Log Service

Product Introduction

Product Introduction

What is Log Service

As a one-stop service for log data, Log Service (Log for short) experiences massive big data scenarios of Alibaba Group. Log Service allows you to quickly complete the collection, consumption, shipping, query, and analysis of log data without the need for development, which improves the Operation & Maintenance (O&M) efficiency and the operational efficiency, and builds the processing capabilities to handle massive logs in the DT (data technology) era.

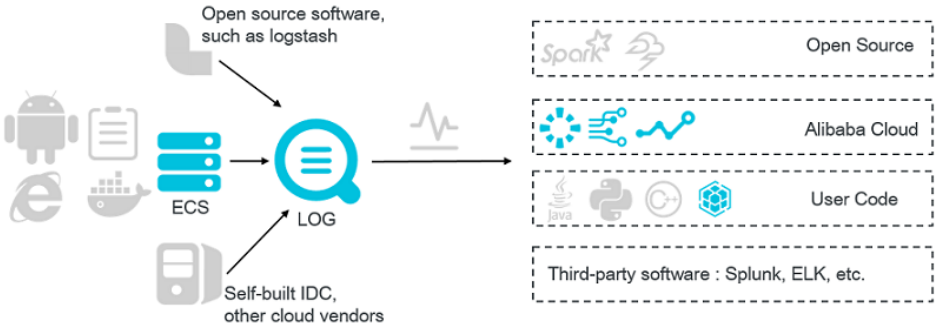
Log Service has the following core functions.

Real-time log collection and consumption (LogHub)

Functions:

- Use Elastic Compute Service (ECS), containers, mobile terminals, open-source softwares, and JS to access real-time log data (such as Metric, Event, BinLog, TextLog, and Click data).
- A real-time consumption interface is provided to interconnect with real-time computing and service.

Purposes: ETL, Stream Compute, monitoring and alarm, machine learning, and iterative computing.

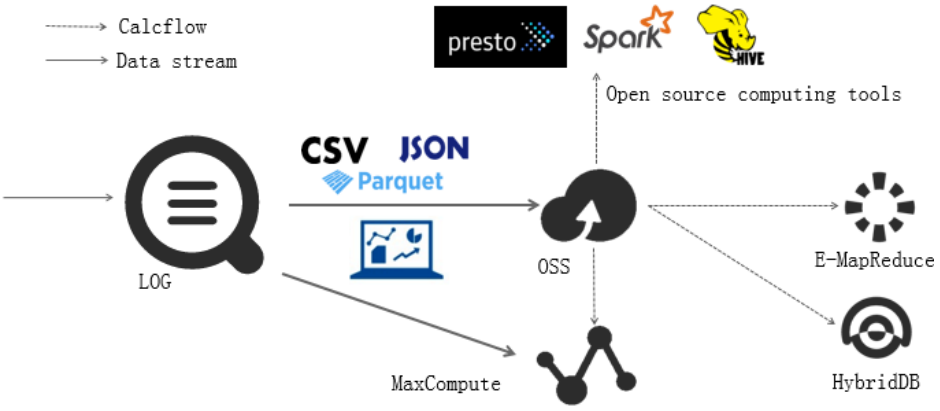


LogShipper

Stable and reliable log shipping ships LogHub data to storage services for storage and big data analysis. Supports various storage methods such as compression, user-defined partitions, row

storage, and column storage.

Purposes: Data warehouse + data analysis, audit, recommendation system, and user profiling.

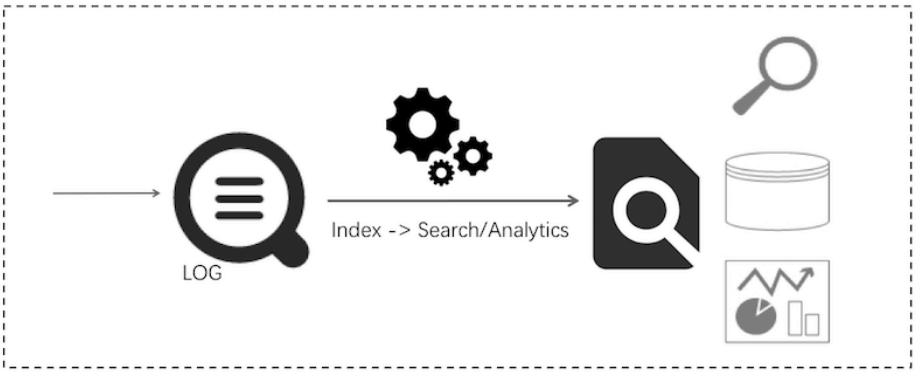


Query and real-time analysis (Search/Analytics)

Index, query, and analyze data in real time.

