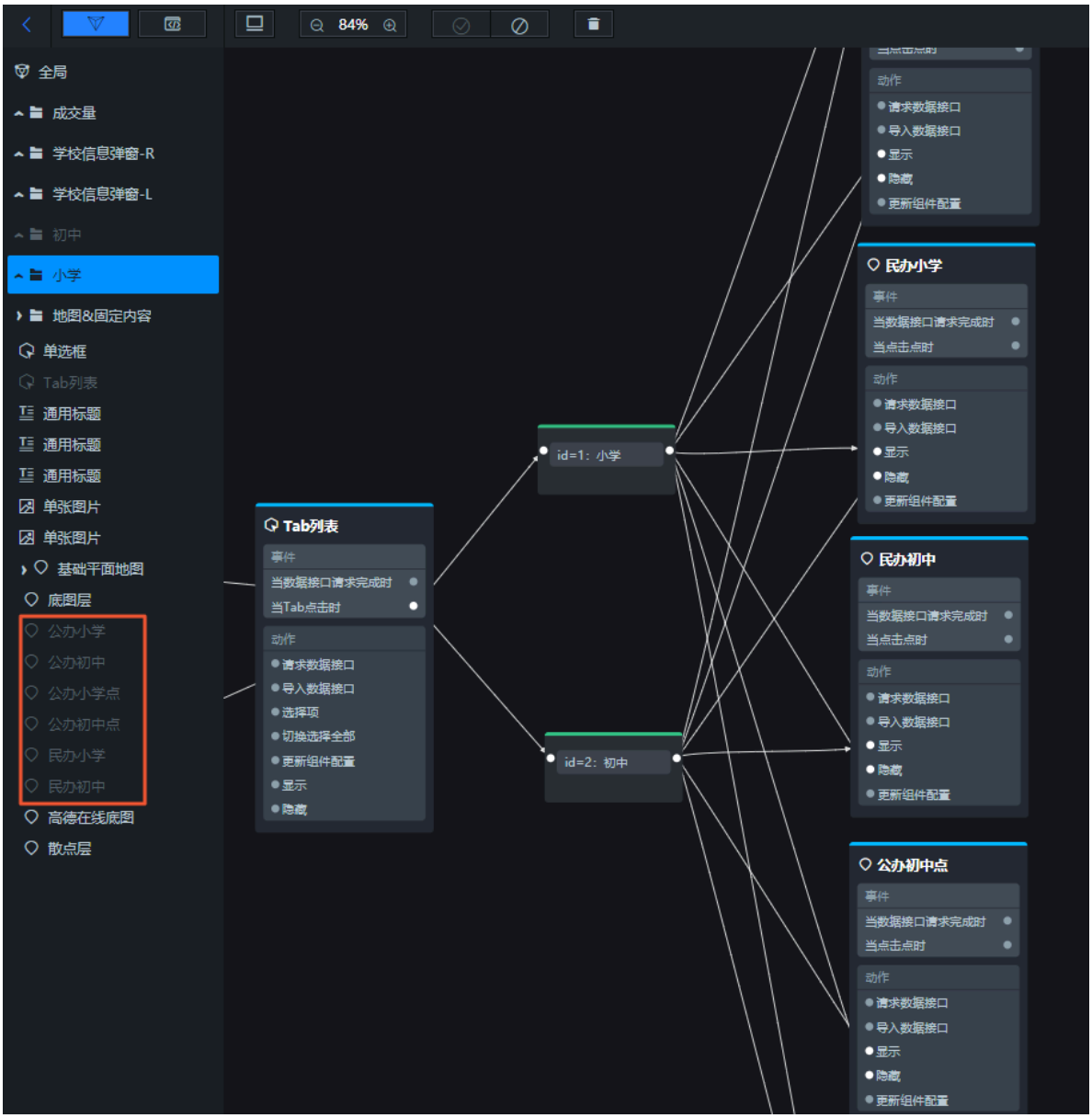
   d) 在上述连线中添加触发器，命名为id=2：初中。

      详细操作方法请参见#unique_98。

3. 按照以下说明完成触发器和节点之间的连线。

   - 将id=1：小学触发器与公办小学点和民办小学点的显示动作连线；与公办初中点和民办初中点的隐藏动作连线。
   - 将id=2：初中触发器与公办小学点和民办小学点的隐藏动作连线；与公办初中点和民办初中点的显示动作连线。

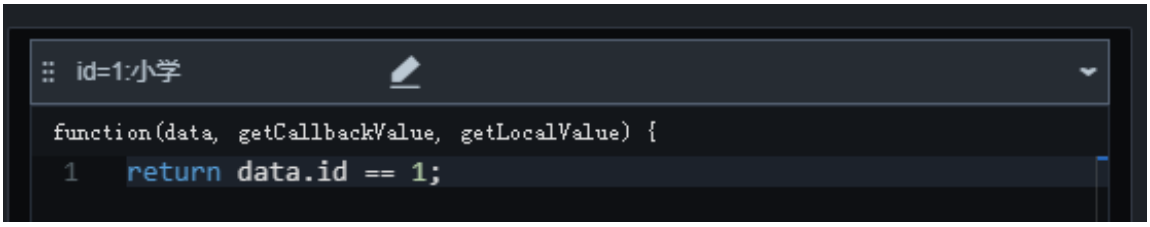
连线完成后的样式如下图所示（截图中只包含了部分节点与连线）。



4. 配置触发器。

   a) 右键单击名称为id=1：小学的触发器，选择编辑。

   b) 在触发器设置对话框的触发器脚本编辑区域，输入转换条件，完成后单击保存。



   当前触发器的示例代码如下。

```
return data.id == 1;
```

   说明:

   以上触发条件的原理是：单击小学Tab时，三个小学相关的地图子组件显示在大屏相应位置；三个初中相关的地图子组件隐藏不显示在大屏中。

   c) 使用同样的方式配置id=2：初中的触发器，示例代码如下。

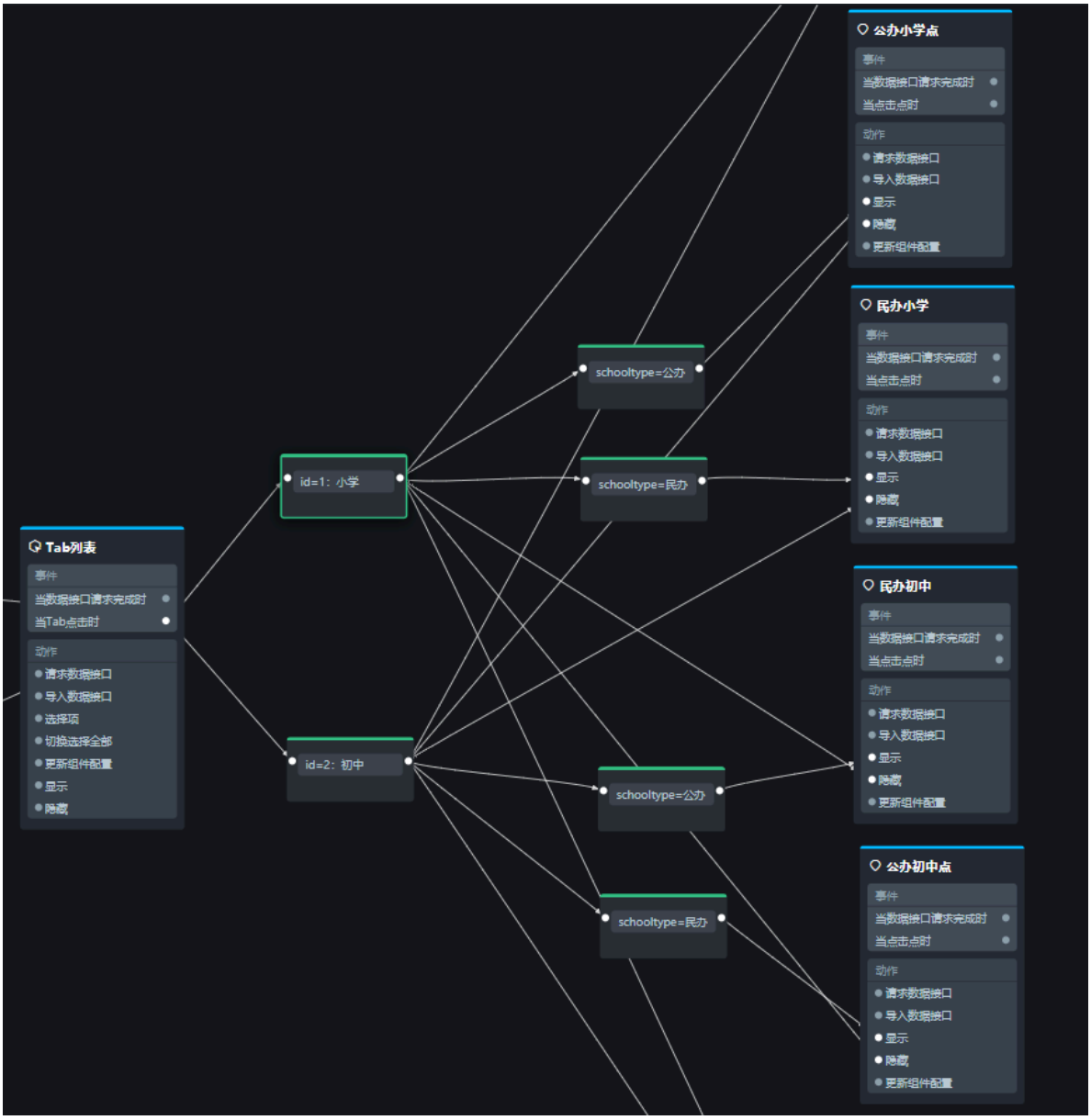
```
return data.id == 2;
```

   由于Tab列表控制地图子组件显隐的同时要考虑到和单选框选项内的数据保持一致，因此需要继续执行以下步骤进行配置。

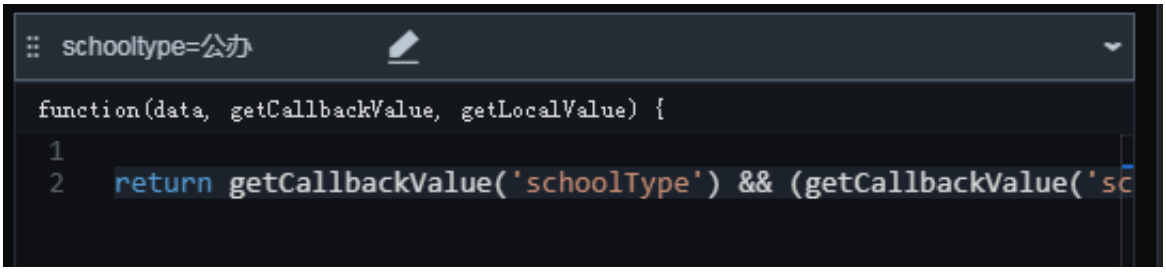
5. 在id=1：小学触发器与公办小学点和民办小学的连线中分别添加两个触发器，并分别命名为schooltype=公办和schooltype=民办。

6. 在id=2：初中的触发器与公办初中点和民办初中的连线中分别添加两个触发器，并分别命名为schooltype=公办和schooltype=民办。

   触发器添加完成后，最终效果实现双重判断标准，部分截图如下所示。

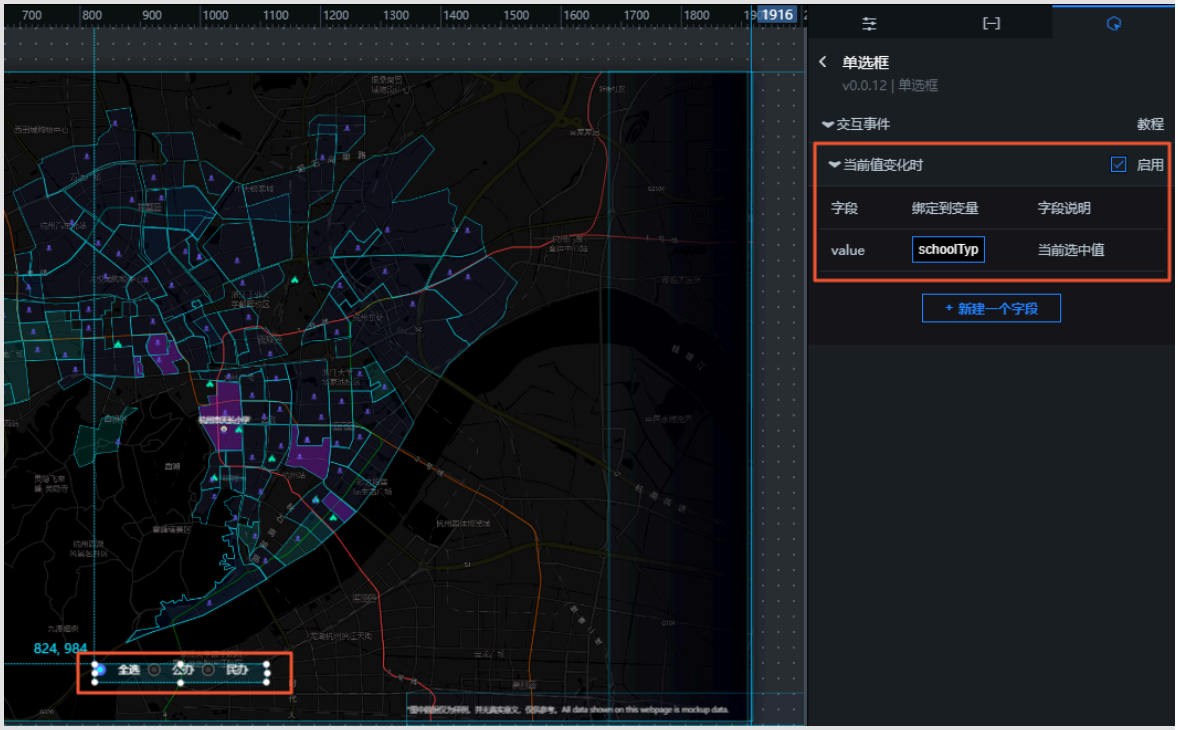


7. 使用同样的方式，按照以下说明，配置用于单选框判断的触发器。



注意:

在进行以下配置前，您需要首先在大屏编辑器内配置单选框组件的交互事件。



在单选框组件的交互面板中，勾选单选框变化响应事件右侧的启用，在value字段对应的绑定到变量输入框中输入schooltype。

- schooltype=公办触发器的触发条件为：

```
return getCallbackValue('schoolType') && (getCallbackValue('
schoolType') == "全选" || getCallbackValue('schoolType') == "公
办");
```

以上代码的作用是获取单选框中全局变量的值，判断当前单选框的状态。

- schooltype=民办触发器的触发条件为：

```
return getCallbackValue('schoolType') && (getCallbackValue('
schoolType') == "全选" || getCallbackValue('schoolType') == "民
办");
```

6.3.3 配置学区地图单选框交互

本文档为您介绍通过单选框实现不同类型学校的散点层数据的切换展示，以及Tab列表和单选框的双重触发判断方法。例如Tab列表选择小学，单选框选择公办，那么最终实现在大屏上展示所有公办小学的散点层信息。

