阿里云
日志服务

产品简介

文档版本：20190912

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

法律声明.....I
通用约定.....I
1 什么是日志服务..... 1
2 产品架构..... 3
3 产品优势..... 5
3.1 功能优势.....5
3.2 成本优势.....5
3.3 查询分析全方位对比（ELK）.....7
3.4 查询方案（ELK/Hive）对比.....25
4 应用场景.....27
5 基本概念.....31
5.1 简介.....31
5.2 日志.....31
5.3 日志组.....34
5.4 项目.....35
5.5 日志库.....35
5.6 分区.....35
5.7 日志主题.....37
6 限制说明.....39
6.1 基础资源.....39
6.2 数据读写.....40
6.3 查询分析与可视化.....41
6.4 保留字段.....42

1 什么是日志服务

日志服务（Log Service，简称 LOG）是针对日志类数据的一站式服务，在阿里巴巴集团经历大量大数据场景锤炼而成。您无需开发就能快捷完成日志数据采集、消费、投递以及查询分析等功能，提升运维、运营效率，建立 DT 时代海量日志处理能力。

日志服务学习路径

日志服务学习路径图为您推荐热门功能的操作指引文档，帮助您快速了解日志服务产品。视频与文档结合，全方位提升您的产品使用及文档阅读体验。

产品介绍

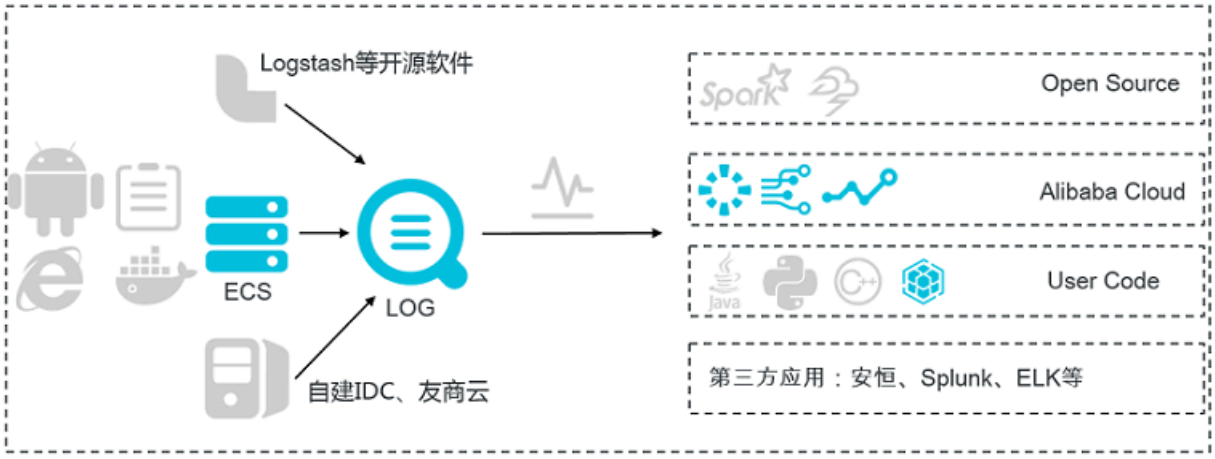
日志服务产品介绍视频：

实时采集与消费（LogHub）

功能：

- 通过ECS、容器、移动端，开源软件，JS等接入实时日志数据（例如Metric、Event、BinLog、TextLog、Click等）。
- 提供实时消费接口，与实时计算及服务对接。

用途：数据清洗（ETL），流计算（Stream Compute），监控与报警， 机器学习与迭代计算。



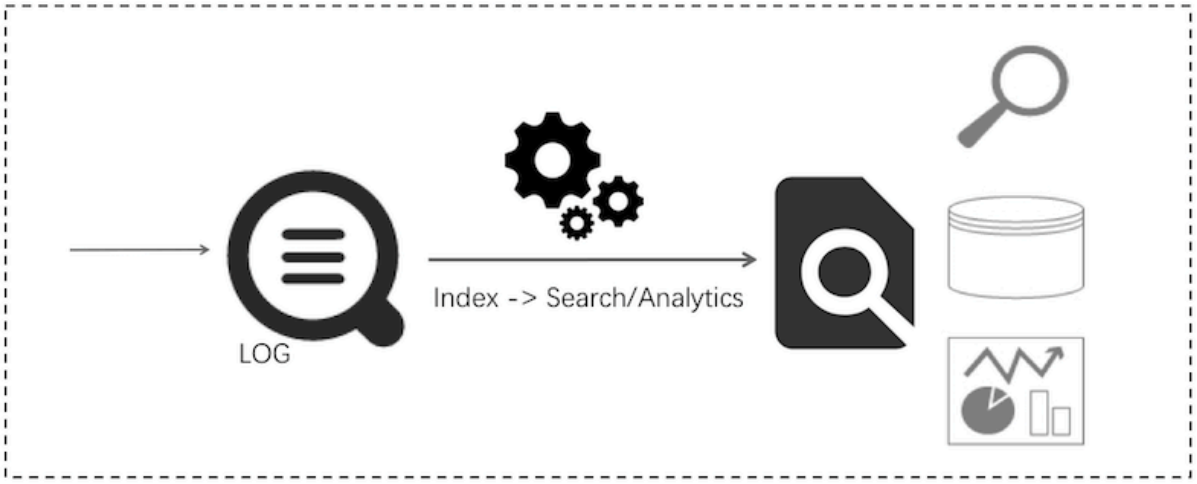
查询与实时分析（Search/Analytics）

实时索引、查询分析数据。

- 查询：关键词、模糊、上下文、范围。
- 统计：SQL聚合等丰富查询手段。
- 可视化：Dashboard + 报表功能。

· 对接：Grafana，JDBC/SQL92。

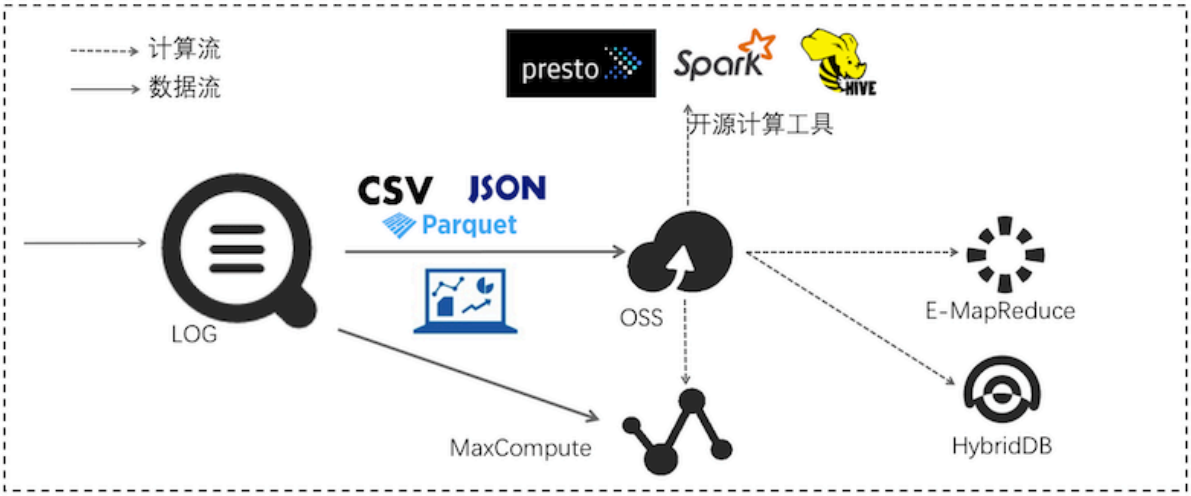
用途：DevOps/线上运维，日志实时数据分析，安全诊断与分析，运营与客服系统。



投递数仓（LogShipper）

稳定可靠的日志投递。将日志中枢数据投递至存储类服务进行存储。支持压缩、自定义Partition、以及行列等各种存储方式。

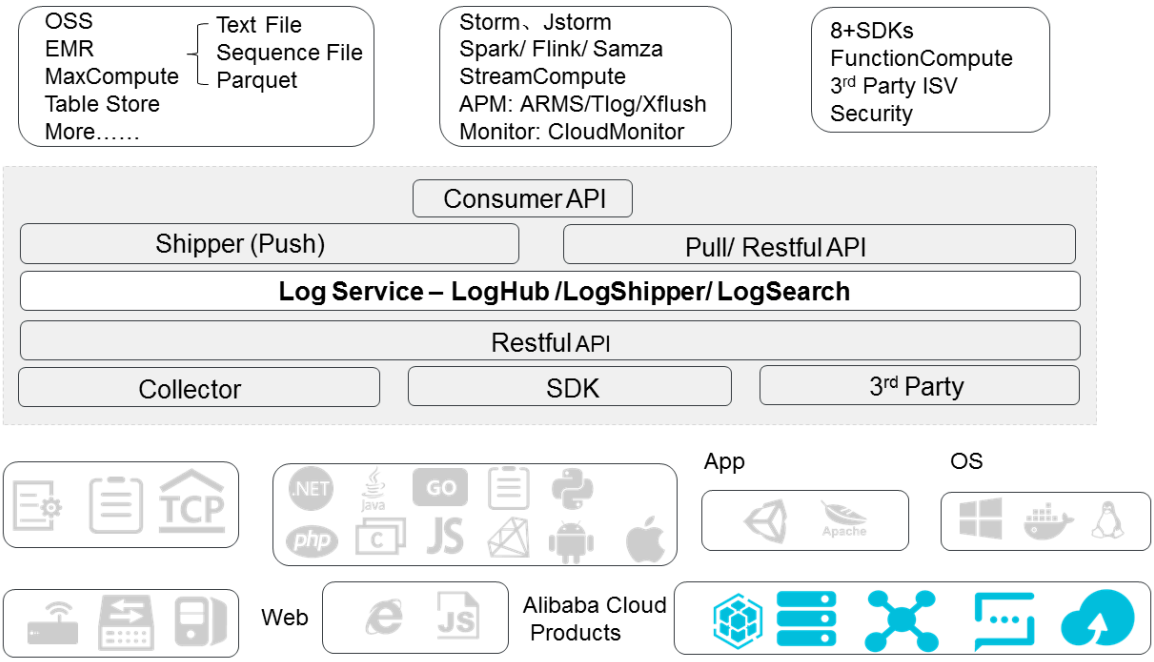
用途：数据仓库 + 数据分析、审计、推荐系统与用户画像。



2 产品架构

日志服务的架构如下图所示。

图 2-1: 产品架构



Logtail

帮助您快速收集日志的Agent。其特点如下所示：

- 基于日志文件、无侵入式的收集日志
  - 只读取文件。
  - 日志文件无侵入。
- 安全、可靠
  - 支持文件轮转不丢失数据。
  - 支持本地缓存。
  - 网络异常重试。
- 方便管理
  - Web端操作。
  - 可视化配置。

- 完善的自我保护
  - 实时监控进程CPU、内存消耗。
  - 限制使用上限。

前端服务器

采用LVS + Nginx构建的前端机器。其特点如下所示：

- HTTP、REST协议
- 水平扩展
  - 流量上涨时可快速提高处理能力。
  - 支持增加前端机。
- 高吞吐、低延时
  - 纯异步处理，单个请求异常不会影响其他请求。
  - 内部采用专门针对日志的Lz4压缩，提高单机处理能力，降低网络带宽。

后端服务器

后端是分布式的进程，部署在多个机器上，完成实时对Logstore数据的持久化、索引、查询以及投递至MaxCompute。整体后端服务的特点如下所示：

- 数据高安全性：
  - 您写入的每条日志，都会被保存3份。
  - 任意磁盘损坏、机器宕机情况下，数据自动复制修复。
- 稳定服务：
  - 进程崩溃和机器宕机时，Logstore会自动迁移。
  - 自动负载均衡，确保无单机热点。
  - 严格的Quota限制，防止单个用户行为异常对其他用户产生影响。
- 水平扩展：
  - 以分区（Shard）为单位进行水平扩展。
  - 用户可以按需动态增加分区来增加吞吐量。