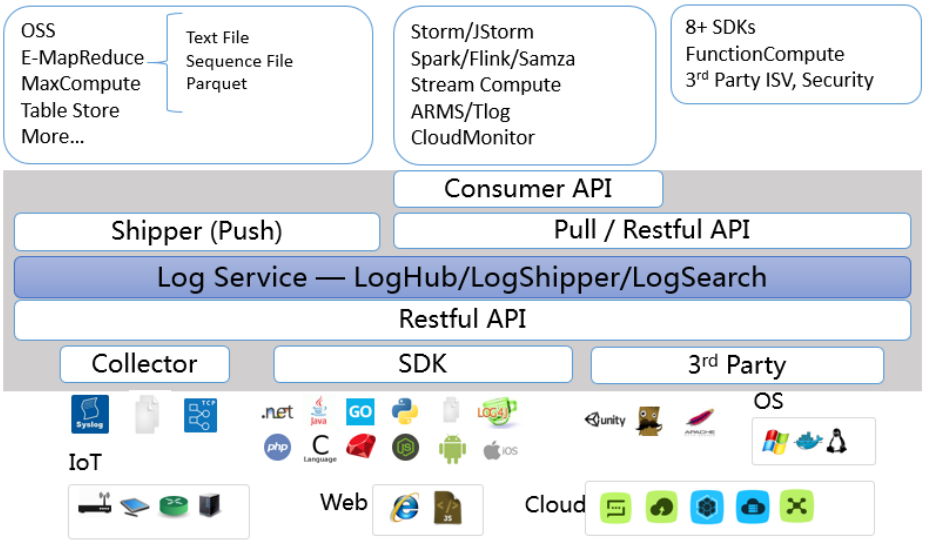
- Query: Keyword, fuzzy match, context, and range.
- Statistics: Rich query methods such as SQL aggregation.
- Visualization: Dashboard and report functions.
- Interconnection: Grafana and JDBC/SQL92.

Purposes: DevOps/online O&M, real-time log data analysis, security diagnosis and analysis, and operation and customer service systems.



Architecture

The Log Service system architecture is as follows.



Logtail

Logtail is an agent that helps you quickly collect logs and has the following features:

- Non-invasive log collection based on log files
  - Only read files.
  - Non-invasion during the reading process.
- Secure and reliable
  - Supports file rotation, so no loss of data.
  - Supports local caching.
  - Provides network exception retry mechanism.
- Convenient management
  - Management on Web.
  - Supports visualization configuration.
- Comprehensive self-protection
  - Monitors the CPU and memory consumed by the process in real time.
  - Restricts the upper limit of memory usage.

Frontend servers

Frontend servers are the frontend machines built with LVS + Nginx and have the following features:

- HTTP and REST protocols
- Horizontal scaling
  - Supports horizontal scaling when traffic increases.
  - Frontend servers can be added to quickly improve processing capabilities.
- High throughput and low latency
  - Pure asynchronous processing. A single request exception does not affect other requests.

- Adopts the Lz4 compression, which is specially for logs, to increase the processing capabilities of individual machines and reduce network bandwidth.

Backend servers

The backend is a distributed process deployed on multiple machines. It provides real-time Logstore data persistence, index, query, and shipping to MaxCompute. The features of the overall backend service are as follows:

- High data security
  - Each log you write is saved in triplicate.
  - Data is automatically replicated and repaired if a disk is damaged or the machine hardware/software has a system error.
- Stable service
  - Logstores are automatically migrated if the process is crashed or the machine does not have a response for a long time.
  - Automatic Server Load Balancer makes sure that traffic is distributed evenly among different machines.
  - Strict quota limits that prevent abnormal behavior of a single user from affecting other users.
- Horizontal scaling
  - Horizontal scaling is performed by using shards as the unit.
  - You can dynamically add shards as needed to increase throughput.

