阿里云开发者社区

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 19:30-20:30	8月8日 19:30-20:30	8月13日 19:30-20:30	8月14日 19:30-20:30	8月20日 19:30-20:30	8月21日 19:30-20:30	8月28日 19:30-20:30	8月29日 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

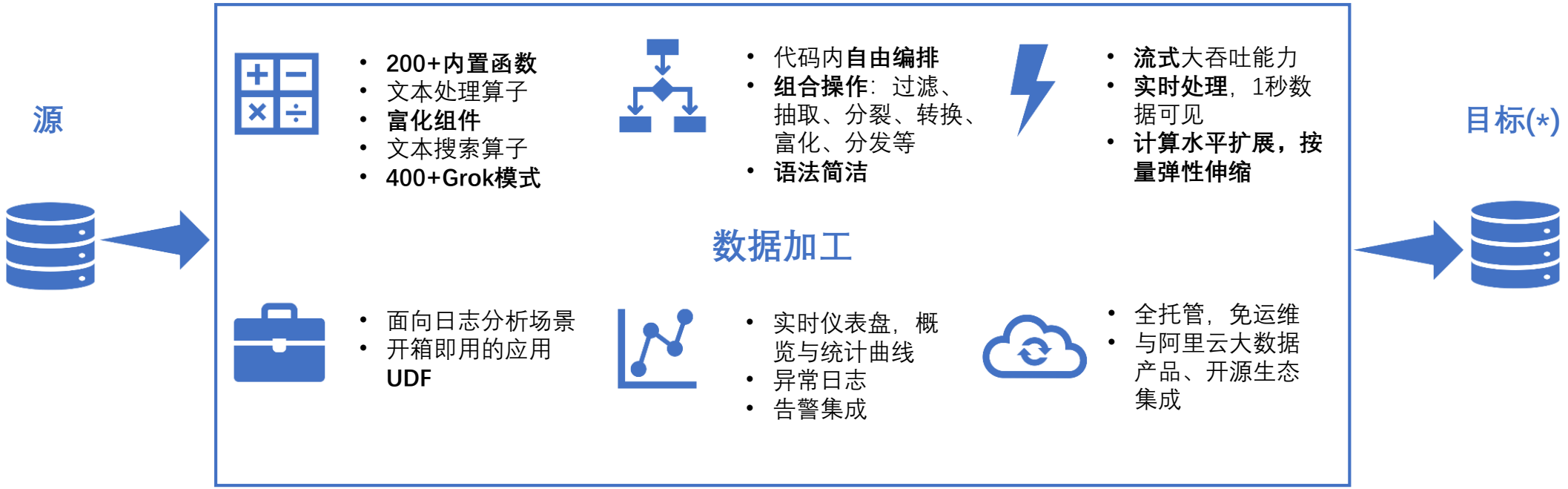
数据加工:非结构化数据解析实践

系列培训五

唐恺

日志服务-数据加工简介

- 功能概述
  - 将各类日志处理为结构化数据，具备全托管、实时、高吞吐的特点
  - 面向日志分析领域，提供丰富算子、开箱即用的场景化UDF（Syslog、非标准json、AccessLog UA/URI/IP解析等）
  - 丰富的阿里云大数据产品（OSS、MC、EMR、ADB等）、开源生态（Flink、Spark等）集成能力，降低数据分析门槛
- 典型场景
  - 数据规整：对混乱格式的日志进行字段提取、格式转换，获取结构化数据以支持后续的流处理、数仓计算
  - 数据富化：日志（例如业务订单）与维表（例如用户信息MySQL表）进行字段join，为日志添加更多维度信息供分析
  - 数据分发：将全量日志按转发规则分别提取到多个下游存储供不同业务使用



相关语法与原理

非结构化数据处理

- 数据规整在做什么?
  - schema on write
  - raw text -> table
- 最常见的两类数据
  - 自描述格式（下次实践主题）
    - json
    - delimiter
    - key-value
    - xml等
  - 非结构化数据（本次实践主题）
    - 正则表达式
    - grok
    - 字符串函数

