



日志服务数据加工系列培训

<<< 主题: 扫平日志分析路上障碍, 实时海量日志加工实践培训 >>>

讲师: 丁来强 (成喆) - 阿里高级技术专家 | 唐恺(风毅) - 阿里技术专家

分享介绍

8月7日	8月8日	8月13日	8月14日	8月20日	8月21日	8月28日	8月29日
19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30
数据加工 介绍与实战	数据加工DSL 核心语法介绍	数据加工DSL 语法实践	数据加工动态 数据分发汇集实践	非结构化数据 解析实践	结构化数据 解析实践	数据映射 富化实践	数据加工 可靠性与排错实践

数据处理DSL: 核心语法介绍

系列培训二

丁来强（成喆）

议题

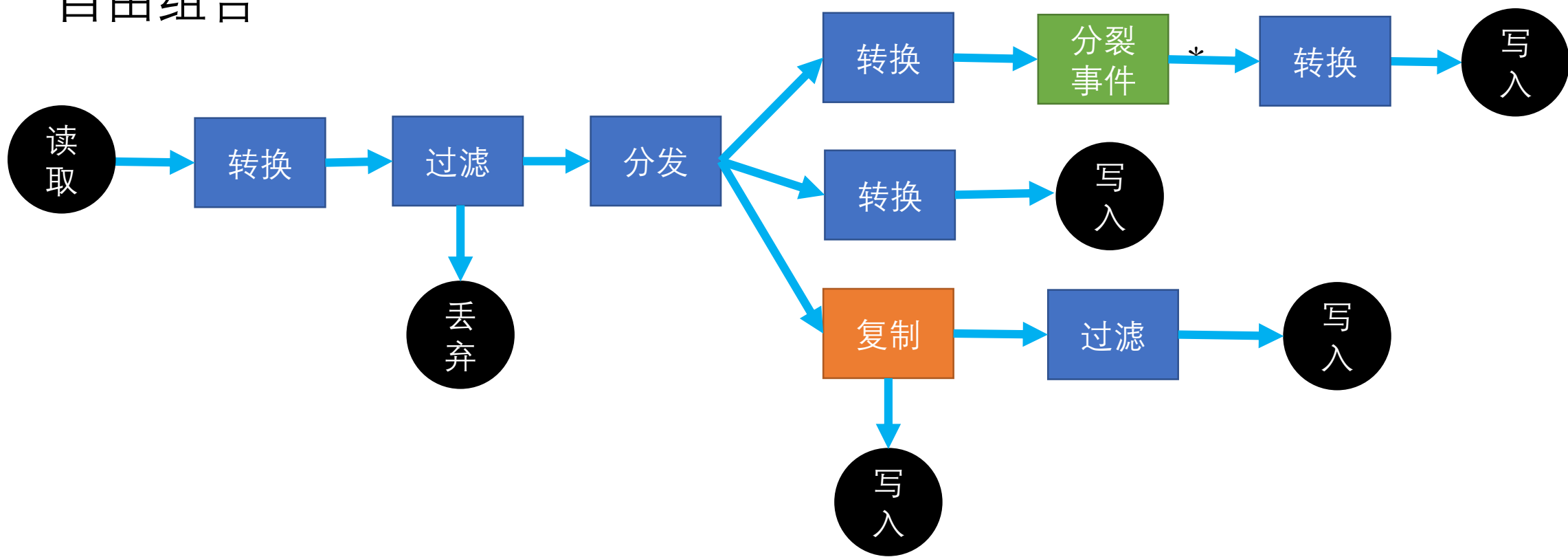
- DSL概览
- DSL语法详解
 - 基础数据结构
 - 基础语法
 - 事件类型
 - 函数框架与机制
- 查询字符串语法

数据加工DSL概览

自由编排DSL

200+函数, 400+GROK模式

自由组合



数据加工DSL能力

- 语法简洁
- 30+全局操作函数
- 200+内置函数
- 数学/文本/时间/IP/处理算子
- OSS/Logstore/RDS-MySQL/IP库/配置资源连接器
- 400+Grok模式
- 类Lucene事件搜索算子
- JMES语法支持
- 代码内自由编排
- 组合操作：过滤、抽取、分裂、转换、富化、分发等

兼容Python的DSL

类别	Python语法	DSL语法
数据结构	数字, 字符串, 布尔	支持, 除""形式字符串不支持
数据结构	元祖, 列表, 集合, 字典	支持, 除集合set如{1,2,3}不支持
数据结构	对象定义	仅支持内置扩展数据结构,如表格, 日期时间对象等
基本语法	操作符, 如加减乘除等	不直接支持, 需要通过函数方式支持
基本语法	注释	支持
基本语法	变量定义赋值	不支持, 需使用无状态方式调用传递
基本语法	条件、循环	不直接支持, 控制逻辑通过内置函数实现
函数	标准Python内置函数	不支持, 使用内置200+函数
函数	函数调用	支持, 除解包调用不支持
函数	自定义函数def或lambda	不支持, 提供200+事件与表达式函数, 支持自由组合调用
模块	导入与使用Python标准库	不支持 (特殊需求可提工单)
模块	线程与进程创建	不支持 (特殊需求可提工单)
模块	导入第三方库	不支持 (特殊需求可提工单)
模块	外部网络连接或命令调用	提供内置的资源连接器

基本数据结构

整数、浮点、布尔、空

- 整数
 - `e_set("f1", 100)`
- 浮点
 - `e_set("f1", 1.5)`
- 布尔
 - `True, False, true, false`
- 空
 - `None` 或 `null`, 表示无(不同于空字符串)
 - 许多函数也以`None`为命名参数的默认值.
 - 函数返回时表示空(全局事件函数以返回`None`表示删除)

字符串

- 字符串
 - "abc"等价于'abc'
 - 字符\用于转义，自身为\\或者r修饰
 - 宽字符友好，例如"你好"的长度也是2.
 - 正则表达式以字符串表示，常用r修饰
- 字节
 - b'abc'不同于字符串的内存编码形式. 某些特殊函数接收或者返回.

列表、元祖、字典

- 列表
 - 也叫数组，形式：[1,2,3,4]
 - 函数可能接受或返回列表，例如e_dict_map、json_select
- 元组(tuple)
 - 形式：(1, 2, 3, 4)
 - 和列表功能一样，某些函数参数使用元组形式
- 字典
 - 形式为 {"key": "value", "k2": "v2", ...} 的键值对组合
 - 关键字一般是字符串，值可以是任何支持的数据结构
 - 存储格式以哈希方式，关键字不能重复. 查找时无序.
 - 事件是一种特殊的字典

日期对象、表格

- 表格
 - 多列的表格结构
 - 可以CSV格式内容构建, 或者从RDS, Logstore拉取
 - 一般传递给接收表格作为参数的函数, 如e_table_map, tab_to_dict等
- 日期时间对象
 - 表示日期时间的内存对象
 - 某些时间类函数会接收或返回

基础语法

注释、换行

- 注释 (# 开头)

```
# 设置默认主题（放在行首的注释）  
e_set("__topic__", "access_log")    # 设置默认主题（放在行尾的注释）
```

- 换行:

- 使用\衔接换行
- 在,的地方可以直接换行（例如函数参数）

```
e_set("__topic__", "this is a very long long long .....\" \\  
      \".....long text")  
  
e_set("__topic__", "v1",  
      "type", "v2",  
      "length", 100)    # 逗号分隔的可以直接换行
```

操作符

场景操作	函数	样例
加 +	op_add	op_add(v("age"), 2)
减 -	op_sub	op_sub(v("age"), 2)
乘 *	op_mul	op_mul(v("size"), 2)
幂 **	op_pow	op_pow(v("size"), 2)
整除 //	op_div_floor	op_div_floor(v("bytes"), 1024)
取模 %	op_mod	op_mod(v("age"), 10)
取负 -	op_neg	op_neg(v("profit"))
判断存在 in	op_in	op_in(["pass", "ok"], v("result"))
判断不存在 not in	op_not_in	op_in(["pass", "ok"], v("result"))
逻辑且 and	op_and	op_and(op_gt(v("age"), 18), op_lt(v("age"), 31))
逻辑或 or	op_or	op_or(op_le(v("age"), 18), op_gt(v("age"), 65))
逻辑否 not	op_not	op_not(op_gt(v("age"), 18))
判断等于 ==	op_eq	op_eq(v("name"), "xiao ming")
判断不等于 !=	op_ne	op_ne(v("name"), "xiao ming")
大于 >	op_gt	op_gt(ct_int(v("age")),)
大于等于 >=	op_ge	op_ge(ct_int(v("age")), 18)
小于	op_lt	op_lt(ct_int(v("age")), 18)
小于等于 <=	op_le	op_le(ct_int(v("age")), 18)
字符串切片 [...]	op_slice	op_slice(v("message"), 0, 20)

```
# *
e_set("a", 3600 * 6)      # 非法
e_set("a", op_mul(3600, 6))  # 合法

# /
e_set("bytes_kb", v("bytes") / 1024)      # 非法
e_set("bytes_kb", op_div_floor(v("bytes"), 1024))  # 合法
```

函数调用方式

- 基本调用方式:

- `e_set("abc", "xyz")`
 - 非命名参数一般都需要传递所有值.

- 命名参数调用方式:

- `e_set("abc", "xyz", mode="fill")`
- 命名参数不传递, 会使用默认值.
- 多个命名参数下, 可以根据选择传递, 顺序不要求(以下一样) :
`e_csv("data", ["f1", "f2", "f3"], sep='#', quote="|")`
`e_csv("data", ["f1", "f2", "f3"], quote="|", sep='#')`
 - 注意: 命名参数的顺序, 始终在非命名参数的后面.

函数调用方式 (2)

- 组合调用

- 参数是其他函数的调用方式

- ```
e_set("abc", v("xyz"))
```

- ```
e_set("abc", str_lower(v("xyz")))
```

- 变参

- 某些函数支持变参传递, 例如:

- ```
e_set("k1", "v1", "k2", "v2",)
```

- 带命名参数时, 命名参数放最后:

- ```
e_set("k1", "v1", "k2", "v2", ..., mode="fill")
```

真假判断

数据类型	True的条件	False的条件
布尔	True, true	False, false
None	-	总是False
数值	非0, 0.0	0, 0.0
字符串	非空	空串
字节	非空	空字节
元组	非空	空元组
列表	非空	空列表
字典	非空	空字典
表格	存在即为True	空对象(None)
日期时间	存在即为True	空对象(None)

```
e_if(True, DROP)           # 总是
e_if(1, DROP)              # 总是
e_if(v("abc"), DROP)       # 存在字段abc, 且字段不为空时
e_if(str_isdigit(v("abc")), DROP) # 存在字段abc, 且字段的内容都是数字时
```

事件类型

基本类型

- 数据加工的日志的数据结构是以**字典**
 - 例如{"__topic__": "access_log", "content": "....."}
- 字典的关键字和值, 对应于日志的字段和值
- 注意: 事件的关键字和值都是字符串.
- 注意: 关键字不能重复.
- **事件类的函数**会自动接收事件

元字段

- 时间字段: `__time__`: 是Unix时间戳的**字符串**
- 其他的有
 - 主题: `__topic__`
 - 源: `__source__`
- 时间字段修改
 - 修改这个字段的值, 等于修改了日志的事件时间.
 - 可以使用**时间函数**对齐进行进一步的各种操作.
 - 删除了这个字段, 等于重置了时间
 - 在输出日志时, 会自动用当前时间.

标记 (tag)

- 不同于一般字段, 格式为__tag__:名称
 - 如果源logstore打开了服务器接收时间的日志, 则会存在tag: __tag__:__receive_time__
 - K8S的日志会存在许多容器类的tag, 例如: __tag__:__container_name__
- 可以自由添加Tag:
 - e_set("__tag__:type", "access_log")
 - 或者在e_output、e_coutput时设置tag

赋值自动转换

- 注意: 事件的关键字和值都是字符串, 提取或设置时, 会自动转换为字符串

类型	样例	转换类型	转换样例
整数	1	字符串	"1"
浮点	1.2	字符串	"1.0"
布尔	True	字符串	"true"
字节	b"123"	使用UTF8反转为字符串	"123"
元组	(1, 2, 3)	json转换	"[1, 2, 3]"
列表	[1,2,3]	json转换	"[1, 2, 3]"
字典	{"1":2, "3":4}	json转换	"{"1": 2, "3": 4}"
日期时间	datetime(2018, 10, 10, 10, 10, 10)	ISO格式转换	2018-10-10 10:10:10

```
e_set("v1", 12.3)
e_set("v2", True)
```

固定标示

- 方便使用

类型	标示	说明
布尔	true	真, 等价于True
布尔	false	假, 等价于False
None	null	无, 等价于None
字符串	F_TAGS	TAG字段正则表达式, 等价于"__tag__:.+"
字符串	F_META	topic, source, TAG字段的正则表达式表示, 等价于__tag__:.+ __topic__ __source__
字符串	F_TIME	time字段的名称, 等价于__time__
字符串	F_PACK_META	pack meta字段的正则表达式表示形式, 等价于"__pack_meta__ __tag__:__pack_id__"
字符串	F_RECEIVE_TIME	服务器接收时间的tag字段, 等价于"__tag__:__receive_time__"

JSON对象

- 一般指JSON表达式函数json_select或者json_parse解析提取后的对象, 并没有真正JSON对象

原始字符串	解析出的JSON对象	实际类型
1	1	整数
1.2	1.2	浮点
true	True	布尔
false	False	布尔
"abc"	"abc"	字符串
null	None	None
["v1", "v2", "v3"]	["v1", "v2", "v3"]	列表
["v1", 3, 4.0]	["v1", 3, 4.0]	列表
{"v1": 100, "v2": "good"}	{"v1": 100, "v2": "good"}	字典
{"v1": {"v11": 100, "v2": 200}, "v3": "good"}	{"v1": {"v11": 100, "v2": 200}, "v3": "good"}	字典

函数框架

内置函数

- 全局操作函数

- 只有全局操作函数才能构建加工规则的每一步骤.
- 接收事件, 处理并返回事件的函数

```
全局函数1(..参数....)  
全局函数2(..参数....)  
全局函数3(..参数....)  
全局函数4(..参数....)
```

- 表达式函数

- 通用型函数, 接收特定参数, 组合调用后作为传递给全局操作函数以定义更加灵活逻辑.

```
全局操作1(表达式函数1(...), ....)  
全局操作2(..., 表达式函数2(...), 表达式函数3(...), ...)
```

两者区别

特点	全局操作函数	表达式函数
可做全局步骤	是	否
接收	自动接收事件	除个别大部分不自动接收事件
返回	0到多条事件	特定数据结构
修改事件	大部分会	不会
可否组合调用	可以	可以

全局操作函数概览

类型	函数	说明
流程控制	e_if	多个条件操作的配对操作
	e_if_else	if-else操作
	e_switch	满足一个条件操作后跳出
	e_compose	组合
事件操作	e_drop	丢弃
	e_keep	保留
	e_split	分裂
	e_output	输出
	e_coutput	复制输出
字段操作	e_drop_fields	删除
	e_keep_fields	保留
	e_rename	重命名

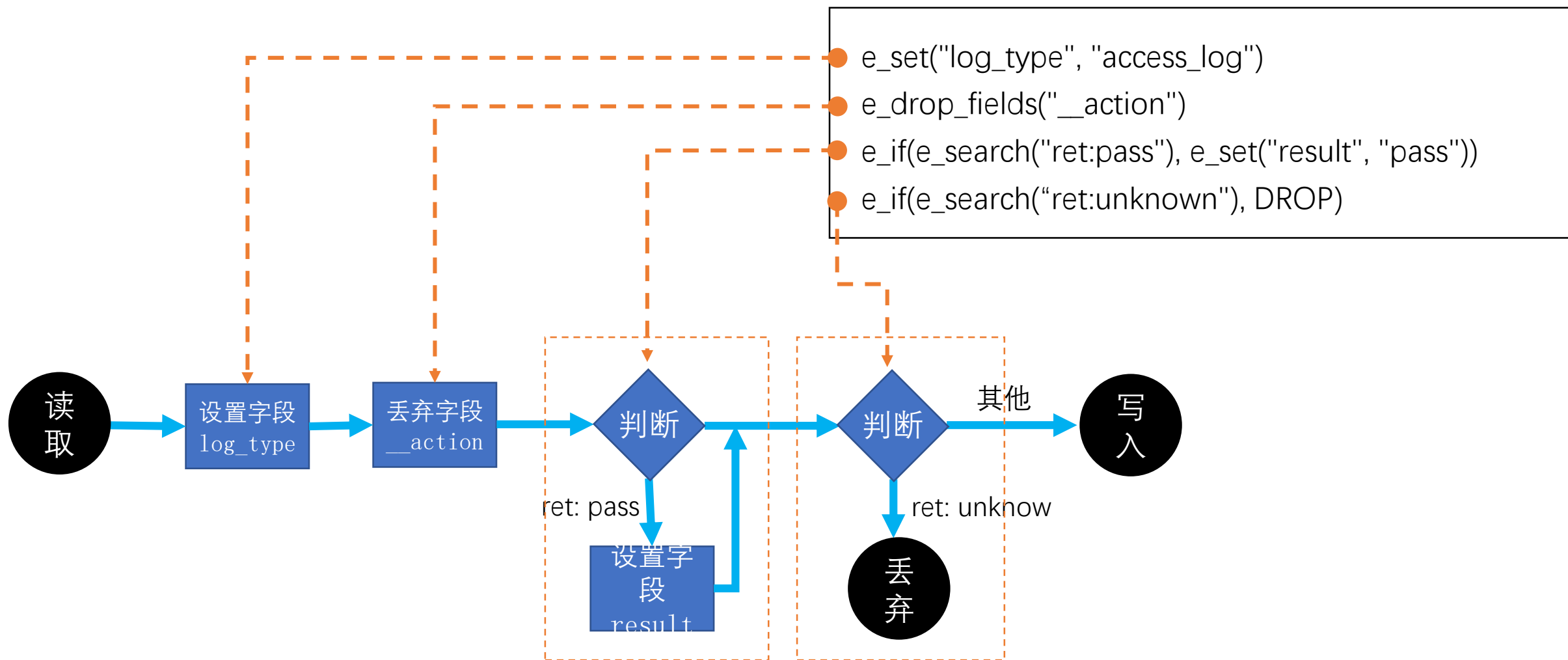
类型	函数	说明
字段值赋值	e_set	赋值
字段值提取	e_regex	正则提取
	e_json	json展开或提取
	e_kv	自动提取键值对
字段值提取	e_kv_delimit	基于分隔符提取键值对
	e_csv	逗号或其他分隔符提取
	e_tsv	tab分隔符提取
	e_psv	pipe分隔符提取
	e_syslogrfc	根据syslog协议提取头
字段富化	e_dict_map	字典映射
	e_table_map	表格映射
	e_search_map	搜索映射
资源操作	res_local_update	设置任务参数上下文

表达式函数概览

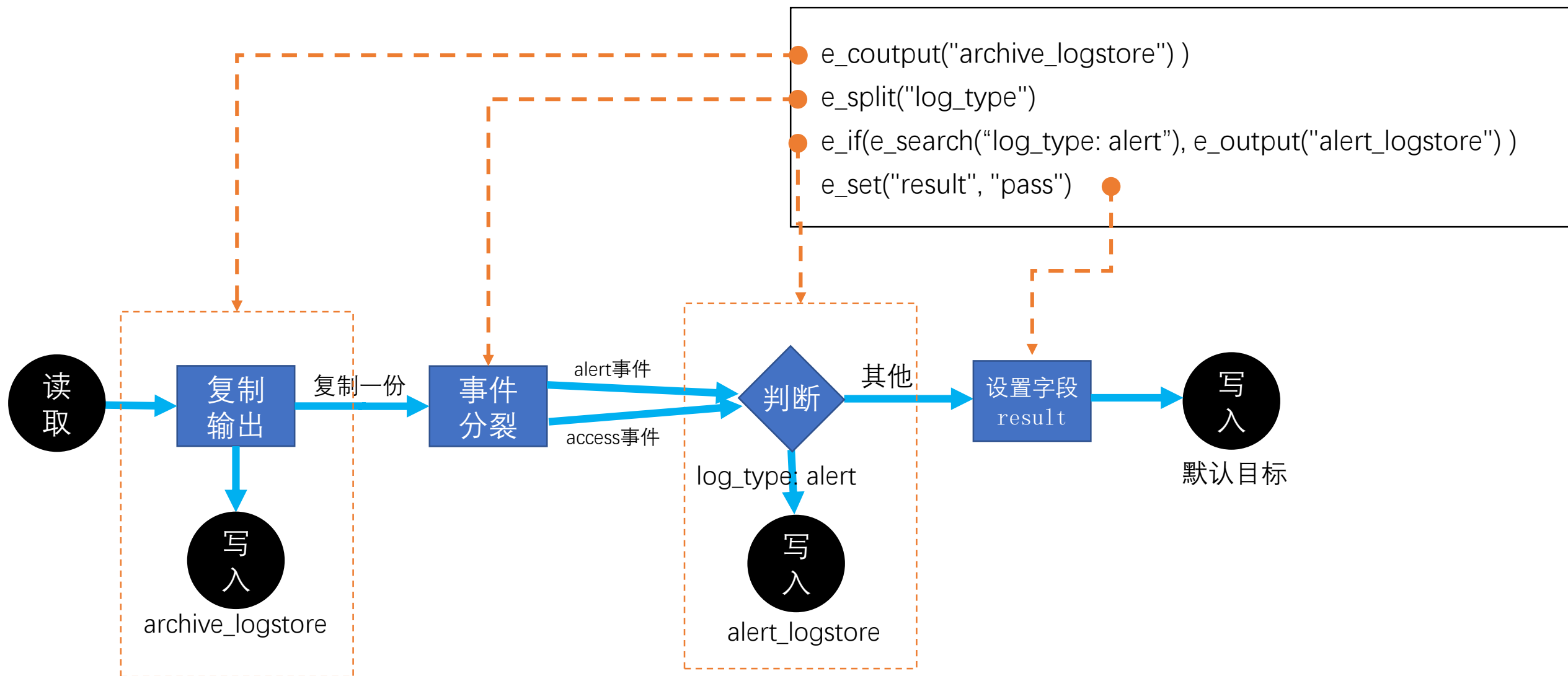
类型	函数	说明
事件检查函数	v, e_has, e_not_has, e_search, e_match, e_match_any, e_match_all等	获取事件字段值, 或判断字段或字段值是否符合特定内容
基础操作函数	部分op_* 函数	比较, 条件判断, 容器类计算, 一般性多值操作
转换函数	ct_*函数	数字,字符串,布尔之间的转换, 数字进制转换
算术函数	部分op_*函数, math_*函数等	数字的+ -*/幂等计算, 数学计算, 多值计算等
字符串函数	str_*函数	字符串的所有相关操作与判断搜索等
日期时间函数	dt_*函数	Unix时间戳, 日期时间对象, 日期时间字符串转化, 时区调整, 圆整等

类型	函数	说明
正则表达式函数	reg_*函数	正则提取, 检索, 替换, 分裂多值等
GROK函数	grok函数	提取grok模式返回对应正则表达式
JSON, XML, Protobuf函数	json_*, xml_*, pb_*函数	对应提取或解析
编码解码类函数	url_*, html_*, md5_*, sha1_*, base64_*函数	相关单向或双向函数
列表函数	lst_*函数	列表相关构建,获取, 修改, 操作等
字典函数	dct_*函数	字典相关构建,获取, 修改, 操作等
资源函数	res_*函数	本地配置, RDS, Logstore等资源获取

规则引擎原理：基本操作



规则引擎原理：输出，复制与分裂



函数浏览

查询字符串语法

相关函数

类型	函数	使用场景
表达式函数-事件判断函数	e_search	使用查询字符串判断事件的字段值是否满足特定条件, 返回True或False
表达式函数-资源函数	res_log_logstore_pull	拉取Logstore资源返回表格结构, 支持使用查询字符串配置黑白名单对行进行过滤筛选.
表达式函数-资源函数	res_rds_mysql	拉取RDS-MySQL资源返回表格结构, 支持使用查询字符串配置黑白名单对行进行过滤筛选.
全局事件函数-搜索表格映射	e_search_table_map	关键字是查询字符串, 值是匹配的值的字典进行映射

功能概览

功能	字段	全文
子串搜索	支持	支持
通配符*?搜索	支持	支持
完全匹配搜索	支持	-
正则表达式搜索	支持	-
数值范围搜索	支持	-
数值比较	支持	-
关系and, or, not以及自由组合	支持	支持

子串搜索

```
e_search("子串")  
e_search("字段名: 子串")
```

全文搜索

样例	场景
e_search("active error")	多个子串搜索, 默认关系是OR
e_search("active error")	搜索完整带空格的子串
e_search("错误")	中文子串

字段搜索

样例	场景
e_search("status: active")	子串搜索
e_search('author: "john smith"')	带空格子串搜索
e_search('fileId: active error')	相当于 field:active OR "error"

通配符搜索

- * ? 匹配: *表示 0个或多个字符串, ? 表示一个字符, 也可以表示一个宽字符如中文.

全文搜索

样例	场景
<code>e_search("active*test")</code>	匹配0到多个, 不需要用双引号括起来
<code>e_search("发生*错误")</code>	中文匹配, 可以匹配发生错误, 发生严重错误等
<code>e_search("active?good")</code>	? 可以不用双引号
<code>e_search("ac*tive?good")</code>	也可以应用于完全匹配
<code>e_search("ac*tive??go*od")</code>	支持多个混合使用

字段搜索

样例	场景
<code>e_search("status: active*test")</code>	匹配0到多个
<code>e_search("status: active?good")</code>	匹配一个

完全匹配

```
e_search("字段名==子串")
```

样例	场景
e_search('author== "john smith"')	字段author必须完全等于john smith
e_search("status== ac*tive?good")	可以与通配符结合使用

正则表达式匹配

```
e_search("字段名~=正则表达式字符串")
```

样例	场景
<code>e_search(r'status~= "\d+")</code>	status字段包含数字
<code>e_search(r'status~= "^\d+\$")</code>	status字段等于数字

数值比较

- 范围类比较

```
e_search('count: [100, 200]')    # >=100 and <=200  
e_search('count: [* , 200]')     # <=200  
e_search('count: [200, *]')     # >=200
```

- 数值直接比较

```
e_search('age >= 18')    # >= 18  
e_search('age > 18')     # > 18  
e_search('age = 18')     # = 18  
e_search('age <= 18')    # <=18  
e_search('age < 18')     # <18
```

逻辑关系

- 支持任意搜索之间的逻辑关系, 也支持用 () 进行嵌套.

逻辑	关键字
且	and AND && 大小写不敏感
或	or OR
否	not ! 大小写不敏感

全局逻辑关系

```
e_search("abc OR xyz") # 大小写不敏感
e_search("abc and (xyz or zzz)")
e_search("abc and not (xyz and not zzz)")
e_search("abc && xyz") # and
e_search("abc || xyz") # or
e_search("abc || !xyz") # or not
```

逻辑关系 (2)

字段逻辑关系

```
e_search("field: (abc OR xyz)") # 字段field包含 abc 或 xyz  
e_search("field: (abc OR not xyz)") # 字段field包含 abc 或 不包含xyz  
e_search("field: (abc && !xyz)") # 字段field包含 abc 且 不包含xyz
```

字段判断

样例	场景
<code>e_search("field: *")</code>	字段存在
<code>e_search("not field:*")</code>	字段不存在
<code>e_search('not field: ""')</code>	字段不存在
<code>e_search('field: "?"')</code>	字段存在, 且值不为空
<code>e_search('field== ""')</code>	字段存在, 且值为空
<code>e_search('field~="."+ ""')</code>	字段存在, 值不为空
<code>e_search('not field~="."+ ""')</code>	字段不存在或值为空
<code>e_search('not field== ""')</code>	字段不存在或值不为空

字段名转义

- **字段名不能使用双引号：** 包含特殊字符时直接使用\转义，
 - 包含中文, 字母, 数字, 下划线, 小横线等情况不需要用双引号, 其他情况需要.
 - `__tag_\:_container_name_:` abc
 - 中文字段: 错误
 - 错误:
"content": abc
- **值用字符串用双引号括起来, 不支持单引号**
 - 错误: `e_search("domain: '/url/test.jsp'")`
 - 正确: `e_search('domain: "/url/test.jsp"')`

搜索值转义

- 值包含"\"时需要用\"转义.
 - 例如:content: "abc\"xy\"z" 合法
- 希望搜索字符*?时, 也需要用\"转义, 否则会被视为通配符匹配
- 中文, 字母, 数字, 下划线, 小横线, *, ?等情况不需要用双引号, 其他情况需要用双引号括起来.
 - status: "\"*\"?()[]:=\" 值含特殊字符的值, 推荐放在"中, 除了*?和\"需要转义外, 其他不用转义
 - status: active\"*test 值只包含*或?, 可以不用双引号
 - status: active\"?test 值只包含*或?, 可以不用双引号
 - content: ()[]:= 非法

注意点

f1: "abc xyz"

表示：字段f1里面搜索子串"abc xyz"

f1: (abc xyz)

f1: abc or f1: xyz # 等价于

表示：字段f1里面搜索abc或xyz

f1: (abc and xyz)

f1: abc and f1: xyz # 等价于

表示：字段f1里面搜索abc且xyz

f1: abc and xyz

(f1: abc) and (xyz) # 等价于

表示：字段f1里面搜索"abc"，且全文搜索xyz

f1: abc xyz

f1: abc or xyz # 等价于

(f1: abc) or (xyz) # 等价于

表示：字段f1搜索abc，或全文搜索xyz

最佳实践

基础

- [函数调用最佳实践](#)
- [事件判断最佳实践](#)
- [日期时间处理最佳实践](#)

分发汇总

- 数据分发: [跨账号多目标logstore数据分发](#)
- 数据汇总: [跨账号多源logstore数据汇总](#)

非结构化文本解析

- 解析syslog协议框架: [解析syslog/Rsyslog的标准格式](#)
- 一般性文本: [使用正则表达式与grok解析Nginx日志](#)
- 动态KV: [动态键值对KV解析](#)
- 特定格式的: [特定格式文本的数据加工](#)

[持续更新](#)

<https://yq.aliyun.com/articles/712381>

结构化文本解析

- 复杂JSON格式加工:
 - [多子键为数组的复杂JSON](#)
 - [多层数组对象嵌套的复杂JSON](#)
- CSV格式的: [解析CSV格式的日志](#)

数据富化

- [构建字典与表格](#)
- [从RDS-MySQL获取数据](#)
- [从其他logstore获取数据](#)
- [使用搜索映射做高级数据富化](#)

Thanks

日志服务数据处理系列培训



欢迎加入
(上海、杭州)



<<< 主题: 扫平日志分析路上障碍, 实时海量日志加工实践培训 >>>

讲师: 丁来强 (成喆) - 阿里高级技术专家 | 唐恺(风毅) - 阿里技术专家

分享介绍

8月7日	8月8日	8月13日	8月14日	8月20日	8月21日	8月28日	8月29日
19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30	19:30-20:30
数据加工 介绍与实战	数据加工DSL 核心语法介绍	数据加工DSL 语法实践	数据加工动态 数据分发汇集实践	非结构化数据 解析实践	结构化数据 解析实践	数据映射 富化实践	数据加工 可靠性与排错实践