Benefits

Fully managed service

- Easy to use. You can access the service for usage in five minutes and use Agents to collect data in any network environment.
- LogHub has all the functions of Kafka, provides complete functional data, such as monitoring and alarms, and supports auto scaling (by PB/day). The use cost is less than 50% of the self-built cost.
- LogSearch/Analytics provides the functions of saving queries, dashboard, and alarm. The use cost is less than 20% of the self-built cost.
- Log Service has more than 30 Access Methods, and interconnects with cloud products (such as Object Storage Service (OSS), E-MapReduce, MaxCompute, Table Store, MNS, CDN, and ARMS) and open-source softwares (Storm and Spark) seamlessly.

Rich ecosystem

- LogHub supports over 30 collectors, including Logstash and Fluent, and can be easily accessed by using embedded devices, Web pages, servers, and programs. It can also be interconnected with consumption systems such as Spark Streaming, Storm, CloudMonitor, and ARMS.
- LogShipper supports a variety of data formats (such as TextFile, SequenceFile, and Parquet) and user-defined partitions. Data can directly interconnect with storage engines such as Presto, Hive, Spark, Hadoop, E-MapReduce, MaxCompute, and HybridDB.
- LogSearch/Analytics has complete query and analysis syntaxes and is compatible with SQL-92. Supports interconnecting with Grafana by using JDBC protocol.

Strong real-timeliness

- LogHub: Data can be used after being written. Logtail (collection agent) can collect and transfer data in real time to the server side within one second (in 99.9% cases).
- LogSearch/Analytics: Data can be queried and analyzed after being written. When multiple query conditions are used, billions of data pieces can be queried within one second. When multiple aggregation conditions are used, hundreds of millions of data pieces can be analyzed within one second.

Complete API/SDK

- Easily supports user-defined management and secondary development.
- All functions can be implemented by using APIs/SDKs. SDKs for multiple languages are provided. Services and millions of devices can be managed in an easy way.
- The query and analysis syntax is simple (compatible with SQL-92). The interfaces can be used to interconnect with the ecological softwares (supports Grafana interconnection solution).

Scenarios

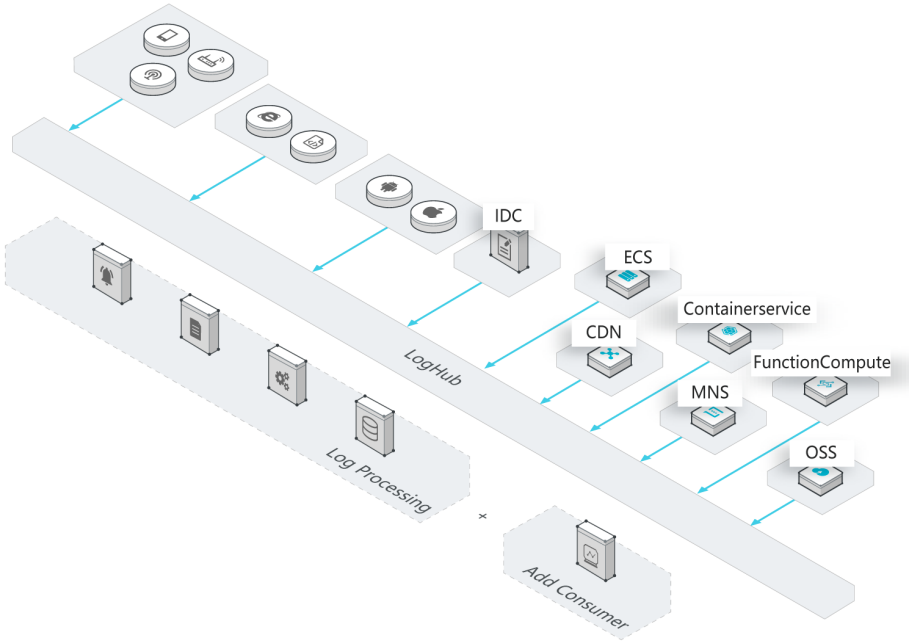
Typical scenarios of Log Service include data collection, real-time computing, data warehousing and offline analysis, product operation and analysis, and Operation & Maintenance (O&M) and management. This document introduces some typical scenarios. For more scenarios, see Best practices.

