Log Service

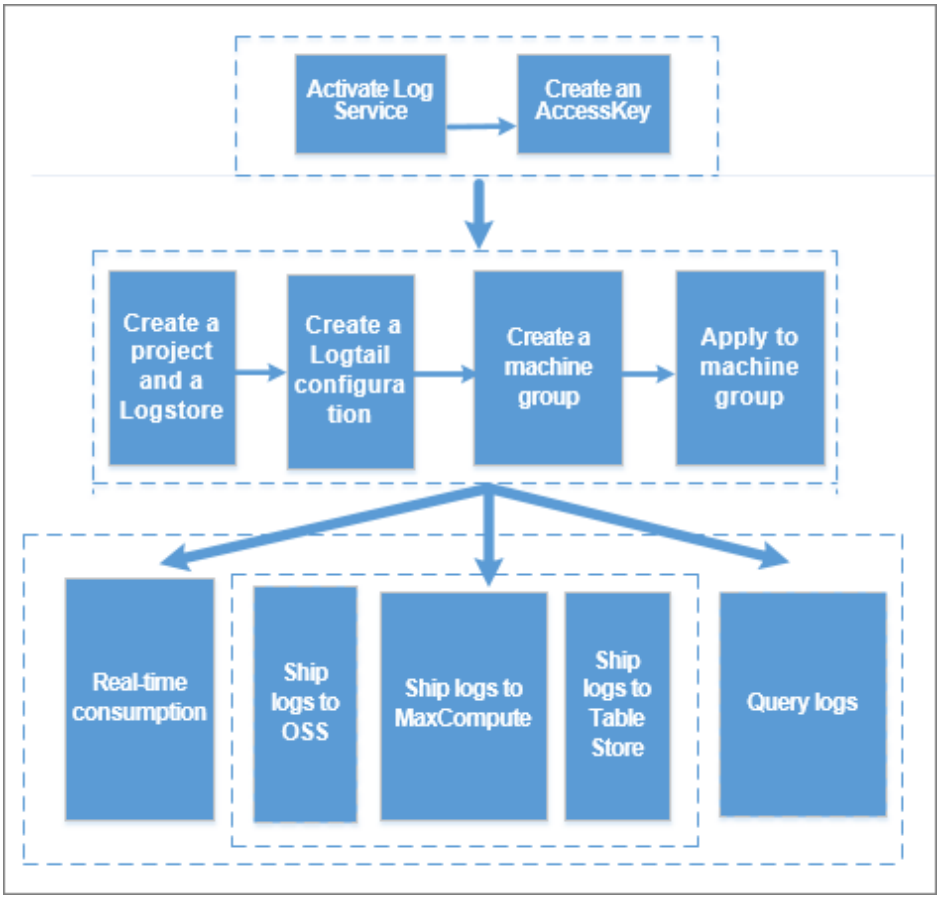
Quick Start

Quick Start

5-minute quick start

Log Service is a platform service provided by Alibaba Cloud to handle the collection, storage, and query of massive logs. You can use Log Service to centrally collect all the logs from the service cluster. It also supports real-time consumption and query.

Workflow of Log Service:



This document demonstrates the basic workflow of configuring Logtail to collect Alibaba Cloud Elastic Compute Service (ECS) logs in the Windows environment. This case is related to the basic functions of Log Service, such as collecting logs and querying logs in real time, and is an entry-level user guide of Log Service.

Step 1 Getting started

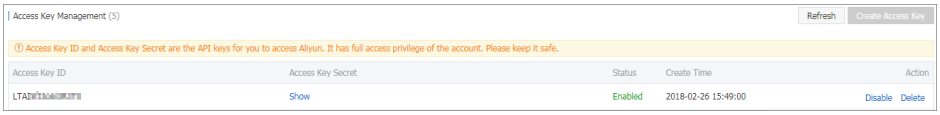
1 Activate Log Service

Use a registered Alibaba Cloud account to log on to the Log Service product page and click **Get it Free**.

2 Create an AccessKey

The AccessKey is a requirement for Logtail to collect logs. Before you use Log Service, create an AccessKey first.

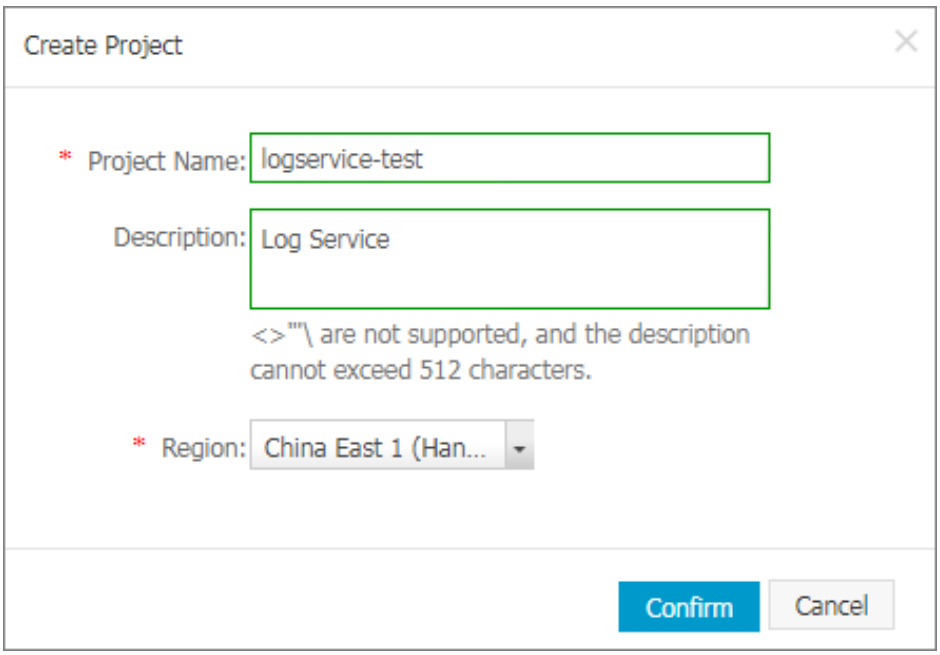
In the Log Service console, hover your mouse over your avatar in the upper-right corner and click **accesskeys** in the displayed drop-down menu. In the dialog box, click **Continue to manage AccessKey** to go to the **Access Key Management** page. Then, create an AccessKey. Make sure the status is set to **Enabled**.



3 Create a project

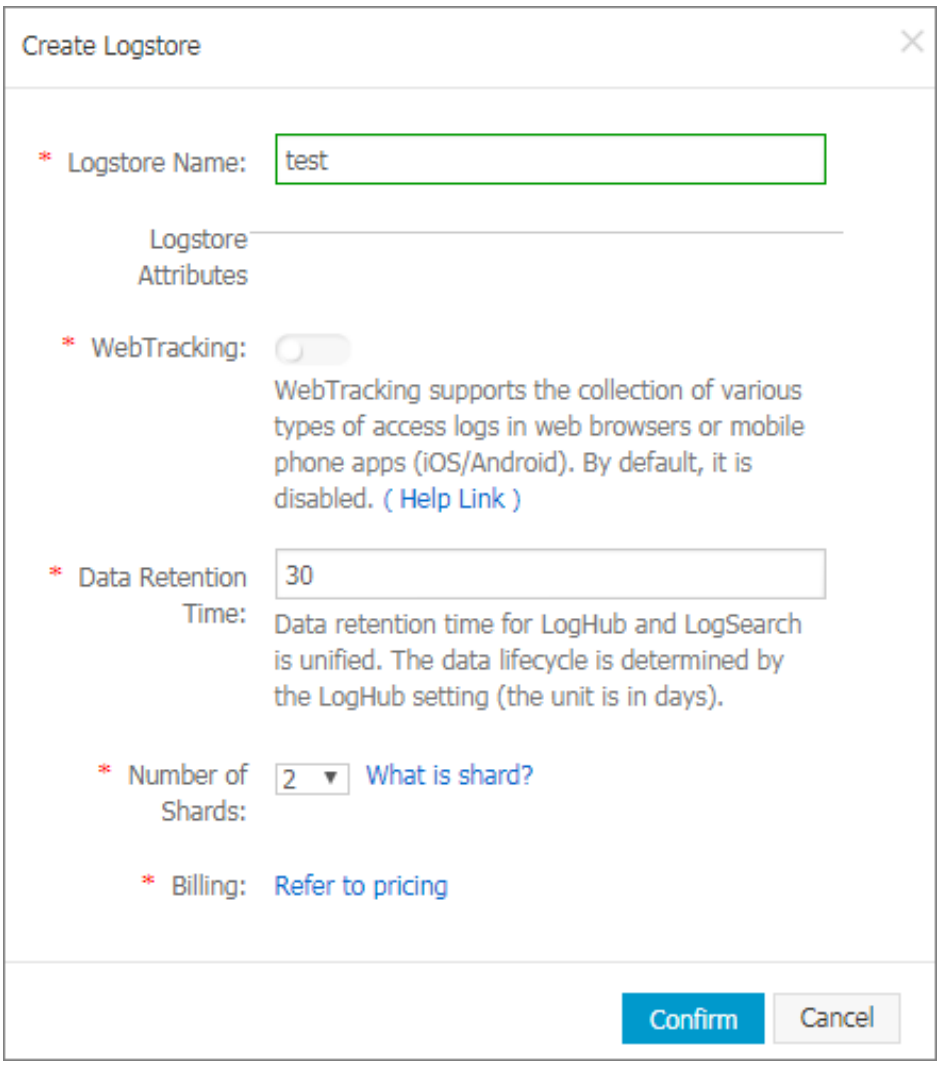
If you have logged on to the Log Service console for the first time, the system prompts you to create a project. You can also click **Create Project** in the upper-right corner to create a project.

When creating a project, you must specify the **Project Name** and **Region** based on your actual needs. Among the regions, cn-shanghai-internal-prod-1 and cn-hangzhou-internal-prod-1 are used for internal Log Service, while the other regions are in the public cloud.



4 Create a Logstore

After creating a project, you will be prompted to create a Logstore. You can also go to the project and click **Create** in the upper-right corner. When creating a Logstore, you must specify how you are going to use these logs.



Step 2 Install Logtail client on ECS instance

1. Download the installation package

Download the Logtail installation package to an ECS instance. Click here to download the Windows installation package.

2. Install Logtail

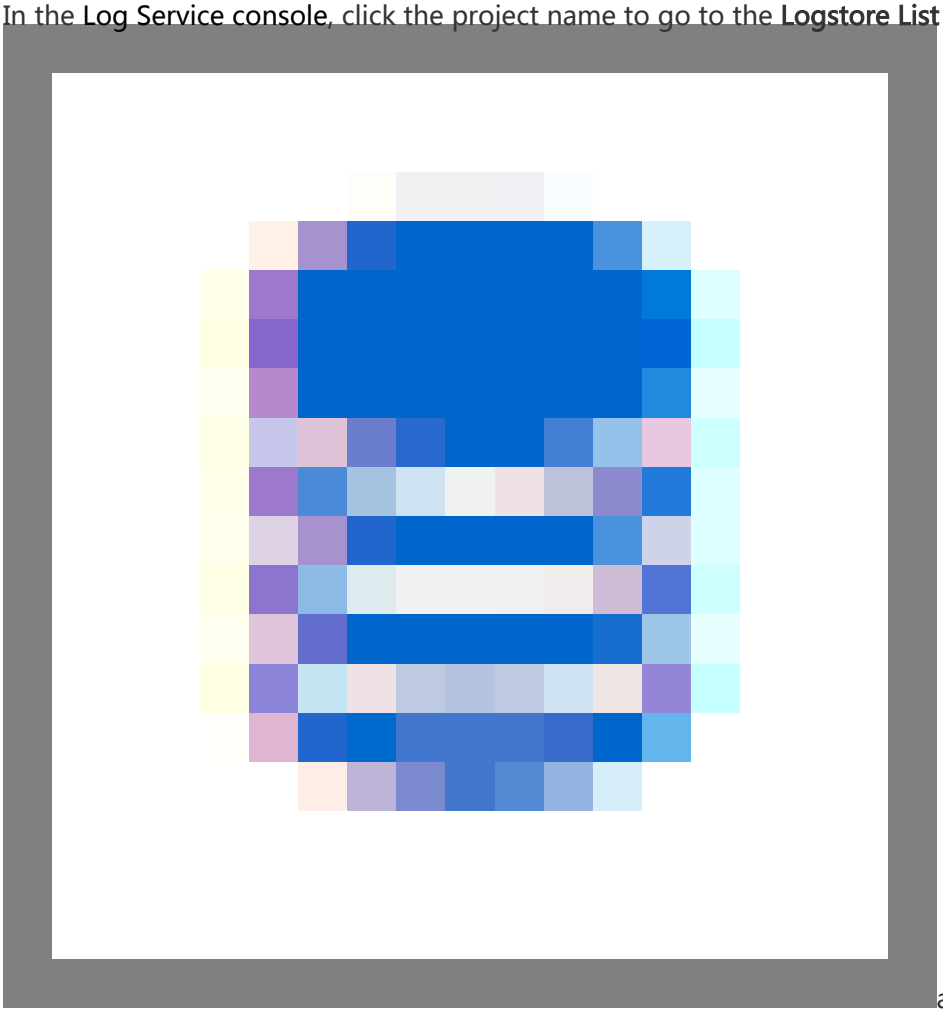
Extract the installation package to the current directory and then enter the logtail_installer directory. Run cmd as an administrator and run the installation command .\logtail_installer.exe install cn_hangzhou.

> **Note:** You must run different installation commands according to the network environment and the region of Log Service. This document uses the ECS classic network in China East 1 (Hangzhou) as an example. For the installation commands of other regions, see Install Logtail on

Windows.

For more information, see Install Logtail on Windows and Install Logtail on Linux.

Step 3 Configure data import wizard

In the Log Service console, click the project name to go to the **Logstore List** page. Click  at the right of the Logstore to enter the Logtail configuration process. You can also click **Manage** at the right of the Logstore to create a configuration in the Logtail configuration list.

Logtail configuration process includes the following steps:

- **Select Data Source**
- **Configure Data Source**
- **Search, Analysis, and Visualization**
- **Shipper & ETL**

The **Search, Analysis, and Visualization** step and **Shipper & ETL** step are optional.

1. Select data source

Log Service supports the log collection of many cloud products, self-built softwares, and custom data. This document uses collecting text logs as an example. For detailed steps and descriptions, see Text logs. For how to collect syslog, see Use Logtail to collect syslog.

Click **Text** under **Other Sources** and then click **Next**.

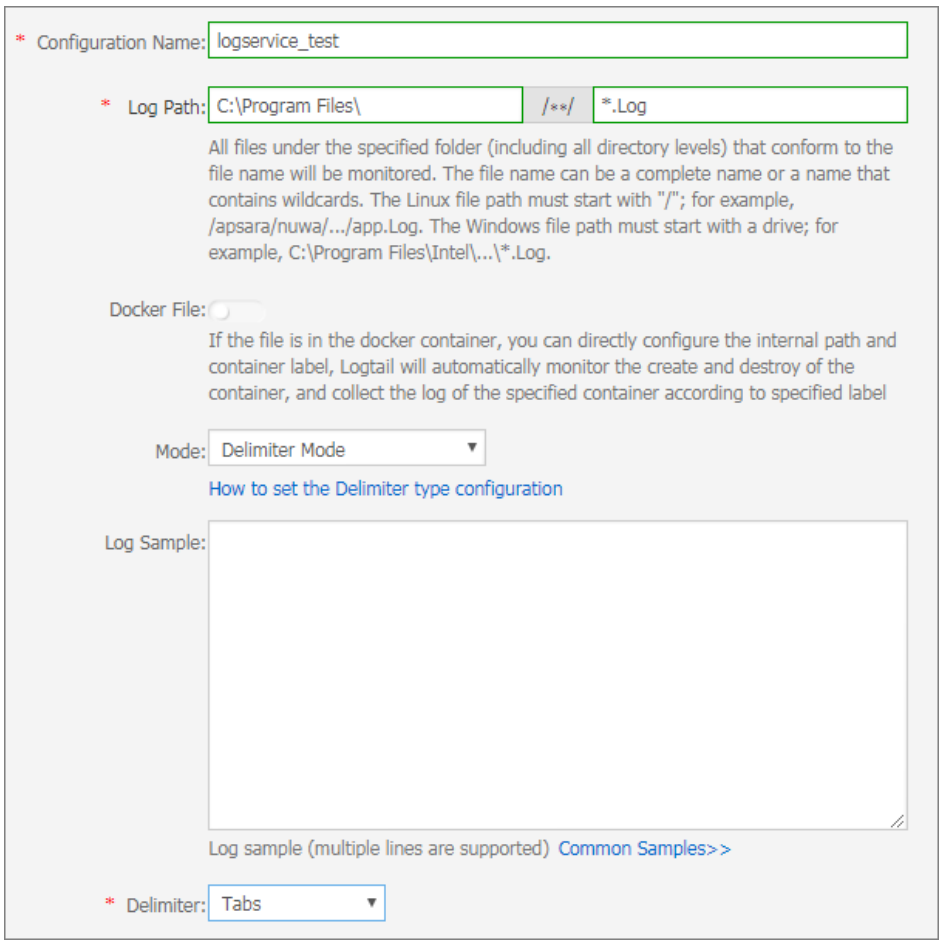
2. Configure data source

Specify the **Configuration Name** and **Log Path**.

As instructed on the page, enter the configuration name, log path, and log file name. The log file name can be a full name, and supports wildcard matching at the same time.

Specify the log collection mode.

Log Service currently supports parsing logs in simple mode, delimiter mode, JSON mode, full mode, or Alibaba Cloud custom mode. This document uses the delimiter mode as an example. For more information about the collection modes, see Collection steps and Other information.



Enter the log sample.

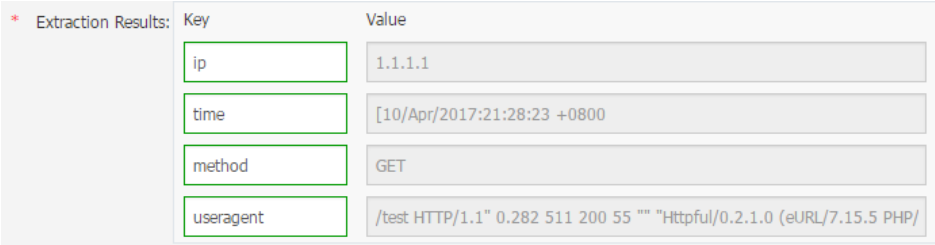
You must enter the log sample if **Delimiter Mode** or **Full Mode** is selected as the log collection mode. Log Service supports parsing the log sample according to your selected configuration when configuring Logtail. If the log sample failed to be parsed, you must modify the delimiter configurations or regular expressions. Enter the log sample to be parsed in the **Log Sample** field.

Specify the delimiter.

You can specify the delimiter as a tab, a vertical line, or a space. You can also customize the delimiter. Select the delimiter according to your log format. Otherwise, logs fail to be parsed.

Specify the key in the log extraction results.

After you enter the log sample and select the delimiter, Log Service extracts log fields according to your selected delimiter, and defines them as **Value**. You must specify the corresponding **Key** for the **Value**.



Configure the advanced options as needed.

Generally, keep the default configurations of the advanced options. For how to configure the advanced options, see the related descriptions in Text logs.

Apply to the machine group.

If you have not created a machine group before, create a machine group as instructed on the page. Then, apply the Logtail configuration to the machine group.

> **Note:** To create Armory to associate with the machine group, jump to the specified internal link as instructed on the page.

After completing these steps, Log Service begins to collect logs from the Alibaba Cloud ECS instance immediately. You can consume the collected logs in real time in the console and by using API/SDK.

To query, analyze, ship, or consume the logs, click **Next**.

> Note:
>
> - It can take up to 3 minutes for the Logtail configuration to take effect, so be patient.
> - To collect IIS access logs, see Use Logstash to collect IIS logs.
> - For the Logtail collection errors, see Query collection errors.

3. Search, analysis, and visualization

After the collection configurations, your ECS logs are collected in real time. To query and analyze the collected logs, configure the indexes in the data import wizard as follows.

You can also click **Search** on the **Logstore List** page to go to the query page. Click **Enable** in the upper-right corner and configure the indexes on the displayed **Search & Analysis** page.

Full text index attributes

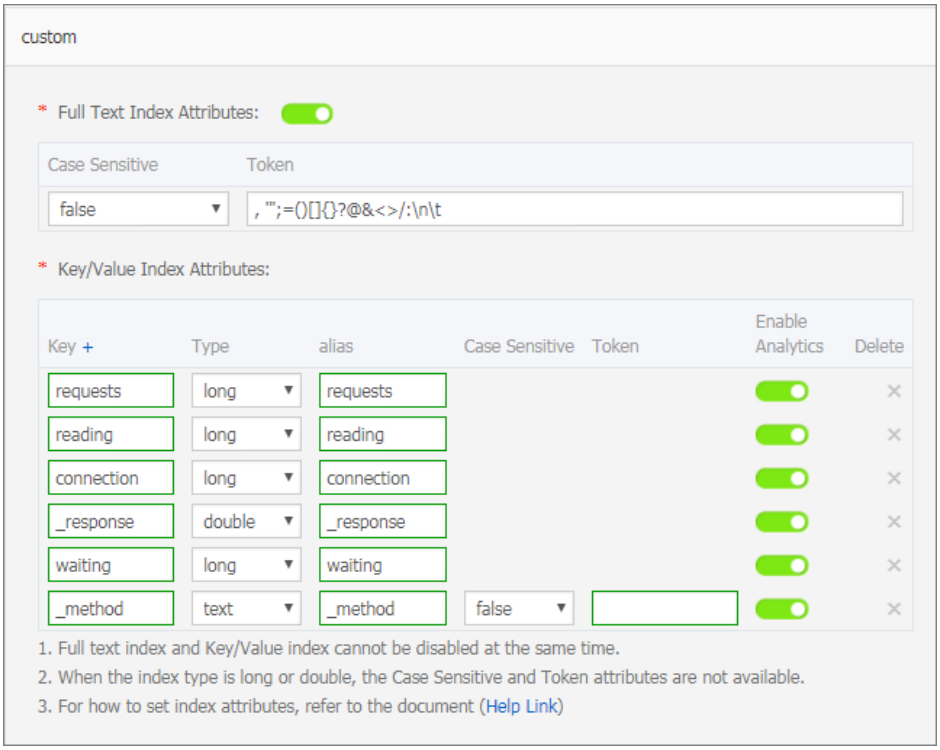
You can enable the **Full Text Index Attributes**. Confirm whether or not to enable **Case Sensitive** and confirm the **Token** contents.

Key/value index attributes

Click the plus icon at the right of **Key** to add a line. Configure the **Key**, **Type**, **Alias**, **Case Sensitive**, and **Token**, and select whether or not to enable analytics.

Note:

- Enable at least one type of index attributes. When both types are enabled, be subject to key/value index attributes.
- When the index type is long or double, the **Case Sensitive** and **Token** attributes are unavailable.
- For how to configure indexes, see Query logs.
- To use Nginx template or MNS template, configure the attributes on the **Search & Analysis** page after clicking **Enable** on the query page.



After configuring the query and analysis, click **Next** if you want to configure the log shipping. To experience the query and analysis, go back to the **Logstore List** page and click **Search** to go to the query page. You can enter the keyword, topic, or query & analysis statement, and select the time range to query logs. Log Service provides you with histograms to preview the query results intuitively. You can click the histogram to query logs in a more detailed time range. For more information, see Query logs.

Log Service also supports querying and analyzing logs in many ways such as quick query and statistical graphs. For more information, see Other functions.

For example, to query all the logs within the last 15 minutes, you can set an empty query condition and select **15 min** as the time range.

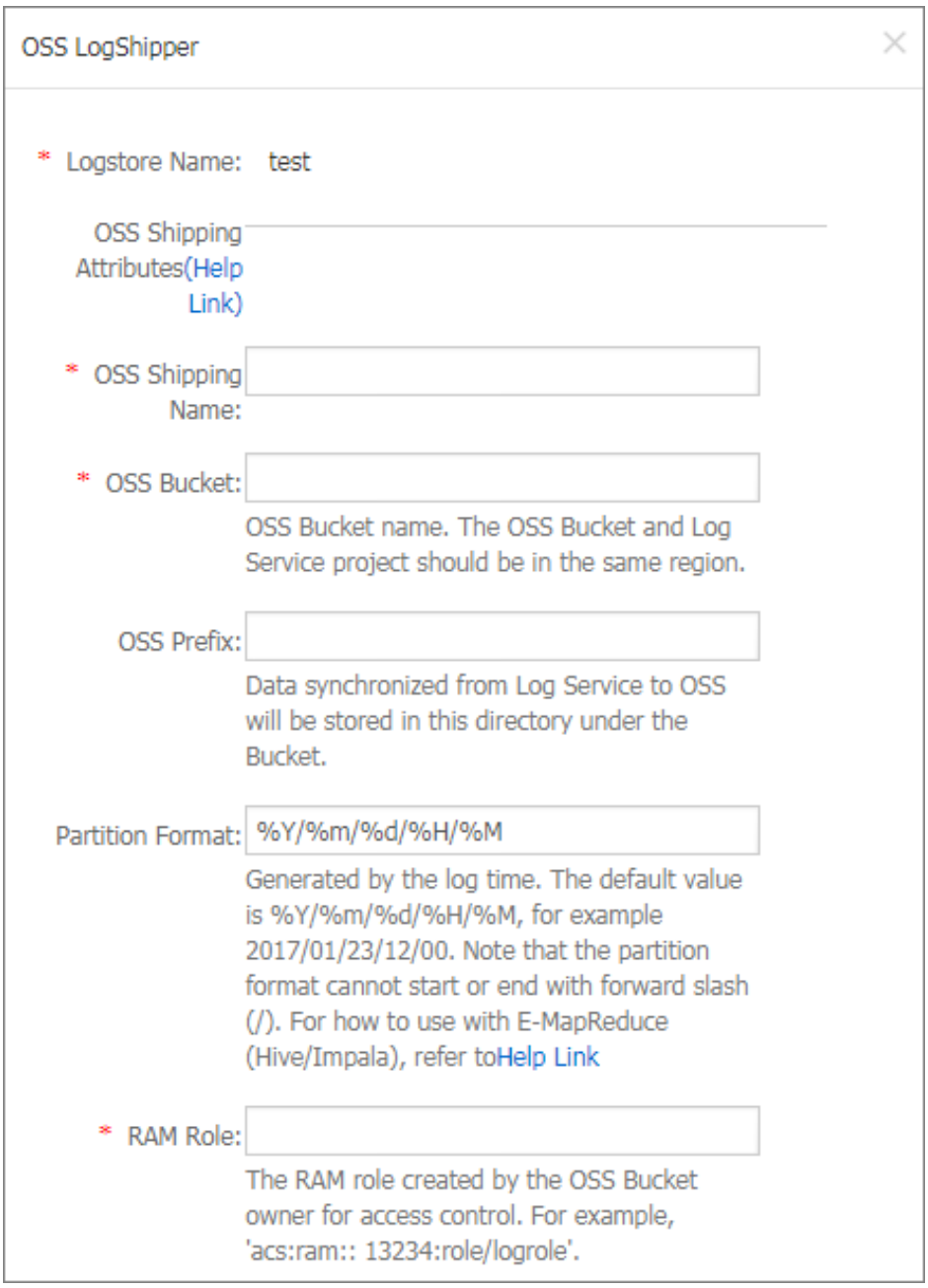
4. Shipper

Log Service not only supports collecting data with multiple sources and formats in batch, managing and maintaining the data, but also supports shipping log data to cloud products such as Object Storage Service (OSS) for calculation and analysis.

To ship logs to OSS, click **Enable**.

In this document, ship the logs to an OSS bucket. See Ship logs to OSS to complete the authentication.

The **OSS LogShipper** dialog box appears after you click **Enable**. Complete the configurations. For descriptions about the configurations, see Ship logs to OSS. Click **Confirm** after the configurations to complete the shipping.



Besides the basic functions such as accessing, querying, and analyzing logs, Log Service also provides many ways to consume logs. For more information, see documents in the **User Guide** section.

Analysis - Nginx access logs

Many webmasters use Nginx as the server to build websites. When analyzing the website traffic data, they must perform a statistical analysis on Nginx access logs to obtain data such as the page views and the access time periods of the website. In the traditional methods such as CNZZ, a js is inserted in

the frontend page and will be triggered when a user accesses the website. However, this method can only record access requests. Stream computing and offline statistics & analysis can also be used to analyze Nginx access logs, which however requires to build an environment, and is subject to imbalance between timeliness and analytical flexibility.

Log Service supports querying and analyzing real-time logs, and saves the analytical results to Dashboard, which greatly decreases the analytical complexity of Nginx access logs and streamlines the statistics of website access data. This document introduces the detailed procedure of log analysis function by analyzing the Nginx access logs.

Scenarios

A webmaster builds a personal website by using Nginx as the server. The PV, UV, popular pages, hot methods, bad requests, client types, and referer tabulation of the website are obtained by analyzing Nginx access logs to assess the website access status.

Log format

We recommend that you use the following log_format configuration for better meeting the analytic scenarios:

```
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" $http_host '
'$status $request_length $body_bytes_sent "$http_referer" '
'"$http_user_agent" $request_time $upstream_response_time';
```

The meaning of each field is as follows.

Field	Meaning
remote_addr	The client address.
remote_user	The client username.
time_local	The server time.
request	The request content, including method name, address, and HTTP protocol.
http_host	The HTTP address used by the user request.
status	The returned HTTP status code.
request_length	The request size.
body_bytes_sent	The returned size.
http_referer	The referer.
http_user_agent	The client name.
request_time	The overall request latency.

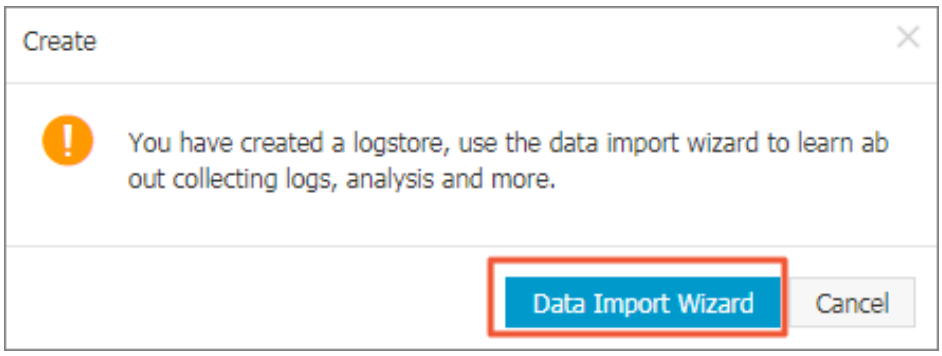
upstream_response_time	The processing latency of upstream services.

Procedure

1. Open the data import wizard

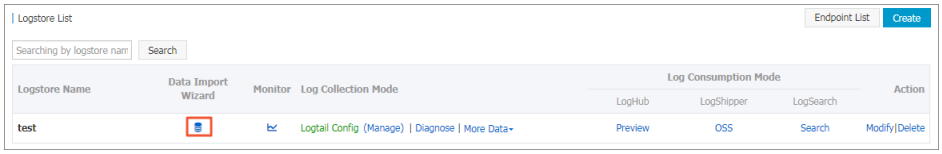
Log Service provides the data import wizard to access data sources fast. To collect Nginx access logs to Log Service, use the following two methods to enter the data import wizard.

Click **Data Import Wizard** after creating a Logstore in an existing project or a newly created project. For how to create a project and a Logstore, see **Create a project** in Manage a project and **Create a Logstore** in Manage a Logstore.



For an existing Logstore, click the **Data Import Wizard** icon

on the **Logstore List** page.

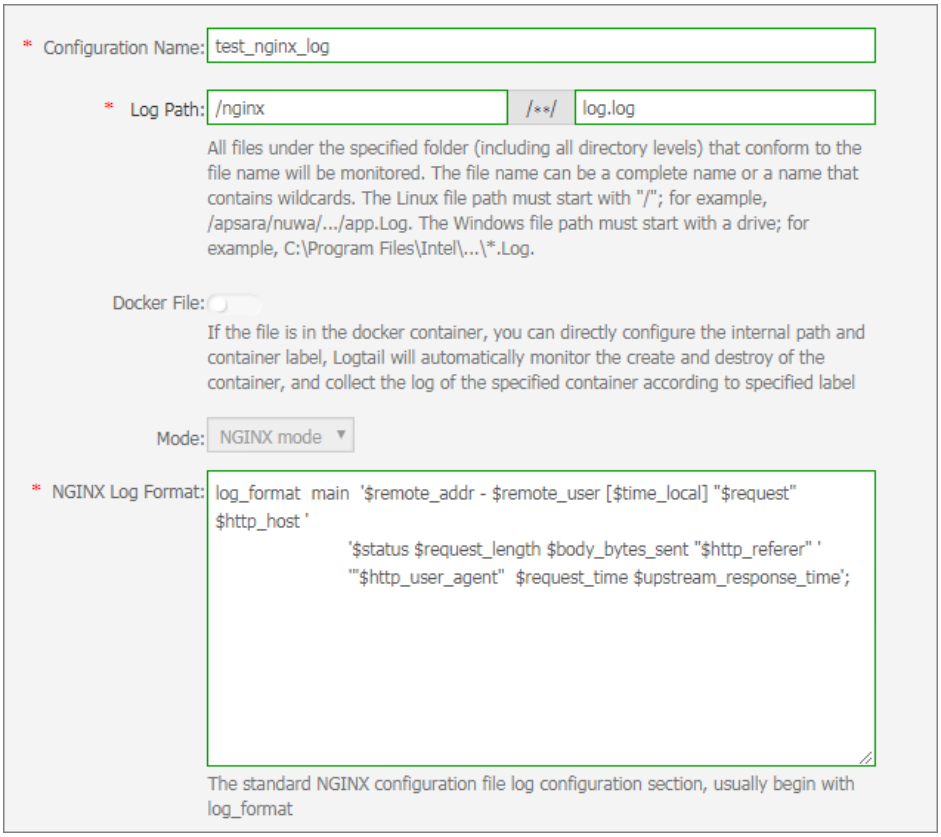


2. Select a data source

Log Service provides many types of data sources, such as cloud service, third-party software, API, and SDK. To analyze the Nginx access logs, select **NGINX ACCESSLOG** under **Third-Party Software**.

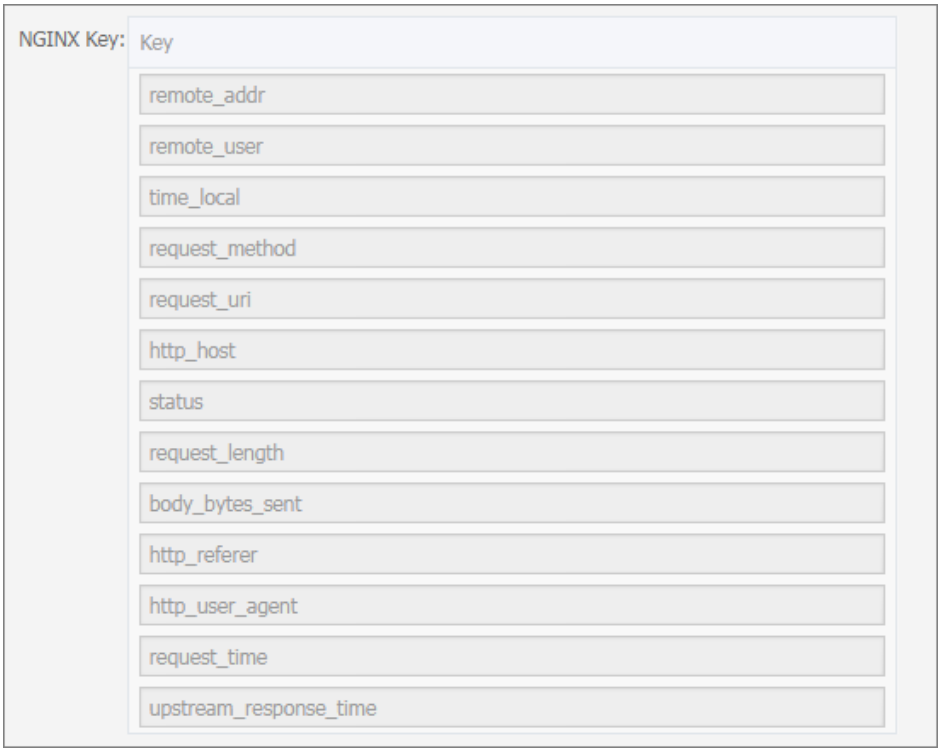
3. Configure the data source

Enter the **Configuration Name** and **Log Path** according to your actual situation. Then, enter the recommended log_format information in the **NGINX Log Format** field.



Log Service automatically extracts the corresponding keys.

Note: $request is extracted as two keys: request_method and request_uri.



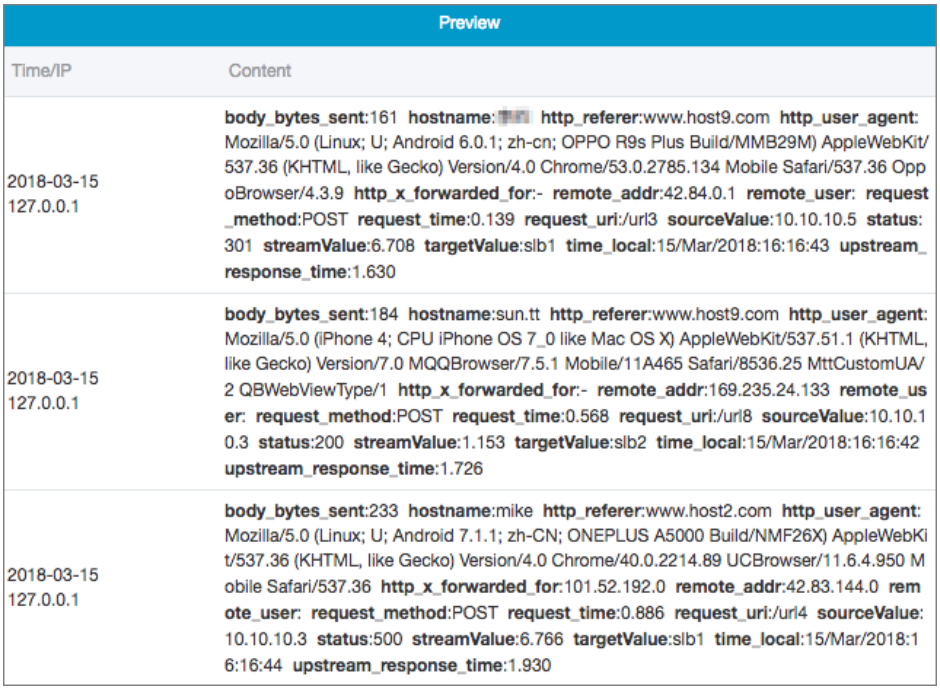
Apply to the machine groups.

If you have not created a machine group, you must create one first. For how to create a machine group, see Create a machine group.

> **Note:** It takes up to three minutes for the Logtail configuration to take effect, so be patient.

4. Search, analysis, and visualization

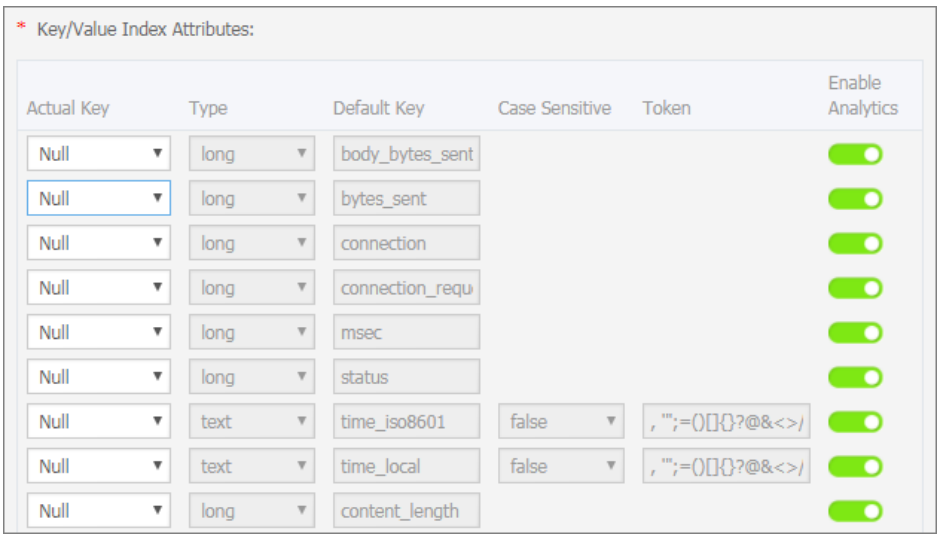
Make sure the heartbeat statuses of the machine groups that apply the Logtail configuration are normal and you can click **Preview** on the right to obtain the collected data.



Preview

Time/IP	Content
2018-03-15 127.0.0.1	**body_bytes_sent**:161 **hostname**:[redacted] **http_referer**:www.host9.com **http_user_agent**:Mozilla/5.0 (Linux; U; Android 6.0.1; zh-cn; OPPO R9s Plus Build/MMB29M) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/53.0.2785.134 Mobile Safari/537.36 OppoBrowser/4.3.9 **http_x_forwarded_for**:- **remote_addr**:42.84.0.1 **remote_user**: **request_method**:POST **request_time**:0.139 **request_uri**:/url3 **sourceValue**:10.10.10.5 **status**:301 **streamValue**:6.708 **targetValue**:slb1 **time_local**:15/Mar/2018:16:16:43 **upstream_response_time**:1.630
2018-03-15 127.0.0.1	**body_bytes_sent**:184 **hostname**:sun.tt **http_referer**:www.host9.com **http_user_agent**:Mozilla/5.0 (iPhone 4; CPU iPhone OS 7_0 like Mac OS X) AppleWebKit/537.51.1 (KHTML, like Gecko) Version/7.0 MQQBrowser/7.5.1 Mobile/11A465 Safari/8536.25 MttCustomUA/2 QBWebViewType/1 **http_x_forwarded_for**:- **remote_addr**:169.235.24.133 **remote_user**: **request_method**:POST **request_time**:0.568 **request_uri**:/url8 **sourceValue**:10.10.10.3 **status**:200 **streamValue**:1.153 **targetValue**:slb2 **time_local**:15/Mar/2018:16:16:42 **upstream_response_time**:1.726
2018-03-15 127.0.0.1	**body_bytes_sent**:233 **hostname**:mike **http_referer**:www.host2.com **http_user_agent**:Mozilla/5.0 (Linux; U; Android 7.1.1; zh-CN; ONEPLUS A5000 Build/NMF26X) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/40.0.2214.89 UCBrowser/11.6.4.950 Mobile Safari/537.36 **http_x_forwarded_for**:101.52.192.0 **remote_addr**:42.83.144.0 **remote_user**: **request_method**:POST **request_time**:0.886 **request_uri**:/url4 **sourceValue**:10.10.10.3 **status**:500 **streamValue**:6.766 **targetValue**:slb1 **time_local**:15/Mar/2018:16:16:44 **upstream_response_time**:1.930

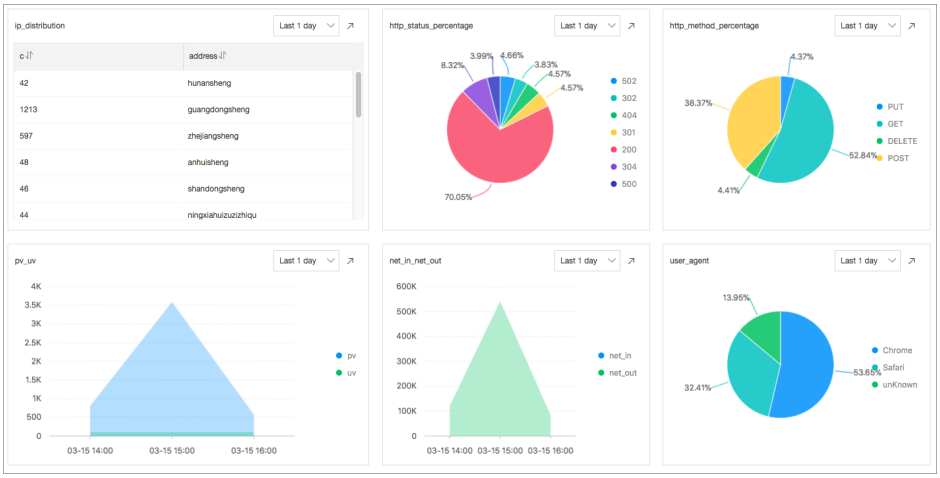
Log Service provides predefined keys for analysis and usage. You can select the actual keys (generated according to the previewed data) to map with the default keys.



Click **Next**. Log Service configures the index attributes for you and creates the nginx-dashboard dashboard for analysis and usage.

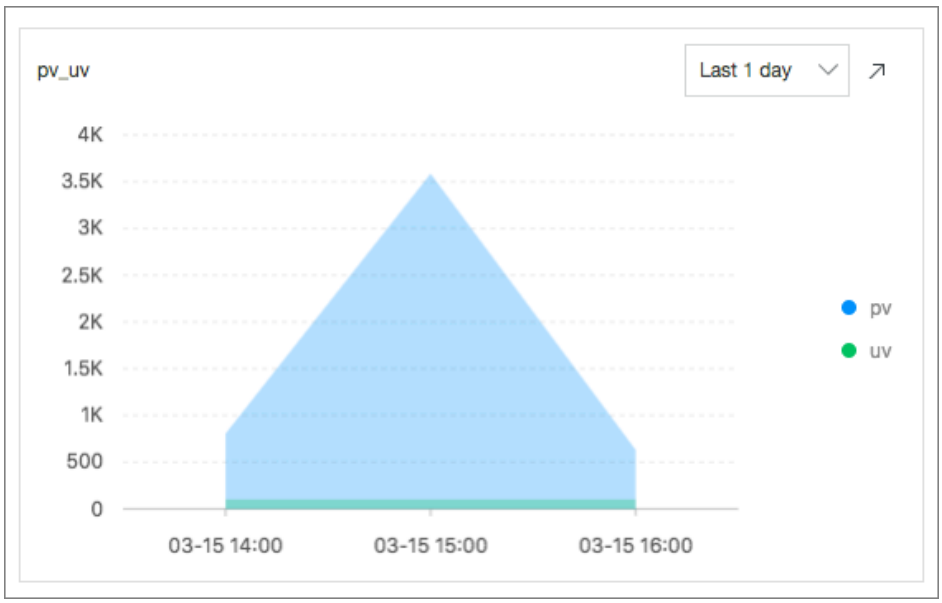
5. Analyze access logs

After the index feature is enabled, you can view the analysis of each indicator on the page where dashboards are generated by default. For how to use dashboards, see Dashboard.



Count PVs/UVs (pv_uv)

Count the numbers of PVs and UVs in the last day.

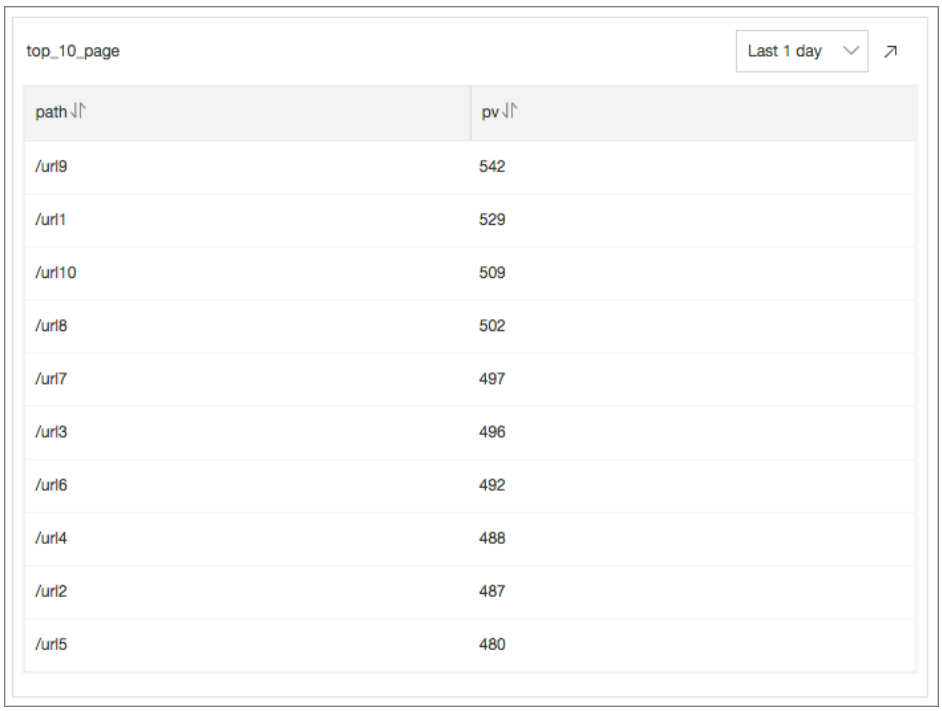


Statistics statement:

```
  * | select approx_distinct(remote_addr) as uv ,
count(1) as pv ,
date_format(date_trunc('hour', __time__), '%m-%d %H:%i') as time
group by date_format(date_trunc('hour', __time__), '%m-%d %H:%i')
order by time
limit 1000
```

Count the top 10 access pages (top_page)

Count the top 10 pages with the most PVs in the last day.



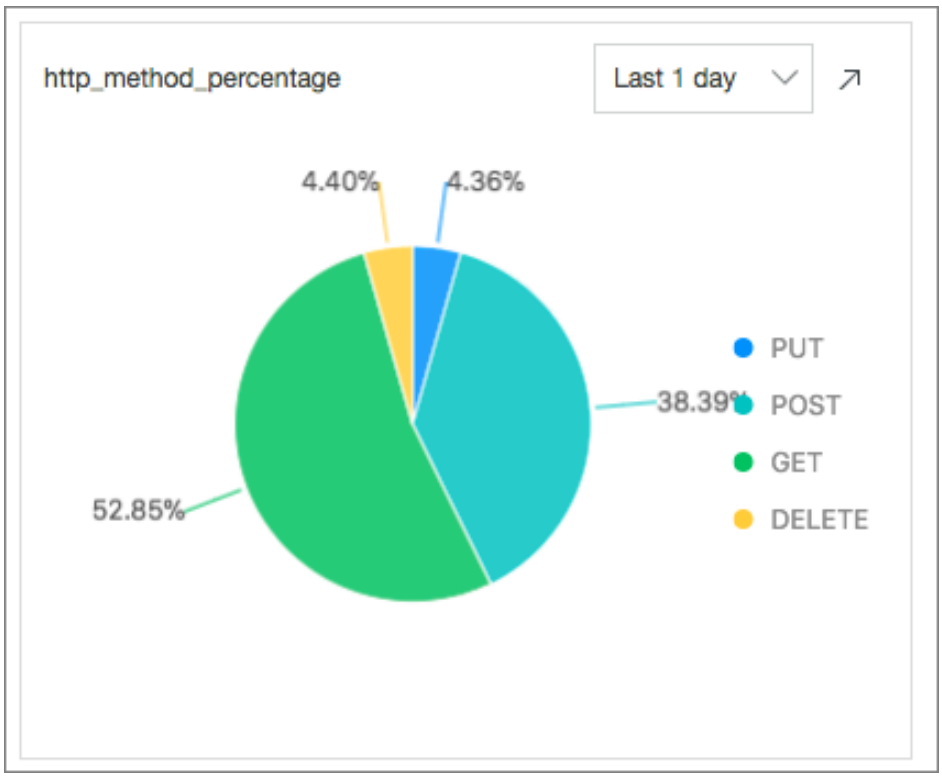
path	pv
/url9	542
/url1	529
/url10	509
/url8	502
/url7	497
/url3	496
/url6	492
/url4	488
/url2	487
/url5	480

Statistics statement:

```
  * | select split_part(request_uri,'?',1) as path,
count(1) as pv
group by split_part(request_uri,'?',1)
order by pv desc limit 10
```

Count the ratios of request methods (http_method_percentage)

Count the ratio of each request method used in the last day.

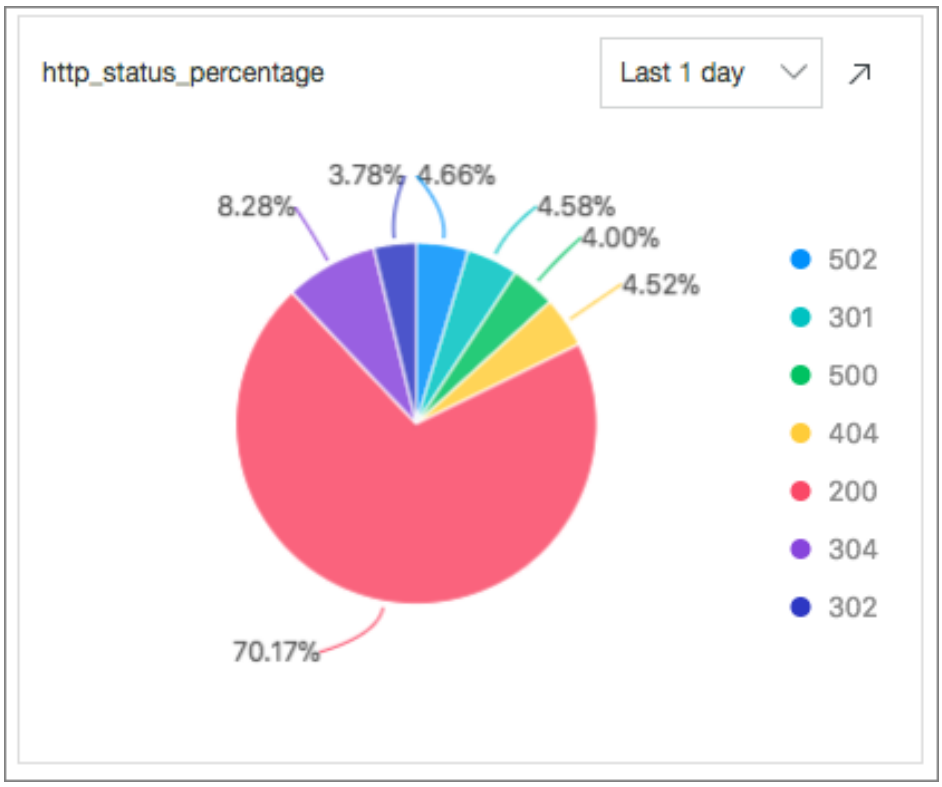


Statistics statement:

```
 * | select count(1) as pv,
request_method
group by request_method
```

Count the ratios of request statuses (http_status_percentage)

Count the ratio of each request status (HTTP status code) in the last day.

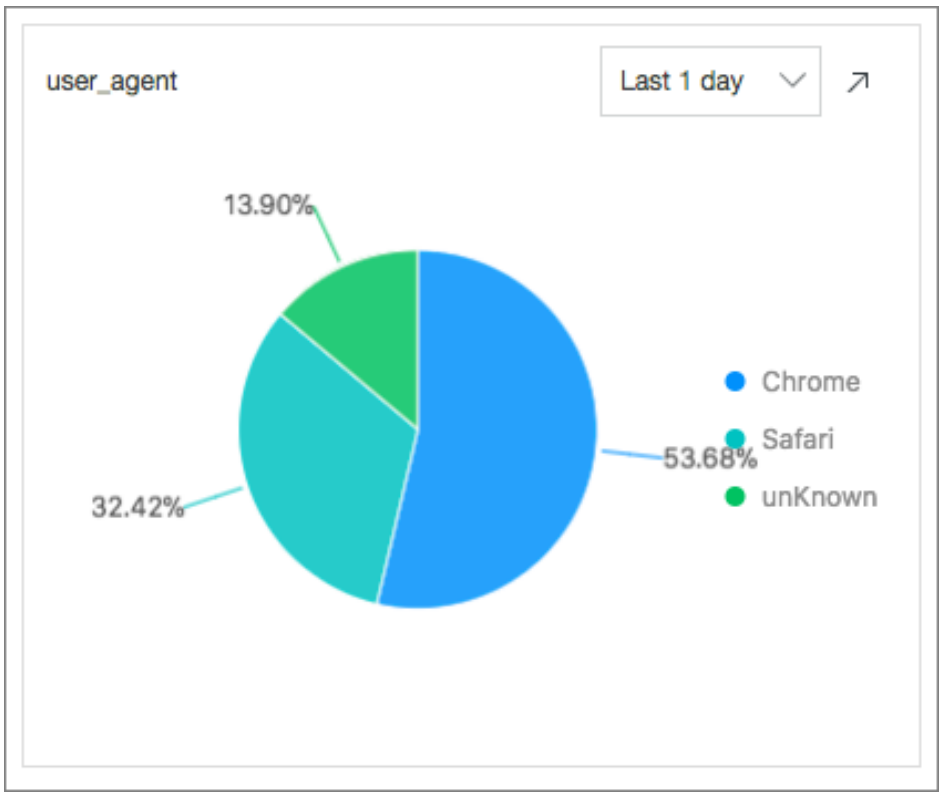


Statistics statement:

```
  * | select count(1) as pv,
status
group by status
```

Count the ratios of request UA (user_agent)

Count the ratio of each browser used in the last day.

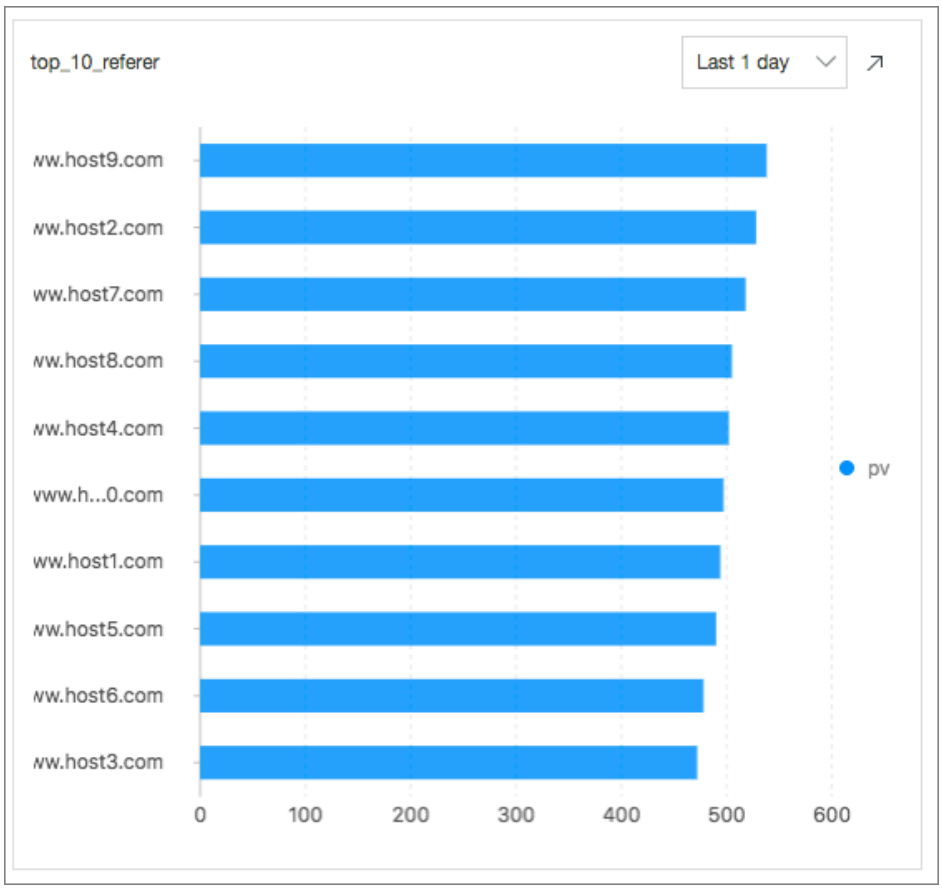


Statistics statement:

```
 * | select count(1) as pv,
case when http_user_agent like '%Chrome%' then 'Chrome'
when http_user_agent like '%Firefox%' then 'Firefox'
when http_user_agent like '%Safari%' then 'Safari'
else 'unKnown' end as http_user_agent
group by case when http_user_agent like '% Chrome%' then 'Chrome'
when http_user_agent like '% Firefox%' then 'Firefox'
when http_user_agent like '% Safari%' then 'Safari'
else 'unKnown' end
order by pv desc
limit 10
```

Count the top 10 referers (top_10_referer)

Count the top 10 referers in the last day.



Statistics statement:

```
 * | select count(1) as pv,
http_referer
group by http_referer
order by pv desc limit 10
```

6. Access diagnosis and optimization

In addition to some default access indicators, webmasters often have to diagnose some access requests to check the latency of request processing, what are the long latencies, and on what pages long latencies occur. Then, you can enter the query page for fast analysis.

Count the average latency and the maximum latency

With the average latency and the maximum latency every five minutes, you can get a picture of the latency issue.

Statistics statement:

```
* | select from_unixtime(__time__ -__time__% 300) as time,
```

```
avg(request_time) as avg_latency ,
max(request_time) as max_latency
group by __time__ -__time__% 300
```

Count the request page with the maximum latency

After knowing the maximum latency, you must identify the request page with the maximum latency to further optimize the page response.

Statistics statement:

```
  * | select from_unixtime(__time__ - __time__% 60) ,
max_by(request_uri,request_time)
group by __time__ - __time__%60
```

Count the distribution of request latencies

Count the distribution of all the request latencies on the website. Place the latencies in ten buckets, and check the number of requests in each latency interval.

Statistics statement:

```
  * | select numeric_histogram(10,request_time)
```

Count the ten longest latencies

In addition to the maximum latency, the second to the tenth longest latencies and their values are also counted.

Statistics statement:

```
  * | select max(request_time,10)
```

Tune the page with the maximum latency

Assume that the maximum access latency occurs on the /url2 page. To tune the /url2 page, count the PVs, UVs, numbers of various methods, statuses, and browsers, the average latency, and the maximum latency of the /url2 page.

Statistics statement:

```
   request_uri:"/url2" | select count(1) as pv,
approx_distinct(remote_addr) as uv,
histogram(method) as method_pv,
```

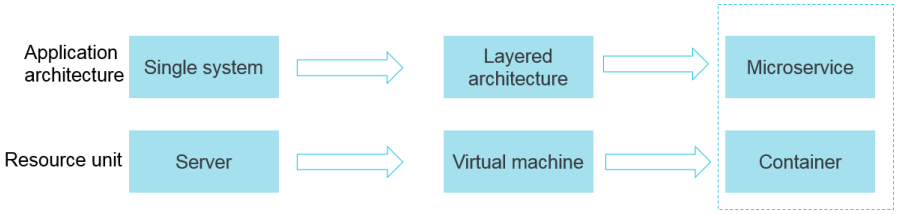
```
histogram(status) as status_pv,
histogram(user_agent) as user_agent_pv,
avg(request_time) as avg_latency,
max(request_time) as max_latency
```

After obtaining the preceding data, you can make targeted and detailed assessments on the access status of this website.

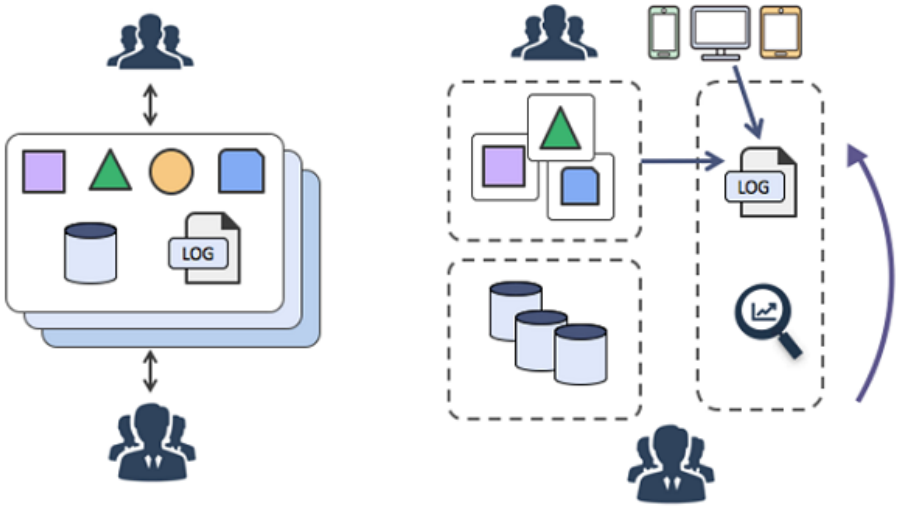
Access - Log4j/Logback/Producer Lib

In recent years, the advent of stateless programming, containers, and serverless programming greatly increased the efficiency of software delivery and deployment. In the evolution of the architecture, you can see the following two changes:

- The application architecture is changing from a single system to microservices. Then, the business logic changes to the call and request between microservices.
- In terms of resources, traditional physical servers are fading out and changing to the invisible virtual resources.



The preceding two changes show that behind the elastic and standardized architecture, the Operation & Maintenance (O&M) and diagnosis requirements are becoming more and more complex. Ten years ago, you could log on to a server and fetch logs quickly. However, the attach process mode no longer exists. Currently, we are facing with a standardized black box.



To respond to these changes, a series of DevOps-oriented diagnosis and analysis tools have emerged. For example, centralized monitoring and log system, and Software as a Service (SaaS) deployment and monitoring.

Centralizing logs solves the preceding issue. Then, logs generated by applications are transmitted to a centralized node server in real time (or quasi-real time). For example, the centralized storage of syslog, Kafka, ELK, and Hbase data is a common mode of centralizing logs.

Advantages of centralization

- Ease of use: Using Grep to query stateless application logs is troublesome. In the centralized storage, the previous long process is replaced by running a search command.
- Separated storage and computing: When customizing machine hardware, you do not have to consider the storage space for logs.
- Lower costs: Centralized log storage can perform load shifting to reserve more resources.
- Security: In case of hacker intrusion or a disaster, critical data is retained as the evidence.

Collector (Java series)

Log Service provides more than 30 data collection methods and comprehensive access solutions for

servers, mobile terminals, embedded devices, and various development languages. Java developers need the familiar log frameworks: Log4j, Log4j2, and Logback Appender.

Java applications currently have two mainstream log collection solutions:

- Java programs flush logs to disks and use Logtail for real-time collection.
- Java programs directly configure the Appender provided by Log Service. When the program is running, logs are sent to Log Service in real time.

Differences between these two solutions

	Flush logs to disks + Use Logtail to collect logs	Use Appender for direct transmission
Real time	Logs are flushed to files and collected by using Logtail	Logs are directly sent to Log Service
Throughput	High	High
Breakpoint transmission	Supported. Depends on the Logtail configuration	Supported. Depends on the memory size
Sensitive to application location	Required when configuring the collection machine group	Not required. Logs are initiatively sent
Local logs	Supported	Supported
Disable collection	Delete Logtail configuration	Modify Appender configuration and restart the application

By using Appender, you can use Config to complete real-time log collection easily without changing any code. The Java-series Appender provided by Log Service has the following advantages:

- Configuration modifications take effect without modifying the program.
- Asynchrony + breakpoint transmission: I/O does not affect main threads and can tolerate certain network and service faults.
- High-concurrency design: Meets the writing requirements for massive logs.
- Supports context query: Supports precisely restoring the context of a log (N logs before and after the log) in the original process in Log Service.

Overview and usage of Appender

The currently provided Appenders are as follows. The underlying layers all use aliyun-log-producer-java to write data.

- aliyun-log-log4j-appender
- aliyun-log-log4j2-appender
- aliyun-log-logback-appender

Differences

Appender name	Description

aliyun-log-log4j-appender	Developed for Log4j 1.x. If your application uses the Log4j 1.x log framework, we recommend that you use this Appender.
aliyun-log-log4j2-appender	Developed for Log4j 2.x. If your application uses the Log4j 2.x log framework, we recommend that you use this Appender.
aliyun-log-logback-appender	Developed for Logback. If your application uses the Logback log framework, we recommend that you use this Appender.
aliyun-log-producer-java	The LogHub class library used for high-concurrency log writing, which is programmed for Java applications. All the provided Appender underlying layers use this Appender to write data. This highly flexible Appender allows you to specify the fields and formats of the data written to LogHub. If the provided Appender cannot meet your business needs, you can develop a log collection program based on this Appender as per your needs.

Step 1 Access Appender

Access the Appender by following the steps in aliyun-log-log4j-appender.

The contents of the configuration file log4j.properties are as follows:

```
log4j.rootLogger=WARN,loghub

log4j.appender.loghub=com.aliyun.openservices.log.log4j.LoghubAppender

#Log Service project name (required parameter)
log4j.appender.loghub.projectName=[your project]
#Log Service Logstore name (required parameter)
log4j.appender.loghub.logstore=[your logstore]
#Log Service HTTP address (required parameter)
log4j.appender.loghub.endpoint=[your project endpoint]
#User identity (required parameter)
log4j.appender.loghub.accessKeyId=[your accesskey id]
log4j.appender.loghub.accessKey=[your accesskey]
```

Step 2 Query and analysis

After configuring the Appender as described in the previous step, logs generated by Java applications are automatically sent to Log Service. You can use LogSearch/Analytics to query and analyze these logs in real time. See the sample log format as follows.

Logs that record your logon behavior:

```
level:  INFO
location: com.aliyun.log4jappendertest.Log4jAppenderBizDemo.login(Log4jAppenderBizDemo.java:38)
message: User login successfully. requestID=id4 userID=user8
thread: main
time: 2018-01-26T15:31+0000
```

Logs that record your purchase behavior:

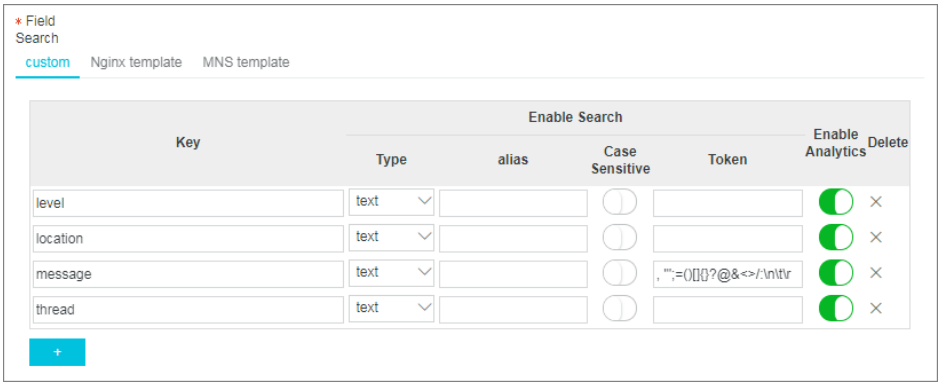
```
level:  INFO
location: com.aliyun.log4jappendertest.Log4jAppenderBizDemo.order(Log4jAppenderBizDemo.java:46)
message: Place an order successfully. requestID=id44 userID=user8 itemID=item3 amount=9
thread: main
time: 2018-01-26T15:31+0000
```

Step 3 Enable query and analysis

You must enable the query and analysis function before querying and analyzing data. Follow these steps to enable the function:

1. Log on to the Log Service console.
2. Click the project name on the **Project List** page.
3. Click **Search** at the right of the Logstore.
4. Click **Enable** in the upper-right corner. If you have enabled the index before, click **Index Attributes** > **Modify**. The **Search & Analysis** page appears.

   Enable the query for the following fields.



Step 4 Analyze logs

See the following five analysis examples:

Count the top three locations where errors occurred most commonly in the last hour.

```
level: ERROR | select location ,count(*) as count GROUP BY  location  ORDER BY count DESC LIMIT 3
```

Count the number of generated logs for each log level in the last 15 minutes.

```
| select level ,count(*) as count GROUP BY level ORDER BY count DESC
```

Query the log context.

You can precisely restore the context information of any log in the original log file. For more information, see Context query.

Count the top three users who have logged on most frequently in the last hour.

```
login | SELECT regexp_extract(message, 'userID=(?<userID>[a-zA-Z\d]+)', 1) AS userID, count(*) as count
GROUP BY userID ORDER BY count DESC LIMIT 3
```

Count the total payment of each user in the last 15 minutes.

```
order | SELECT regexp_extract(message, 'userID=(?<userID>[a-zA-Z\d]+)', 1) AS userID,
sum(cast(regexp_extract(message, 'amount=(?<amount>[a-zA-Z\d]+)', 1) AS double)) AS amount
GROUP BY userID
```