

业务实时监控服务 ARMS

常见问题

常见问题

日志清洗常见问题

智能切分日期格式

智能切分会自动识别特定格式的日期字段，本文介绍了具体定义智能切分适配的日期格式。

时间必须有精确的年月日时分秒字段。可根据以下说明格式可选添加毫秒字段、可选添加时区字段。

格式规范说明参考 SimpleDateFormat。

常见日期格式

常见日期格式为日志中常见的时间格式，通常为由规律分隔符隔开的年月日时分秒数据，年月日与时分秒中可用分隔符或大写字母“T”隔开。秒字段后可用分隔符“,”或“.”后接毫秒字段。最后可加上标准时区字段。

yyyy-MM-ddTHH:mm:ss

其中“T”可替换为空格或其他分隔符，“:”可替换为其他分隔符，年、月、日之间和时、分、秒之间的分隔符可替换为“/”、“-”等常用分隔符，例如：

```
1963-04-17T23:51:12
1963/04/17 12:12:53
```

yyyy-MM-dd HH:mm:ss,SSS

其中“,”可替换为“.”，其他位置可替换字符同上，例如：

```
1963/04/17 12:12:53,172
1963/04/17T12:12:53.172
```

yyyy-MM-dd HH:mm:ssZ

例如：

```
1963/04/17 12:12:53+0800
1963/04/17T12:12:53-0700
```

yyyy-MM-dd HH:mm:ss,SSS

例如：

```
1963/04/17 12:12:53,999+0800
1963/04/17T12:12:53.878-0700
```

固定标准时间格式

固定时间表示格式，SimpleDateFormat 中能翻译成日期正则式的较常用日期格式，例如年月日中没有分隔符的时间格式，以及月日年形式的常用时间格式等。

类型一：日期和时间之中均无分隔符

yyyyMMddHHmmss

```
19551020233546
```

yyyyMMdd HHmmss

```
19551020 233546
```

类型二：日期格式倒序，日在前、月在中、年在后

dd-MM-yyyy HH:mm:ss

```
17-04-1963 23:59:23
```

SimpleDateFormat 中部分常见日期格式

SimpleDateFormat 文档中出现的部分常见日期格式。详见SimpleDateFormat官方文档说明。时间字段必须

精确到年月日时分秒的日期。

yyyy.MM.dd G 'at' HH:mm:ss z

2001.07.04 AD at 12:08:56 PDT

EEE, MMM d, '' yy

Wed, Jul 4, '01

yyyyy.MMMMM.dd GGG hh:mm aaa

02001.July.04 AD 12:08 PM

EEE, d MMM yyyy HH:mm:ss Z

Wed, 4 Jul 2001 12:08:56 -0700

yyMMddHHmmssZ

010704120856-0700

yyyy-MM-dd' T' HH:mm:ss.SSSZ

2001-07-04T12:08:56.235-0700

yyyy-MM-dd' T' HH:mm:ss.SSSXXX

2001-07-04T12:08:56.235-07:00

其他特定日期格式

- 如 Nginx 日期格式

25/Aug/2012:16:41:07 +0800

关于自定义清洗的常见问题

本文介绍了关于自定义清洗的一些常见问题。

特殊字符（不可见字符）作为分隔符如何切分？

假如不可见字符是 ascii 码的 1，则对应的分隔符为\u0001。如果是 2，则为\u0002.....依此类推。
若要用 tab 作为分隔符，请使用\t。

特殊字符在流程配置页面上如何测试？

因为无法在浏览器里粘贴不可见字符，因此您只能暂时将测试文本的分隔符替换为可见字符进行测试。
全部流程测试通过后，再将分隔符替换为不可见字符进行部署。

下图中 Key 的列表中如何添加更多的 Key？



请单击积木块上的蓝底白色五角星。

JSON 格式的日志可以切分吗？

可以，不过没有专属 JSON 切分器来支持，需要配合使用逻辑表达式 + KV 切分器来实现。详情请参考自定义切分的使用中的 KV 切分器说明。

支持跨行日志（如异常堆栈）吗？

暂不支持。

自定义切分的使用

除了智能切分器，ARMS 还提供了自定义清洗配置器，用于自定义日志清洗流程来满足复杂的切分需求。

自定义清洗配置器采用所见即所得的可视化配置方式。本文介绍了如何使用自定义清洗配置器。

配置界面简介

自定义清洗配置界面包含三个部分：

- 原始日志区域
- 可视化清洗流程编辑区域
- 清洗结果预览区域



在可视化流程编辑区域（以下简称“编辑区”），可以使用一个或多个“积木块”组装出日志清洗的逻辑，将原始日志区域中的日志“清洗”为结构化的键值对（Key-Value Pair）。

在编辑区编辑的过程中，可以随时单击清洗结果预览区域上方的**日志切分预览**按钮，预览每一行原始日志通过当前编辑区的流程清洗后的键值对结果。

预览结果并确认清洗流程准确无误后，单击页面下方的**下一步**按钮启动任务。ARMS 将使用在编辑区保存的流程来处理每一条从数据源中消费的日志。

示例一

本示例演示如何配置一个简单的日志清洗流程。

假如某系统的交易日志如下：

```
2016-11-08 11:00:01|user_abc|123456|下单
2016-11-08 11:00:02|user_abc|123456|支付
2016-11-08 11:10:01|user_abc|123456|退货
```

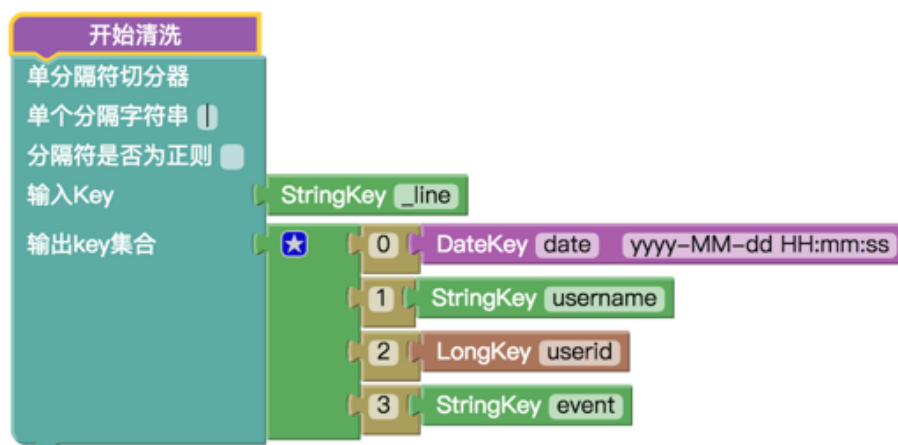
以第一行数据为例，假设目标切分结果为如下几个键值对：

Key 类型	Key	Value
Date	date	2016-11-08 11:00:01
String	username	user_abc
Long	userid	123456
String	event	下单

通过观察发现，键值对之间均使用竖线（|）进行分隔，因此可以使用**单分隔符切分器**来切分键值对。

在编辑区左侧的工具栏中，将**单分隔符切分器**拖拽至编辑区，如下图所示。

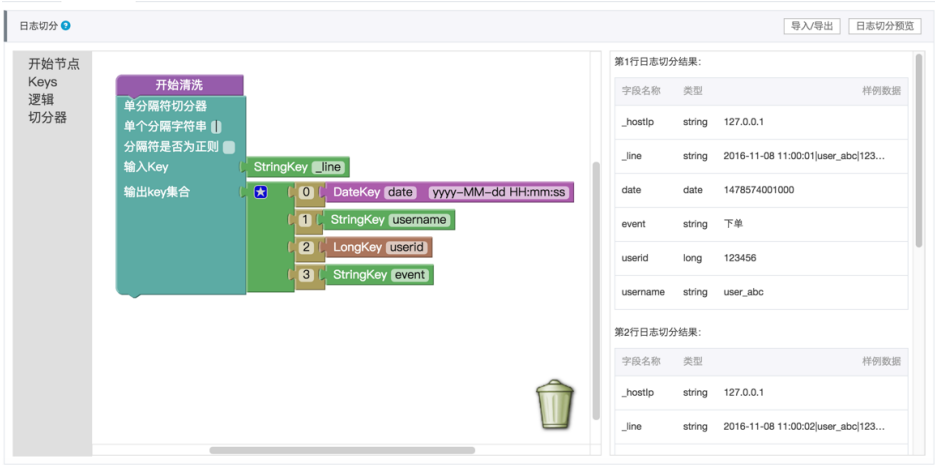
注意：切分器必须连接在**开始清洗**模块下才能生效。



配置切分说明：

- 以 “|” 为分隔符，“_line” 作为入参（_line 代表原始日志行本身）。
- 切分后，0 号位置的字符串按照 yyyy-MM-dd HH:mm:ss 的格式转为 date（内部实际将以 Long 型保存）。
- 1 号位置的字符串转为 key 为 username 的 String。
- 2 号位置转为 key 为 userid 的 Long。
- 3 号位置保存为 key 为 event 的 String。

点击**日志切分预览**，可以观察到每行日志的切分结果如下：



最终生成的切分模型如下所示：

切分模型预览：		
字段名称	类型	样例数据
username	string	user_abc
_hostIp	string	127.0.0.1
event	string	下单
_line	string	2016-11-08 11:00:01 user_abc 123456 下单
date	date	1478574001000
userid	long	123456

单击**保存**或**下一步**，清洗流程生效。

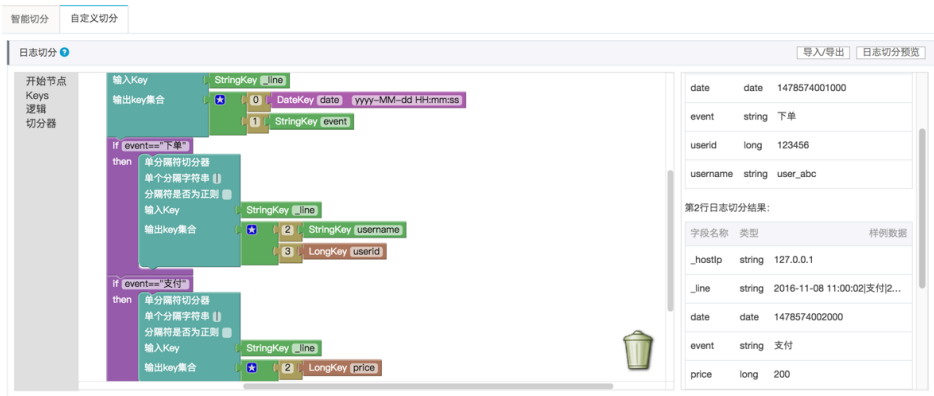
到此为止，一个简单日志切分就完成了。

示例二

本示例演示如何对复杂的日志进行切分清洗。

```
2016-11-08 11:00:01|下单|user_abc|123456
2016-11-08 11:00:02|支付|200
2016-11-08 11:10:01|退货
```

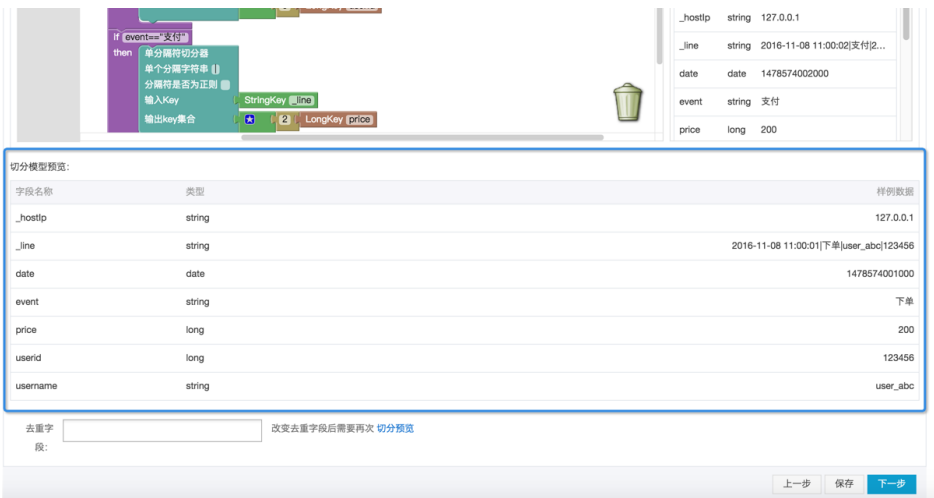
在此样例中，不同的日志由于操作类型不一样，字段的个数、同位置各字段的含义也不同，这在业务系统中十分常见。此时可以通过 if-then/if-else（其语法参考 [aviator](#)）逻辑来区分不同类型的日志。



配置切分说明：

- 首先使用单分隔符切分器切分出date与event字段。
- 接着使用逻辑模块中的 if-then 模块，在 event 为不同值时，分别使用不同的单分隔符切分器配置切分出不同的字段。
- 从右侧的预览结果中也可以看到，第一行与第二行切分出的键值对组合是不一样的。第一行中有 username 与 userid，而第二行中有 price。

注意：在自定义切分最下方的**最终切分模型预览**，将包含上图中右侧单行切分预览区域中**所有字段的并集**。



以上两个例子仅使用了一种切分器（单分隔符切分器）。ARMS 还内置了单分隔符、多分隔符、顺序、K-V、JSON 等多种切分器，针对不同的场景可以单独或组合使用，详情请参考内置切分器。

关于自定义切分的其他问题

自定义清洗中的系统字段请参考日志清洗中系统字段介绍。

自定义切分遇到的问题请参考自定义清洗常见的问题。

If-then/if-else 的语法请参考 [aviator](#)。

内置切分器

本文介绍了 ARMS 的内置切分器。

ARMS 内置单分隔符、多分隔符、顺序、K-V、JSON 等多种切分器，针对不同的场景可以单独或组合使用。

单分隔符切分器

说明

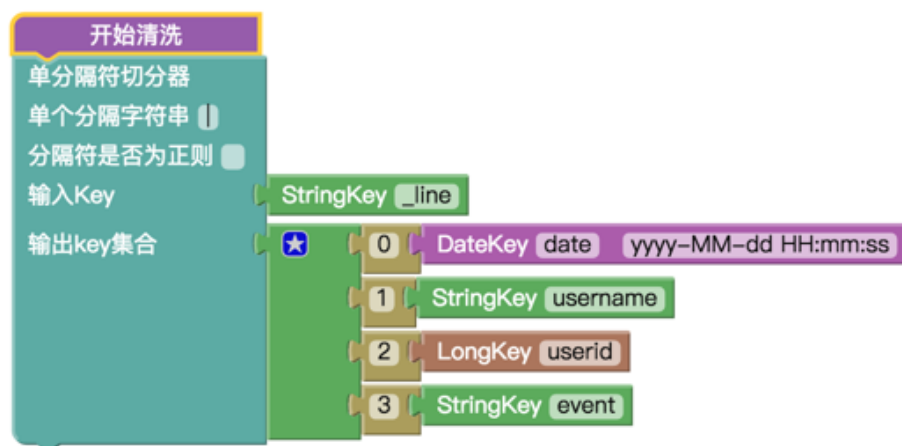
将单一字符串作为输入，当匹配用户指定分隔字符串时进行切分，切分结果为多个键值对。

实例

样例日志如下所示：

```
2016-11-08 11:00:01|user_abc|123456|下单
2016-11-08 11:00:02|user_abc|123456|支付
2016-11-08 11:10:01|user_abc|123456|退货
```

其自定义切分如下所示：



配置说明：

- 以 "|" 为分隔符

- 切分后 0 号位置的字符串按照 yyyy-MM-dd HH:mm:ss 的格式转为 date (内部实际将以 Long 型保存)
- 1 号位置的字符串转为 key 为 username 的 String
- 2 号位置转为 key 为 userid 的 Long
- 3 号位置保存为 key 为 event 的 String

注意：

对不可见字符的支持：本切分器支持不可见字符的切分，例如 \t、\u001f (ascii char=31)。一些不可见字符的切分将无法在页面测试中被正确验证（因为页面文本框内无法正确输入不可见字符），所以建议在测试页面先使用可见字符进行调试，然后换成不可见字符保存后进行部署。

多字符分隔符：本切分器分隔符可以为多字符，如a::b::c为输入，以::为分隔符切分。

切分模型包含 _line 字段与 _hostIp 字段，具体请参考日志清洗中系统字段介绍。

多分隔符切分器

说明

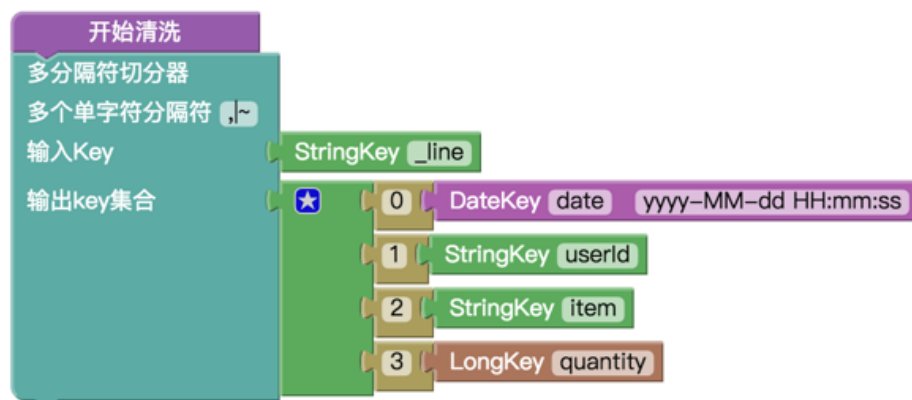
将单一字符串作为输入，当匹配任意一个用户指定分隔字符时进行切分，切分结果为多个键值对。

实例

样例日志：

```
2014-07-25 17:25:00,aaa|b~1
2014-07-25 17:26:00,aaa|b~1
```

自定义切分如下所示：



每行日志的切分如下所示：

日志切分

开始节点
Keys
逻辑
切分器

开始清洗

多分隔符切分器
多个单字符分隔符

输入Key

输出key集合

StringKey _line

0 DateKey date yyyy-MM-dd HH:mm:ss

1 StringKey userid

2 StringKey item

3 LongKey quantity

导入/导出

日志切分预览

第1行日志切分结果：

字段名称	类型	样例数据
_hostip	string	127.0.0.1
_line	string	2014-07-25 17:25:00,aaa b~1
date	date	1406280300000
item	string	b
quantity	long	1
userid	string	aaa

第2行日志切分结果：

字段名称	类型	样例数据
_hostip	string	127.0.0.1
line	string	2014-07-25 17:26:00,aaa b~1

切分模型如下所示：

切分模型预览：		
字段名称	类型	样例数据
_hostip	string	127.0.0.1
item	string	b
userid	string	aaa
_line	string	2014-07-25 17:25:00,aaa b~1
date	date	1406280300000
quantity	long	1

顺序分隔符切分器

说明

针对较复杂的切分场景（如 accesslog），ARMS 提供了**顺序分隔符切分器**，以 O(n) 的复杂度来切分日志。

实例

样例日志：

```
117.74.77.48 835158 - [26 May 2014:14:05:28 +0800] "GET
http://trade.taobao.com/trade/detail/trade_snap.htm?spm=a1z0f.2.100003.9.8BpicQ&trade_id=664793864233811"
302 0
117.74.87.58 835158 - [26 May 2014:14:05:28 +0800] "GET
http://trade.taobao.com/trade/detail/trade_snap.htm?spm=a1z0f.2.100003.9.8BpicQ&trade_id=664793864233811"
302 0
```

我们希望切出 IP 字段、平均响应时间、时间、请求的 URL：

```
=> [ url -
http://trade.taobao.com/trade/detail/trade_snap.htm?spm=a1z0f.2.100003.9.8BpicQ&trade_id=66479
3864233811 ] [ _hostIp - 127.0.0.1 ] [ ip - 117.74.77.48 ] [ date - 1401084328000 ] [ rt - 835158 ]
```

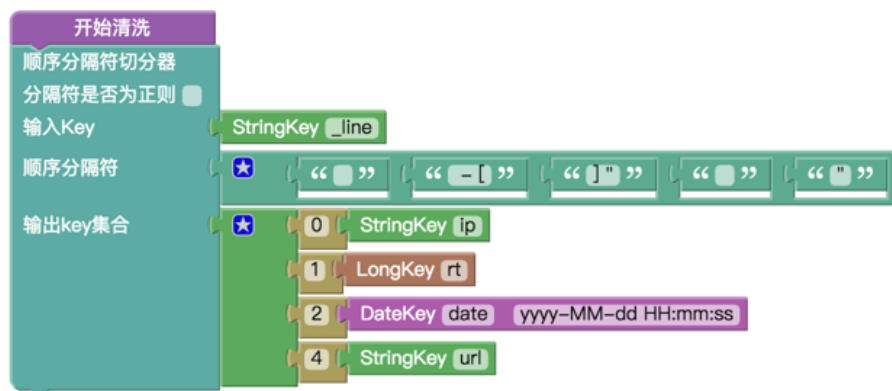
我们会发现：

用单分隔符切分器显然无法满足需求。

用多分隔符切分器也不能满足需求：我们如果想切出rt=835158这组 KV（键值对），必须以“ ”（空格）作为切分字符之一，但是这样做会错误地把26 May 2014:14:05:28 +0800这个字符串切成几段，因此我们将无法得到date - 1401084328000。

如果使用正则表达式来匹配 KV，效率太低，需要遍历多次。

ARMS 提供了“顺序分隔符切分”的方式。使用如下配置，就可以得到本示例正确的结果。具体配置方式如下所示：



先定义 5 个分隔符字符串，并依次将输入字符串切分为 6 段子串，随后将指定的子串分别赋给不同的 Key，即可达成目标切分效果。

KV 切分器

如果需要切分如下所示的多组 Key=Value 形式的日志：

```
key1=aaaa;key2=bbbb;key3=cccc;key4=dddd;....  
key1=aaaa;key2=bbbb;key3=cccc;key4=dddd;....
```

且各个 Key-Value 的位置不固定时，可以使用 KV 切分器：

切分配置如下：



注意：

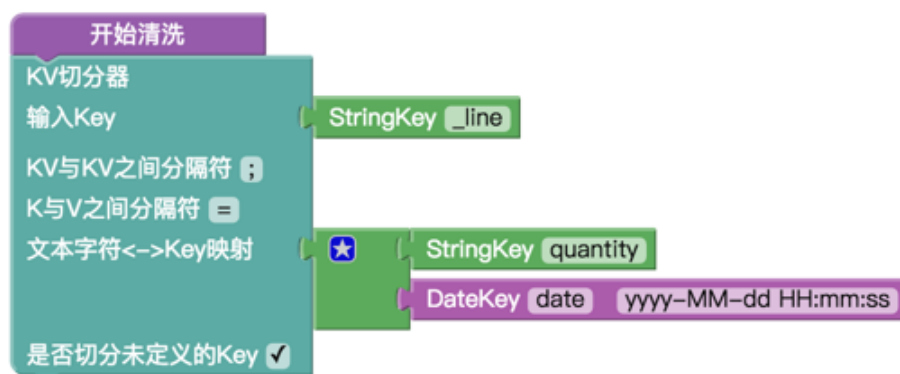
在没有配置文本字符 <-> Key 映射时，KV 切分器将视所有的 Key 为“未定义key”，所有 Key 对应的 Value 将被切为 StringKey 类型。

如果取消勾选**是否切分未定义的 Key**，那么未定义 Key 将不会被切分。换言之，如果没有配置文本字符 <-> Key 映射，并且未勾选**是否切分未定义的 Key**，本切分器将不执行任何操作。

接下来展示一个需要将指定的 Key 转换成对应类型的例子。假设有日志如下：

```
name=abc;item=iphone6;quantity=15;date=2014-07-25 17:25:00;....
name=abc;item=iphone6;quantity=15;date=2014-07-25 17:25:00;....
```

需要将 quantity 变为 LongKey，date 变为 DateKey：



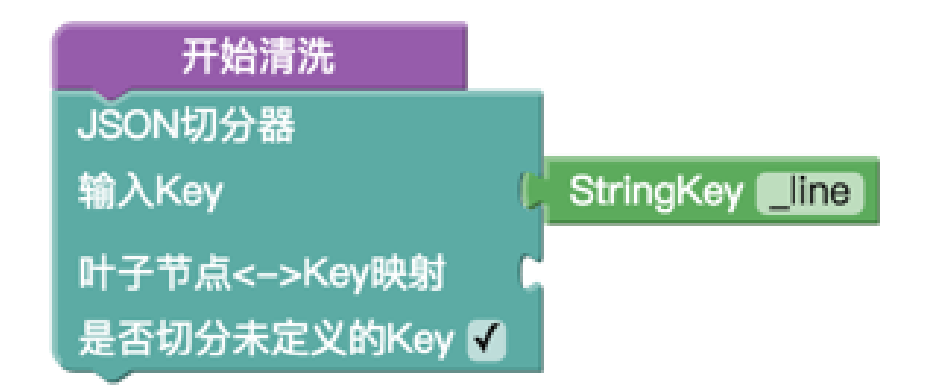
JSON 切分器

如果日志中包含 JSON 字符串，使用 JSON 切分器可以快速切分出该 JSON 中的“叶子节点”。

例一：全自动切分

```
{
  "title": "Example Schema",
  "type": "object",
  "quantity": 200,
  "date": "2015-12-12 12:12:12",
  "properties": {
    "firstName": {
      "type": "string"
    },
    "lastName": {
      "type": "string"
    },
    "age": {
      "description": "Age in years",
      "type": "integer",
      "minimum": 0
    }
  },
  "required": ["firstName", "lastName"]
}
```

将该 JSON 放在一行文本中，使用 JSON 切分器进行切分：



切分结果如下：

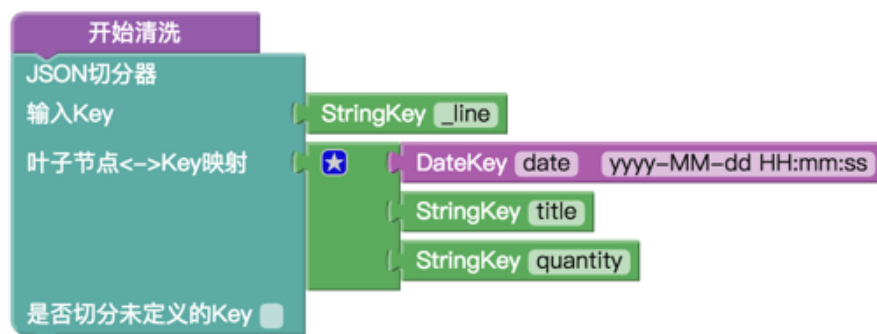
切分模型预览：		
字段名称	类型	样例数据
description	string	Age in years
_hostIp	string	127.0.0.1
minimum	long	0
quantity	long	200
type	string	string
date	string	2015-12-12 12:12:12
_line	string	{"title": "ExampleSchema", "type": "object", "quantity": 200, "date": "2015-12-12 12:12", "properties": {"firstName": {"type": "string"}, "lastName": {"ty...
title	string	ExampleSchema

由于勾选了**是否切分未定义的 Key**，JSON 切分器将自动切出所有的“非 Array 类型的叶子节点”，并自动转成对应的类型（StringKey 或 LongKey）。

注意：本例中的required字段由于是 Array 类型，所以没有切分。另外，date这个字段默认转化为 String 类型。

例二：自定义切分

如果只想切分 JSON 中指定的某些字段，可以这样配置：



其切分模型如下：

切分模型预览：		
字段名称	类型	样例数据
_hostIp	string	127.0.0.1
_line	string	{ "title": "ExampleSchema", "type": "object", "quantity": 200, "date": "2015-12-12 12:12:12", "properties": { "firstName": { "type": "string" }, "lastName": { "t...
date	date	1449893532000
quantity	string	200
title	string	ExampleSchema

本例中取消勾选了**是否切分未定义的 Key**，自定义了title、date和quantity三个字段，并指定了类型。ARMS 会只切分自定义的字段，并转成对应的类型（本例中date字段被转成了 ARMS 内置的 Long 型）。

自定义切分中 if/assign 的语法介绍

ARMS 默认支持 Aviator 表达式引擎，您可以完成对数据处理时的逻辑控制、赋值操作等。

条件判断 if-else

在**自定义切分**标签页的左侧操作面板中选择**逻辑**，系统提供两种类型的逻辑判断积木块 “if/else” 和 “if” ，如下图所示：

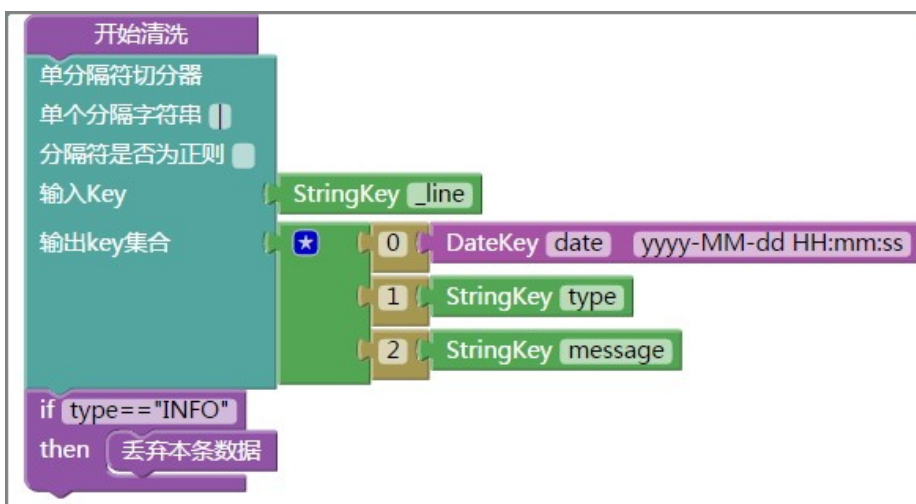


下图演示一些简单的逻辑表达式使用例子，用户日志如下所示：

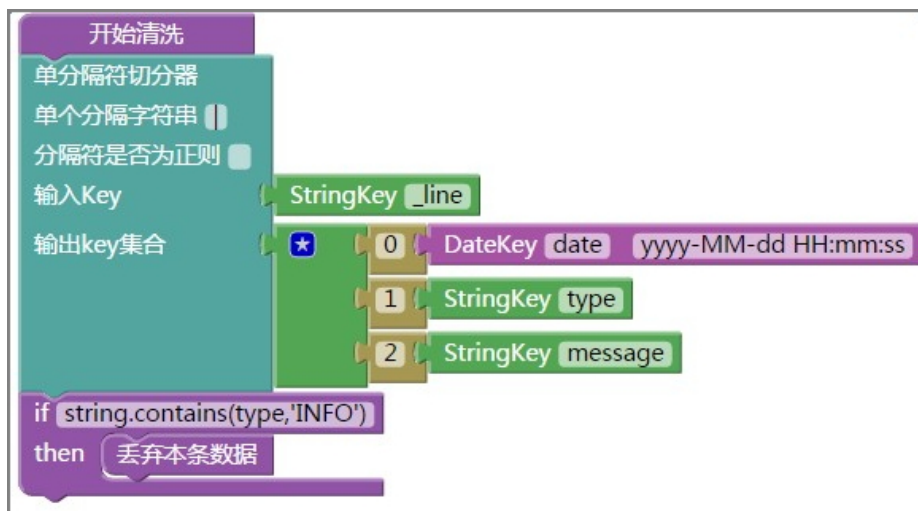
```
2017-01-09 16:02:49|ERROR|it is error...
2017-01-09 16:03:49|INFO|it is ok...
2017-01-09 16:03:49|INFO_1|it is ok...
```

日志按照 “|” 切分之后的字段分别为date、type、message。

示例一：当type为 INFO 时，该行日志直接丢弃；



示例二：当type包含 INFO 的时候，该行日志丢弃；

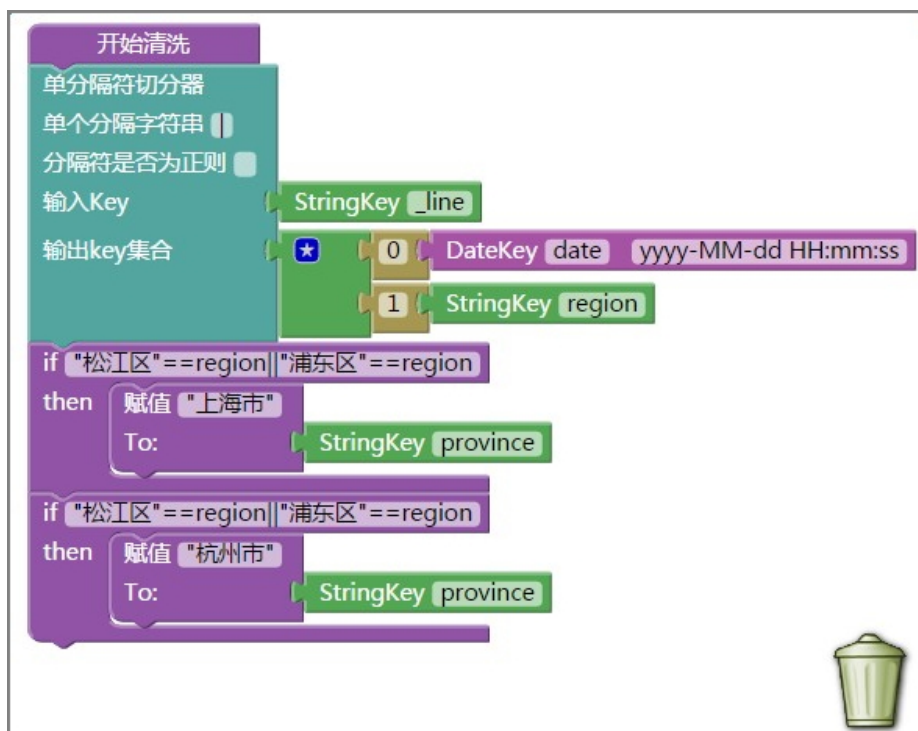


赋值

接下来演示一个简单的赋值使用例子，用户日志如下所示：

```
2017-01-09 16:02:49|松江区
2017-01-09 16:03:49|浦东区
2017-01-09 16:03:49|上城区
2017-01-09 16:03:49|下城区
```

日志按照 “|” 切分之后的的字段分别为date、region。如果region为 “松江区” 或者 “浦东区” ，则添加一个新的字段province（String 类型），其值为 “上海市”；如果region为 “上城区” 或者 “下城区” ，则添加一个新的字段province（String 类型），其值为 “杭州市”。



内置函数（转自 Aviator 官方文档）

Aviator 表达式是一个开源产品（[点此查看官方文档](#)）。常见内置函数的说明如下：

函数	描述
sysdate()	返回当前日期对象 java.util.Date
rand()	返回一个介于0-1的随机数，Double 类型
now()	返回 System.currentTimeMillis
long(v)	将值的类型转为 Long
double(v)	将值的类型转为 Double
date_to_string(date,format)	将 Date 对象转化化特定格式的字符串
string_to_date(source,format)	将特定格式的字符串转化为 Date 对象
string.contains(s1,s2)	判断 s1 是否包含 s2，返回 Boolean
string.length(s)	求字符串长度,返回 Long
string.startsWith(s1,s2)	s1 是否以 s2 开始，返回 Boolean
string.endsWith(s1,s2)	s1 是否以 s2 结尾,返回 Boolean
string.substring(s,begin[,end])	截取字符串 s，从 begin 到 end，end 如果忽略的话，将从 begin 到结尾，与 java.util.String.substring 一样
string.indexOf(s1,s2)	java 中的 s1.indexOf(s2)，求 s2 在 s1 中的起始索引位置，如果不存在为-1

<code>string.split(target,regex,[limit])</code>	Java 里的 <code>String.split</code> 方法一致
<code>string.replace_first(s,regex,replacement)</code>	Java 里的 <code>String.replaceFirst</code> 方法
<code>string.replace_all(s,regex,replacement)</code>	Java 里的 <code>String.replaceAll</code> 方法
<code>math.abs(d)</code>	求 d 的绝对值
<code>math.sqrt(d)</code>	求 d 的平方根
<code>math.pow(d1,d2)</code>	求 d1 的 d2 次方
<code>math.log(d)</code>	求 d 的自然对数
<code>math.log10(d)</code>	求 d 以 10 为底的对数
<code>math.sin(d)</code>	正弦函数
<code>math.cos(d)</code>	余弦函数
<code>math.tan(d)</code>	正切函数

日志清洗中系统字段介绍

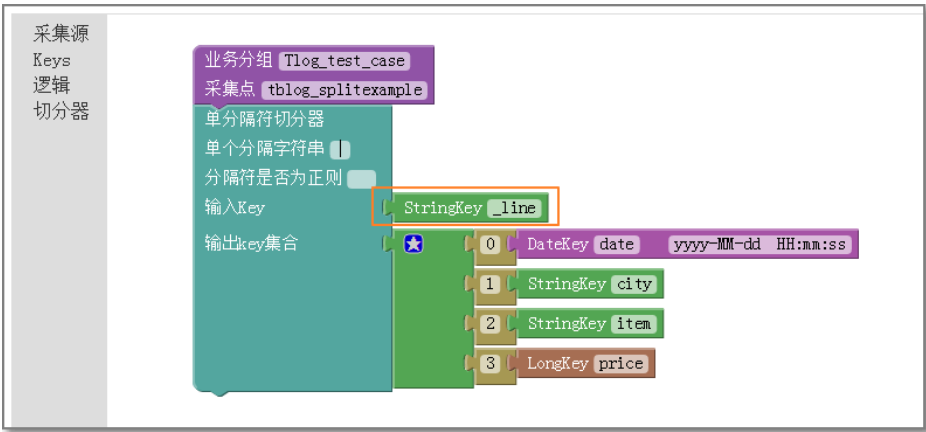
在“日志清洗”环节，除了用户定义的字段，通常系统会添加一些默认的字段，主要包括 `_line`、`_hostIp` 和 `_sysTime`。

`_line`

`_line` 字段表示每一行日志。例如，日志格式如下：

```
2016-07-28 12:12:12|北京|电子产品|211
2016-07-28 12:13:12|北京|电子产品|211
2016-07-28 12:14:12|北京|电子产品|211
```

进行第一次切分时，其输入 Key 为每一行日志 `_line`，其自定义切分形式如下：



_hostIp

_hostIp字段表示每一行日志的来源 IP，目前 ARMS 支持 ECS、LogHub 和 SDK 数据源，其支持关系如下表所示。

数据源	是否支持	备注
ECS 数据源	支持	无
LogHub 数据源	视情况而定	如果 Loghub 中的数据是通过 SDK 写入则不支持；如果 LogHub 中的数据是 SLS 抓取的则支持。
SDK 数据源	不支持	无

目前智能切分不提供_hostIp，只有在自定义切分模式下才提供_hostIp。以上面的日志为例来说明，单击**日志切分预览**：

字段名称	类型	样例数据
_line	string	2016-07-28 12:12:12 北冥 电子产品 211
date	date	1469679132000
city	string	北冥
price	long	211
_hostIp	string	127.0.0.1
item	string	电子产品

上图中_hostIp字段为 127.0.0.1（单击**日志切分预览**后，无论什么数据源，_hostIp字段均为 127.0.0.1），是因为处于本地模式。任务真正运行时会产生真实数据。

_sysTime

_sysTime字段表示日志的处理时间，如果您的日志中没有自己的业务时间，可以选择_sysTime时间字段进行聚合计算。

任务运行错误处理

TLogParseException 类型转换异常

异常原因：通常用户刚开始使用 ARMS 的时候，会使用 ARMS 的智能切分功能对样本数据进行切分。对于样本数据中是数字的字段，智能切分器会将其置为 LongKey，而实际数据过来时，发现数据有部分是字符串型的，这个时候就会报类型转换异常。

解决方案：

1. 重新进入监控任务编辑页面。
2. 在第 2 步**日志清洗**页面，选择**自定义切分**。
3. 根据错误提示，调整相应的类型转换异常字段。比如 String 无法转为 Long 的状况下，可以点开 Keys，找到 StringKey 替换 LongKey 即可。

FlowControlException 维度限流异常

异常原因：数据集大维度状况下，计算存储资源消耗会非常严重。在有限资源的状况下，ARMS 为了保证大部分监控任务能正常运行，对数据集维度个数进行了一定的限制，当前维度个数限制为 1000 个。当数据集的维度大小超出 1000 个时，ARMS 就会提示维度限流异常。

解决方案：

1. 根据错误提示中提到的数据集 ID，在**监控管理**>**数据集管理**页面搜索确定出现异常的数据集。
2. 查看维度的设置，看是否是自己真正需要观测的维度。如果不是自己需要关注的维度，请调整数据集，将维度配置的地方设置为**无**。
3. 回到监控任务页面，点击**暂停**按钮，再点击**恢复**按钮，新的配置即可生效。

如果确定配置的维度是自己需要关注的维度，请在 ARMS 控制台首页点击**联系我们**，与我们的客服人员联系，申请 VIP。我们会为您开通独立的计算存储资源，解决维度限流异常。

ExpressionRuntimeException 表达式异常

异常原因：通常是由于实际的数据与配置的切分器出现了不匹配。

解决方案：

1. 暂停监控任务，进入监控任务编辑页面。
2. 进入第 2 步**日志清洗**页面。
3. 将出现异常的日志，贴入日志抓取结果下的文本框。
4. 逐步拉开降低切分器的复杂度，点击**日志切分预览**，查看切分是否正常，最终找到不符合预期的部分。
5. 通过调整切分器或者调整输出数据解决该异常。

JSONException JSON 切分异常

异常原因：通常出现这个异常的原因是出现了异常的 JSON 数据，导致 JSON 切分器无法正常工作。

解决方案：根据错误提示，调整自己的日志输出为符合切分规则的 JSON。

RAM 子账号访问控制台

ARMS 支持阿里云主子账号（RAM）访问体系，本文介绍如何授权子账号访问 ARMS。

前提条件

使用 RAM 子账号访问 ARMS 控制台，子账号需要满足以下两个条件：

- 需要为该子账号创建 AccessKeyId/AccessSecretKey。
- 必须启用该子账号的控制台登录功能，为该子账号设置登录用户名、密码。

操作方法请参考 RAM 文档。

授权子账号访问 ARMS 服务控制台

满足登录前提条件的子账号，还需要授权才能使用 ARMS 服务控制台。目前 ARMS 仅支持授予子账号 ARMS 服务资源完全访问权限。

注意：Open API 暂时不支持子账号访问。

操作步骤如下：

1. 登录 RAM 控制台，在子账号管理界面，选择要授权的子账号。操作方法参考 RAM 文档给用户授权。
2. 在授权策略页面选择 AliyunARMSFullAccess。

授权完成后，即可用子账号登录 ARMS 控制台。

区域化架构总览和注意事项

ARMS 在部署之初主要部署了杭州 Region。为了快速满足异地 Region 的用户需求，不同 Region 的用户数

据通过跨 Region 传输到 ARMS 杭州 Region 的计算资源和存储资源。随着用户日益增加，以及对监控响应时效的逐渐提升，ARMS 已近期开始 Region 化。本文详细介绍 Region 前后架构对比以及对用户的影响。

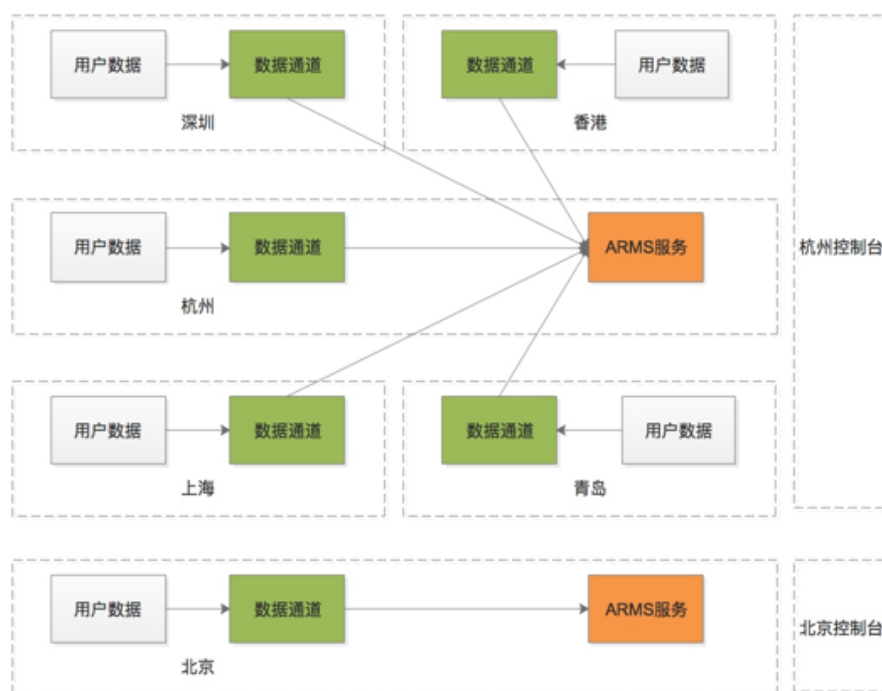
目前 ARMS 公有云支持杭州、北京、上海、青岛、深圳、香港地域。

原北京控制台近期将会下线，请尽快将北京控制台的任务迁移到 ARMS 中心控制台。

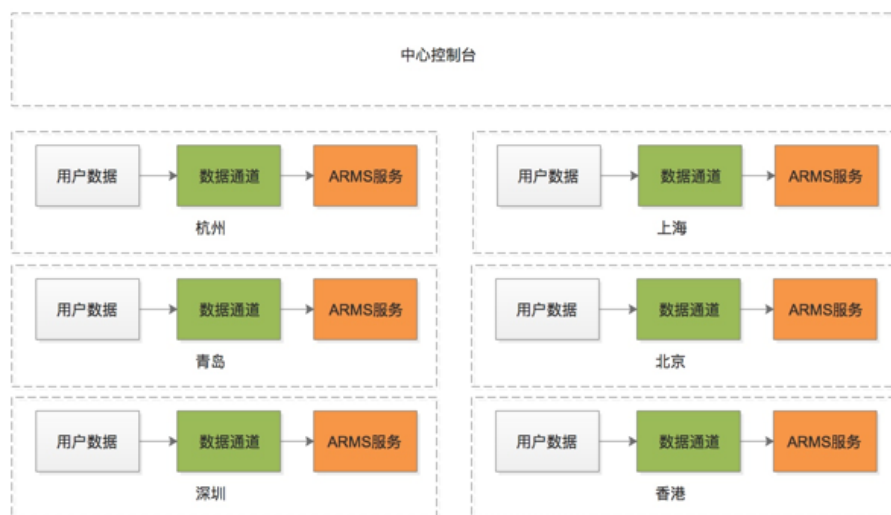
ARMS Region 化前后架构对比

- 在最初 ARMS 部署杭州 Region 时，所有用户数据将上传到本地的数据通道。当需要计算时，本地数据通道将通过跨 Region 的网络传输到杭州 ARMS 的计算资源和存储资源。
- 在本次多地部署 Region 化之后，以深圳为例，相关的数据会从本地的数据通道直接进入到深圳的 ARMS 开通的计算资源和存储资源。**数据原则上不出 Region，所以原杭州中心控制台的用户如果有数据源在深圳地话，会受到影响，详情见下文。**
- 此外，在Region 化后，公有云国内只提供一个中心控制台，所以对原北京控制台的用户也会产生很大影响。

下图展示了公有云国内 Region 化之前的架构：



下图展示了公有云国内 Region 化之后的架构：



任务迁移方法

如何将原杭州中心控制台的非杭州数据源任务迁移到中心控制台下的对应Region任务。

在进行 Region 化之前，原中心控制台同时支持杭州、上海、青岛、深圳、香港。但是 ARMS 底层服务只是部署在杭州，所以存在跨 Region 数据传输的问题。原中心控制台的任务虽然统一定义为杭州任务，由于历史原因，部分用户使用了非杭州 Region 的数据源。这种情况下，请注意以下事项。

注意事项：

如果您的杭州任务使用了非杭州 Region 的 ECS、非杭州 Region 的 LogHub、非杭州 Region 的 SDK、非公网 Region 或者非杭州 Region 的 MQ 数据源，请尽快将原有任务停止，并建立相应 Region 的任务；

- 如果您的杭州任务满足上述条件，并且进入流程进行了除数据源配置以外的更改，同时保存了配置，可能导致用户任务不可用（因为杭州任务不支持非杭州数据源的配置）；

如果您没有修改任务或者启停任务，任务将正常运行。但是建议您尽快处理这种跨 Region 数据源配置的任务，在 2018-01-30 之前系统将对这些异常任务进行停止服务。

如何将北京控制台的任务迁移到中心控制台

由于国内采用中心控制台，原北京控制台即日起用户将无法建立新的任务，老的任务可以继续运行一段时间，但是2018-01-30前停止提供服务，请尽快将老的任务迁移到中心控制台。

迁移步骤：

如果您的任务处于运行状态或者暂停状态，请将任务全部停止；

在任务列表中选择要迁移的任务，在**更多>导出模板**中导出任务的模板配置；

进入中心控制台，在**自定义监控>监控任务管理**在右侧**新建监控任务**下**导入自定义模板**，导入自定义模板请参考如下文档，新建的任务请选择北京区域，如下图所示：



任务新建成功之后，需要在流程中配置相关的数据源。如果没有数据源，请到数据源管理模块进行相关的同步操作等。

返回监控任务列表后，点击**启动**。到此完成迁移工作。

注意事项：

迁移时一定要先停止北京控制台的任务，否则中心控制台的任务拉取日志可能会出现异常；

如果登陆到中心控制台，发现自己的数据源（ECS、LogHub）少了，请点击同步操作；

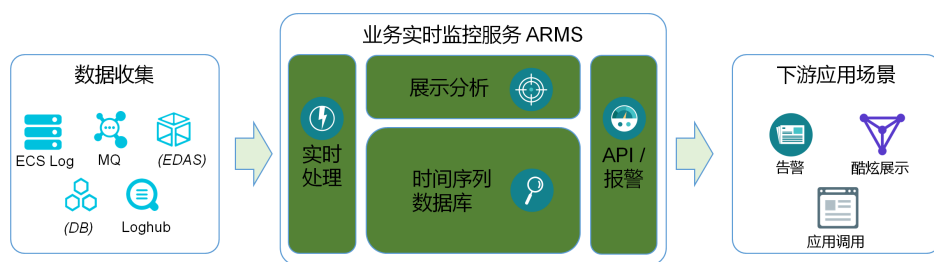
迁移到中心控制台后，原北京控制台的数据将丢失。

云产品对接

本文阐述了 ARMS 在云产品的对接问题。

ARMS 产品对接总览

ARMS 在阿里云上可以和各产品进行对接，如下图所示。



其中：

- ECS Log、MQ、Loghub 等产品作为数据源可与 ARMS 无缝对接，详情可查看 ARMS。
- 其他下游产品，例如 DataV，其对接方式比较特殊，详述如下。

DataV 对接

前提条件：

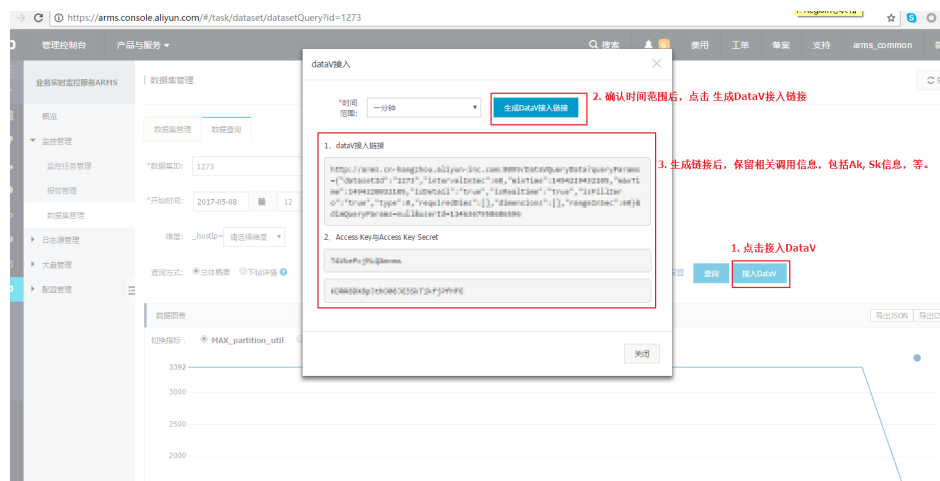
- 开通 ARMS
- 开通 DataV 基础版

接入步骤：

确定要进行查询的数据

在 ARMS 控制台的左侧菜单栏中，选择自定义监控 > 数据集管理。选择要查询的数据集，补全相关参数，并单击查询数据按钮。

验证查询结果。确认无误后，单击“接入 DataV”按钮，并确认生成信息。如下图所示：

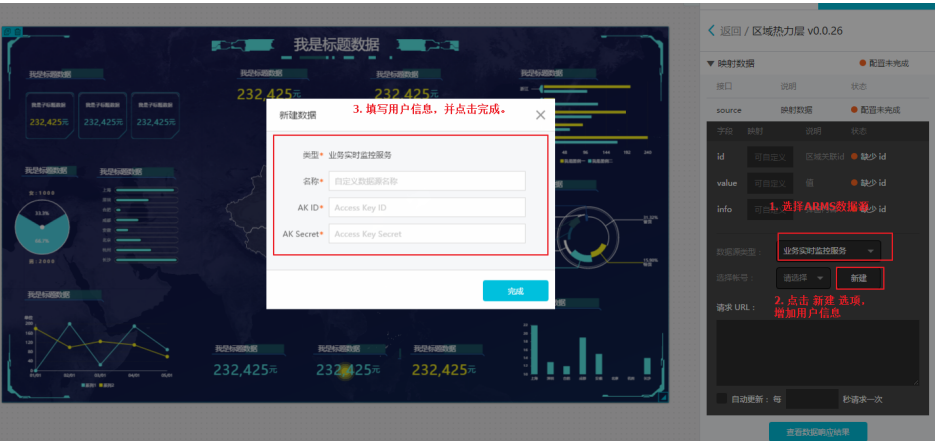


在 DataV 中选择 ARMS 数据源进行展示

在 DataV 中某个具体要展示的数据层，选择 ARMS 数据源。

如果第一次使用，则需要在界面中创建账号。如下图所示，单击新建账号，将第一步中在 ARMS 上

生成的 Ak、Sk 填入，并填写一个方便识别的用户名称。



创建用户以后，将在第一步中 ARMS 生成函数 URL 填入到 DataV 中，单击验证，并根据结果填写字段映射表。



至此，ARMS 接入完成。

常见问题

问：我在 ARMS 中下钻维度采用的是省市区的中文名字，如四川省、上海市。在 API 输出时，API 调用采用下钻详情输出，也输出了所有省市的正确结果。但是如何在 DataV 的热力图中进行展示？

答：DataV 的地图热力图在地区 ID 方面仅支持区域码。如果要在 DataV 中进行展示，需要将省市转换成区域码。可以打开省市地区码映射，将其内容拷贝下来，在 ARMS 中建立一个维度的映射表，并在 ARMS 结果输出时，将结果关联到映射表做转义。详情请参考映射表管理。

ARMS 商用通知

ARMS “前端监控”和“应用监控”产品预计于 2018 年 2 月 9 日前后正式上线商用功能。2018 年 2 月 12 日正式开始收费。

商用版开通细则

- 正式商用后，请您在控制台 arms.console.aliyun.com 或产品详情页 aliyun.com/product/arms 开通按量付费。商用后将有三天的缓冲期，在缓冲期内开通按量付费将不会被收取费用，缓冲期于 2018 年 2 月 12 日截止。
- ARMS “前端监控”和“应用监控”服务将继续免费提供。没有开通付费服务的用户，在商用后三天以内不受影响。三天以后，服务将自动转为免费服务。监控服务每天超过免费额度后将会被自动截停。用户需要在控制台上手动执行恢复任务操作才能恢复任务。恢复任务操作需要用户满足以下条件。
 - 开通相应付费服务，或
 - 等待翌日免费流量套餐恢复
- 如需进一步了解 ARMS 产品功能以及付费服务和免费服务的区别，请点击[此处](#)。
- 如需进一步了解 ARMS 服务类别和开通流程，请点击[此处](#)。
- 关于 ARMS 商用版执行价格，请点击[此处](#)。

商用版开通重要注意事项（必读）

每日消费限流

ARMS 前端监控和应用监控采用按量付费和资源包收费模式，为帮助用户防止因异常消费造成资金损失，ARMS 提供按每日核算的消费限流封顶的功能。您可以前往应用监控和前端监控控制台设置每日限流。以应用监控为例，设置方法如图所示。



修改需谨慎每天消费上限(Agent*Hour):

48

温馨提示:

- 1.调整日自定义使用量上限,限额调整最小值为48 Agent*Hour,最大值为48000 Agent*Hour;
- 2.调整日自定义使用量上限时,新的设定值不能低于当日实际使用量;
- 3.若有特殊需求可在概览页通过钉钉与我们联系;

确认

取消

系统报警联系人设置

商用版发布后,针对可能会出现每日限流问题,ARMS 提供了系统报警功能。为接收相关系统报警,您需要在 ARMS 控制台的报警管理中修改报警联系人设置,将报警联系人设置为系统报警联系人。报警联系人的设置如图所示。

*姓名:

zuoyu

手机号码:

18621867001

邮箱:

dinghui.yan@alibaba-inc.com

手机号码和邮箱至少有一项必填;同一手机号码或邮箱不能重复存储

是否接收系统通知:

☒

确定

关闭