Data collection and consumption

The LogHub function of Log Service enables access to massive real-time log data (including Metric, Event, BinLog, TextLog, and Click data) at the lower costs.

Advantages of the solution:

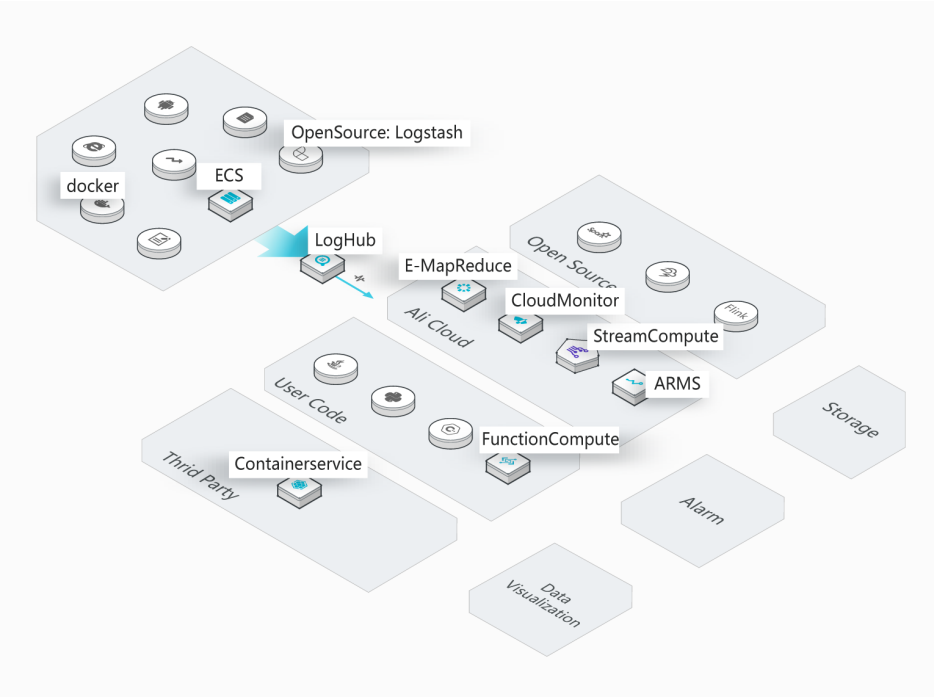
- Easy to use: Over 30 real-time data collection methods are provided for you to quickly build your platform. The powerful configuration and management capabilities can ease O&M workload. Nodes are available across China and the rest of the world.
- Auto scaling: It helps easily cope with traffic peaks and business growth.



ETL/Stream Processing

LogHub can interconnect with various real-time computing and services, provides complete progress monitoring and alarm notification functions, and supports SDK/API-based custom consumption.