正则表达式相关函数

类型	函数	功能	匹配方式
全局操作函数	e_regex	使用正则从从字段值中提取值	部分
	e_keep_fields	使用正则匹配字段名	完全
	e_drop_fields	使用正则匹配字段名	完全
	e_rename	使用正则匹配字段名	完全
	e_kv	使用正则提取关键字与值	部分
	e_search_dict_map	关键字是搜索字符串, 支持正则	部分
	e_search_table_map	表格字段是搜索字符串, 支持正则	部分
表达式函数	e_match	使用正则匹配值	参数控制, 默认完全
	e_search	接受搜索字符串, 支持正则	部分
	regex_select	使用正则从值中提取值	部分
	regex_findall	使用正则搜索匹配值	部分
	regex_match	使用正则匹配值	参数控制, 默认部分
	regex_replace	对值正则替换值	部分
	regex_split	使用正则做分隔符	部分

正则提取日志

- 提取的字段名必须满足字符条件，否则会被丢弃
- 正则规则：

```
u'_*[\u4e00-\u9fa5\u0800-\u4e00a-zA-Z][\u4e00-\u9fa5\u0800-\u4e00\\w\\.\\-]*'
```

  - 字符集：中文、字母、数字、_、-、.
  - 可以下划线开头
  - 非_的第一个字符必须是中文或字母

```
# 保留
_test_
字段名
字段名12
field1
字_段.名
a_b-c.d123

# 丢弃
__1__
1abc
1中文
a@b
```

值提取： regex_select

说明：根据正则表达式和配置提取特定的值

语法：regex_select(值, r"正则表达式", mi=None, gi=None)

参数名称	字段属性	是否必填	说明
值	任意	是	填入要匹配的值
正则表达式	String	是	填入正则表达式
mi	Number	否	默认None，第几个match
gi	Number	否	默认None，match内第几个group

值提取： regex_select

```
e_keep(e_search("http_user_agent: Mozilla"))
e_set("http_user_agent_kv_m0", regex_select(v("http_user_agent"), r"(\w+)/(\d+.\d+)"))
e_set("http_user_agent_kv_m0_g0", regex_select(v("http_user_agent"), r"(\w+)/(\d+.\d+)", mi=0, gi=0))
e_set("http_user_agent_kv_m0_g1", regex_select(v("http_user_agent"), r"(\w+)/(\d+.\d+)", mi=0, gi=1))
e_set("http_user_agent_kv_m1", regex_select(v("http_user_agent"), r"(\w+)/(\d+.\d+)", mi=1))
e_keep_fields(r"http_user_agent.*")
```

预览任务开始时间： **2019-08-20 13:59:52**

原始日志 | 数据加工 new

>	输出目标	时间	内容
1	target0	08-20 15:00:10	__source__ : __topic__ : http_user_agent : Mozilla/5.0 (iPhone; CPU iPhone OS 12_3_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Mobile/15E148 MicroMessenger/7.0.5(0x17000523) NetType/4G Language/zh_CN http_user_agent_kv_m0 : Mozilla/5.0 http_user_agent_kv_m0_g0 : Mozilla http_user_agent_kv_m0_g1 : 5.0 http_user_agent_kv_m1 : AppleWebKit/605.1

值提取： regex_findall

说明：根据正则表达式获得符合条件的所有值的一个列表

语法：regex_findall(值, r"正则表达式")

参数名称	字段属性	是否必填	说明
值	任意	是	填入要匹配的值
正则表达式	String	是	填入正则表达式

值提取： regex_findall

```
1 e_set("client_ip_parts", regex_findall(v("client_ip"),"\d+"))
2 e_keep_fields(r"client_ip.*")
```

预览任务开始时间： **2019-08-20 14:00:30**

原始日志 | 数据加工 new

	输出目标	时间	内容
1	target0	08-20 14:19:23	__source__ : __topic__ : client_ip : 171.216.213.162 client_ip_parts : ["171", "216", "213", "162"]

匹配判断： regex_match

说明：判断是否匹配正则表达式

语法：regex_match(值, r"正则表达式", full=False)

参数名称	字段属性	是否必填	说明
值	任意	是	填入要匹配的值
正则表达式	String	是	填入正则表达式
full	Bool	否	默认False

匹配判断： regex_match

```
e_keep(e_search("request_method:POST"))
e_set("request_method_lable_1", regex_match(v("request_method"), r"POST"))
e_set("request_method_lable_2", regex_match(v("request_method"), r"POST", full=True))
e_set("request_method_lable_3", regex_match(v("request_method"), r"O.*T"))
e_set("request_method_lable_4", regex_match(v("request_method"), r"O.*T", full=True))
e_keep_fields(r"request_method.*")
```

预览任务开始时间： 2019-08-20 13:54:36

原始日志 | 数据加工 new

>	输出目标	时间	内容
1	target0	08-20 14:10:39	__source__ : __topic__ : request_method : POST request_method_lable_1 : true request_method_lable_2 : true request_method_lable_3 : true request_method_lable_4 : false

替换：regex_replace

说明：根据正则表达式获得替换(或删除)字符串的新的字符串

语法：regex_replace(值, r"正则表达式", replace="", count=0)

参数名称	字段属性	是否必填	说明
值	任意	是	填入要被替换的值
正则表达式	String	是	填入正则表达式
replace	String	否	替换为什么字符串, 默认为空串
count	Number	否	最多替换次数后停止, 默认为0, 表示替换所有

替换： regex_replace

```
e_keep(e_search("http_user_agent: Mozilla"))
e_set("http_user_agent_masked_all", regex_replace(v("http_user_agent"), r"\d+.\d+", "<major_version>.<minor_version>"))
e_set("http_user_agent_masked_first", regex_replace(v("http_user_agent"), r"\d+.\d+", "<major_version>.<minor_version>", count=1))
e_keep_fields(r"http_user_agent.*")
```

预览任务开始时间： **2019-08-20 14:13:43**

原始日志 | 数据加工 new

>	输出目标	时间	内容
1	target0	08-20 14:29:00	__source__ : __topic__ : http_user_agent : Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.131 Safari/537.36 http_user_agent_masked_all : Mozilla/<major_version>.<minor_version> (Windows NT <major_version>.<minor_version>; WOW64) AppleWebKit/<major_version>.<minor_version> (KHTML, like Gecko) Chrome/<major_version>.<minor_version>.<major_version>.<minor_version> Safari/<major_version>.<minor_version> http_user_agent_masked_first : Mozilla/<major_version>.<minor_version> (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.131 Safari/537.36

切分： regex_split

说明：返回根据正则表达式分裂值的一个列表

语法：regex_split(值, r"正则表达式", maxsplit=0)

参数名称	字段属性	是否必填	说明
值	任意	是	填入要分裂的值
正则表达式	String	是	填入正则表达式
maxsplit	Number	否	默认为0

切分： regex_split

```
1 e_keep(e_search("http_user_agent: Mozilla"))
2 e_set("http_user_agent_split_all", regex_split(v("http_user_agent"), r"\s*\([^)]*\)\s*"))
3 e_set("http_user_agent_split_one", regex_split(v("http_user_agent"), r"\s*\([^)]*\)\s*", maxsplit=1))
4 e_keep_fields(r"http_user_agent.*")
```

预览任务开始时间： **2019-08-20 14:35:52**

原始日志 | 数据加工 new

>	输出目标	时间	内容
1	target0	08-20 14:51:09	__source__ : __topic__ : http_user_agent : Mozilla/5.0 (Linux; Android 7.1.1; OPPO A83 Build/N6F26Q; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/62.0.3202.84 Mobile Safari/537.36 http_user_agent_split_all : ["Mozilla/5.0", "AppleWebKit/537.36", "Version/4.0 Chrome/62.0.3202.84 Mobile Safari/537.36"] http_user_agent_split_one : ["Mozilla/5.0", "AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/62.0.3202.84 Mobile Safari/537.36"]

字段提取：e_regex

说明：使用正则表达式，从事件现有字段值中提取一个或多个值出来，赋值到事件字段中

语法：

e_regex(源字段名，有命名捕获正则)

e_regex(源字段名，无捕获正则，目标字段名)

e_regex(源字段名，无捕获正则，目标字段名数组)

e_regex(源字段名，有捕获正则，目标字段名数组)

e_regex(源字段名，有捕获正则，目标键值字典)

e_regex(...上述参数...mode="fill-auto")

参数名称	字段属性	是否必填	说明
字段名	字段名	是	任意字符（特殊字段名的设置，可以参考事件类型；注意：如果字段在事件中不存在，则不进行任何操作.）
正则表达式	String	是	参考正则表达式（非捕获：有时需要使用GROUP来做逻辑，但是又不是捕获时，使用前缀?:，例如 \w+@\w+\.\w(?:\.\cn)?）
mode	String	否	默认fill-auto，请参考覆盖模式（无任何匹配时，不进行任何操作）

字段提取： e_regex

```
#e_regex(源字段名，有命名捕获正则)
e_regex("http_user_agent", r"\((?P<first_system_information>[^)]+)\)[^()]*\((?P<first_platform_details>[^)]+)\).*")
#e_regex(源字段名，无捕获正则，目标字段名)
e_regex("http_user_agent", r"\([^)]+\)", "second_system_information")
#e_regex(源字段名，无捕获正则，目标字段名数组)
e_regex("http_user_agent", r"\([^)]+\)", ["third_system_information", "third_platform_details"])
#e_regex(源字段名，有捕获正则，目标字段名数组)
e_regex("http_user_agent", r"\(([^)]+)\)[^()]*\(([^)]+)\).*", ["forth_system_information", "forth_platform_details"])
#e_regex(源字段名，有捕获正则，目标键值字典)
e_regex("http_user_agent", r"\(([^)]+)\)[^()]*\(([^)]+)\).*", {r"fifth_kv": r"[\1] [\2]"})
```

预览任务开始时间： **2019-08-20 14:37:49**

原始日志 | 数据加工 new

>	输出目标	时间	内容
1	target0	08-20 16:09:02	__source__ : __topic__ : fifth_kv : [Windows NT 10.0; WOW64] [KHTML, like Gecko] first_platform_details : KHTML, like Gecko first_system_information : Windows NT 10.0; WOW64 forth_platform_details : KHTML, like Gecko forth_system_information : Windows NT 10.0; WOW64 http_user_agent : Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36 second_system_information : (Windows NT 10.0; WOW64) third_platform_details : (KHTML, like Gecko) third_system_information : (Windows NT 10.0; WOW64)

字段提取：grok

说明：正则表达式较为复杂, 推荐使用GROK，还可以和正则混合使用

语法：grok(pattern, escape=False, extend=None)

参数名称	字段属性	是否必填	说明
pattern	String	是	grok语法, 参考grok语法, 具体的GROK模式, 参考grok模式
escape	Bool	否	是否将其他非grok pattern中的正则相关特殊字符做转义, 默认不转义, 参考grok与正则混合
extend	Dict	否	用户自定义grok表达式，默认为None，参考样例7

字段提取： grok

```
e_keep(op_eq(v("__tag__:__path__"), "/var/log/messages_traditionfileformat"))
e_regex('content', grok('%{SYSLOGBASE} %{GREEDYDATA:message}'))
```

预览任务开始时间： 2019-08-19 17:40:20

原始日志 | 数据加工 new

>	输出目标	时间	内容
1	target0	08-20 11:56:32	__source__ : 172.16.157.214 __tag__:__client_ip__ : 47.97.157.202 __tag__:__hostname__ : iZbp1a65x3r1vhpe94fi2qZ __tag__:__path__ : /var/log/messages_traditionfileformat __tag__:__receive_time__ : 1566273395 __topic__ : content : Aug 20 11:56:32 iZbp1a65x3r1vhpe94fi2qZ root: I'm a huge fan of Weibo, the most popular social networking site in China. I'm on it 24/7. logsource : iZbp1a65x3r1vhpe94fi2qZ message : I'm a huge fan of Weibo, the most popular social networking site in China. I'm on it 24/7. program : root timestamp : Aug 20 11:56:32

字符串函数

类型	函数	说明
多字符串操作	str_format	以格式化字符串形式格式化多个字符串
多字符串操作	str_join	以分隔字符串链接多个字符串
多字符串操作	str_zip	对2个(值或表达式的)字符串进行并发分裂在合并成一个字符串返回
编码 解码	str_encode	对字符串进行进行编码
编码 解码	str_decode	对字符串进行进行编码
编码 解码	str_hex_escape_encode	对字符串进行hex反转解码
排序、倒叙、替换	str_logtash_config_normalize	对字符串logstash格式转换
排序、倒叙、替换	str_translate	返回字符串用字符集映射后的版本
更多参考文档		

字符串函数

```
1 e_set("host_splits", str_split(v("host"), "."))
2 e_set("host_and_method", str_format("{} -> {}",v("request_method"), v("host")))
3 e_keep_fields(r"host.*", "request_method")
4
```

预览任务开始时间： **2019-08-20 14:37:49**

原始日志 | 数据加工 new

	输出目标	时间	内容
1	target0	08-20 16:34:55	__source__ : __topic__ : host : xncz9tgcfc3.lobxon.com host_and_method : POST -> xncz9tgcfc3.lobxon.com host_splits : ["xncz9tgcfc3", "lobxon", "com"] request_method : POST

综合实践：Syslog解析

grok与regex结合处理混乱syslog格式

```
e_switch(
    op_eq(v("__tag__:__path__"), "/var/log/messages_traditionfileformat"),
    e_compose(
        e_regex('content', grok('%{SYSLOGBASE} %{GREEDYDATA:message}')),
        e_set("__topic__", "traditionfileformat")
    ),
    op_eq(v("__tag__:__path__"), "/var/log/messages_fileformat"),
    e_compose(
        e_regex('content',grok('%{TIMESTAMP_ISO8601:timestamp} %{SYSLOGHOST:hostname} %{SYSLOGPROG} %{GREEDYDATA:message}')),
        e_set("__topic__", "fileformat")
    ),
    op_eq(v("__tag__:__path__"), "/var/log/messages_fluent3164"),
    e_compose(
        e_regex('content', grok('%{POSINT:priority}>%{SYSLOGTIMESTAMP:timestamp} %{SYSLOGHOST:hostname} %{WORD:ident}(?P<pid>(\[(\d+)\]|[^:])): (?P<level>(\[(\w+)\]|[^ ])) %{GRE\
EDYDATA:message}')),
        e_set("__topic__", "rfc3164")
    ),
    op_eq(v("__tag__:__path__"), "/var/log/messages_protocol23format"),
    e_compose(
        e_regex('content',grok('%{POSINT:priority}>%{NUMBER:version} %{TIMESTAMP_ISO8601:timestamp} %{SYSLOGHOST:hostname} %{PROG:program} - - - %{GREEDYDATA:message}')),
        e_set("__topic__", "protocol23format")
    ),
    default=e_drop()
)
```

总结

非结构化数据解析实践总结

- 如何开始
  - 求助《最佳实践》： 文档、云栖，钉钉群
  - 学习语法（正则）， 善用工具
- 三板斧

手段	上手难度	解析能力	调试难度
grok	低	强	低
正则表达式函数	较高	非常强	高
字符串函数	低	中	低

阿里云开发者社区

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 19:30-20:30	8月8日 19:30-20:30	8月13日 19:30-20:30	8月14日 19:30-20:30	8月20日 19:30-20:30	8月21日 19:30-20:30	8月28日 19:30-20:30	8月29日 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