前提条件

完成#unique_90操作，本文档将在其基础上继续添加节点和连线。

背景信息

需要实现的交互：通过控制组件单选框内的全选/公办/私办选项和Tab列表内的小学/初中选项，控制目标组件散点层的显隐效果。

可实现方案有两种：

- 使用触发器控制对应散点层的显隐样式，本文使用此方案，操作步骤如下。您也可以通过转换器来实现，详情请参见#unique_97。
- 切换散点层内的数据data/[]，比较复杂，不建议使用此方案。

操作步骤

1. 在节点编程配置页面，将左侧地图&固定内容下的单选框节点拖至画布中。
2. 按照以下说明进行连线。

   将单选框的当前值变化时事件分别与公办小学点和公办初中点的显示动作连线。
3. 在连线中添加两个触发器，分别命名为小学和初中。

   具体操作方法请参见#unique_98。

4. 按照以下说明继续添加其他连线和触发器。

a) 将小学触发器分别与民办小学的显示和隐藏动作连线。

b) 在每条连线中分别添加一个触发器，命名为全选、公办和民办。

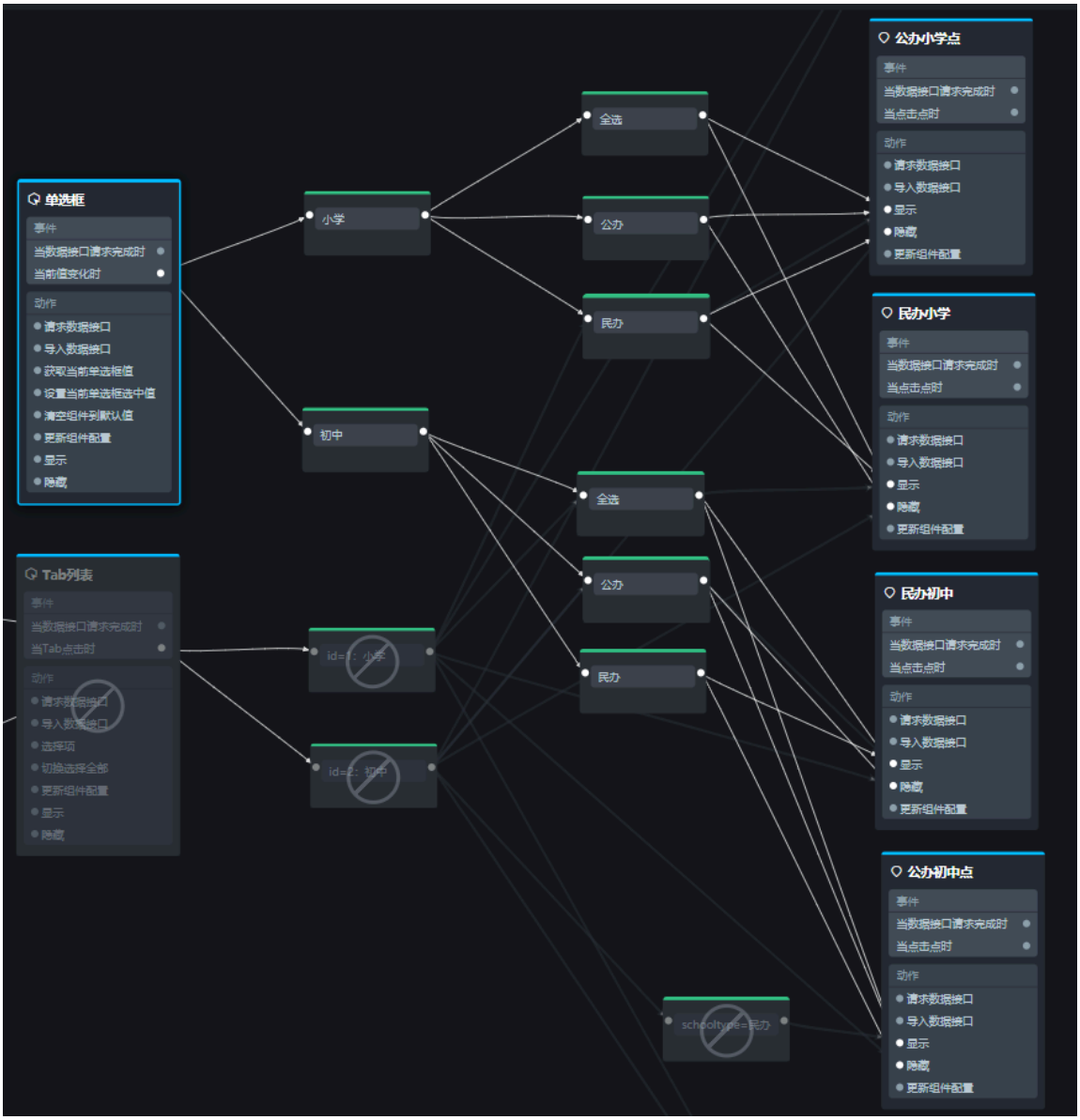
c) 按照以下说明进行连线。

最终的连线效果如下。

- 将全选触发器分别与公办小学点和民办小学的显示动作连线。
- 将公办触发器分别与公办小学点的显示动作和民办小学的隐藏动作连线。
- 将民办触发器分别与公办小学点的隐藏动作和民办小学的显示动作连线。

d) 使用同样的方式对初中触发器进行连线并添加触发器。

最终的连线效果如下。



连线的原理为：全选触发器同时连接公办和民办的显示动作；公办触发器连接公办的显示动作和民办的隐藏动作；民办触发器连接公办的隐藏动作和民办的显示动作。

5. 配置触发器。

a) 配置小学触发器。

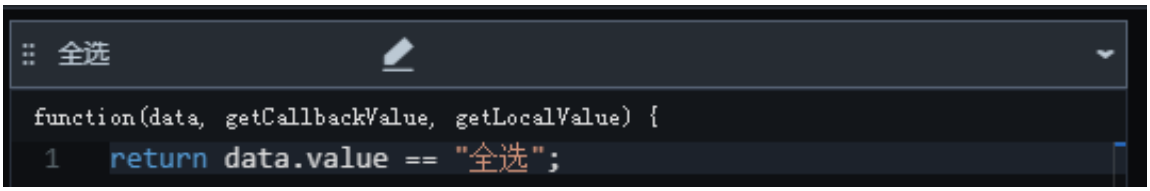

```
return getCallbackValue('tabID') && getCallbackValue('tabID') == "
1"; //初中触发情况反之为== "2"。
```

以上代码用于获取Tab列表中小学的触发判断条件。在当前小学触发条件下，小学相关的散点层组件都显示在大屏相应位置；初中相关的散点层组件不显示在大屏中。

b) 配置初中触发器。

```
return getCallbackValue('tabID') && getCallbackValue('tabID') == "
2";
```

c) 配置全选触发器。




```
return data.value == "全选"; //公办或民办的触发情况反之为== "公办"或"民
办"。
```

以上代码用于获取单选框组件中全选状态的触发判断条件。在当前全选触发条件下，全部散点层都显示在大屏相应位置，反之则显示部分散点层。

d) 配置公办触发器。

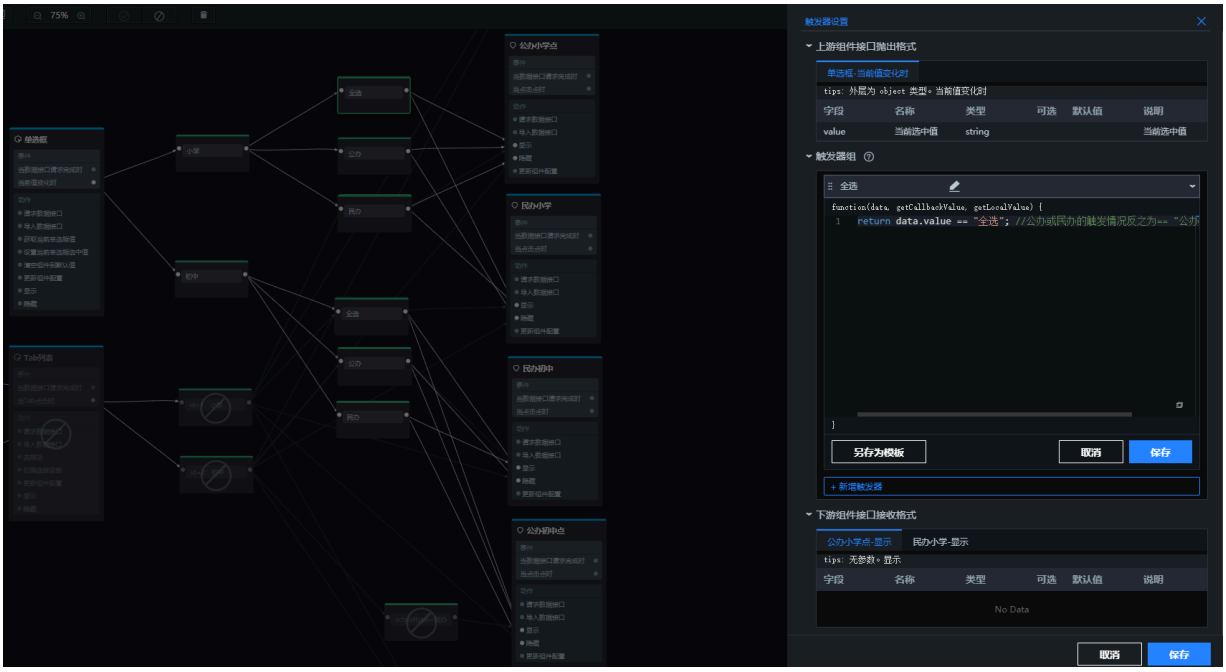
```
return data.value == "公办";
```

e) 配置民办触发器。

```
return data.value == "民办";
```

预期结果

所有连线和触发器都配置完成后，最终效果如下图所示。



以上配置实现了双重触发判断，即首先判断Tab列表中选择的是小学还是中学，在此基础上再判断单选框中选择的是民办、公办还是全选，最终组合筛选出对应的散点显示在大屏的地图区域中。

6.3.4 配置学区地图区域热力层交互

本文档为您介绍当鼠标划过地图的区域热力层子组件时，切换展示当前区域对应的学校数据的方法。

前提条件

完成配置学区地图单选框交互操作，本文档将在其基础上继续添加节点和连线。

背景信息

本文档包括以下两个场景的配置。

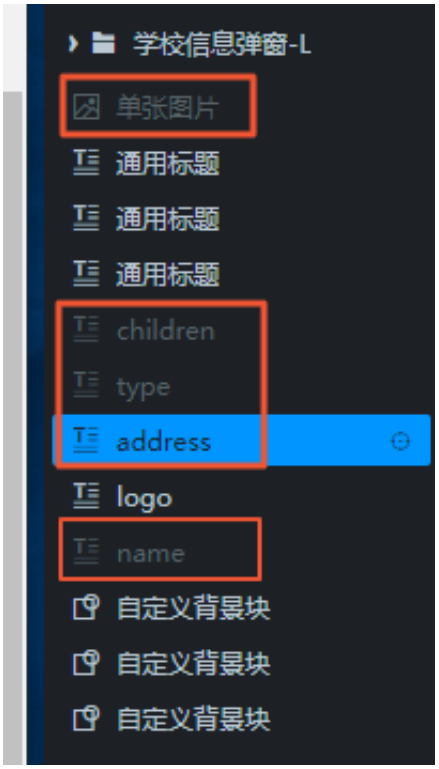
- 切换展示学校基本信息。
- 切换展示学校对应的学区房信息。

切换展示学校基本信息

需要实现的交互：鼠标滑过学校区域时，在大屏左下角切换展示学校的基本信息，包括学校的名称、地址和属性。

实现方案：划过地图的区域热力层子组件时，获取当前区域内对应学校的全量数据，将数据字段导入对应的组件中，并显示在大屏左下角。

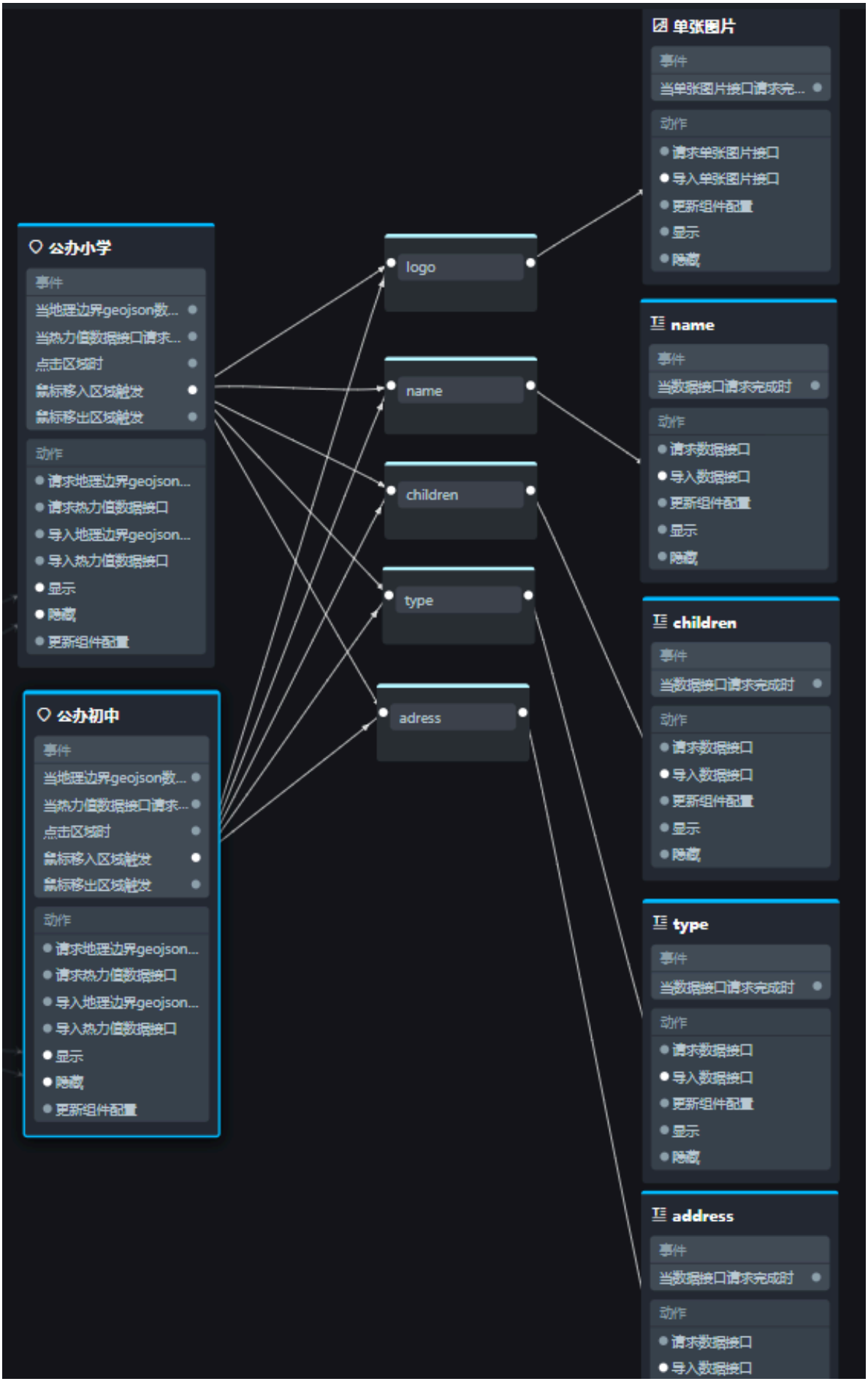
1. 在节点编程配置页面，将左侧学校信息弹窗-L下的单张图片、name、children、address和type这五个节点拖至画布中。



2. 将公办小学的鼠标移入区域触发事件分别与上一步中添加的五个节点的导入接口动作连线。
3. 在连线中各添加一个转换器，分别命名为logo、name、children、address和type。具体操作方法请参见#unique_98。

4. 将公办初中的鼠标移入区域触发事件分别与上一步中添加的五个转换器连线。

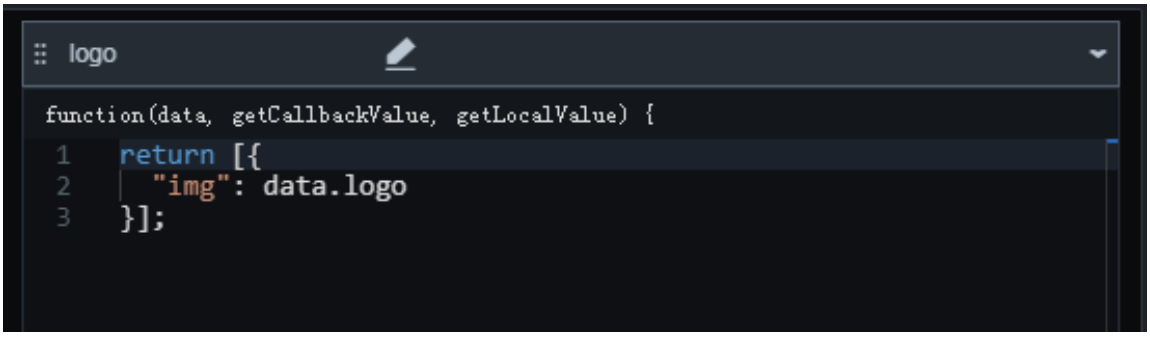
   连线完成后的结果如下图所示。


公办小学
事件
当地理边界geojson数...
当热力值数据接口请求...
点击区域时
鼠标移入区域触发
鼠标移出区域触发
动作
请求地理边界geojson...
请求热力值数据接口
导入地理边界geojson...
导入热力值数据接口
显示
隐藏
更新组件配置
公办初中
事件
当地理边界geojson数...
当热力值数据接口请求...
点击区域时
鼠标移入区域触发
鼠标移出区域触发
动作
请求地理边界geojson...
请求热力值数据接口
导入地理边界geojson...
导入热力值数据接口
显示
隐藏
更新组件配置
logo
name
children
type
adress
单张图片
事件
当单张图片接口请求完...
动作
请求单张图片接口
导入单张图片接口
更新组件配置
显示
隐藏
name
事件
当数据接口请求完成时
动作
请求数据接口
导入数据接口
更新组件配置
显示
隐藏
children
事件
当数据接口请求完成时
动作
请求数据接口
导入数据接口
更新组件配置
显示
隐藏
type
事件
当数据接口请求完成时
动作
请求数据接口
导入数据接口
更新组件配置
显示
隐藏
address
事件
当数据接口请求完成时
动作
请求数据接口
导入数据接口

5. 配置转换器。

a) 右键单击名称为logo的转换器，选择编辑。

b) 在转换器设置对话框的转换器脚本编辑区域，输入转换条件，完成后单击保存。




```
return [{
  "img": data.logo
}];
```

转换器从上游获取的data结构样式如下显示，转换器代码配置时候可按需选择。

```
{
  "id":xxx,
  "name":"xxx小学",
  "distance":"",
  "public_or_private":"",
  "is_primary":1,
  "is_middle":0,
  "is_nursery":0,
  "is_high":0,
  "location":"",
  "address":"",
  "logo":""
}
```

c) 使用同样的方式，配置其他四个转换器，示例代码如下。

- name转换器：

```
return [{
  "value": data.name
}];
```

- children转换器：

```
let res = `${data.is_primary && data.is_primary == 1 ? "小
学" : ""}${data.is_middle && data.is_middle == 1 ? "/初中" :
 ""}${data.is_high && data.is_high == 1 ? "/高中" : ""}${data.
is_nursery && data.is_nursery == 1 ? "/幼儿园" : ""}`;
return [{
  "value": `${res[0] == "/" ? res.substring(1) : res}`
}];
```

- type转换器：

```
return [{
  "value": `学校属性：${data.public_or_private}`
```

```
}];
```

- address转换器：

```
return [{
  "value": `学校地址：${data.address}`
}];
```

切换展示学校对应的学区房信息

需要实现的交互：鼠标滑过学校区域时，在大屏右上角切换展示对应学区房成交量排行榜信息和房价趋势图。

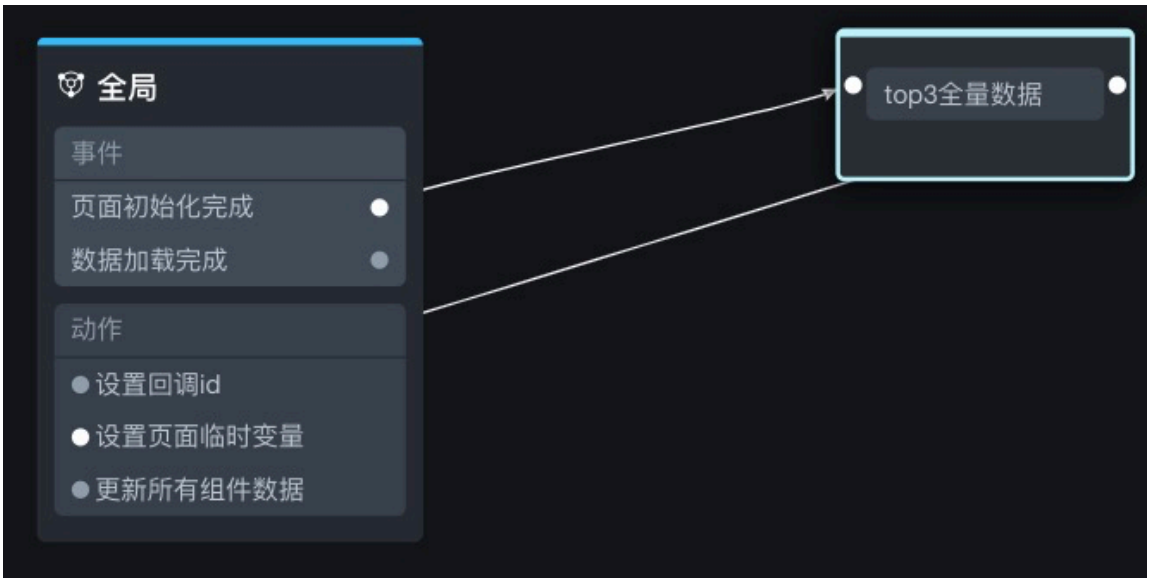
实现方案：划过地图的区域热力层子组件时，获取当前区域内对应学校的id，从临时变量（全量学区数据）中过滤出对应数据并分发给组件，显示在大屏右上角。

1. 配置全局节点的临时变量。

   a) 在节点编程配置页面，将左侧的全局节点拖至画布中。

   b) 将全局节点的页面初始化完成事件和设置页面临时变量动作进行连线。

   c) 在连线中添加一个转换器，并命名为top3全量数据。



   d) 配置转换器，设置临时变量的数据结构。

   - 一号标记：临时变量名。
   - 二号标记：学校对应的ID。
   - 三号标记：学区信息。

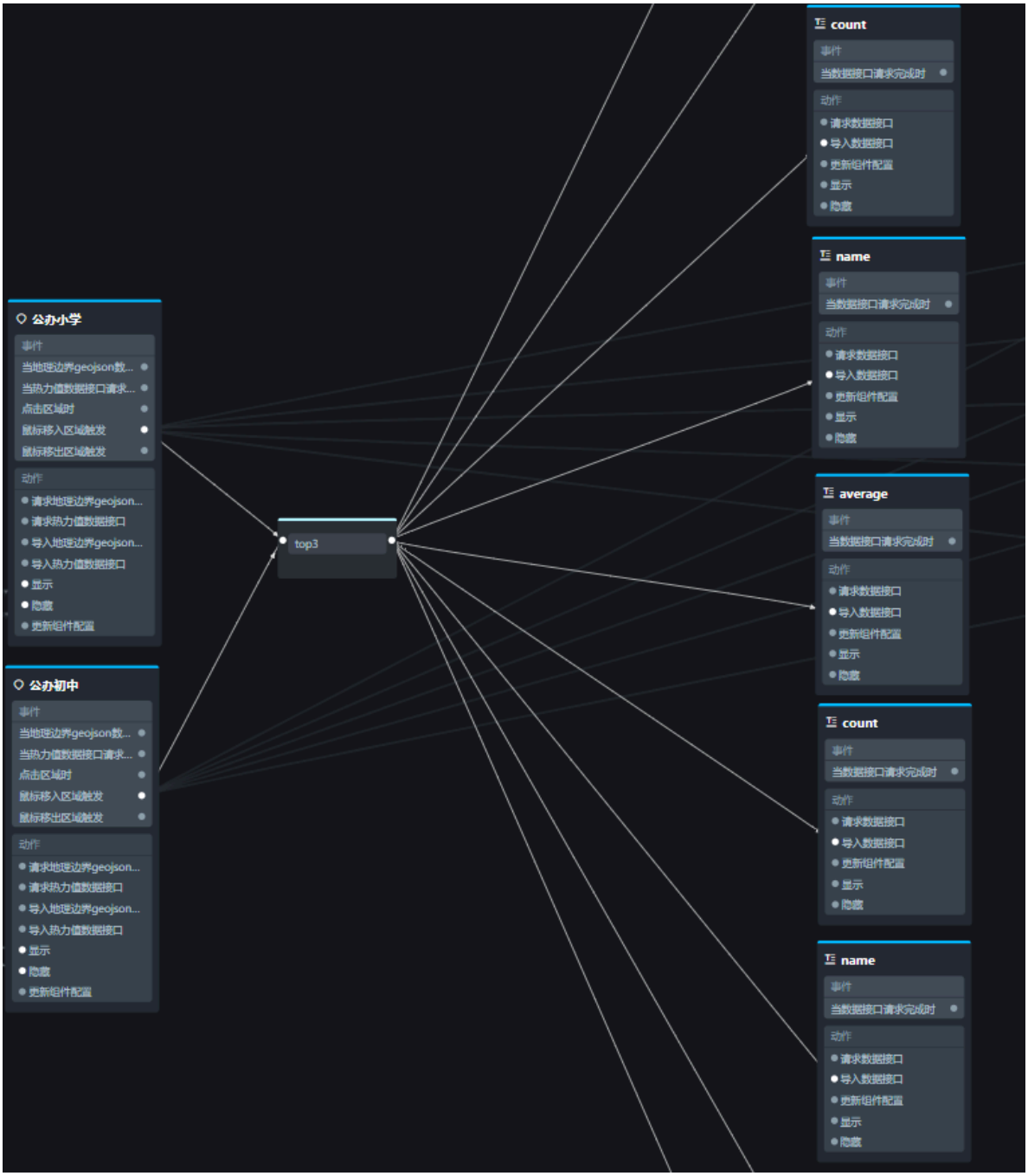
可单击此处下载上图中的示例代码。

2. 在节点编程配置页面，将左侧成交量 > NO.1、成交量 > NO.2和成交量 > NO.3文件夹下的name、average和count共九个节点都拖至画布中。
3. 将公办小学或公办初中与上一步中拖入的任意一个节点连线。
4. 在连线中添加转换器，并命名为top3。

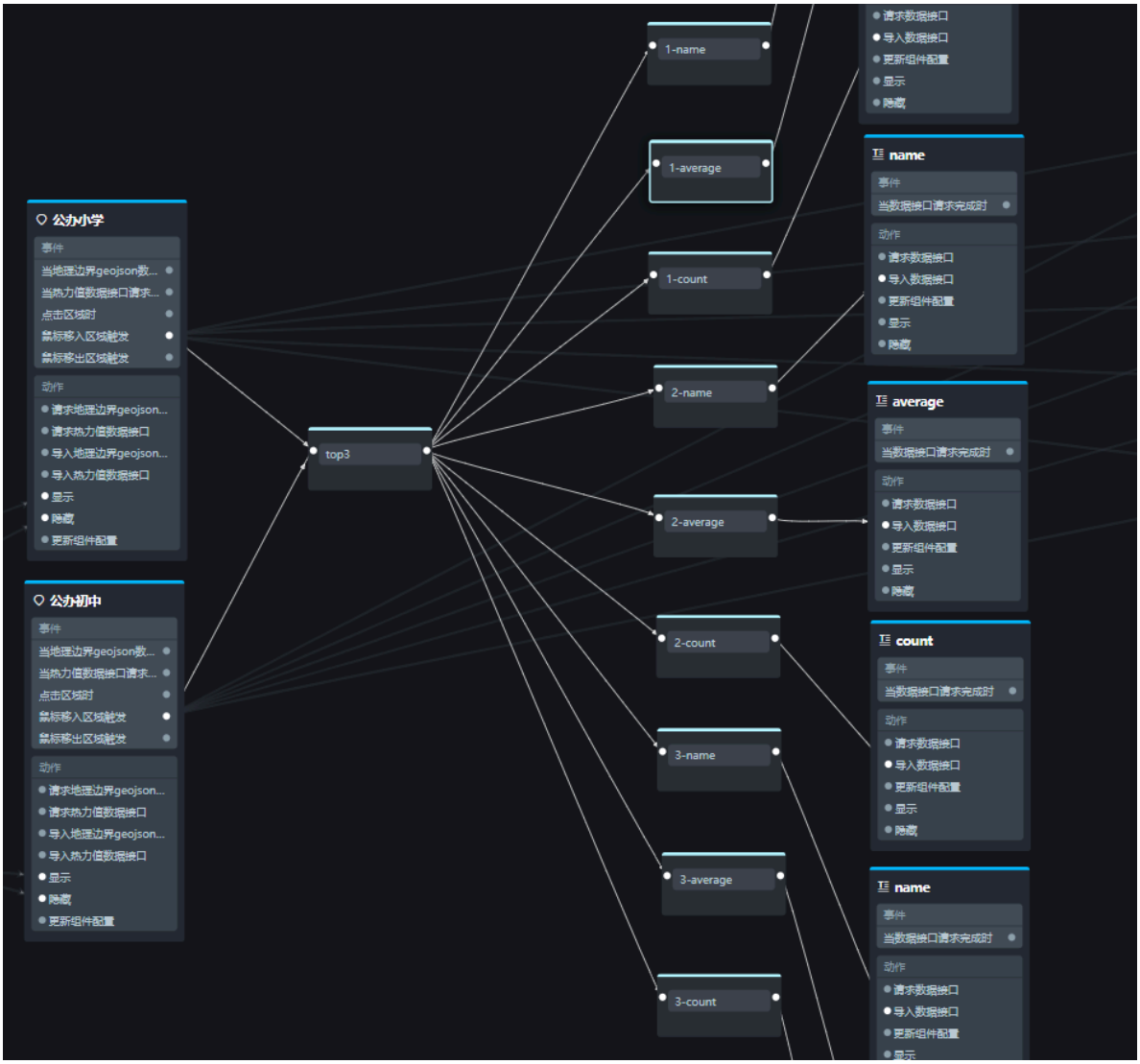
5. 按照以下说明完成top3转换器的连线。

   - 将top3转换器分别与公办小学和公办初中的鼠标移入区域触发事件连接。
   - 将top3转换器分别与第二步中所有的name、average和count节点的导入数据接口动作连线。

   部分连线结果如下。

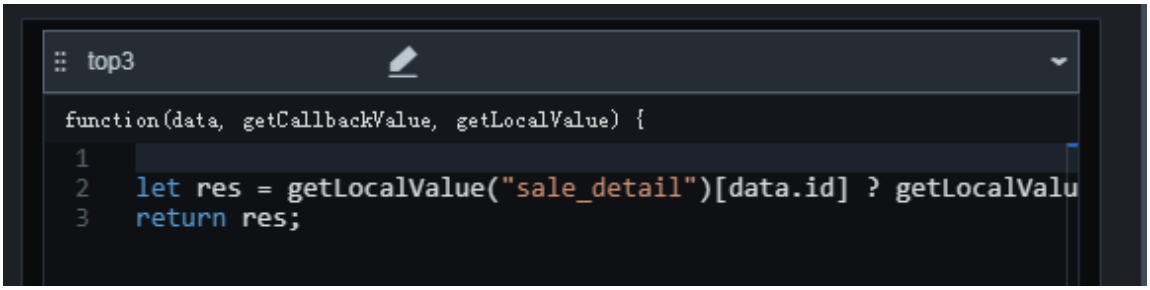


6. 在上一步的九条连线中各添加一个转换器，分别命名为1-name、1-count、1-average、2-name、2-count、2-average、3-name、3-count和3-average。
   连线完成后，结果如下图所示。



7. 配置转换器。

   · 配置top3转换器。




```
let res = getLocalValue("sale_detail")[data.id] ? getLocalValue("
sale_detail")[data.id].list : [];
```

```
return res;
```

- 配置1-name转换器。

```
return data.length >= 1 ? [
  {
    value: `${data[0].district}区 ${data[0].name}`
  }
] : [{ value: "" }];
```

- 配置1-count转换器。

```
return data.length >= 1 ? [
  {
    value: `成交量: ${data[0].count}`
  }
] : [{ value: "" }];
```

- 配置1-average转换器。

```
return data.length >= 1 ? [
  {
    value: `平均单价: ${(data[0].sumPrice / data[0].area).toFixed(2
)}万`
  }
] : [{ value: "" }];
```

- 配置2-name转换器。

```
return data.length >= 2 ? [
  {
    value: `${data[1].district}区 ${data[1].name}`
  }
] : [{ value: "" }];
```

- 配置2-count转换器。

```
return data.length >= 2 ? [
  {
    value: `成交量: ${data[1].count}`
  }
] : [{ value: "" }];
```

- 配置2-average转换器。

```
return data.length >= 2 ? [
  {
    value: `平均单价: ${(data[1].sumPrice / data[1].area).toFixed(2
)}万`
  }
] : [{ value: "" }];
```

- 配置3-name转换器。

```
return data.length >= 3 ? [
  {
    value: `${data[2].district}区 ${data[2].name}`
  }
```

```
] : [{ value: "" }];
```

- 配置3-count转换器。

```
return data.length >= 3 ? [
  {
    value: `成交量: ${data[2].count}`
  }
] : [{ value: "" }];
```

- 配置3-average转换器。

```
return data.length >= 3 ? [
  {
    value: `平均单价: ${(data[2].sumPrice / data[2].area).toFixed(2
)}万`
  }
] : [{ value: "" }];
```

说明:

此步骤使用到了节点编程的数据分发功能，详情请参见#unique_99。

6.3.5 配置学区地图轮播列表交互

本文档为您介绍当单击轮播列表组件时，切换展示对应学校在地图上的位置信息和数据的方法。

背景信息

需要实现的交互：当单击用来展示学校排名的轮播列表组件内的某一行时，在地图组件上定位该学校位置，并通过散点层子组件返回该学校位置上的数据。

实现方案：轮播列表组件的数据中包含学校的位置数据。当单击轮播列表组件的某一行时，获取该行的位置数据，基础平面地图父组件对应更新数据修改经纬度，散点层返回对应的点位数据。

操作步骤

1. 在节点编程配置页面，将左侧小学文件夹下的小学学校列表、初中下的初中学校列表、地图&固定内容下的基础平面地图和地图&固定内容 > 基础平面地图下的散点层四个组件节点拖至画布中。
2. 按照以下说明连线。
   - 将初中学校列表的当点击单行时事件与散点层的导入数据接口连线。
   - 将小学学校列表的当点击单行时事件与基础平面地图的导入数据接口连线。
3. 在上一步的两条连线中各添加一个转换器，并分别命名为提取散点数据和地图父组件定位缩放。具体操作方法请参见#unique_98。

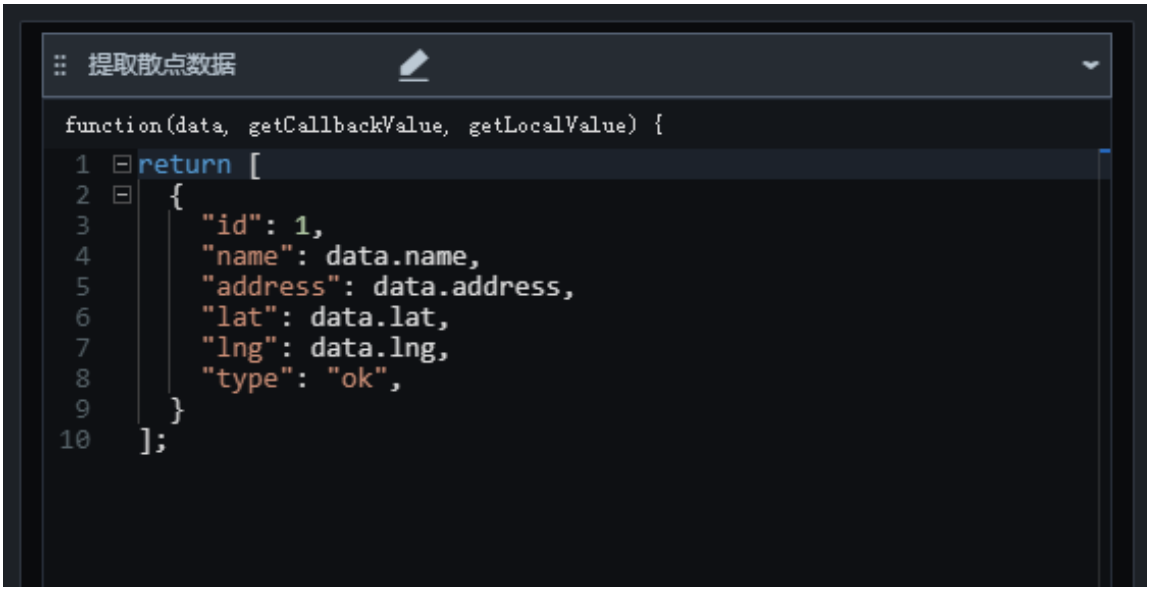
4. 将初中学校列表和小学学校列表的当点击单行时事件分别与提取散点数据和地图父组件定位缩放转换器连线。

   连线完成后，效果如下图所示。

5. 配置转换器。

a) 右键单击名称为提取散点数据的转换器，选择编辑。

b) 在转换器设置对话框的转换器脚本编辑区域输入转换条件，完成后单击保存。



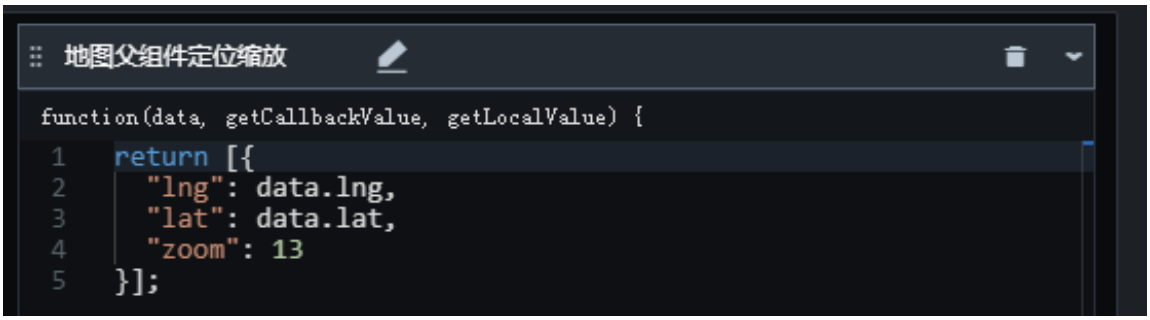
转换器的示例代码如下。

```
return [
  {
    "id": 1,
    "name": data.name,
    "address": data.address,
    "lat": data.lat,
    "lng": data.lng,
    "type": "ok",
  }
];
```

说明:

由于轮播列表组件中的数据中包含了所有的地理信息，所以当前转换器需要提取的数据都可以直接从上游的轮播列表组件抛出的数据中获取。

c) 使用同样的方式配置地图父组件定位缩放转换器。



转换器的示例代码如下。

```
return [{
```

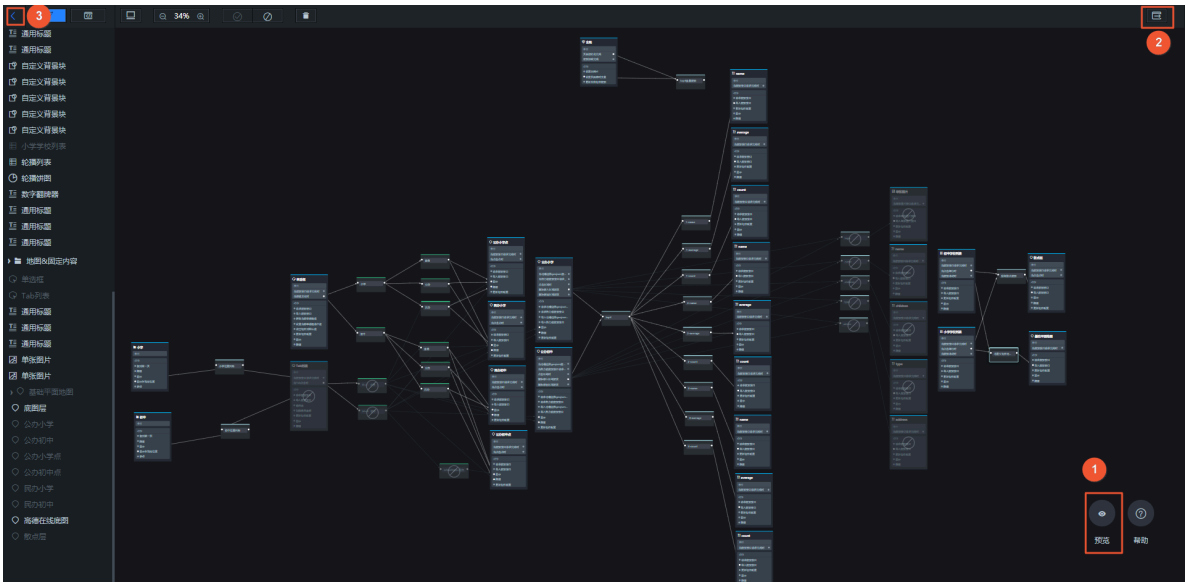
```
  "lng": data.lng,
  "lat": data.lat,
  "zoom": 13
}];
```

6.4 查看大屏效果

本文档为您介绍通过节点编程功能配置完所有节点、连线和触发器/转换器后，预览演示效果，并将交互应用到大屏编辑器页面，最终实现在线演示交互式学区房大屏的方法。

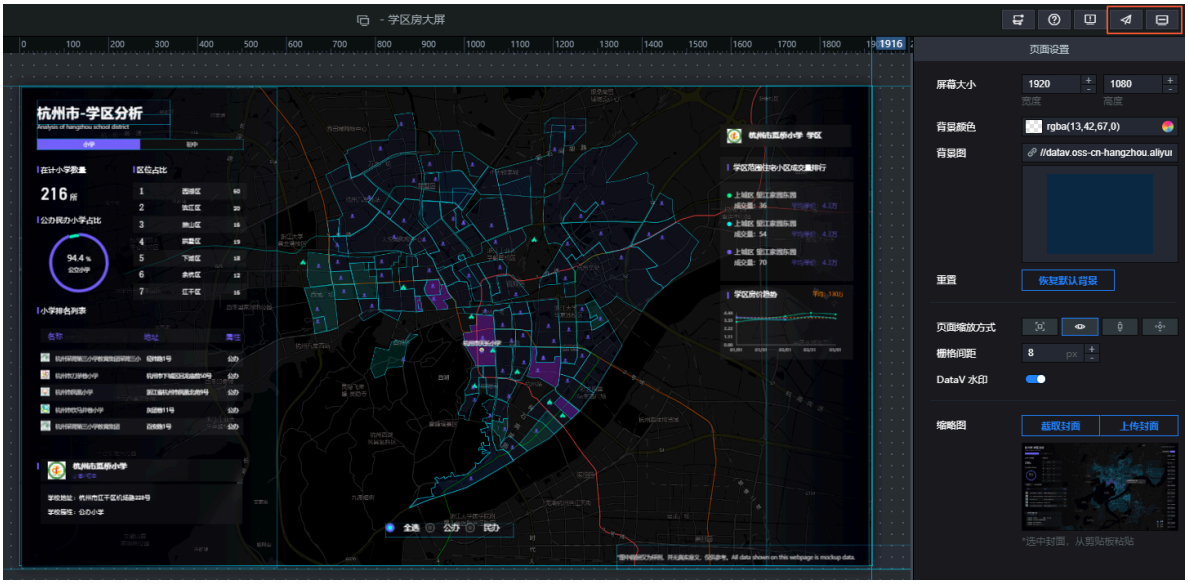
操作步骤

1. 单击节点编程配置页面右下角的预览图标。
2. 在效果预览页面，测试大屏的交互效果。预览无误后，退出预览页继续执行以下步骤。
3. 单击页面右上角的应用图标，将节点编程内的效果应用到大屏。
4. 单击页面左上角的返回图标。



5. 在大屏编辑器页面，单击右上角的预览图标，查看学区房大屏的展示效果。

6. 单击页面右上角的发布图标，将学区房大屏发布到线上环境，实现在线大屏演示。



预期结果

最终效果如下图所示。