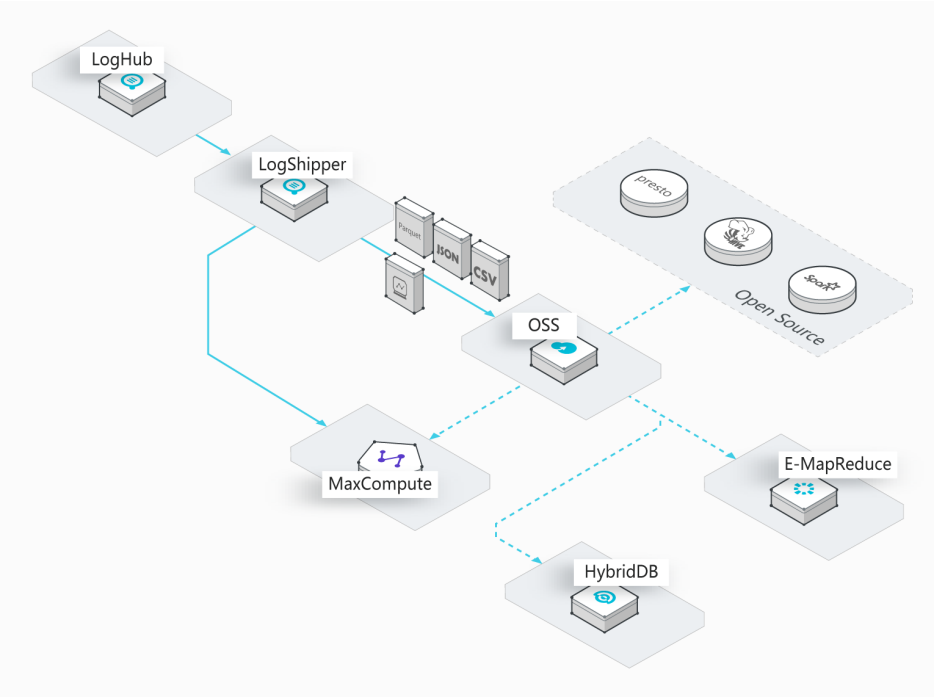
- Easy to operate: It provides various SDKs and programming frameworks and can interconnect with various stream computing engines seamlessly.
- Comprehensive functions: Rich monitoring data and alarm postponing mechanism are provided.
- Auto scaling: PB-grade elasticity and zero latency.



Data warehouse

LogShipper ships LogHub data to storage services and supports various storage formats such as compression, user-defined partitions, row storage, and column storage.

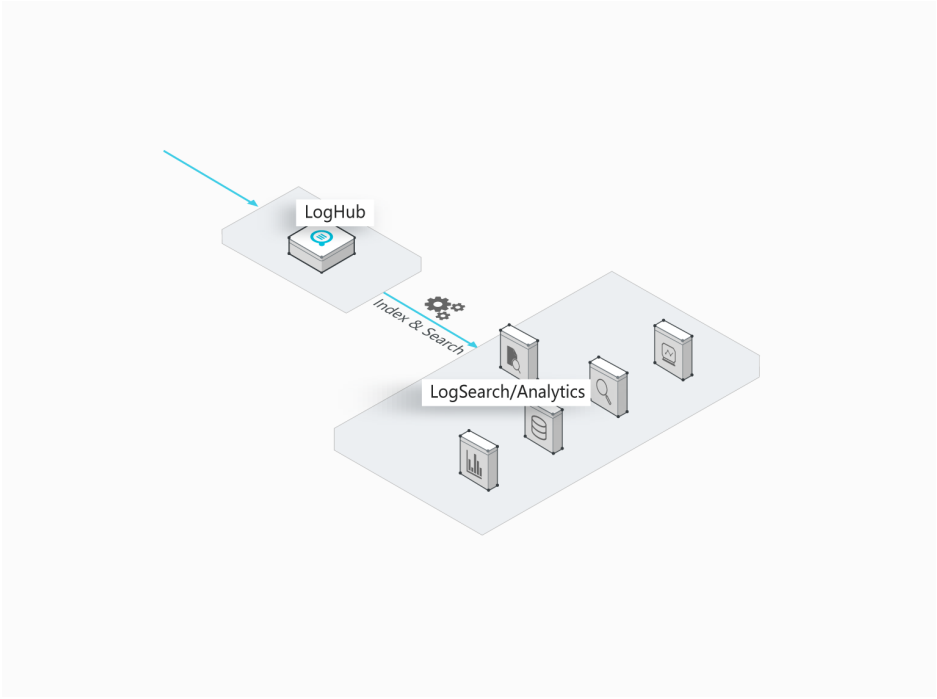
- Massive data: No upper limit is configured for the amount of data.
- Rich storage formats: Various storage formats are supported, such as row storage, column storage, and TextFile.
- Flexible configuration: Configurations such as user-defined partitions are supported.



Real-time query and analysis of logs

LogAnalytics supports indexing LogHub data in real time and provides rich query methods such as keywords, fuzzy match, context, range, and SQL aggregation.

- Strong real-timeliness: Data can be queried after being written.
- Massive amount and low cost: Supports PB/day indexing capabilities, and the cost is 15% of the self-built solution.
- Strong analysis capabilities: Supports multiple query methods. Supports SQL aggregation and analysis. Visualization and alarm notification functions are provided.