3 产品优势

3.1 功能优势

全托管服务

- 应用性强，5分钟即可接入服务进行使用，Agent支持任意网络下数据采集。
- LogHub覆盖Kafka 100%功能，提供完整监控、报警等功能数据，并支持弹性伸缩（可支持PB/Day规模），使用成本为自建50%以下。
- LogSearch/Analytics 提供快速查询、仪表盘和报警功能，使用成本为自建 20%以下。
- 30+接入方式，与云产品 （OSS/E-MapReduce/MaxCompute/Table Store/MNS/CDN/ARMS等）、开源软件（Storm、Spark）无缝对接。

生态丰富

- LogHub 支持30+采集端，包括Logstash、Fluent等，无论是嵌入式设备，网页，服务器，程序等都能轻松接入。在消费端，支持与Storm、Spark Streaming、云监控等对接。
- LogShipper 支持丰富数据格式（TextFile、SequenceFile、Parquet等），支持自定义Partition，数据可以直接被Presto、Hive、Spark、Hadoop、E-MapReduce、MaxCompute、HybridDB、DLA等处理。
- LogSearch/Analytics 查询分析语法完整、兼容SQL92、支持JDBC协议与Grafana对接。

实时性强

- LogHub：写入即可消费；Logtail（采集Agent）实时采集传输，1秒内到服务端（99.9%情况）。
- LogSearch/Analytics：写入即可查询分析，在多个查询条件下1秒可查询10亿级数据，多个聚合条件下1秒可分析1亿级数据。

完整API/SDK

- 轻松支持自定义管理及二次开发。
- 所有功能均可通过API/SDK实现，提供多种语言SDK，可轻松管理服务和百万级设备。

3.2 成本优势

成本优势

日志服务产品在日志处理的三种场景下具有以下成本优势：

- LogHub：
  - 与购买云主机 + 云磁盘搭建 Kafka 相比，对于 98% 场景下用户价格有优势。对小型网站而言，成本为 kafka 的30% 以下。
  - 提供 RESTful API，可以直接针对移动设备提供数据收集功能，节省了日志收集网关服务器的费用。
  - 免运维，随时随地弹性扩容使用。
- LogShipper：
  - 无需任何代码/机器资源，灵活配置与丰富监控数据。
  - 规模线性扩展 （PB级/Day），功能当前免费。
- LogSearch/Analytics：
  - 与购买云主机 + 自建 ELK 相比，成本为自建的 15% 以下，并且查询能力与数据规模有极大提升。与日志管理软件相比，能无缝支持各种流行流计算 + 离线计算框架，日志流动畅通无阻。

成本对比

以下是在计费模型下，日志服务功能与自建方案的对比，仅供参考。

日志中枢（LogHub vs Kafka）

-	关注点	LogHub	自建中间件（如Kafka）
使用	新增	无感知	需要运维动作
	扩容	无感知	需要运维动作
	增加备份	无感知	需要运维动作
	多租户使用	隔离	可能会相互影响
费用	公网采集（10GB/天）	2 元/天	16.1 元/天
	公网采集（1TB/天）	162 元/天	800 元/天
	内网采集（数据量小）	-	-
	内网采集（数据量中）	-	-
	内网采集（数据量大）	-	-

日志存储与查询引擎

关注点		LogSearch	ES（Lucene Based)	NoSQL	Hive
规模	规模	PB	TB	PB	PB
成本	存储（元/GB *天）	0.0115	3.6	0.02	0.035
	写入（元/GB）	0.35	5	0.4	0
	查询（元/GB）	0	0	0.2	0.3
	速度-查询	毫秒级-秒级	毫秒级-秒级	毫秒级	分钟级
	速度-统计	弱+	较强	弱	强
延时	写入->可查询	实时	分钟级	实时	十分钟级

说明:

此处价格对比主要基于ECS上部署软件，并且设置为3份副本后的计算结果。

更多内容请参考：#unique_8。

3.3 查询分析全方位对比（ELK）

通过将阿里云日志服务与ELK Stack进行全面对比，帮助您更好的了解阿里云日志服务的主要功能和优势。

背景信息

提到日志实时分析，很多人都会想到基于ELK Stack（Elastic/Logstash/Kibana）来搭建。ELK方案开源，在社区中有大量的内容和使用案例。

阿里云 日志服务 是阿里巴巴集团对日志场景的解决方案产品，前身是2012年初阿里云在研发飞天操作系统过程中用来监控+问题诊断的产物，但随着用户增长与产品发展，慢慢开始向面向Ops（DevOps，Market Ops，SecOps）日志分析领域发展，期间经历双十一、蚂蚁双十二、新春红包、国际业务等场景挑战，成为同时服务国内外的产品。

面向日志分析场景

Apache Lucene是Apache软件基金会一个开放源代码的全文检索引擎工具包，一个全文检索引擎的架构，提供了完整的查询引擎和索引引擎、部分文本分析引擎。2012年Elastic把Lucene基础库包成了一个更好用的软件，并且在2015年推出ELK Stack（Elastic Logstash Kibana）解决集中式日志采集、存储和查询问题。Lucene设计场景是Information Retrial，面对是Document类型，因此对于Log这种数据有一定限制，例如规模、查询能力、以及智能聚类LogReduce等定制化功能。

LOG提供日志存储引擎是阿里内部自研究技术，经过3年万级应用锤炼，每日索引数据量达PB级，服务万级开发者每天亿次查询分析。在阿里集团内阿里云全站，SQL审计、鹰眼、蚂蚁云图、飞猪Tracing、阿里云谛听等都选择LOG作为日志分析引擎。

而日志查询是DevOps最基础需求，业界的调研《50 Most Frequently Used Unix Command》也验证了这一点，tar排名第一、Grep这个命令排名第二，由此可见日志查询对程序员的重要性。

在日志查询分析场景，以如下点对ELK 与 LOG 做一个全方位比较：

- 易用：上手及使用过程中的代价。
- 功能（重点）：主要针对查询与分析。
- 性能（重点）：对于单位大小数据量查询与分析需求，延时如何。
- 规模（重点）：能够承担的数据量，扩展性等。
- 成本（重点）：同样功能和性能，使用分别花多少钱。

易用性

对日志分析系统而言，有如下使用过程：

- 采集：将数据稳定写入。
- 配置：如何配置数据源。
- 扩容：接入更多数据源，更多机器，对存储空间，机器进行扩容。
- 使用：这部分在功能这一节介绍。
- 导出：数据能否方便导出到其他系统，例如做流计算、放到对象存储中进行备份。
- 多租户：如何将数据分享给其他人使用，使用是否安全等。

以下是比较结果：

项目	分项	自建ELK	Log Analytics
采集	协议	Restful API	· Restful API · JDBC
	客户端	Logstash/Beats/FluentD，生态十分丰富	· Logtail（为主） · 其他（例如Logstash）
配置	单元	提供Index概念用以区分不同日志	· Project · Logstore 提供两层概念，Project相当于命名空间，可以在Project下建立多个Logstore。
	属性	API + Kibana	· API + SDK · 控制台
扩容	存储	· 增加机器 · 购买云盘	无需操作
	计算	新增机器	无需操作

项目	分项	自建ELK	Log Analytics
	配置	· 通过配管系统应用机器 · （Logstash在Beta版本中已经提供中心化配置功能）	控制台/API 操作，无需配管系统
	采集点	通过配管系统控制，将配置和Logstash安装到机器组	控制台/API 操作，无需配管系统
	容量	不支持动态扩容	动态扩容、弹性伸缩
导出	方式	· API · SDK	· API · SDK · 类Kafka接口消费 · 各流计算引擎消费（Spark, Storm，Flink） · 流计算类库消费（Python、Java）
多租户	安全	商业版	· https · 传输签名 · 多租户隔离 · 访问控制
	流控	无流控	· Project级 · Shard级
	多租户	Kibana支持	原生提供账号与权限级管理

整体而言：

· ELK有非常多的生态和写入工具，安装、配置等都有较多工具可以使用。

· LOG是托管服务，从接入、配置、使用上集成度非常高，普通用户5分钟就可以接入。

· LOG是SaaS化服务，在过程中不需要担心容量、并发等问题，弹性伸缩，免运维。

功能（查询+分析）

查询主要是将符合条件的日志快速命中，分析功能是对数据进行统计与计算。

例如我们需要所有状态码大于200的读请求，根据IP统计次数和流量。这样的分析请求可以转化为两种操作：

- 查询到指定结果，对结果进行统计分析。
- 不进行查询，直接对所有日志进行分析。

```
1. Status in (200,500] and Method:Get*
2. select count(1) as c, sum(inflow) as sum_inflow, ip group by Ip
```

- 查询基础对比

  该对比基于Elastic 6.5 Indices。

类型	子类	ELK	LOG
文本	索引查询	支持	支持
	分词	支持	支持
	中文分词	支持	支持
	前缀	支持	支持
	后缀	支持	-
	模糊	支持	可通过SQL支持
	Wildcast	支持	可通过SQL支持
数值	long	支持	支持
	double	支持	支持
Nested	Json	支持	-
Geo	Geo	支持	可通过SQL支持
IP	IP查询	支持	可通过SQL支持

对比结论

- ES 支持的数据类型丰富度，原生查询能力比LOG更完整。
- LOG 能够通过SQL方式（如下）来代替字符串模糊查询，Geo等比较函数，但性能会比原生查询稍差。

```
子串命中
* | select content where content like '%substring%' limit 100

正则表达式匹配
* | select content where regexp_like(content, '\d+m') limit 100

JSON内容解析与匹配
* | select content where json_extract(content, '$.store.book')='
mybook' limit 100

如果设置json类型索引也可以使用：
```

```
field.store.book='mybook'
```

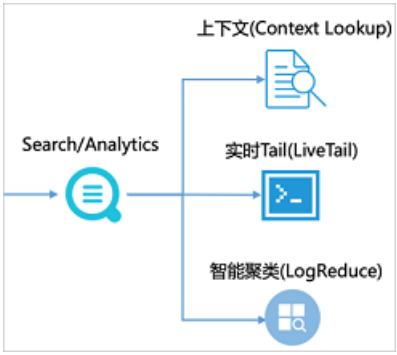
- 查询扩展能力

  在日志分析场景中，光有检索还不够，需要能够围绕查询做进一步的工作：

  - 定位到错误日志后，想通过上下文查看是什么参数引起了错误。
  - 定位到错误后，想看看之后有没有类似错误，类似tail -f 原始日志文件，并进行grep。
  - 通过关键词搜索到大量日志（例如百万条），其中90%都是已知问题干扰调查线索。

  LOG 针对以上问题提供闭环解决方案：

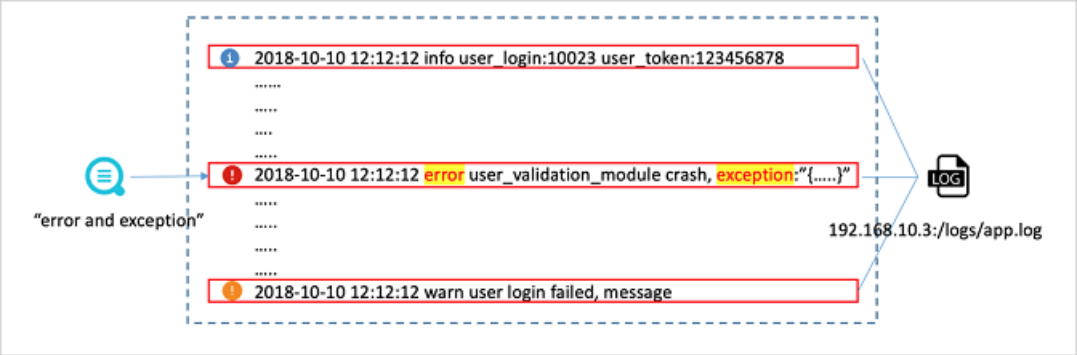
  - 上下文查询（Context Lookup）：原始上下文翻页，免登服务器。
  - LiveTail功能（Tail-f）：原始上下文tail-f，更新实时情况。
  - 智能聚类（LogReduce）：根据日志Pattern动态归类，合并重复模式，洞察异常。



1. #unique_10功能

   在传统的运维方式中，如果需要对日志文件进行实时监控，需要到服务器上对日志文件执行`tail -f`命令，如果实时监控的日志信息不够直观，可以加上`grep`或者`grep -v`进行关键词

过滤。LOG在控制台提供了日志数据实时监控的交互功能LiveTail，针对线上日志进行实时监控分析，减轻运维压力。



Livetail特点如下：

- 智能支持Docker、K8S、服务器、Log4J Appender等来源数据。
- 监控日志的实时信息，标记并过滤关键词。
- 日志字段做分词处理，以便查询包含分词的上下文日志。

2. 智能聚类（LogReduce）

业务的高速发展对系统稳定性提出了更高的要求，各个系统每天产生大量的日志，存在如下担忧：

- 系统有潜在异常，但被淹没在海量日志中。
- 机器被入侵，有异常登录，却后知后觉。
- 新版本上线，系统行为有变化，却无法感知这些问题，是因为信息太多、太杂，不能良好归类，同时记录信息的日志，往往还都是无主题，格式多样，归类难度更大。LOG提供实时日志智能聚类（LogReduce）功能根据日志的相似性进行归类，快速掌握日志全貌：

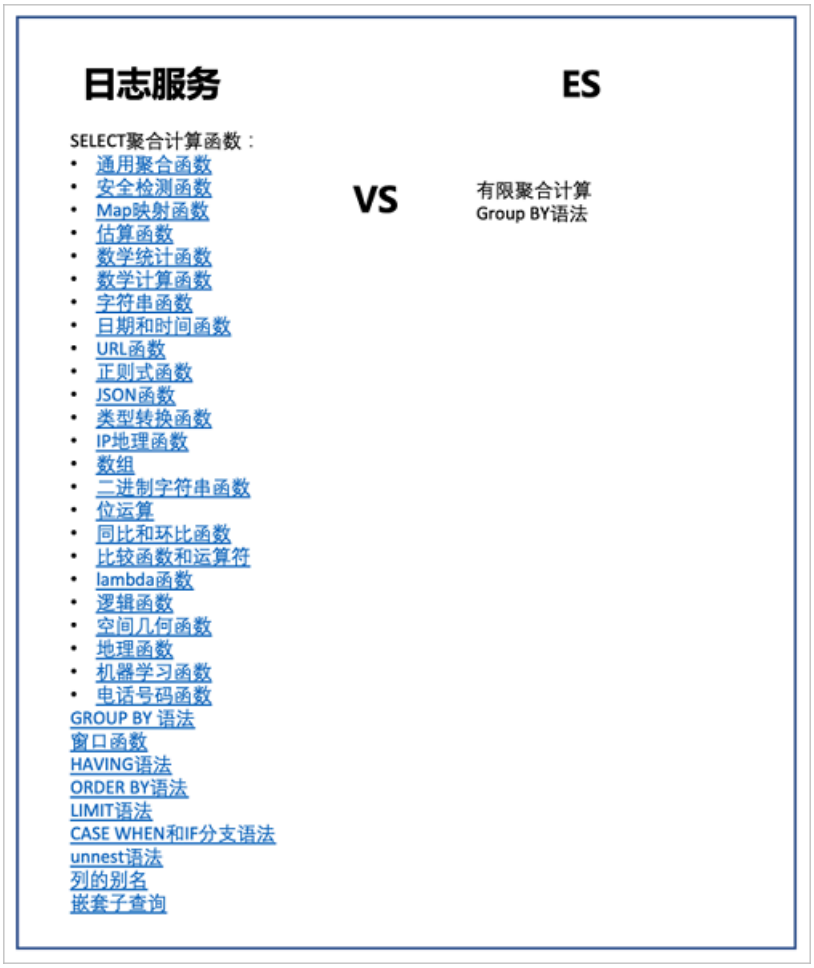
  ■ 支持任意格式日志：Log4J、Json、单行（syslog）。

- 日志经任意条件过滤后再Reduce；对日志Reduce后Pattern，根据signature反查原始数据。
- 不同时间段Pattern比较。
- 动态调整Reduce精度。
- 亿级数据，秒级出结果。

分析能力对比

ES在docvalue之上提供一层聚合（Aggregation）语法，并且在6.x版本中提供SQL语法能够对数据进行分组聚合运算。 LOG支持完整SQL92标准（提供restful 和 jdbc两种协议），除基本聚合功能外，支持完整的SQL计算，并支持外部数据源联合查询（Join），机器学习，模式分析等函数。

> **说明:**
> 以下分析基于ES 6.5 Aggregation和LOG分析语法。



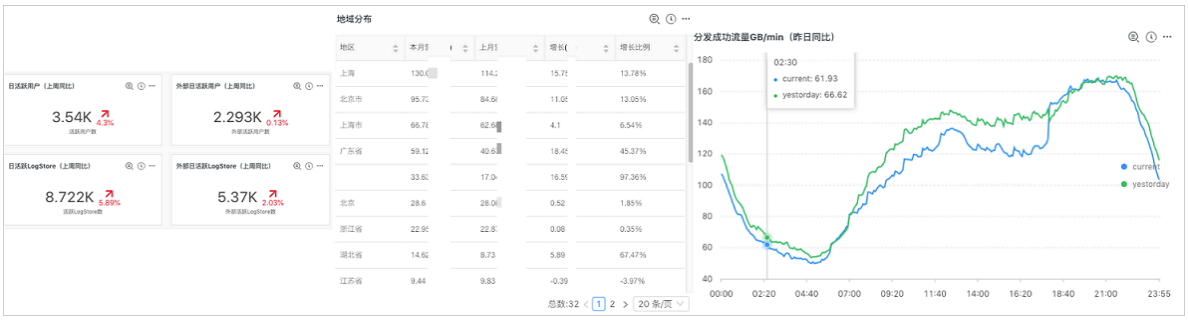
除SQL92标准语法外，我们根据实际日志分析需求，研发一系列实用的功能：

- #unique_13

  同比环比函数能够通过SQL嵌套对任意计算（单值、多值、曲线）计算同环比（任意时段），以便洞察增长趋势。

```
* | select compare( pv , 86400) from (select count(1) as pv from log
)
```

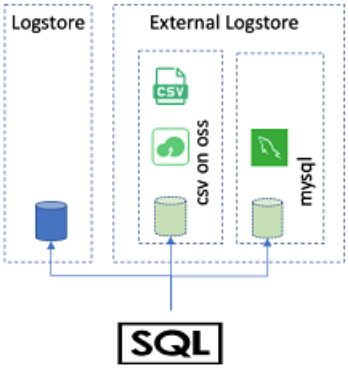
```
*|select t, diff[1] as current, diff[2] as yestoday, diff[3] as
percentage from(select t, compare( pv , 86400) as diff from (select
 count(1) as pv, date_format(from_unixtime(__time__), '%H:%i') as t
from log group by t) group by t order by t) s
```



- 外部数据源联合查询（Join）

  可以在查询分析中关联外部数据源

  - 支持Logstore，MySQL，OSS（CSV）等数据源。
  - 支持left，right，out，innerjoin。
  - SQL查询外表，SQLJoin外表。



  Join外表的样例：

```
sql
创建外表：
* | create table user_meta ( userid bigint, nick varchar, gender
 varchar, province varchar, gender varchar,age bigint) with (
endpoint='oss-cn-hangzhou.aliyuncs.com',accessid='LTA288',accesskey
 ='EjsowA',bucket='testossconnector',objects=ARRAY['user.csv'],type
='oss')

使用外表
* | select u.gender, count(1) from chiji_accesslog l join user_meta1
 u on l.userid = u.userid group by u.gender
```

- 地理位置函数

  针对IP地址、手机等内容，内置地理位置函数方便分析用户来源，包括：

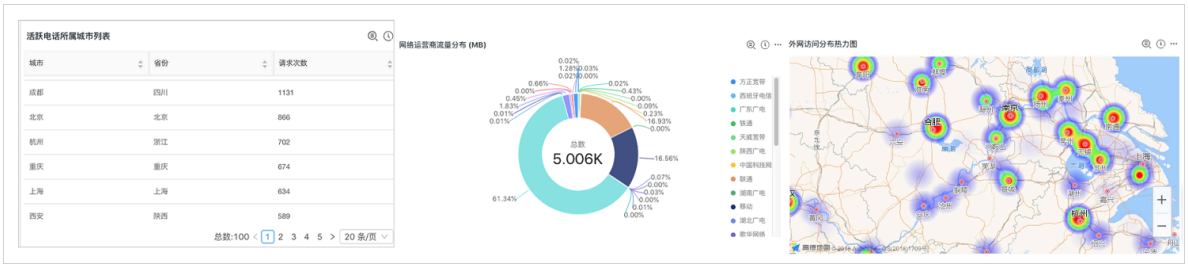
  - IP：国家、省市、城市、经纬度、运营商。
  - Mobile：运营商、省市。
  - GeoHash：Geo位置与坐标转换。

  查询结果分析样例：

```
sql
* | SELECT count(1) as pv, ip_to_province(ip) as province WHERE
ip_to_domain(ip) != 'intranet' GROUP BY province ORDER BY pv desc
limit 10
```

```
* | SELECT mobile_city(try_cast("mobile" as bigint)) as "城市",
mobile_province(try_cast("mobile" as bigint)) as "省份", count(1) as
 "请求次数" group by "省份", "城市" order by "请求次数" desc limit 100
```

查询结果：



- GeoHash
- 电话号码
- IP函数

· 安全分析函数

依托全球白帽子共享安全资产库，提供安全检测函数，您只需要将日志中任意的IP、域名或者URL传给安全检测函数，即可检测是否安全。

- security_check_ip
- security_check_domain
- security_check_url

- 机器学习与时序检测函数

  新增机器学习与智能诊断系列函数：

  - 根据历史自动学习其中规律，并对未来的走势做出预测。
  - 实时发现不易察觉的异常变化，并通过分析函数组合推理导致异常的特征。
  - 结合环比、告警功能智能发现/巡检。该功能适用在智能运维、安全、运营等领域，帮助更快、更有效、更智能洞察数据。

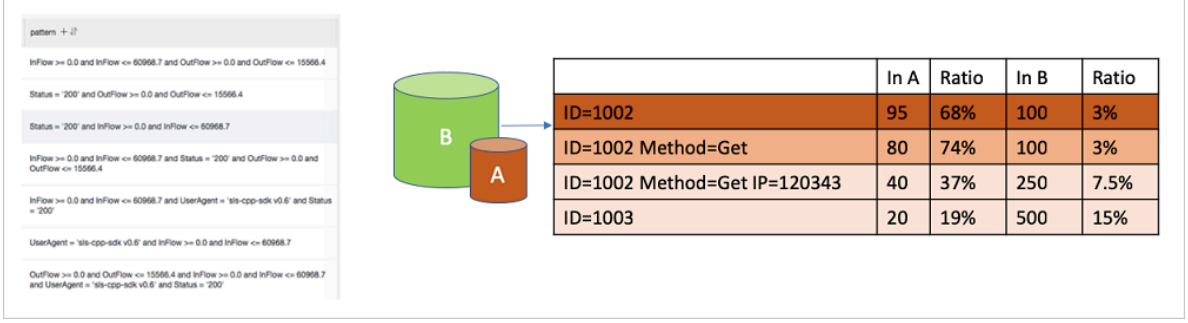
  提供的功能有：

  - 预测：根据历史数据拟合基线。
  - 异常检测\变点检测\折点检测：找到异常点。
  - 多周期检测：发现数据访问中的周期规律。
  - 时序聚类：找到形态不一样的时序。

- 模式分析函数

  模式分析函数能够洞察数据中的特征与规律，帮助快速、准确推断问题：

  - 定位频繁集。例如：错误请求中90%由某个用户ID构成。
  - 定位两个集合中最大支持因素，例如：
    - 延时>10S请求中某个ID构成比例远远大于其他维度组合。
    - 并且该ID在对比集合（B）中的比例较低。
    - A和B中差异明显。



	In A	Ratio	In B	Ratio
ID=1002	95	68%	100	3%
ID=1002 Method=Get	80	74%	100	3%
ID=1002 Method=Get IP=120343	40	37%	250	7.5%
ID=1003	20	19%	500	15%

性能

针对相同数据集，分别对比写入数据及查询，和聚合计算能力。

- 实验环境
  - 测试配置

类别	自建ELK	LOG
环境	ECS 4核16GB * 4台 + 高效云盘或SSD	-
Shard	10	10
拷贝数	2	3（默认配置，对用户不可见）

  - 测试数据
    - 5列double，5列long，5列text，字典大小分别是256，512，768，1024，1280
    - 以上字段完全随机（测试日志样例如下）。
    - 原始数据大小：50 GB。
    - 日志行数：162640232（约为1.6亿条）。

    以上字段完全随机，如下为一条测试日志样例：

```
timestamp:August 27th 2017, 21:50:19.000
long_1:756,444 double_1:0 text_1:value_136
long_2:-3,839,872,295 double_2:-11.13 text_2:value_475
long_3:-73,775,372,011,896 double_3:-70,220.163 text_3:value_3
long_4:173,468,492,344,196 double_4:35,123.978 text_4:value_124
long_5:389,467,512,234,496 double_5:-20,10.312 text_5:value_1125
```

- 写入测试结果

  ES采用bulk api批量写入，LogSearch/Analytics 用PostLogstoreLogs API批量写入，结果如下：

类型	项目	自建ELK	LOG
延时	平均写入延时	40 ms	14 ms
存储	单拷贝数据量	86G	58G
	膨胀率：数据量/原始数据大小	172%	116%

说明：

日志服务产生计费的存储量包括压缩的原始数据写入量（23G）和索引流量27G，共50G存储费用。

从测试结果来看

- 日志服务写入延时好于ES，40ms vs 14 ms。
- 空间：原始数据50G，因测试数据比较随机所以存储空间会有膨胀（大部分真实场景下，存储因压缩会比原始数据小）。ES胀到86G，膨胀率为172%，在存储空间超出日志服务 58 %。这个数据与ES推荐的存储大小为原始大小2.2倍比较接近。

· 读取（查询+分析）测试

- 测试场景

  选取两种比较常见的场景：日志查询和聚合计算。分别统计并发度为1，5，10时，两种case的平均延时。

  ■ 针对全量数据，对任意text列计算group by，计算5列数值的avg/min/max/sum/count，并按照count排序，取前1000个结果，例如：

  ```
  select count(long_1) as pv,sum(long_2),min(long_3),max(long_4),
  sum(long_5)
    group by text_1 order by pv desc limit 1000
  ```

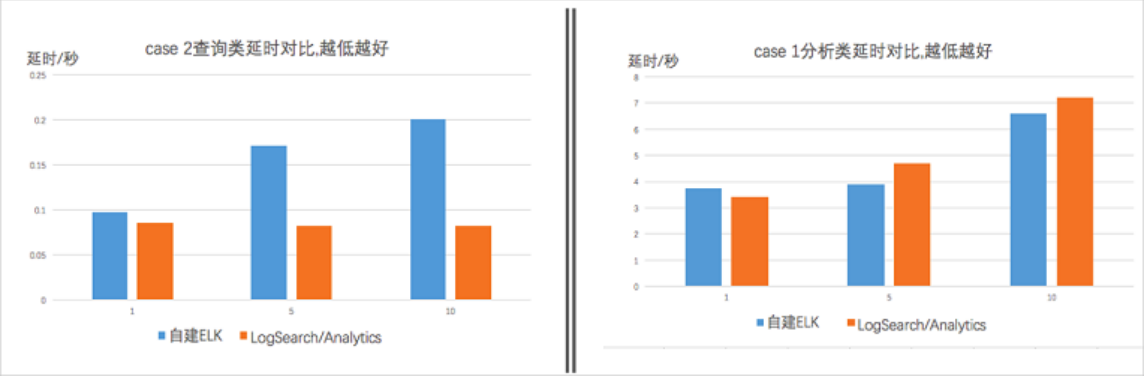
  ■ 针对全量数据，随机查询日志中的关键词，例如查询“value_126”，获取命中的日志数目与前100行，例如：

  ```
  value_126
  ```

- 测试结果

类型	并发数	ES延时（单位为秒）	日志服务延时（单位为秒）
case1：分析类	1	3.76	3.4
	5	3.9	4.7
	10	6.6	7.2
case2：查询类	1	0.097	0.086
	5	0.171	0.083

类型	并发数	ES延时（单位为秒）	日志服务延时（单位为秒）
	10	0.2	0.082



- 结果分析

  - 从结果看，对于1.6亿数据量这个规模，两者都达到了秒级查询与分析能力。
  - 针对统计类场景（case 1），ES和日志服务延时处同一量级。ES采用SSD云盘，在读取大量数据时IO优势比较高。
  - 针对查询类场景（case 2），LogAnalytics在延时明显优于ES。随着并发的增加，ELK延时对应增加，而LogAnalytics延时保持稳定甚至略有下降。

规模与成本

- 规模能力
  - 日志服务一天可以索引PB级数据，一次查询可以在秒级过几十TB规模数据，在数据规模上可以做到弹性伸缩与水平扩展。
  - ES比较适合服务场景为：写入GB-TB/Day、存储在TB级。主要受限于2个原因：
    - 单集群规模：比较理想为20台左右，据了解业界比较大为100节点一个集群，为了应对业务往往拆成多个集群。
    - 写入扩容：shard创建后便不可再修改，当吞吐率增加时，需要动态扩容节点，最多可使用的节点数便是shard的个数。
    - 存储扩容：主shard达到磁盘的上限时，要么迁移到更大的一块磁盘上，要么只能分配更多的shard。一般做法是创建一个新的索引，指定更多shard，并且rebuild旧的数据。

用户案例（规模带来的问题）

客户A是国内最大资讯类网站之一，有数千台机器与百号开发人员。运维团队原先负责一套ELK集群用来处理Nginx日志，但始终处于无法大规模使用状态：

- 一个大Query容易把集群打爆，导致其他用户无法使用。
- 在业务高峰期间，采集与处理能力打满集群，造成数据丢失，查询结果不准确。
- 业务增长到一定规模，因内存设置、心跳同步等节点经常内存失控导致OOM 不能保证可用性与准确性，开发最终没有使用起来，成为一个摆设。

在2018年6月份，运维团队开始运行日志服务方案：

1. 使用Logtail来采集线上日志，将采集配置，机器管理等通过API集成进客户自己运维与管控系统。
2. 将日志服务查询页面嵌入统一登录与运维平台，进行业务与账户权限隔离。
3. 通过控制台内嵌方案满足开发查询日志需求，通过Grafana插件调用日志服务统一业务监控，通过DataV连接日志服务进行大盘搭建。

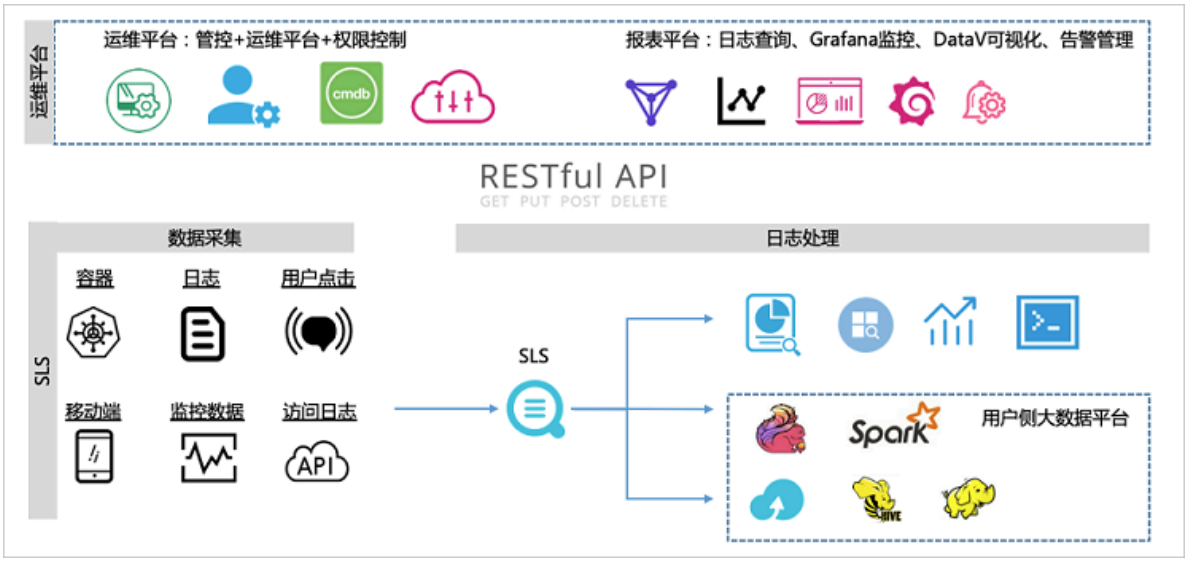
说明：

详细说明请参见文档：

- #unique_20
- #unique_21
- #unique_22

- #unique_23

整体架构如下图：



平台上线2个月后：

- 每天查询的调用量大幅上升，开发逐步开始习惯在运维平台进行日志查询与分析，提升了研发的效率，运维部门也回收了线上登录的权限。
- 除Nginx日志外，把App日志、移动端日志、容器日志也进行接入，规模是之前10倍。
- 除查询日志外，也衍生出很多新的玩法，例如通过Jaeger插件与控制台基于日志搭建了Trace系统，将线上错误配置成每天的告警与报表进行巡检。
- 通过统一日志接入管理，规范了各平台对接总线，不再有一份数据同时被采集多次的情况，大数据部门Spark、Flink等平台可以直接去订阅实时日志数据进行处理。

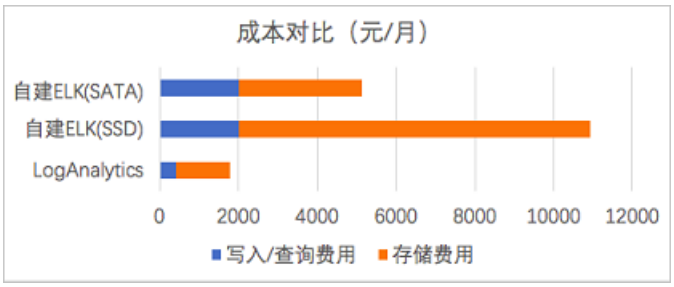
· 成本

以上述测试数据为例，一天写入50GB数据（其中23GB 为实际的内容），保存90天，平均一个月的耗费。

- 日志服务（LogSearch/LogAnalytics）计费规则参见#unique_24，包括读写流量、索引流量、存储空间等计费项，查询功能免费。

计费项目	值	单价	费用（元）
读写流量	23G * 30	0.2 元/GB	138
存储空间（保存90天）	50G * 90	0.3 元/GB*Month	1350
索引流量	27G * 30	0.35 元/GB	283

计费项目	值	单价	费用（元）
总计	-	-	1771



- ES费用包括机器费用，及存储数据SSD云盘费用

  ■ 云盘一般可以提供高可靠性，因此我们这里不计费副本存储量。

  ■ 存储盘一般需要预留15%剩余空间，防止空间写满，因此乘以一个1.15系数。

计费项目	值	单价	费用（元）
服务器	4台4核16G（三个月）（ecs.mn4.xlarge）	包年包月费用：675 元/Month	2021
存储	86 * 1.15 * 90 （这里只计算一个副本）	SSD：1 元/GB*M	8901
	-	SATA：0.35 元/GB*M	3115
总计			12943 （SSD）
			5135 （SATA）

同样性能，使用LogSearch/Analytics与ELK（SSD）费用比为 13.6%。在测试过程中，我们也尝试把SSD换成SATA以节省费用（LogAnalytics与SATA版费用比为 34%），但测试发现延时会从40ms上升至150ms，在长时间读写下，查询和读写延时变得很高，无法正常工作了。

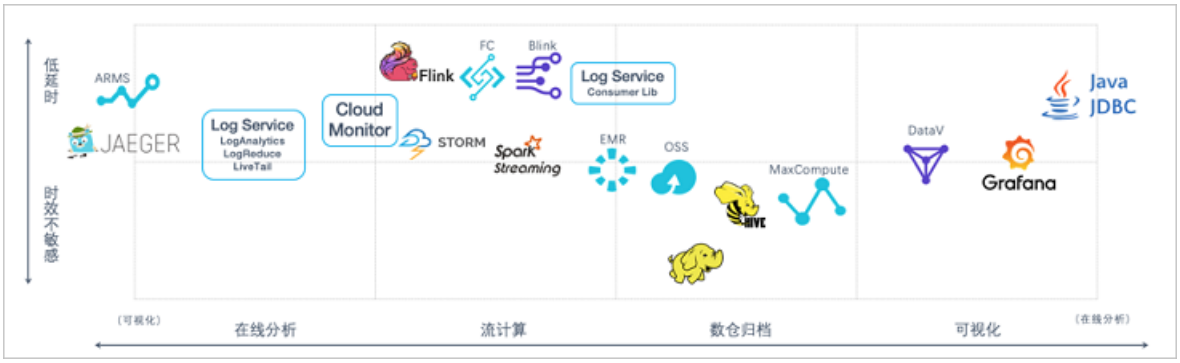
· 时间成本（Time to Value）

除硬件成本外，日志服务在新数据接入、搭建新业务、维护与资源扩容成本基本为0。

- 支持各种日志处理生态，可以和Spark、Hadoop、Flink、Grafana等系统无缝对接。
- 在全球化部署（有20+ Region），方便拓展全球化业务。
- 提供30+日志接入SDK，与阿里云产品无缝打通集成。

日志服务采集和可视化可以参见如下文章，非核心功能不展开做比较。

- 采集
- 可视化



总结

ES支撑更新、查询、删除等更通用场景，在搜索、数据分析、应用开发等领域有广泛使用，ELK组合在日志分析场景上把ES灵活性与性能发挥到极致；日志服务是纯定位在日志类数据分析场景的服务，在该领域内做了很多定制化开发。一个服务更广，一个场景更具针对性。当然离开了场景纯数字的比较没有意义，找到适合自己场景的才重要。

深入了解日志服务：

· 日志服务产品
· 了解更多

3.4 查询方案（ELK/Hive）对比

DevOps 场景日志查询方案对比：ELK（搜索类）、Hadoop/Hive

在互联网大潮中，为应对不断加速的软件、服务交付需求，无论是在创业团队还是大型互联网公司，都已经转向或逐步转向 DevOps 模式，通过开发（Dev）和运维支持（Ops）之间有效协作，解决跨部门协作、快速响应客户需求、进行持续交付。

日志在 DevOps 下显得越发重要，无论是在问题调查、安全审计、运营支撑等各方面，日志都起到了重要的支撑作用。一个合适的日志解决方案，对于 DevOps 显得尤为重要。

我们从如下方面考察 LogSearch 与 ELK、Hadoop/Hive 类方案的对比：

- 延时：日志产生后，多久可查询
- 查询能力：单位时间扫描数据量
- 查询功能：关键词查询、条件组合查询、模糊查询、数值比较、上下文查询
- 弹性：快速应对百倍流量上涨
- 成本：每 GB 费用
- 可靠性：日志数据安全不丢失

常用方案以及对比

- 自建 ELK：通过 Elastic、Logstash、Kibana 进行对比
- 离线 Hadoop + Hive：将数据存储在 Hadoop，利用 Hive 或 Presto 进行查询（非分析）
- 使用日志服务（LogSearch）

以应用程序日志和 Nginx 访问日志为例（每天 10GB），对比几种方案。

功能项	ELK 类系统	Hadoop + Hive	日志服务
可查延时	1~60 秒(由 refresh_interval 控制)	几分钟~数小时	实时
查询延时	小于 1 秒	分钟级	小于 1 秒
超大查询	几十秒~数分钟	分钟级	秒级（查询 10 亿日志）
关键词查询	支持	支持	支持
模糊查询	支持	支持	支持
#unique_26	不支持	不支持	支持
数值比较	支持	支持	支持
连续字符串查询	支持	支持	不支持
弹性	提前预备机器	提前预备机器	秒级 10 倍扩容
写入成本	写入5 元/GB，查询免费	写入免费，查询一次 0.3 元/GB	写入 0.5 元/GB，查询免费
存储成本	≤ 3.36 元/GB*天	≤ 0.035 元/GB*天	≤ 0.016 元/GB*天
可靠性	设置拷贝数	设置拷贝数	SLA > 99.9%，数据 > 99.99999999%

4 应用场景

日志服务的典型应用场景包括：数据采集与消费、数据清洗与流计算 （ETL/Stream Processing ）、数据仓库对接（Data Warehouse）、日志实时查询与分析。

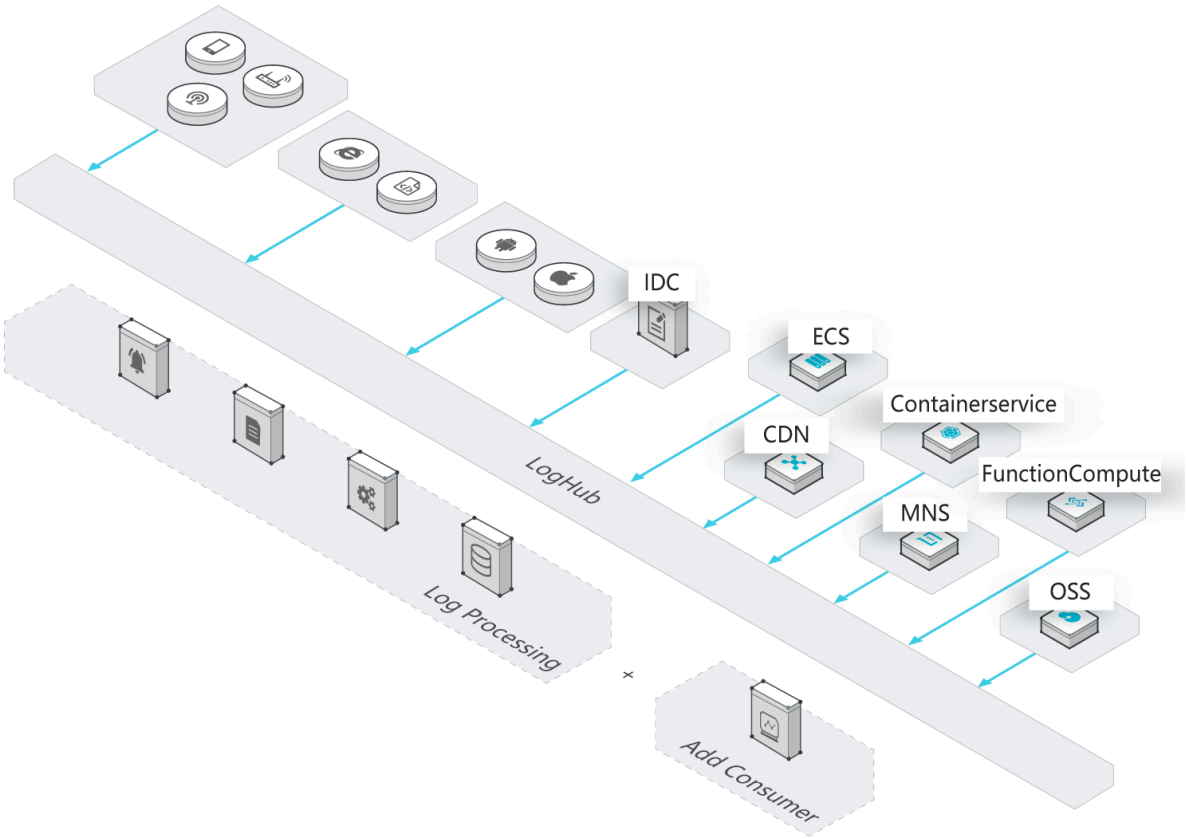
数据采集与消费

通过日志服务LogHub功能，可以大规模低成本接入各种实时日志数据（包括Metric、Event、BinLog、TextLog、Click等）。

方案优势：

- 使用便捷：提供30+实时数据采集方式，让您快速搭建平台；强大配置管理能力，减轻运维负担。
- 弹性伸缩：无论是流量高峰还是业务增长都能轻松应对。

图 4-1: 数据采集与消费

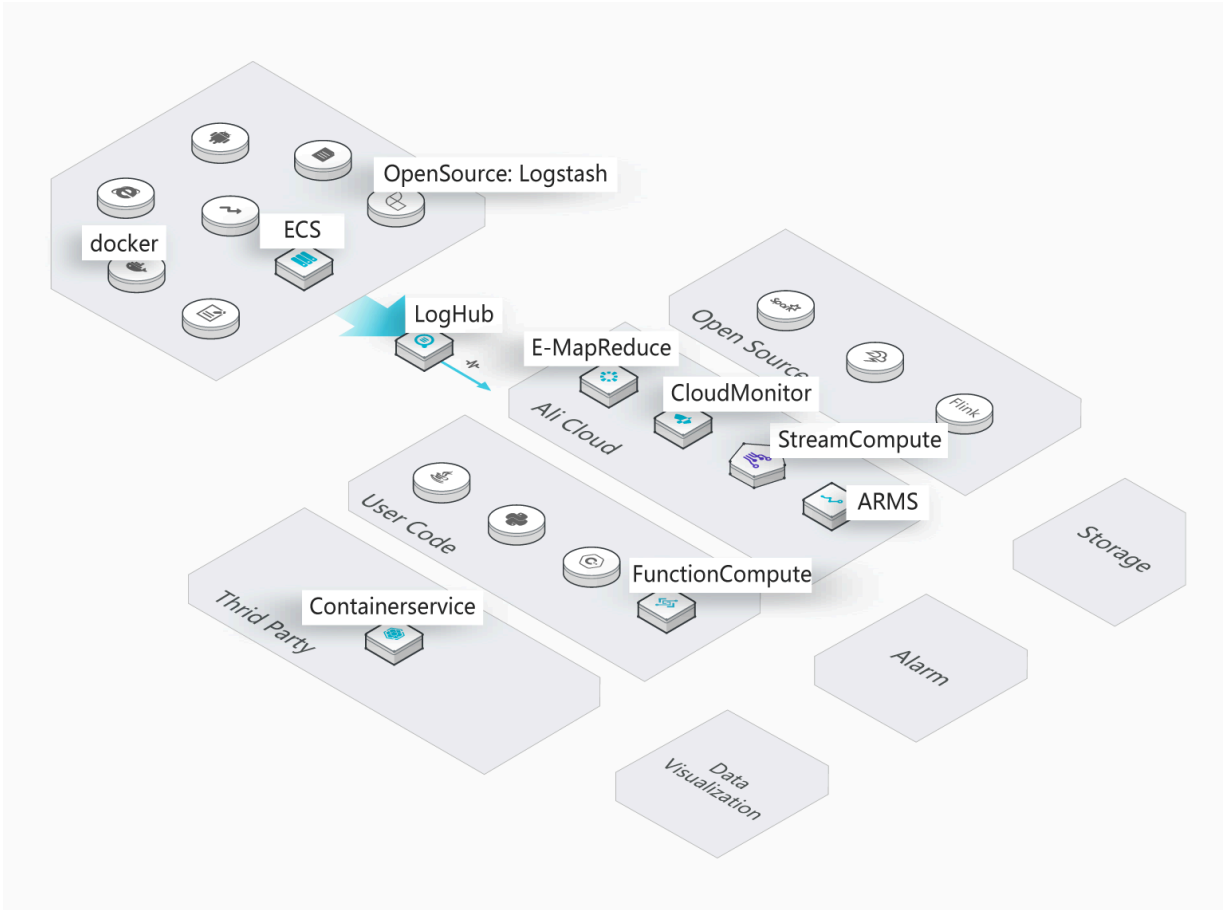


数据清洗与流计算 （ETL/Stream Processing）

日志中枢（LogHub）支持与各种实时计算及服务对接，并提供完整的进度监控，报警等功能，并可以根据SDK/API实现自定义消费。

- 操作便捷：提供丰富SDK以及编程框架，与各流计算引擎无缝对接。
- 功能完善：提供丰富监控数据，以及延迟报警机制。
- 弹性伸缩：PB级弹性能力，0延迟。

图 4-2: 数据清洗与流计算



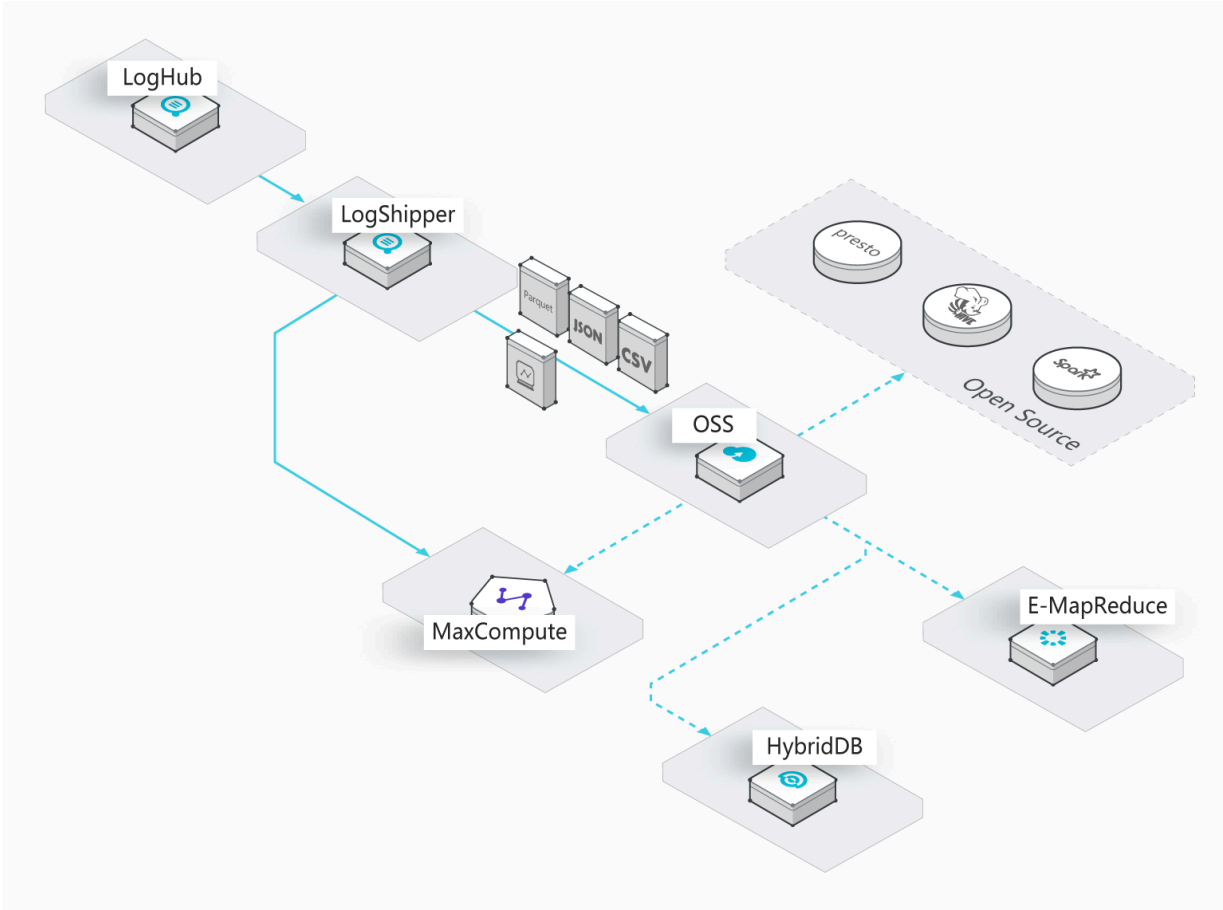
数据仓库对接（Data Warehouse）

日志投递（LogShipper）功能可以将日志中枢（LogHub）中数据投递至存储类服务，过程支持压缩、自定义Partition、以及行列等各种存储格式。

- 海量数据：对数据量不设上限。
- 种类丰富：支持行、列、TextFile等各种存储格式。

· 配置灵活：支持用户自定义Partition等配置。

图 4-3: 数据仓库对接



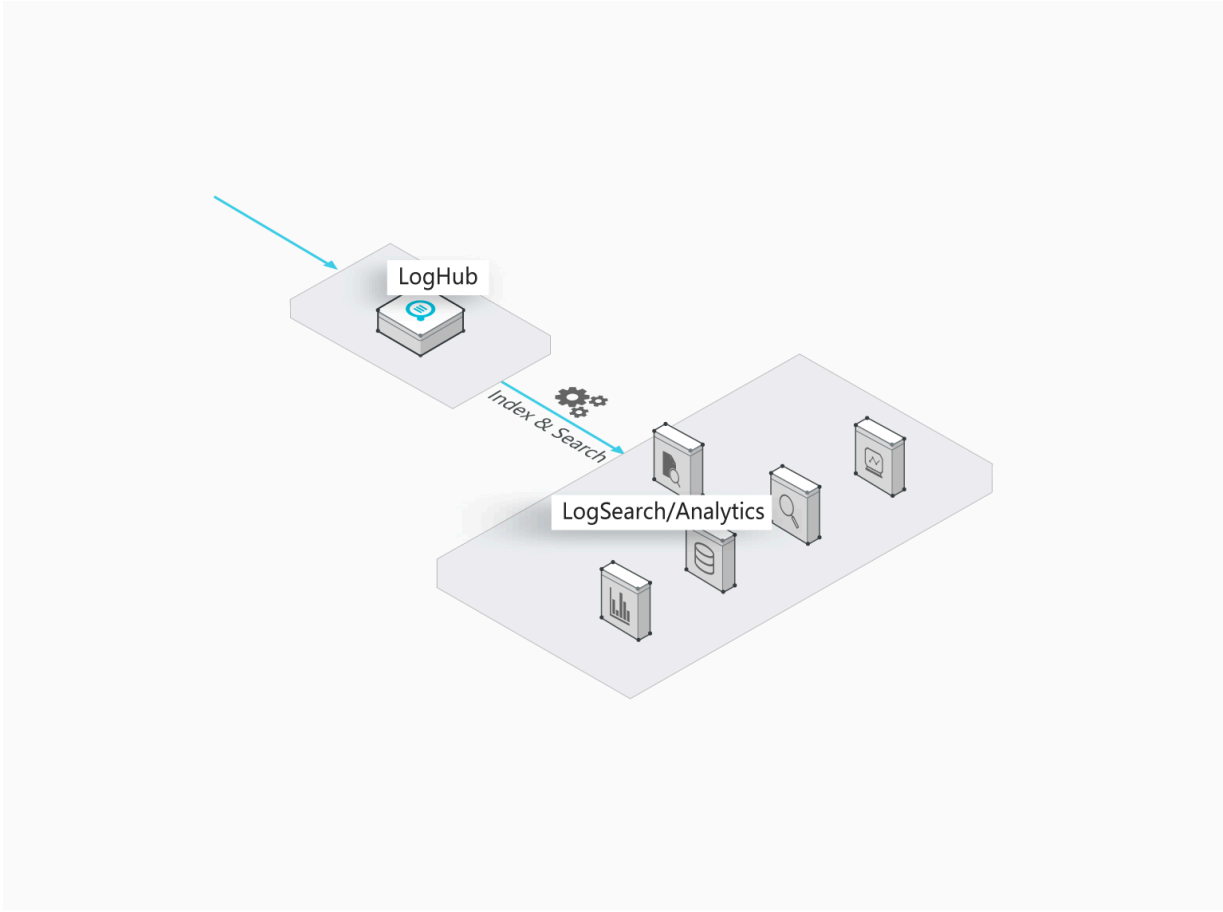
日志实时查询与分析

实时查询分析（LogAnalytics）可以实时索引LogHub中数据，提供关键词、模糊、上下文、范围、SQL聚合等丰富查询手段。

· 实时性强：写入后即可查询。
· 海量低成本：支持PB/Day索引能力，成本为自建方案15%。

· 分析能力强：支持多种查询手段，及SQL进行聚合分析，并提供可视化及报警功能。

图 4-4: 日志实时查询与分析



5 基本概念

5.1 简介

日志（Log）

日志（Log）是系统在运行过程中变化的一种抽象，其内容为指定对象的某些操作和其操作结果按时间的有序集合。文件日志（LogFile）、事件（Event）、数据库日志（BinLog）、度量（Metric）数据都是日志的不同载体。在文件日志中，每个日志文件由一条或多条日志组成，每条日志描述了一次单独的系统事件，是日志服务中处理的最小数据单元。

日志组（LogGroup）

日志组（LogGroup）是一组日志的集合，写入与读取的基本单位。

日志主题（Topic）

一个日志库内的日志可以通过日志主题（Topic）来划分。用户可以在写入时指定日志主题，并在查询时指定查询的日志主题。

项目（Project）

项目（Project）是日志服务中的资源管理单元，用于资源隔离和控制。您可以通过项目来管理某一个应用的所有日志及相关的日志源。它管理着用户的所有日志库（Logstore），采集日志的机器配置等信息，同时它也是用户访问日志服务资源的入口。

日志库（Logstore）

日志库（Logstore）是日志服务中日志数据的收集、存储和查询单元。每个日志库隶属于一个项目，且每个项目可以创建多个日志库。

分区（Shard）

每个日志库分若干个分区（Shard），每个分区由MD5左闭右开区间组成，每个区间范围不会相互覆盖，并且所有的区间的范围是MD5整个取值范围。

5.2 日志

日志（Log）是系统在运行过程中变化的一种抽象，其内容为指定对象的某些操作和其操作结果按时间的有序集合。文件日志（LogFile）、事件（Event）、数据库日志（BinLog）、度量（Metric）数据都是日志的不同载体。在文件日志中，每个日志文件由一条或多条日志组成，每条日志描述了一次单独的系统事件，是日志服务中处理的最小数据单元。

日志服务采用半结构数据模式定义一条日志。该模式中包含主题（Topic）、时间（Time）、内容（Content）、来源（Source）和标签（Tags）五个数据域。

与此同时，日志服务对日志各字段的格式有不同要求，具体如下表所示：

数据域	含义	格式
主题（Topic）	用户自定义字段，用以标记一批日志。例如访问日志可根据不同站点进行标记。	包括空字符串在内的任意字符串，长度不超过128字节。默认情况下，该字段为空字符串。
时间（Time）	日志中的保留字段，用以表示日志产生的时间，一般由日志中的时间信息直接提取生成。	整型，Unix标准时间格式。单位为秒，表示从1970-1-1 00:00:00 UTC计算起的秒数。
内容（Content）	用以记录日志的具体内容。内容部分由一个或多个内容项组成，每一个内容项为一个Key-Value对。	Key为UTF-8编码字符串，包含字母、下划线和数字，且不以数字开头。长度不超过128字节。不可以使用如下关键字： · `__time__` · `__source__` · `__topic__` · `__partition_time__` · `_extract_others_` · `__extract_others__` Value为任意字符串，长度不超过1024*1024字节。
来源（Source）	日志的来源地，例如产生该日志机器的IP地址。	任意字符串，长度不超过128字节。默认情况下该字段为空。
标签（Tags）	日志的标签，包括： · 用户自定义标签：您通过API #unique_37写入数据时添加的标签。 · 系统标签：服务端为您添加的标签，包括`__client_ip__`和`__receive_time__`。	字典格式，Key和Value均为字符串类型。在控制台查询日志时，以`__tag__:`为前缀展示。

实际使用场景中，日志的格式多样。为了帮助理解，以下以一条nginx原始访问日志如何映射到日志服务日志数据模型为例说明。假设用户nginx服务器的IP地址为`10.249.201.117`，以下为该服务器的一条原始日志：

```
10.1.168.193 - - [01/Mar/2012:16:12:07 +0800] "GET /Send?AccessKeyId=
8225105404 HTTP/1.1" 200 5 "-" "Mozilla/5.0 (X11; Linux i686 on x86_64
; rv:10.0.2) Gecko/20100101 Firefox/10.0.2"
```

把该条原始日志映射到日志服务日志数据模型，如下：

数据域	内容	说明
Topic	“”	沿用默认值，即空字符串。
Time	1330589527	日志产生的精确时间,表示从1970-1-1 00:00:00 UTC计算起的秒数。从原始日志中的时间戳转换而来。
Content	Key-Value对	日志具体内容。
Source	“10.249.201.117”	使用服务器IP地址作为日志源。
Tags	无	由用户添加或者服务端添加。

用户可以自己决定如何提取日志原始内容并组合成Key-Value对，例如下表：

Key	Value
ip	“10.1.168.193”
method	“GET”
status	“200”
length	“5”
ref_url	“- “
browser	“Mozilla/5.0 (X11; Linux i686 on x86_64; rv:10.0.2) Gecko/20100101 Firefox/10.0.2”

5.3 日志组

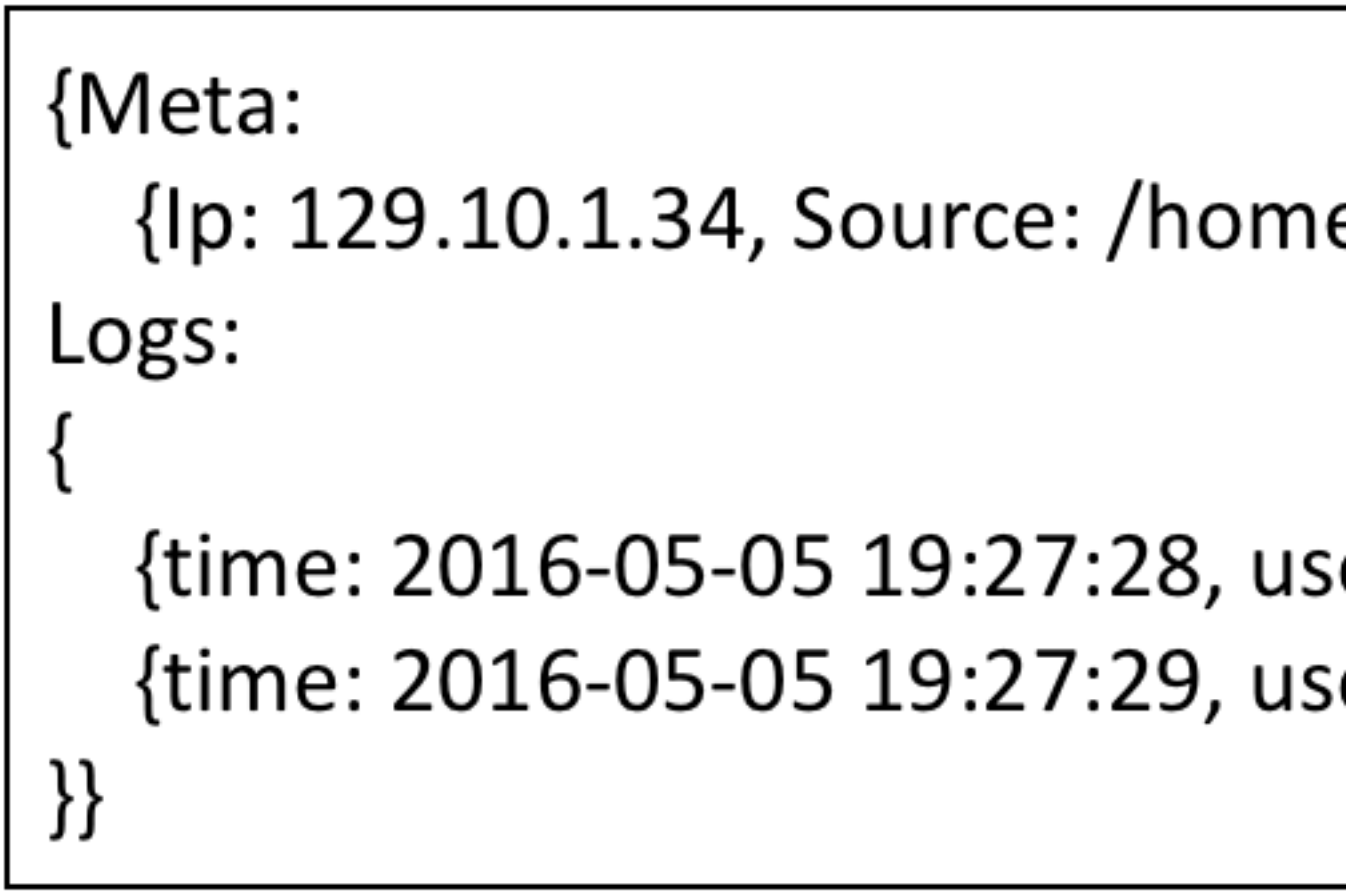
日志组（LogGroup）是一组日志的集合，是API/SDK写入与读取数据的基本单位。一个LogGroup中的数据包含相同Meta（IP、Source等信息）。

通过日志服务API/SDK写入数据时，多条日志会打包成一个LogGroup，以LogGroup为单位发往日志服务。相对于以日志为单位读写数据，LogGroup的读写方式可以最大限度地减少读取与写入次数，提高业务效率。

日志服务的基本数据模型请参考#unique_40/unique_40_Connect_42_section_lfd_woz_kjz。

日志组的限制为：5 MB空间。

图 5-1: 日志组



5.4 项目

项目（Project）是日志服务中的资源管理单元，用于资源隔离和控制。您可以通过项目来管理某一个应用的所有日志及相关的日志源。它管理着用户的所有日志库（Logstore），采集日志的机器配置等信息，同时它也是用户访问日志服务资源的入口。

具体来说，项目可以提供如下功能：

- 帮助您组织、管理不同的Logstore。在实际使用中，您可能需要使用日志服务集中收集、存储不同项目、产品或者环境的日志。您可以把不同项目、产品或者环境的日志分类管理在不同的项目中，方便后续的日志消费、导出或者索引。同时，项目还是日志访问权限管理的载体。
- 提供您日志服务资源的访问入口。每创建一个项目，日志服务会为该项目分配一个独有的访问入口。该访问入口支持通过网络写入、读取及管理日志。

5.5 日志库

日志库（Logstore）是日志服务中日志数据的采集、存储和查询单元。每个日志库隶属于一个项目，且每个项目可以创建多个日志库。您可以根据实际需求为某一个项目生成多个日志库，其中常见的做法是为一个应用中的每类日志创建一个独立的日志库。例如，用户有一个“big-game”游戏应用，服务器上有三种日志：操作日志（operation_log）、应用程序日志（application_log）以及访问日志（access_log），用户可以首先创建名为“big-game”的项目，然后在该项目下面为这三种日志创建三个日志库，分别用于它们的采集、存储和查询。

无论是写入或者查询日志，您都需要指定操作的 Logstore。如果您希望投递日志数据到 MaxCompute 做离线分析，其数据投递也是以 Logstore 为单元进行数据同步，即一个 Logstore 内的日志数据投递到一张 MaxCompute 的 Table。

具体来说，日志库提供如下功能：

- 采集日志，支持实时日志写入
- 存储日志，支持实时消费
- 建立索引，支持日志实时查询
- 提供投递到 MaxCompute 的数据通道

5.6 分区

Logstore读写日志必定保存在某一个分区（Shard）上。每个日志库（Logstore）分若干个分区，每个分区由MD5左闭右开区间组成，每个区间范围不会相互覆盖，并且所有的区间的范围是MD5整个取值范围。

分区范围

创建Logstore时，指定分区个数，会自动平均划分整个MD5的范围。每个分区均有范围，可用MD5方式来表示，且必定包含于以下范围中：[00000000000000000000000000000000,ffffffffffffffffffffffffffffffff）。

分区的范围均为左闭右开区间，由以下Key组成：

- BeginKey：分区起始的Key值，分区范围中包含该Key值。
- EndKey：分区结束的Key值，分区范围中不包含该Key值。

分区的范围用于支持指定Hash Key的模式写入，以及分区的分裂和合并操作。在向分区读写数据过程中，读必须指定对应的分区，而写的过程中可以使用负载均衡模式或者指定Hash Key的模式。负载模式下，每个数据包随机写入某一个当前可用的分区中，在指定Hash Key模式下，数据写入分区范围包含指定Key值的分区。

例如，某Logstore共有4个分区，且该Logstore的MD5取值范围是[00,FF)。各个分区范围如下表所示。

分区号	范围
Shard0	[00,40)
Shard1	[40,80)
Shard2	[80,C0)
Shard3	[C0,FF)

当写入日志时，通过指定Hash Key模式指定一个MD5的Key值是5F，日志数据会写入包含5F的Shard1分区上；如果指定一个MD5的Key值是8C，日志数据会写入包含8C的Shard2分区上。

分区的读写能力

每个分区可提供一定的服务能力：

- 写入：5MB/s，500次/s
- 读取：10MB/s，100次/s

建议您根据实际数据流量规划分区个数，流量超出读写能力时，及时分裂分区以增加分区个数，从而达到更大的读写能力；如您的流量远远达不到分区的最大读写能力时，建议您合并分区以减少分区个数，从而节约分区租赁费用。

例如，如果您有两个readwrite状态的分区，最大可以提供10MB/s的数据写入服务，但如果您实时写入数据流量达到14MB/s，建议分裂其中一个分区，使readwrite分区数量达到3个。如果您实时写入数据流量仅为3MB/s，那么一个分区即可满足需要，建议您合并两个分区。

说明:

- 当写入的API持续报告403或者500错误时，通过Logstore云监控查看流量和状态码判断是否需要增加分区。
- 对超过分区服务能力的读写，系统会尽可能服务，但不保证服务质量。

分区的状态

分区的状态包括：

- readwrite：可以读写
- readonly：只读数据

创建分区时，所有分区状态均为readwrite状态，分裂或合并操作会改变分区状态为readonly，并生成新的readwrite分区。分区状态不影响其数据读取的性能，同时，readwrite分区保持正常的数据写入性能，readonly状态分区不提供数据写入服务。

在分裂分区时，需要指定一个处于readwrite状态的ShardId和一个MD5。MD5要求必须大于分区的BeginKey并且小于EndKey。分裂操作可以从一个分区中分裂出另外两个分区，即分裂后分区数量增加2。在分裂完成后，被指定分裂的原分区状态由readwrite变为readonly，数据仍然可以被消费，但不可写入新数据。两个新生成的分区状态为readwrite，排列在原有分区之后，且两个分区的MD5范围覆盖了原来分区的范围。

在合并操作时，必须指定一个处于readwrite状态的分区，指定的分区不能是最后一个readwrite分区。服务端会自动找到所指定分区的右侧相邻分区，并将两个分区范围合并。在合并完成后，所指定的分区和其右侧相邻分区变成只读（readonly）状态，数据仍然可以被消费，但不能写入新数据。同时新生成一个 readwrite 状态的分区，新分区的MD5范围覆盖了原来两个分区的范围。

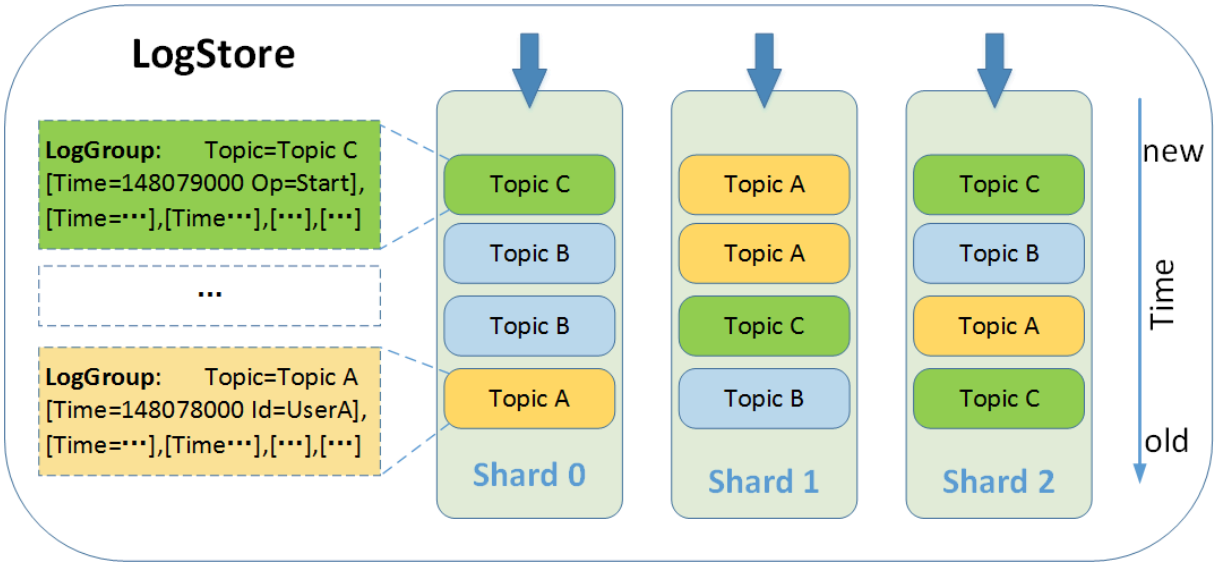
5.7 日志主题

一个日志库内的日志可以通过日志主题（Topic）来划分。您可以在写入时指定日志主题，并在查询时指定查询的日志主题。例如，一个平台用户可以使用用户编号作为日志主题写入日志。这样在查询时可利用日志主题让不同用户仅看到自己的日志。如果不需要划分一个日志库内的日志，让所有日志使用相同的日志主题即可。

说明:

空字符串是一个有效的日志主题（Topic），且无论是写入还是查询日志时，默认的日志主题都是空字符串。所以，如果不需要使用日志主题，最简单的方式就是在写入和查询日志时都使用默认日志主题，即空字符串。

下图描述了日志库、日志主题和日志之间的关系：


LogStore
LogGroup: Topic=Topic C
[Time=148079000 Op=Start],
[Time=…],[Time…],[…],[…]
…
LogGroup: Topic=Topic A
[Time=148078000 Id=UserA],
[Time=…],[Time…],[…],[…]
Topic C
Topic B
Topic B
Topic A
Shard 0
Topic A
Topic A
Topic C
Topic B
Shard 1
Topic C
Topic B
Topic A
Topic C
Shard 2
new
Time
old

6 限制说明

6.1 基础资源

分类	限制说明	备注
Project	每个账号下最多可创建50个Project。	如您有更大的使用需求，请提工单申请。
Logstore	一个Project中最多可创建200个Logstore。	如您有更大的使用需求，请提工单申请。
Shard	· 一个Project中最多可创建200个Shard。 · 控制台创建Logstore时，一个Logstore最多可创建10个Shard；API创建Logstore时，最多可创建100个。但可以通过分裂操作来增加Shard。	如您有更大的使用需求，请提工单申请。
Logtail配置（LogtailConfig）	每个Project最多可创建100个Logtail配置。	如您有更大的使用需求，请提工单申请。
日志保存时间	支持永久保存。 您也可以自定义日志保存时间，取值范围为1~3000。	-
机器组（MachineGroup）	每个Project最多可创建100个机器组。	如您有更大的使用需求，请提工单申请。
协同消费组（ConsumerGroup）	每个Logstore最多可创建10个协同消费组。	可以删除不使用消费组。
快速查询（SavedSearch）	每个Project最多可创建100个快速查询。	-
仪表盘（Dashboard）	· 每个Project最多可创建50个仪表盘。 · 每个仪表盘最多可包含50张分析图表。	-
LogItem	长度最大为1 MB。	以上为API限制参数，如通过Logtail采集日志，单个LogItem最大为 512KB。

分类	限制说明	备注
LogItem（Key）	长度最大为128 Bytes。	-
LogItem（Value）	长度最大为1 MB。	-
日志组（LogGroup）	每个日志组最大长度为5 MB。	-
告警	每个Project最多可以创建100个告警。	如果您有更大的使用需求，请提工单申请。

说明:

通过日志服务命令行工具CLI可以查看当前的Logstore、Shard、仪表盘等基础资源使用情况，详细说明请查看使用CLI查看基础资源使用状况。

6.2 数据读写

分类	限制项	限制说明	备注
Project	写入流量保护	写入流量最大为30 GB/min。	如超过限制，状态码会返回403，提示Inflow Quota Exceed。如您有更大的使用需求，请提工单申请。
	写入次数保护	写入次数最大为600000 次/min。	如超过限制，状态码会返回403，提示Write QPS Exceed。如您有更大的使用需求，请提工单申请。
	读取次数保护	读取次数最大为600000 次/min。	如超过限制，状态码会返回403，提示Read QPS Exceed。如您有更大的使用需求，请提工单申请。
Shard	写入流量	写入流量最大为5 MB/s。	非硬性限制，超过时系统会尽可能服务，但不保证服务质量。
	写入次数	写入次数最大为500 次/s。	非硬性限制，超过时系统会尽可能服务，但不保证服务质量。
	读取流量	读取流量最大为10 MB/s。	非硬性限制，超过时系统会尽可能服务，但不保证服务质量。

分类	限制项	限制说明	备注
	读取次数	读取次数最大为100次/s。	非硬性限制，超过时系统会尽可能服务，但不保证服务质量。

6.3 查询分析与可视化

分类	分类	限制说明	备注
查询（Search）	关键词个数	关键词，即单词查询时布尔逻辑符外的条件个数。每次查询最多30个。	例如"a and b or c and d ..."。
	单个字段长度	单个字段（Value）长度最大为10 KB，超出部分不参与查询。	如果单个字段长度大于10 KB，有一定几率无法通过关键词查询到日志，但数据仍然是完整的。
	单个Project并发	单个Project并发最大为100个。	-
	返回的查询结果条数	每次查询时，默认最多返回100条查询结果。	可以通过翻页读取完整的查询结果。
	单条日志内容显示	由于网页浏览器性能原因，对于超过1w个字符的日志，日志服务只会对前10,000个字符进行DOM切词处理。	如果超出10,000个字符，控制台会提示“该日志存在超过10000个字符的日志数据，部分显示上会有降级处理”。
	模糊查询	模糊查询最多返回100个符合关键词的日志。	-
	查询结果排序	默认按照分钟级时间从最新开始展示。	-
SQL分析（Analytics）	单个字段（Value）最大长度	单个字段（Value）最大长度为2 KB，超出部分不参与查询。	超出限制时查询结果可能不精确，但数据仍然是完整的。
	单个Project并发	单个Project并发不超过15个。	-
	每次分析的返回结果	· 每次分析默认返回100条日志数据。 · 每次分析的返回结果最大100 MB或100,000条。	· 可以通过limit指定返回日志的条数。 · 可以通过翻页方式获取分批读取结果。

分类	分类	限制说明	备注
	字段长度	使用分析语句时，字段长度最大支持2 KB。	分析日志时，超出2 KB的字段会被截断。
	double精度	Double类型最多用52位表示，如果浮点数编码位数超过52位，会有精度损失。	例如，0.1+0.2会表示为0.30000000000000004。建议您使用round(0.1+0.2,2)来取两位有效数值。

6.4 保留字段

日志服务中有部分字段为保留字段，使用API写入数据，或添加Logtail采集配置时，请不要将字段名称设置为日志服务的保留字段。

注意事项

在采集日志或投递数据到其他云产品时，日志服务可以将日志来源、时间戳等信息以Key-Value对的形式添加到日志中，其中字段名称为`__source__`等固定名称，这些字段是日志服务的保留字段。

- 使用API写入日志数据或添加Logtail配置时，请不要将Key即字段名称设置为这些保留字段，否则可能会造成字段名称重复、查询不精确等问题。
- 所有`__tag__`前缀的字段均不支持投递。
- 日志服务为日志数据增加的字段按照#unique_24正常收费，为其开启索引时也会产生少量索引流量及存储费用。

保留字段

目前，日志服务的保留字段包括：

表 6-1: 日志服务保留字段

保留字段名称	数据格式	索引与统计设置	说明
`__time__`	整型，Unix标准时间格式。 例如，`__time__: 1523868463`。	· 索引设置：`__time__`通过API中的参数from和to选择，无需添加该字段的索引。 · 统计设置：当用户为任何一列开启统计后，默认为`__time__`开启统计。	使用API/SDK写入日志数据时指定的日志时间，该字段可用于日志投递、查询、分析。

保留字段名称	数据格式	索引与统计设置	说明
__source__	字符串格式。	· 索引设置：开启索引后，日志服务默认为__source__创建索引，索引数据类型为text类型，分词字符为空。查询时输入source:127.0.0.1或者__source__:127.0.0.1。 · 统计设置：当用户为任何一列开启统计后，默认为__source__开启统计。	日志来源设备。该字段可用于日志投递、查询、分析、自定义消费。
__topic__	字符串格式。	· 索引设置：开启索引后，日志服务默认为__topic__创建索引，索引数据类型为text类型，分词字符为空。查询时输入__topic__:XXX。 · 统计设置：当用户为任何一列开启统计后，默认为__topic__开启统计。	日志主题（Topic）。如果您设置了日志主题，日志服务会自动为您的日志添加日志主题字段，Key为__topic__，Value为您的主题内容。该字段可用于日志投递、查询、分析、自定义消费。
__partition_time__	字符串格式。	日志内容中不存在该字段，无需设置索引。	投递MaxCompute的日志分区时间列。由__time__计算得到，用于日志投递MaxCompute时设置日期格式分区列，详细说明请参考#unique_51。
__extract_others__	字符串，可反序列化成JSON Map。	日志内容中不存在该字段，无需设置索引。	日志中投递MaxCompute的未配置字段组装为一个JSON Map。用于日志投递MaxCompute时打包其它未单独配置的字段，详细说明请参考#unique_51。
_extract_others_	字符串，可反序列化成JSON Map。	日志内容中不存在该字段，无需设置索引。	与__extract_others__相同，建议使用__extract_others__。

保留字段名称	数据格式	索引与统计设置	说明
`__tag__:__client_ip__`	字符串格式。	· 索引设置：开启索引后，日志服务默认为所有标签（Tag）字段创建索引，索引数据类型为text类型，分词字符为空，在查询时要完全命中，或采用模糊查询。 · 统计设置：默认没有为该列开启统计。如需开启统计，请手动添加`__tag__:__client_ip__`的索引，并开启统计功能。	日志来源设备的公网IP，该字段为系统标签（Tag）。开启记录外网IP功能后，服务端接收日志时为原始日志追加该字段。可用于日志查询、分析、自定义消费。 对该字段进行SQL分析时，需要给该字段加上双引号。
`__tag__:__receive_time__`	字符串，可转换为整型的Unix标准时间格式。	· 索引设置：开启索引后，日志服务默认为所有标签（Tag）创建索引，索引数据类型为text类型，分词字符为空，在查询时要完全命中，或采用模糊查询。 · 统计设置：默认没有为该列开启统计。如需开启统计，请手动添加`__tag__:__receive_time__`的索引，并开启统计功能。	日志到达服务端的时间，该字段为系统标签（Tag）。开启记录外网IP功能后，服务端接收日志时为原始日志追加该字段。可用于日志查询、分析、自定义消费。
`__tag__:__path__`	字符串格式。	· 索引设置：开启索引后，日志服务默认为`__tag__:__path__`创建索引，索引数据类型为text类型，分词字符为空。查询时输入`__tag__:__path__:XXX`。 · 统计设置：默认没有为该列开启统计。如需开启统计，请手动添加`__tag__:__path__`的索引，并开启统计功能。	Logtail采集的日志文件路径，Logtail为日志自动填加该字段。可用于日志查询、分析、自定义消费。 对该字段进行SQL分析时，需要给该字段加上双引号。

保留字段名称	数据格式	索引与统计设置	说明
`__tag__:__hostname__`	字符串格式。	· 索引设置：开启索引后，日志服务默认为`__tag__:__hostname__`创建索引，索引数据类型为text类型，分词字符为空。查询时输入`__tag__:__hostname__:XXX`。 · 统计设置：默认没有为该列开启统计。如需开启统计，请手动添加`__tag__:__hostname__`的索引，并开启统计功能。	logtail采集数据的来源机器主机名，Logtail为日志自动填加该字段。可用于日志查询、分析、自定义消费。 对该字段进行SQL分析时，需要给该字段加上双引号。
`__raw_log__`	字符串格式。	请手动添加并设置该字段的索引，索引数据类型为text，并根据需求选择是否开启统计。	解析失败的原始日志。关闭丢弃解析失败日志功能后，Logtail在解析日志失败时上传原始日志。其中Key为`__raw_log__`、Value为日志内容。可用于日志投递、查询、分析、自定义消费。
`__raw__`	字符串格式。	请手动添加并设置该字段的索引，索引数据类型为text，并根据需求选择是否开启统计。	解析成功的原始日志。开启上传原始日志功能后，Logtail会将原始日志作为`__raw__`字段，和解析后的日志一并上传。一般用于审计、合规审查等场景。可用于日志投递、查询、分析、自定义消费。