Basic concepts

Overview

Log

Log is an abstraction of system changes during the running process. The log content is a time-ordered collection of some operations and the corresponding operation results of specified objects. LogFile, Event, BinLog, and Metric data are different carriers of logs. In LogFile, every log file is composed of one or more logs, and every log describes a single system event. The log is the minimum data unit processed in Log Service.

Log group

A log group is a collection of logs and is the basic unit for writing and reading.

Log topic

Logs in a Logstore can be classified by log topics. You can specify the topic when writing and

querying logs.

Project

The project is the resource management unit in Log Service and is used to isolate and control resources. You can manage all the logs and the related log sources of an application by using projects. Projects manage the information of all your Logstores and the log collection machine configuration, and serve as the portals where you can access the Log Service resources.

Logstore

The Logstore is a unit in Log Service for the collection, storage, and query of log data. Each Logstore belongs to a project, and each project can create multiple Logstores.

Shard

Each Logstore is divided into several shards and each shard is composed of MD5 left-closed and right-open intervals. Each interval range does not overlap with others and the total range of all the intervals is the entire MD5 value range.

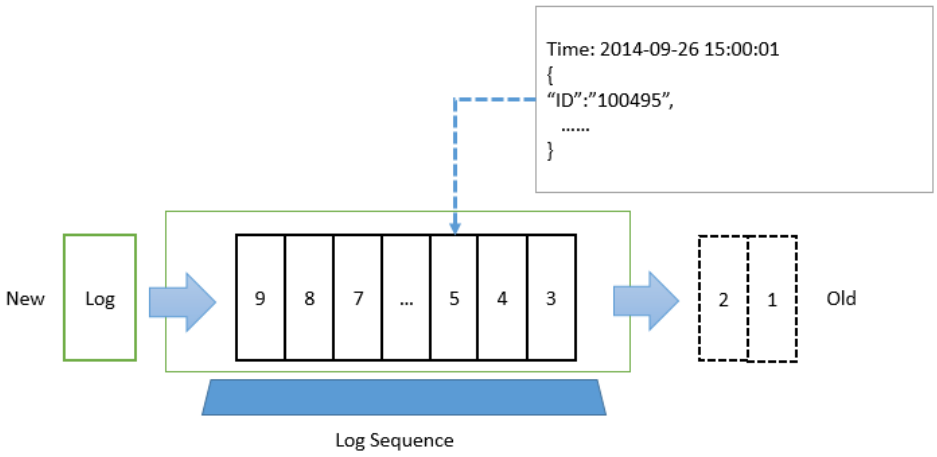
Log

Half a century ago, the term “log” was associated with a thick notebook written by a ship captain or operator. Nowadays, with the advent of computers, logs are generated and used everywhere. Servers, routers, sensors, GPS devices, orders, and various IoT devices describe the world we live from different angles by generating and using logs. With the computing power, we continuously update our recognition to the whole world and system by collecting, processing, and using logs.

What is a log?

Consider an example of a ship captain’s log. In addition to a recorded timestamp, a log can contain almost all sorts of information, such as a text record, an image, weather conditions, and the sailing course. After centuries passed, now the “ship captain’s log” has been expanded to various areas such as orders, payment records, user accesses, and database operations.

The reason why logs are widely used and enduring is that logs are the simplest storage abstraction. Logs are a collection of chronological records that can only be added. The following figure is what logs (time-series data) look like.



We can add a record to the end of a log and read the log records from left to right. Each record has a unique log record number with a sequence.

The log sequence is determined by "time". From the preceding figure, we can see that the log time sequence is from right to left. The new event is recorded, and the old event is gradually out of sight. But a log is a record of events. This is the foundation of recognition and reasoning to computers, humans, and the whole world.

Logs in Log Service

A log is an abstraction of system changes during the running process. The log content is a time-ordered collection of some operations and the corresponding operation results of specified objects. LogFile, Event, BinLog, and Metric data are different carriers of logs. In LogFile, every log file is composed of one or more logs, and every log describes a single system event. A log is the minimum data unit processed in Log Service.

Log Service defines a log by using the semi-structured data mode. This mode includes the following four data fields: Topic, Time, Content, and Source.

Meanwhile, Log Service has different format requirements for different log fields. For more information, see the following table.

Data field	Meaning	Format
Topic	A custom field used to mark multiple logs. For example, access logs can be marked according to sites.	Any string up to 128 bytes, including a null string. By default, this field is a null string.
Time	A reserved field in the log used to indicate the log generation time. Generally this field is generated directly based on the time in the log.	An integer in the standard UNIX time format. The unit is in seconds. This field indicates the number of seconds since 1970-1-1 00:00:00 UTC.
Content	A field used to record the specific log content. The log content is composed of one	The key is a UTF-8 encoded string up to 128 bytes, and can contain letters,

	or more content items, and each content item is a key-value pair.	underscores, and numbers. It cannot start with a number or use any of the following keywords: __time__, __source__, __topic__, __partition_time__, _extract_others_, and __extract_others__. The value can be any string up to 1024*1024 bytes.
Source	A field used to indicate the source of the log. For example, the IP address of the machine where the log is generated.	Any string up to 128 bytes. By default, this field is null.

Various log formats are used in actual usage scenarios. For better understanding, the following example describes how to map an original Nginx access log to the Log Service log data model. Assume that the IP address of your Nginx server is 10.249.201.117. The following is an original log of this server.

```
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"
```

Map the original log to the Log Service log data model as follows:

Data field	Content	Description
Topic	“”	Use the default value (null string).
Time	1330589527	The precise log generation time, indicating the number of seconds since 1970-1-1 00:00:00 UTC. The time is converted from the timestamp of the original log.
Content	Key-value pair	Specific log content.
Source	“10.249.201.117”	Use the IP address of the server as the log source.

You can decide how to extract the original log contents and combine them into key-value pairs. The following table is shown as an example.

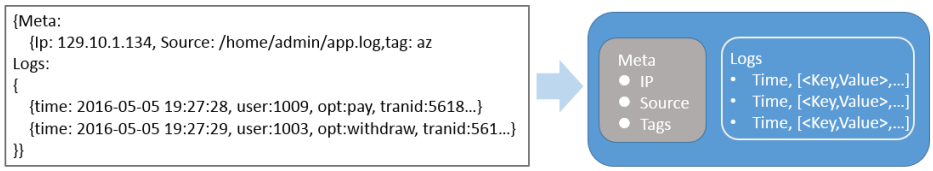
Key	Value
ip	“10.1.168.193”
method	“GET”
status	“200”

length	"5"
ref_url	"- "
browser	"Mozilla/5.0 (X11; Linux i686 on x86_64; rv:10.0.2) Gecko/20100101 Firefox/10.0.2"

Log Group

A log group is a collection of logs and is the basic unit for writing and reading.

The maximum capacity of a log group is up to 4096 logs or 10 MB.



Project

The project is the resource management unit in Log Service and is used to isolate and control resources. You can manage all the logs and the related log sources of an application by using projects. Projects manage the information of all your Logstores and the log collection machine configuration, and serve as the portals where you can access the Log Service resources.

Specifically, projects provide the following functions:

Projects help you organize and manage different Logstores. In actual use, you might use Log Service to centrally collect and store the logs of the different projects, products, or environments. You can classify different logs for management in different projects to facilitate subsequent usage, export, or index of logs. In addition, projects are the carriers of the log access permission management.

Projects serve as the portals where you can access the Log Service resources. Log Service allocates a unique access point for each created project. The access point supports writing, reading, and managing logs by using the network.

In the Log Service console, you can:

- Create a project.
- View project list.

- Manage projects.
- Delete a project.

Logstore

The Logstore is a unit in Log Service to collect, store, and query the log data. Each Logstore belongs to a project, and each project can create multiple Logstores. You can create multiple Logstores for a project according to your actual needs. Typically, an independent Logstore is created for each type of logs in an application. For example, you have a game application “big-game”, and three types of logs are on the server: operation_log, application_log, and access_log. You can first create a project named “big-game”, and then create three Logstores under this project for these three types of logs to collect, store, and query logs respectively.

You must specify the Logstore for writing and querying logs.

Specifically, Logstores provide the following functions: