Alibaba Cloud Log Service

プロダクト概要

Document Version20190712

目次

1 Log Service とは

ログデータのワンストップサービスとして、Log Service (Log) は Alibaba Group のビッグデータシナリオで実証済みです。Log Service を使用することにより、開発することなく、迅速にログデータを収集、処理、送信、照会/分析することができます。保守管理 (O&M) と運用の効率が上がり、DT (データ技術) 時代において膨大なログを処理できます。

Log Service のラーニングパス

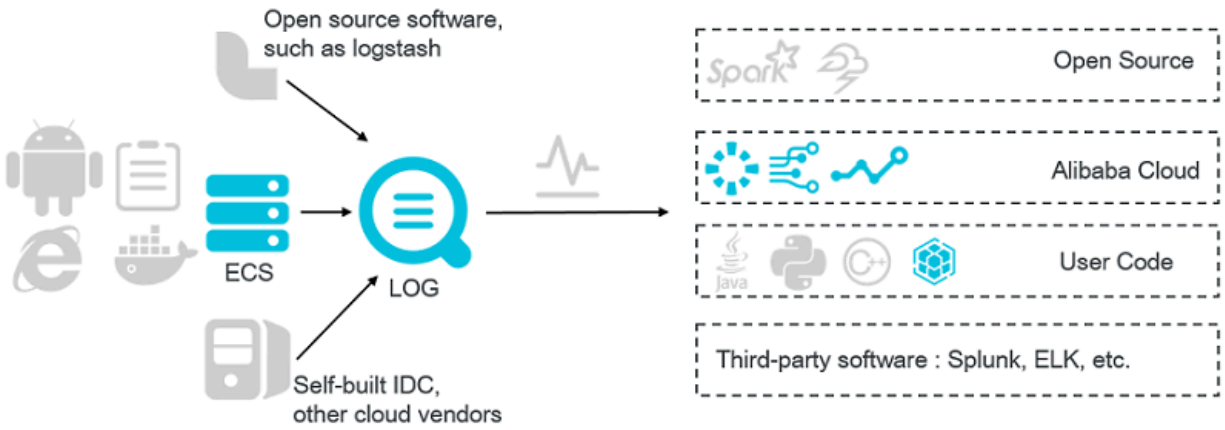
Log Service のラーニングパスは、推奨機能に関するドキュメントであり、Log Service に関する知識を迅速に蓄積するのに役立ちます。ドキュメントおよびビデオが用意されており、習得しやすくなっています。

リアルタイムにログを収集/処理 (LogHub)

機能：

- Elastic Compute Service (ECS)、コンテナ、モバイル端末、オープンソースソフトウェア、JS を使用してリアルタイムログデータ (Metric、Event、 BinLog 、TextLog、Click など) にアクセスします。
- リアルタイム処理やリアルタイムサービスと連携できるよう、リアルタイム処理インターフェイスが用意されています。

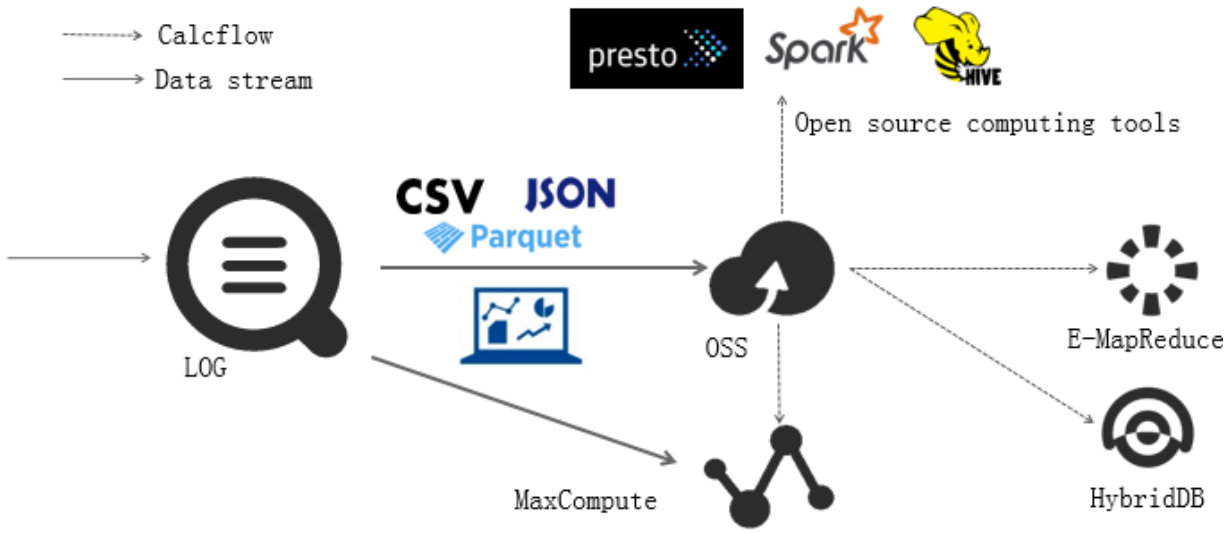
目的： ETL、ストリーム処理、モニタリングとアラーム、機械学習、反復計算。



LogShipper

安定した信頼性の高いログ送信 (shipping) により、 LogHub のデータはストレージサービスに送信、格納され、ビッグデータを分析できます。。 圧縮、カスタム化パーティション、行ストレージ、列ストレージといったさまざまなストレージ方法に対応しています。

目的：データウェアハウス+ データ分析、モニタリング、推奨システム、ユーザープロファイリング

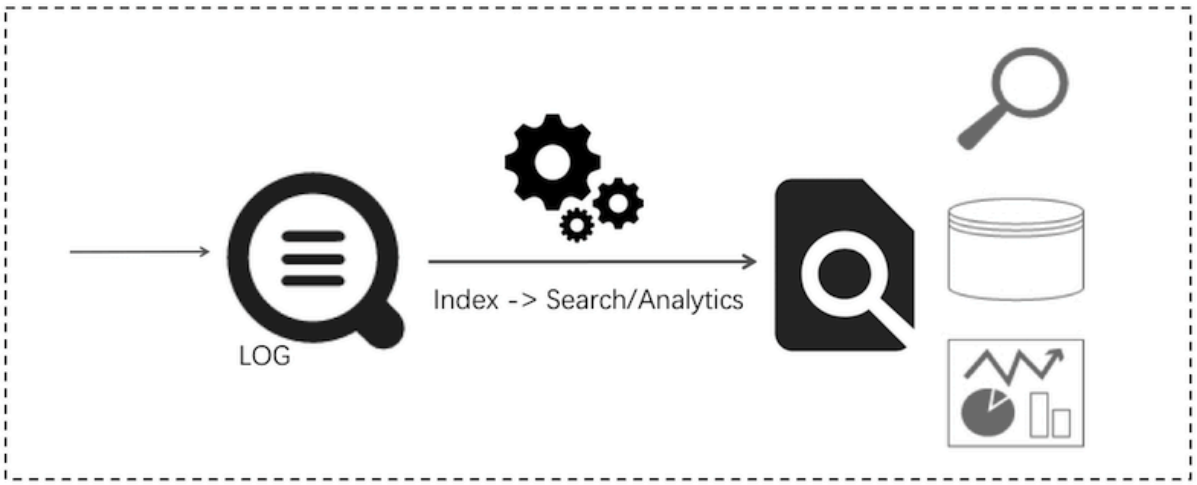


リアルタイムに照会/分析（照会/分析)

リアルタイムにデータのインデックスが作成され、照会、分析できます。

- クエリ: キーワード検索、あいまい検索、コンテキスト検索、および範囲検索
- 統計: SQL 集約といった豊富なクエリ方法
- 視覚化: ダッシュボードおよびレポート機能
- 相互接続: Grafana および JDBC/SQL9

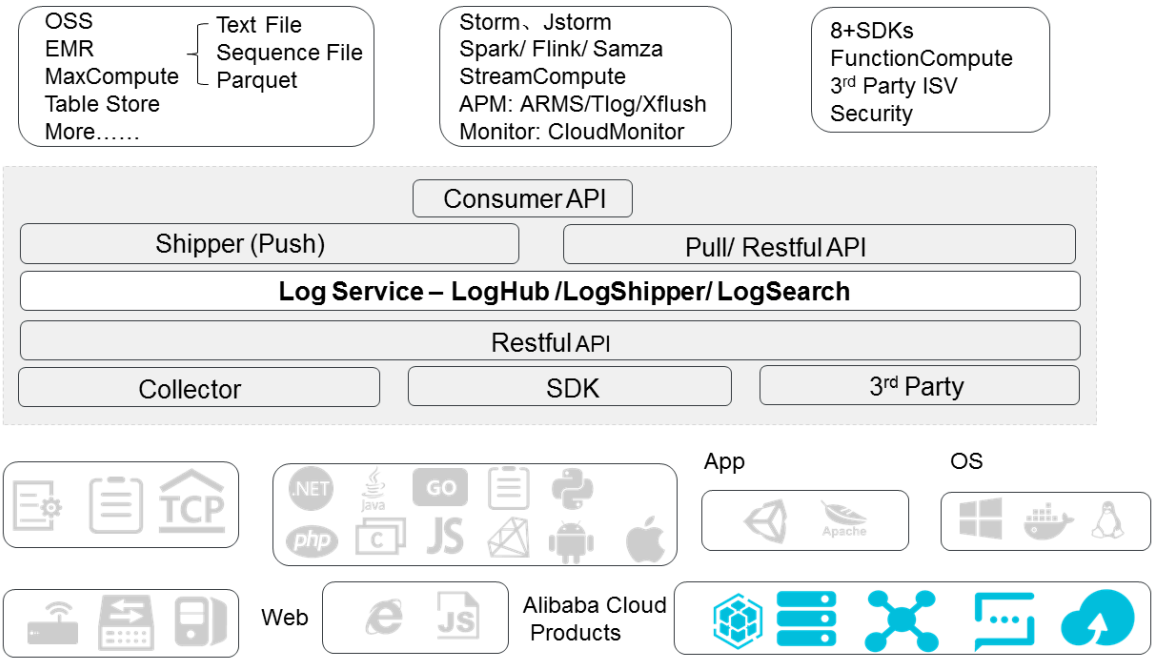
目的: DevOps /オンライン O&M 、リアルタイムなログデータ分析、セキュリティ診断と分析、運用とカスタマーサービスシステム



2 製品アーキテクチャ

Log Service システムアーキテクチャは、下図のとおりです。

図 2-1 : 製品アーキテクチャ



Logtail

Logtail は、ログを迅速に収集するのに役立つエージェントで、次の機能があります。

- ログファイルから排他制御でログ収集
  - 読み取りファイルのみ
  - 読み取り処理中は排他制御
- 安全で信頼性が高い
  - ログローテーションにより、データを損失することがない
  - ローカルにキャッシュ
  - ネットワークの例外があった場合には再試行
- 便利な管理
  - Web 上で管理
  - 視覚化の設定可

- 包括的なセルフプロテクション
  - プロセスの消費 CPU と消費メモリをリアルタイムにモニタリング
  - メモリ使用量に上限あり

フロントエンドサーバー

フロントエンドサーバは、 LVS + Nginx で構築されており、次の機能があります。

- HTTP プロトコルおよび REST プロトコル
- 水平スケーリング
  - トラフィック増量時に水平スケーリング
  - フロントエンドサーバーを追加することにより、処理能力の向上に迅速に対応
- 高スループットで低遅延
  - 純粋な非同期処理 - リクエストの例外により、他のリクエストに影響を及ぼさない
  - ログ専用 Lz4 圧縮の採用により、各マシンの処理能力を向上し、ネットワーク帯域幅を削減

バックエンドサーバー

バックエンドは、複数のマシンに処理が分散されます。Logstore のデータは、MaxCompute にリアルタイムに保存、インデックス作成、クエリ、転送します。バックエンドサービス全体は次のように機能します。

- 高いデータセキュリティ
  - 各ログを 3 か所に格納
  - ディスクの損壊、マシンのハードウェア/ソフトウェアにシステムエラーが発生した場合、データを自動複製、自動修復
- 安定したサービス
  - プロセスのクラッシュや長時間マシン応答のない場合は、自動的に Logstore を移行
  - 自動 Server Load Balancer により、トラフィックを各マシンに均等に分配
  - 異常なユーザー操作が、他のユーザーに影響を及ぼすことのないよう厳格に制限
- 水平スケーリング
  - シャード単位に水平スケーリング
  - スループット向上に、シャードを動的に追加可能

3 利点

3.1 利点

完全管理型サービス

- 優れたアクセシビリティで、サービスの利用開始までに 5 分とかかりません。また、エージェントを使用して、あらゆるネットワーク環境でデータを収集できます。
- LogHub は Kafka のすべての機能を備えており、モニタリングやアラームといった有用データを提供し、自動スケーリング（PB /日単位）に対応します。 使用コストは、自社開発コストの 50% 未満に収まります。
- LogSearch/Analytics には、クエリの保存、ダッシュボード、およびアラーム機能があります。使用料は、自社開発時の 20% 未満です。
- Log Service に接続する方法は、30 以上あります。シームレスにクラウド製品 (OSS、E-MapReduce、MaxCompute、Table Store、MNS、CDN) やオープンソースソフトウェア (Storm や Spark) と連携できます。

豊富なエコシステム

- LogHub は、LogStash や Fluentd といった 30 以上ものログ収集ツールに対応しています。組み込みデバイス、Web ページ、サーバー、プログラムから簡単に接続できます。また、Spark Streaming、Storm、 CloudMonitor、ARMS といった処理システムとも連携できます
- LogShipper は、さまざまなデータ形式 (TextFile、SequenceFile、Parquet など) に対応しており、シャードをカスタム化することもできます。また、Presto、Hive、Spark、Hadoop、E-MapReduce、HybridDB といったストレージエンジンより直接データを処理することができます。
- LogSearch/Analytics は、クエリ構文と分析構文が完備されており、SQL-92 と互換性があります。JDBC プロトコルで Grafana との連携も可能です。

高性能なリアルタイム処理

- LogHub: データが書き込まれると同時に使用できます。 Logtail (データ収集エージェント) は、データを 1 秒以内に収集し、サーバー側に転送します (99.9 % のケースで)。
- LogSearch/Analytics: データが書き込まれると同時に照会し、分析することができます。複数のクエリ条件を指定した場合でも、膨大なデータから 1 秒以内に分析することができます。複数の集約条件を指定した場合も、膨大なデータから 1 秒以内に分析することができます。

完全な API / SDK

- カスタム化した管理や二次開発を行えます。
- API / SDK ですべての機能を実装できます。あらゆる言語の SDK が用意されており、容易にサービスを管理できます。
- クエリ構文と分析構文 (SQL-92 互換) はシンプルです。インターフェイスを使用して、オープンソフトウェアに簡単に接続できます (Grafana との連携も可)。

3.2 コストメリット

コストメリット

Log Serviceは、3つのログ処理シナリオにより、下記のコスト上の利点があります：

- LogHub:
  - 購入したクラウドホスト+クラウドディスクを使用してカフカを構築する場合と比較して、98%のシナリオでコスト効率の高い選択肢となります。 小規模のウェブサイトのカフカコストを30% 削減します。
  - RESTful APIの提供により、モバイルデバイス上のデータ収集をサポートし、ログ収集用のゲートウェイサーバーのコストを削減します。
  - いつでもどこにいても、メンテナンスフリーと自動スケーリングができます。
- LogShipper:
  - コード/マシンリソースが不要で、柔軟な構成、豊富な監視データ。
  - リニアスケーラビリティ（1日あたりPB級）、現在無料で利用可能。
- LogSearch/Analytics:
  - クラウドホスト+自己ビルドELKを購入するコストの15%以下で、クエリ機能とデータ処理規模を大幅に強化します。 詳しくは比較レポートをご参照ください。 上記のログ管理ソフトウェアソリューションよりも優れた選択肢としては、さまざまな ストリームコンピューティング+オフラインコンピューティングフレームワークをシームレスに統合することで、ログの妨げにならないようにします。

コスト比較

ご参考として、以下に課金モデルにおけるLog Serviceと自己構築ソリューションの比較例を記します。

LogHub (LogHub vs Kafka)

-	フォーカス	LogHub	自立ミドルウェア（Kafkaなど）
Use	New	Imperceptible	O&M required
	Expansion	Imperceptible	O&M required
	Increase backups	Imperceptible	O&M required
	Multitenancy	Quarantine	Might affect each other
#用	Internet collection (10 GB/day)	USD 2/day	USD 16.1/day
	Internet collection (1 TB/day)	USD 162/day	USD 800/day
	Intranet collection (small data size)	-	-
	Intranet collection (moderate data size)	-	-
	Intranet collection (large data size)	-	-

ログストレージとクエリエンジン

フォーカス	LogSearch	ES (Lucene Based)	NoSQL	Hive	-
Scale	Scale	PB	TB	PB	PB
Cost	Store (USD/ GB per day)	0.0115	3.6	0.02	0.035
	Write (USD/ GB)	0.35	5	0.4	0
	Query (US $/ GB)	0	0	0.2	0.3
	Speed-query	Millisecond level-second level	Millisecond level-second level	Within milliseconds	In minutes
	Speed-statistics	Weak+	Relatively strong	Weak	Strong

フォーカス	LogSearch	ES (Lucene Based)	NoSQL	Hive	-
Latency	Write-> queryable	Real time	In minutes	Real time	Ten-minute level

注：

ここでの価格比較は、ソフトウェアソリューションがECSに導入され、3つのコピーが構成されているという事実に基づいて計算されています。

詳しくはクエリスキーム(ELK_Hive)の比較-ユーザーガイドをご参照ください#unique_6

3.3 ログクエリと分析で LogSearch/Analytics とELKを比較する

リアルタイムのログ分析について語るとき、人々は ELK（Elastic、Logstash、Kibana）を使用してそれを実装することを考えます。 ELK Stack は、コミュニティで豊富なコンテンツとユースケースを蓄積してきたオープンソースのソリューションです。

Alibaba Cloud Log Service の新バージョンには、LogSearch / LogAnalytics の拡張機能が追加され、ログデータのリアルタイムのインデックス作成、クエリ、および分析をサポートし、さまざまな面でクエリのパフォーマンスとデータ量の計算を最適化します。 このドキュメントでは包括的な比較を行い、お客様の関心を持つ点を分析します。

- 使いやすさ：機能を使い始めたときのコスト。
- 機能（重要）：クエリと分析。
- パフォーマンス（重要）：単位データ量に対するクエリと分析の要件と遅延状況。
- スケール（重要）：処理可能なデータ量及びスケーラビリティ。
- コスト（重要）：同じ機能とパフォーマンスを使用するためのコスト。

使いやすさ

ログ分析システムは、以下の手順で使用されます：

1. 収集：安定した方法でデータを書き込みます。
2. 構成：データソースを構成する方法。
3. 拡張：より多くのデータソースとマシンにアクセスし、 ストレージスペースとマシンを拡張できます。
4. 使用：機能セクションに説明があります。
5. エクスポート：ストリーミングコンピューティングやバックアップ用 OSS への格納などの操作のためにデータを他のシステムに便利にエクスポートできるかどうか。

6. マルチテナント：データを他人と共有する方法、およびデータを安全に使用できるかどうか。

比較結果

項目	サブ項目	自己構築 ELK	LogSearch/ Analytics：
収集	プロトコル	Restful API	Restful API
	エージェント	Logstash / Beats / Fluentd、豊かなエコシステム	Logtail（メイン）+ その他（Logstash など）
構成	ユニット	索引を使用してログを区別する	Project + Logstore。2 つのレベルの概念を提供。 Project は名前空間と見なされ、1 つの Project に複数の Logstore を作成できます。
	属性	API + kikana	API + SDK + コンソール
拡張	ストレージ	マシンを追加してクラウドディスクを購入	操作は必要ありません
	マシン	マシンを追加	操作は必要ありません
	構成	構成管理システムを使用して Logstash を構成し、マシンに適用	構成管理システムを使用せずに、コンソールまたは API を使用して操作を実行
	収集ポイント	構成管理システムを使用して構成と Logstash をマシングループにインストール	構成管理システムを使用せずに、コンソールまたは API を使用して操作を実行
エクスポート	方法	API / SDK	API/SDK + ストリームコンピューティングエンジン（Spark、Storm、Flink、CloudMonitor）+ ストレージ（OSS）
マルチテナント	安全性	なし（非商用版）	HTTPS + 送信署名 + マルチテナント隔離 + アクセス制御

項目	サブ項目	自己構築 ELK	LogSearch/ Analytics：
	使用	同一アカウント	サブアカウント、ロール、製品、および一時的な権限付与

結論：

ELK には多くのエコシステムと書込みツールがあり、多くのインストールと設定ツールをサポートしています。 LogSearch / Analytics は、アクセス、構成、および使用法に関して高度な統合性を備えたホスティングサービスです。 通常のユーザーは、容量と並行処理問題を気にすることなく、5 分で LogSearch / Analytics にアクセスできます。 請求方法は従量課金で、自動スケーリングがサポートされています。

機能（クエリと分析）

クエリ機能により、検索条件を満たすログをすばやく検出します。 分析機能はデータの統計と計算を実行します。

たとえば、ステータスコードが 200 を超えるすべての読み取りリクエストの数とトラフィックに関する統計を IP アドレスで収集することを目的とした分析要件があるとします。 この分析要件は、2 つの操作に変換できます。指定された結果のクエリと結果に対する統計分析です。 場合によっては、クエリなしですべてのログを直接分析できます。

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

クエリ機能の比較

種類	サブ項目	自己構築 ELK	LogSearch/ Analytics
テキスト	インデックスクエリ	サポート	サポート
	単語分割	サポート	サポート
	中国語の単語分割	サポート	未サポート
	プレフィックス	サポート	サポート
	TLD	サポート	
	ファジー	サポート	サポート
	Wildcast	サポート	未サポート
数値	long	サポート	サポート

種類	サブ項目	自己構築 ELK	LogSearch/Analytics
	double	サポート	サポート
Nested	JSON クエリ	サポート	
Geo	Geo クエリ	サポート	直接サポートしていません。 指定範囲クエリを使用して同じ効果を得ることができます。
IP	IP アドレスクエリ	サポート	直接サポートしていません。 文字列クエリを使用しても同じ効果が得られます。
コンテキスト	コンテキストクエリ		サポート
	コンテキストフィルタ		サポート

Elasticsearch は、より多くのデータ型とより高度なクエリ方法をサポートしています。LogSearch/Analytics は独自の機能（たとえば、コンテキストクエリやプログラムログの拡張）を持つ一般的なクエリのほとんどをサポートします。

分析能力の比較

- ES 5.5 aggregation
- #unique_8

種類	サブ項目	自己構築 ELK	LogSearch/Analytics
インターフェイス	方法	API/SDK	API/SDK + SQL92
	その他のプロトコル		JDBC
Agg	Bucketing	サポート	サポート
	Metric	サポート	サポート
	Matrix	サポート	サポート
	Pipeline	一部サポート	フルサポート
算術演算	数値		サポート
	String		サポート
	推定		サポート

種類	サブ項目	自己構築 ELK	LogSearch/Analytics
	数理統計		サポート
	日付変換		サポート
GroupBy	Agg	サポート	サポート
	Having コンディション		サポート
ソート	ソート		サポート
Join	複数テーブルの Join		サポート

LogSearch / LogAnalytics は Elasticsearch と比較して優れた機能のセットを提供し、SQL-92 を完全にサポートします。 LogSearch / LogAnalytics は、SQL 記述シナリオで直接使用できます。

パフォーマンス

同じデータセットを使用して、データの書き込み、データクエリ、集計の観点から、自己構築の ELK と LogSearch / Analytics を比較します。

実験環境

1. テストの構成

種類	自己構築 ELK	LogSearch/Analytics
環境	Elastic Compute Service（ECS）インスタンス（4 コア， 16 GB）x 4 + 効率的な SSD クラウドディスク	-
シャード	10	10
コピー数	2	3（既定の構成、ユーザーには表示されません）

2. テストデータ

   - 5 列の double 型データ
   - 5 列の long 型データ。
   - それぞれ 256、512、768、1024、および 1280 の辞書サイズを持つ 5 列のテキスト型データ。

   上記のフィールドはランダムです。 テストログの例は次の通りです。

```
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
```

3. データセットのサイズ

   - 生データのサイズ：50 GB
   - キーを削除した生データのサイズ：27 GB（LogSearch / Analytics は、このサイズをストレージの請求単位として使用します）。
   - ログ行数：162,640,232（約 1 億 6000 万ログ）

テスト結果を書き込む

Elasticsearch は Bulk API を使用してデータをバッチで書き込み、LogSearch / LogAnalytics は PostLogstoreLogs API を使用してバッチ書き込みを実行します。 各機能の詳細は次の通りです：

種類	項目	自己構築 ELK	LogSearch/Analytics
待ち時間	平均書き込み待ち時間	40 ms	14 ms
ストレージ	一度にコピーされたデータボリューム	86 GB	58 GB
	拡張率：データ量／生データサイズ	172%	121%

 注：

請求書を生成する LogSearch / Analytics のストレージサイズには、書き込まれた圧縮生データの量（23 GB）とインデックス作成トラフィック（27 GB）が含まれ、合計で 50 GB になります。

テスト結果によると：

- LogSearch / Analytics（14 ms）は、Elasticsearch（40 ms）より書き込み待ち時間が短くなっています。
- スペース：生データのサイズは 50 GB です。 テストデータがランダムであるため、ストレージスペースが拡張してしまいます。（ほとんどの実際のシナリオでは、圧縮後のストレージスペースは生データのサイズよりも小さくなります。） Elasticsearch が占有するストレージ容量は 86 GB まで拡張され、拡張率は 172% です。 LogSearch / Analytics が占有するストレージ容量の 58% 以上あります。

読み取り（クエリ + 分析）テスト

テストシナリオ

例として 、2 つの一般的なシナリオを使用します：ログクエリと集計。 並行性がそれぞれ 1、5 、および 10 の場合の 2 case の平均待ち時間をカウントします。

1. 全データの任意の text 列に対して GROUP BY 計算を実行します。 5 つの列の avg/min/max/sum/count を計算し、値を count 順に並べ替えます。 最初の 1000件 の結果を取得します。 例:

```
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
```

2. 完全なデータの場合は、次のようなキーワードを使用してログをランダムにクエリします。 value_126。 クエリ条件と最初の 100 ログ行を満たすログの数を取得します。 例:

```
value_126
```

試験結果

種類	並行数	Elasticsearch の待ち時間（単位：秒）	LogSearch / Analytics の待ち時間（単位：秒）
case 1：解析クラス	1	3.76	3.4
	5	3.9	4.7
	10	6.6	7.2
case 2：クエリ	1	0.097	0.086

種類	並行数	Elasticsearch の待ち時間（単位：秒）	LogSearch / Analytics の待ち時間（単位：秒）
	5	0.171	0.083
	10	0.2	0.082

結果分析

- テスト結果によると、1 億 5 千万データの規模で、Elasticsearch と LogSearch / Analytics の両方が数秒以内にデータのクエリと分析を行うことができます。
- case 1（統計）では、Elasticsearch と Log Service は待ち時間に関して同じパフォーマンスレベルにあります。 SSD クラウドディスクを使用した Elasticsearch は、大量のデータを読み取るときに LogService よりもI/O が速いです。
- case 2（クエリ）では、 LogSearch / Analytics は Elasticsearch よりも待ち時間がはるかに短いです。 並行性の増加につれて、ELK の待ち時間は増加しますが、LogSearch / Analytics の待ち時間は安定したままか、もしくは減少する傾向です。

スケール

1. LogSearch / Analytics は、1 日に数 PB のデータをインデックス付けし、一度に数秒以内に数十 TB のデータをクエリすることができ、データ量の自動スケーリングおよび水平方向のスケーリングをサポートします。
2. Elasticsearch は、1 日に GB-TB レベルのデータを書き込み、TB レベルのデータを保存するのに適用できます。主な制限は以下のとおりです：
   - 単一クラスタスケール：理想的な条件は、1 つのクラスタに約 20 台のマシンが含まれていることです。 業界では、1 つのクラスタに最大 100 のノードを含めることができます。
   - 書き込み機能の拡張：シャードの作成後に書き込み機能を変更することはできません。 スループット率が上がると、ノードは動的に拡張されます。 使用できる最大のノード数は、シャードの数です。
   - ストレージ拡張：シャードが一度にディスク容量の上限に達すると、それをより容量の大きい別のディスクにマイグレーションするか、より多くのシャードを割り振る必要があります。通常は、インデックスを作成し、さらにシャードを指定して、既存のデータを再作成することができます。

     各シャードは分散ストレージにあるため、LogSearch/Analytics には拡張の問題がありません。 スループット率が上がると、処理能力の水平方向の拡大縮小のためにシャードを動的に分割することができます。

コスト

このセクションでは、前述のテストデータに基づいて、50 GB のデータが毎日書き込まれ、90 日間保存された場合の平均月額コストを計算します（実際のデータサイズは 27 GB です）。

1. 1. Log Service LogSearch / Analytics の請求額には、読み書きトラフィック、インデックストラフィック、およびストレージスペースが含まれます。 クエリ機能は無料です。 詳細は、次をご参照ください： Billing method

請求項目	値	単価	費用（USD）
書き読みトラフィック	23 GB x 30	USD 0.2/GB	138
ストレージスペース（90 日間保管されたデータ）	50 GB x 90	USD 0.3/GB x Month	1350
インデックストラフィック	27 GB x 30	USD 0.0875/GB	283
合計	-	-	1771

2. Elasticsearch のコストには、マシンのコストとデータストレージに使用される SSD クラウドディスクのコストが含まれます。

   - 通常、クラウドディスクは高い信頼性を提供します。 そのため、コピーの保管には請求しません。
   - ストレージディスクの場合は通常、書き込まれたデータによる全容量の占有を避けるために、15 % の使用可能容量を確保する必要があります。 そのため、係数 1.15 が掛けられます。

請求項目	値	単価	費用（USD）
サーバー	4 コア、16 GB x 4（3 ヶ月）（ecs.mn4.xlarge）	毎月または年間サブスクリプションの費用：USD 675 /月	2021
ストレージ	86 * 1.15 * 90（ここでは 1 つのコピーのみが計算されます）	SSD：1 USD / GB / 月	8901
	-	SATA：0.35 USD / GB / 月	3115
合計			12943 (SSD)
			5135 (SATA)

同じパフォーマンスで、LogSearch / Analytics と ELK（SSD）のコスト比は 13.6% です。 テストプロセス中、SSD を SATA に置き換えるとコストが削減されます（LogSearch / Analytics と ELK の SATA とのコスト比は 34 % です)。 ただし、待ち時間は 40ms から150ms に増加します。長期間の読み取りおよび書き込みの後、クエリおよび読み取り/書き込みの待ち時間が大幅に増加し、クエリおよび分析機能が異常になります。

最後に

オープンソースの ELK と比較して、LogSearch / Analytics は同じクエリ速度を提供する上、より高いスループット、より強力な分析能力、及び 87% のコスト削減を提供します。 それ以外にも、従量課金と O&M 不要に対応しているため、ビジネス分析に集中できます。

LogSearch / Analytics に加えて、Log Service は LogHub と LogShipper 機能も提供し、リアルタイムのデータ収集、ストリームコンピューティングシステム（Spark、Storm、Flink）およびオフライン分析システム（E-MapReduce、Presto、Hive）との互換をサポートし、 ワンストップのリアルタイムデータソリューションを提供しております。

3.4 ログクエリソリューションの比較

DevOps シナリオでの Log Service と ELK（検索クラス）および Hadoop / Hive との比較

ソフトウェアとサービスの配信に対する需要の急増に対処するために、起業初期のチームも大手 IT 企業も DevOps モードに転換しつつあります。 開発（Dev）と保守（Ops）とのコラボレーションにより、部門を超える共同作業の実施や、顧客の要求に対する迅速な対応、及び持続的デリバリーを実現できます。

DeveOps モードでは、ログは問題調査、セキュリティ監査、運用サポートなどの面で重要なサポートの役割を果たします。 適切なログソリューションは、 DevOps にとって非常に重要です。

以下の点で LogSearch と ELK および Hadoop / Hive ソリューションを比較します。

- ログが生成後、どれくらい経てばユーザーがクエリを実行できるか
- クエリ能力：単位時間当たりにスキャンされるデータ量。
- クエリ機能：キーワードクエリ、条件組み合わせクエリ、あいまいクエリ、数値比較、コンテキストクエリ。
- トラフィックの 100 倍の増加に対する迅速な対応
- コスト：GB あたりのコスト。
- 信頼性：ログデータは安全であり、紛失しません。

一般的な解決策と比較

- 自作 ELK：比較には Elastic、Logstash、Kibana を使用します。

- オフライン Hadoop + Hive：データは Hadoop に格納され、Hive または Presto がクエリに使用されます（分析ではありません）。
- Log Service（LogSearch）を使用します。

例としてアプリケーションログと Nginx アクセスログを使用してこれらのソリューションを比較します（1 日あたり 10 GB）。

機能	ELK システム	Hadoop + Hive	Log service
クエリ完了までの待ち時間	1 ~ 60 秒（refresh_interval により制御）	数分から数時間	リアルタイム
クエリ待ち時間	1 秒未満	分単位	1秒未満
超大クエリ	数十秒から数分	分単位	秒単位（10 億ログのクエリ）
キーワードクエリ	サポート	サポート	サポート
あいまい検索	サポート	サポート	サポート
#unique_11	未サポート	未サポート	サポート
コンテキストクエリ	サポート	サポート	サポート
連続文字列クエリ	サポート	サポート	未サポート
拡張性	マシンを予め準備	マシンを予め準備	秒単位で 10 倍拡張
書込みコスト	書き込み料 5 USD / GB。 クエリは無料	書き込みは無料 1クエリーにつき 0.3 USD / GB	書き込み料 0.5 USD / GB クエリは無料
ストレージコスト	⩽ 3.36 USD / GB *日	⩽ 0.035 USD / GB *日	⩽ 0.016 USD / GB *日
信頼性	コピー数を設定する	コピー数を設定する	SLA > 99.9%、 データ > 99.99999999%

4 シナリオ

Log Service アプリケーションによくあるシナリオとして、データ収集、リアルタイム処理、データウェアハウスのオフライン分析、プロダクトの運用と分析、O&M と管理があります。一般的なアプリケーションのシナリオを以下に示します。 ベストプラクティスにもシナリオをいくつか記載されていますので、ご参照ください。

データの収集と処理

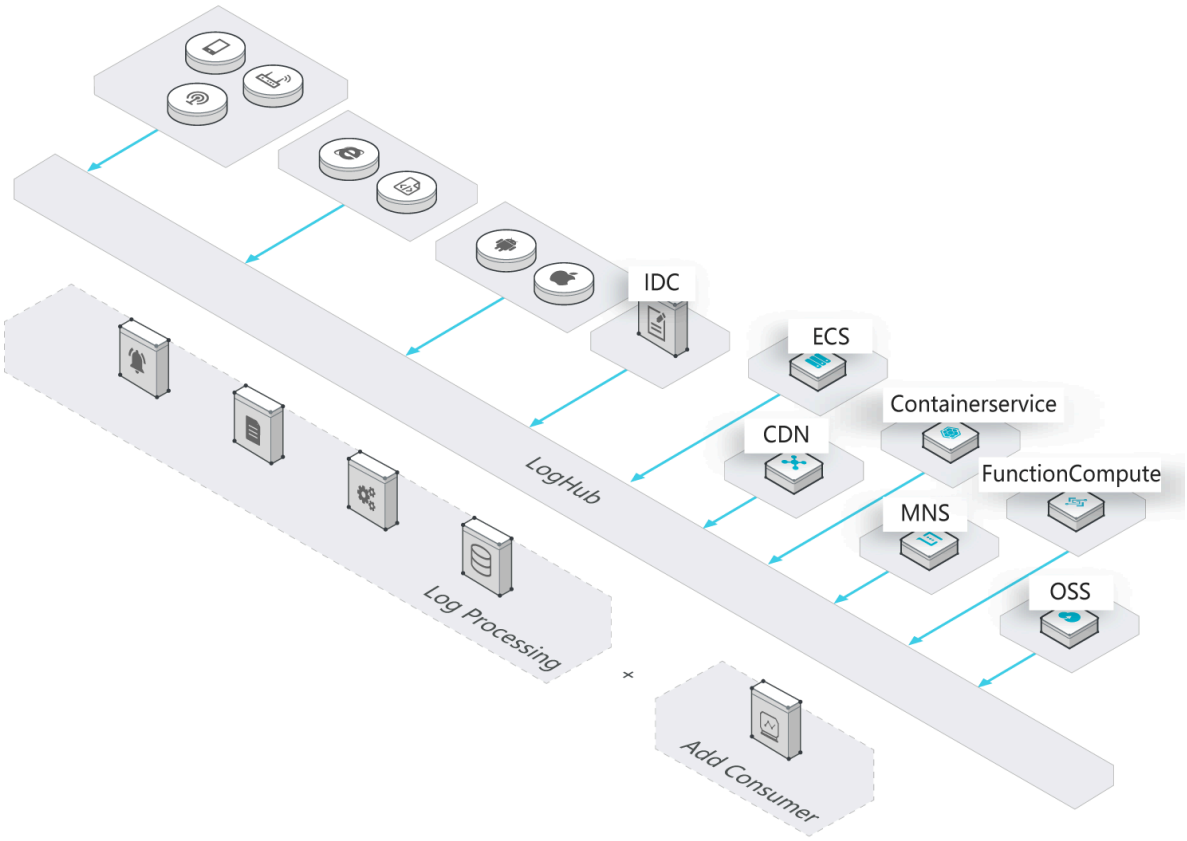
Log Service の LogHub 機能により、大量のリアルタイムログデータ (Metric、BinLog、TextLog、Click を含む) に低コストにアクセスできます。

ソリューションの利点：

- 使いやすい: リアルタイムにデータを収集する方法が 30 以上用意されており、すばやく環境を構築することができます。高性能な設定と管理機能により、保守管理にかかる負荷を軽減させることができます。中国、また、世界のあらゆるリージョンにあるノードをご利用いただけます。

・ 自動スケーリング: トラフィック量の増加やサービス拡充に容易にご対応いただけます。

図 4-1 : データの収集と処理



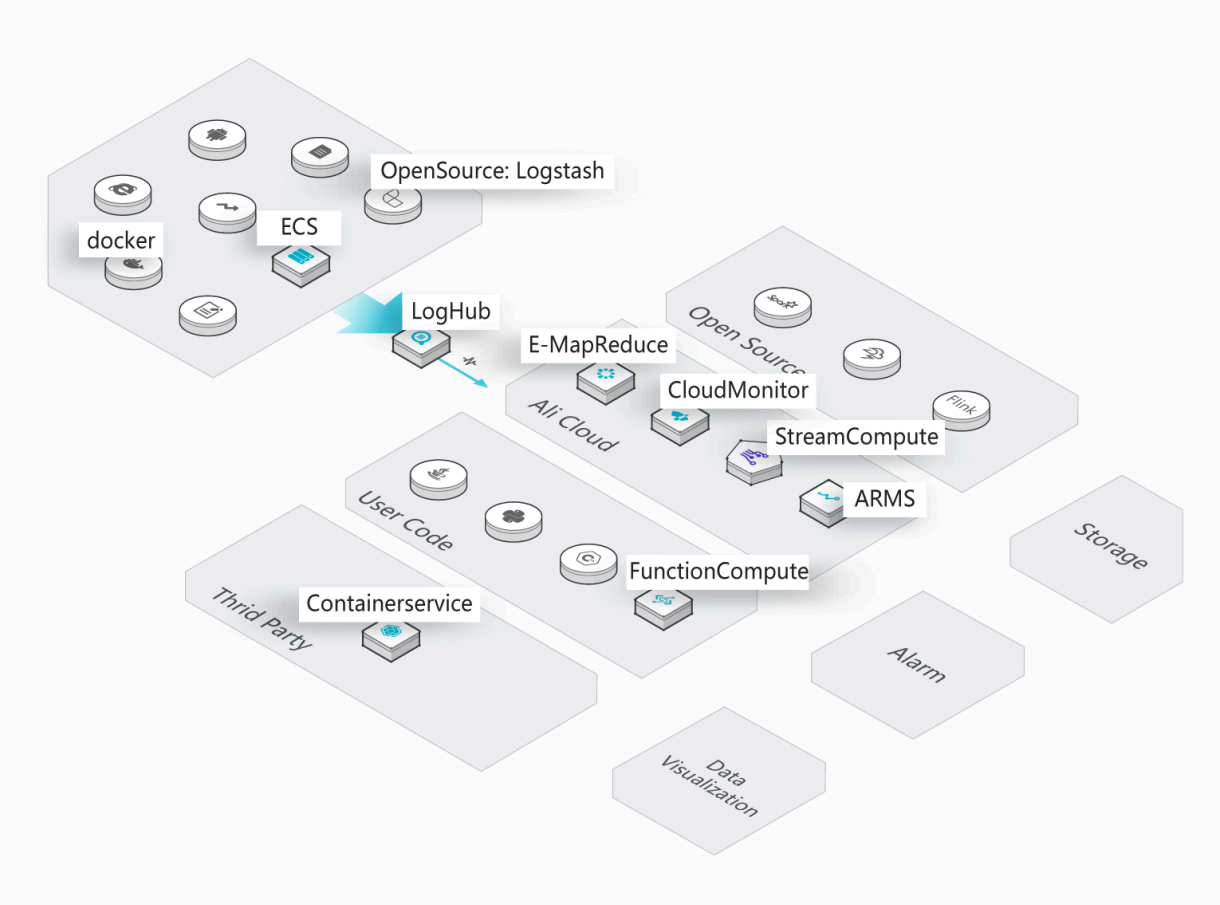
ETL/ストリーム処理

LogHub は、あらゆるリアルタイム処理サービスと組み合わせることができます。進捗モニタリングとアラーム機能を完備しており、SDK/API でカスタマイズ処理できます。

・ 簡単な操作性: さまざまな SDK とプログラミングフレームワークが用意されており、シームレスにさまざまなストリーム処理エンジンと連携できます。
・ 機能が豊富: さまざまなモニタリングデータ、アラーム機能が用意されています。

- 柔軟な拡張性: 遅延なしに PB 単位に拡張できます。

図 4-2 : データクリーニングとストリーム処理



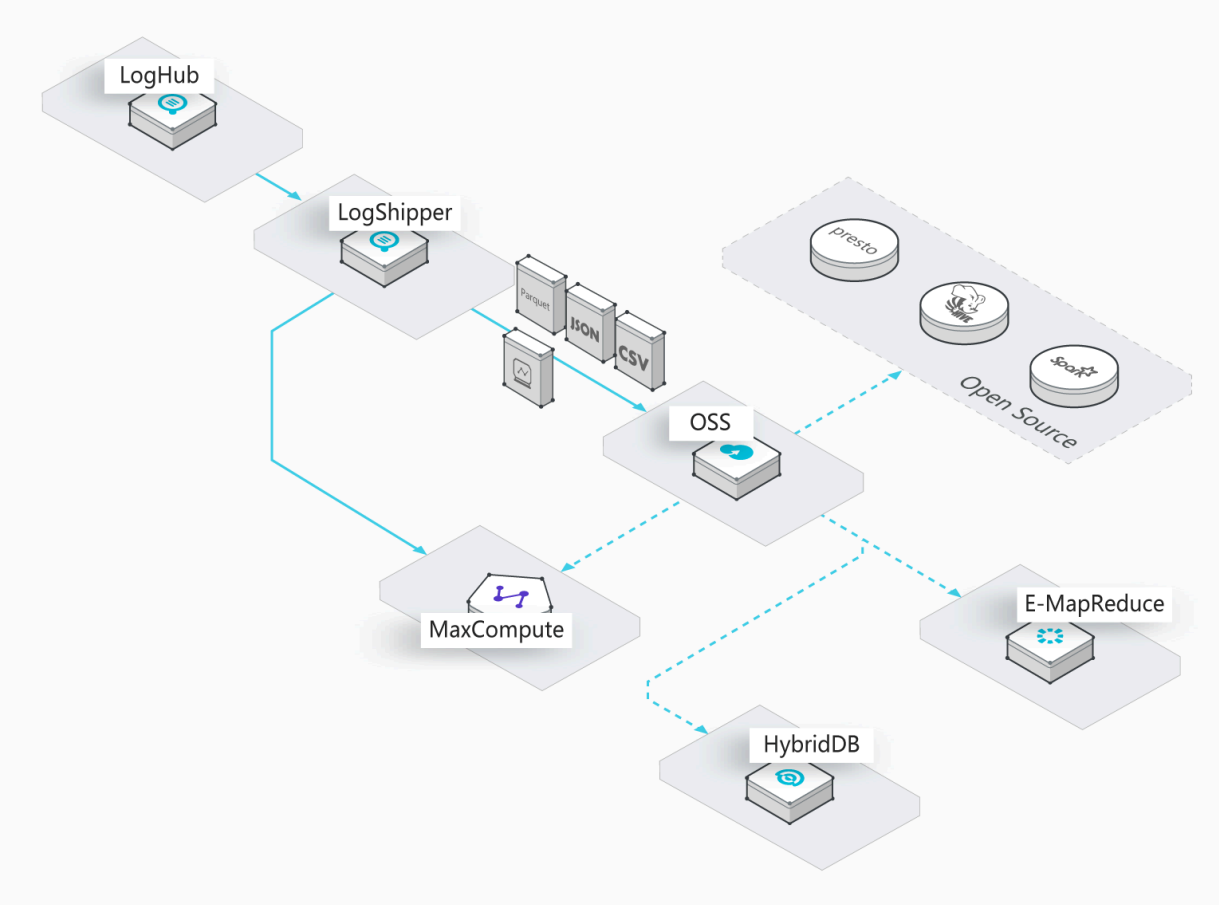
データウェアハウス

LogShipper により、LogHub 内のデータがストレージサービスに送信 (ship) されます。圧縮、カスタム化シャード、行ストレージ、列ストレージといったあらゆるストレージ形式に対応します。

- 大量のデータ: データ量に上限なし
- 豊富なストレージ形式: 行ストレージ、列ストレージ、TextFile といった、さまざまなストレージ形式に対応

- 柔軟に設定: シャードのカスタム化といった設定が可能

図 4-3 : データウェアハウスにドッキング



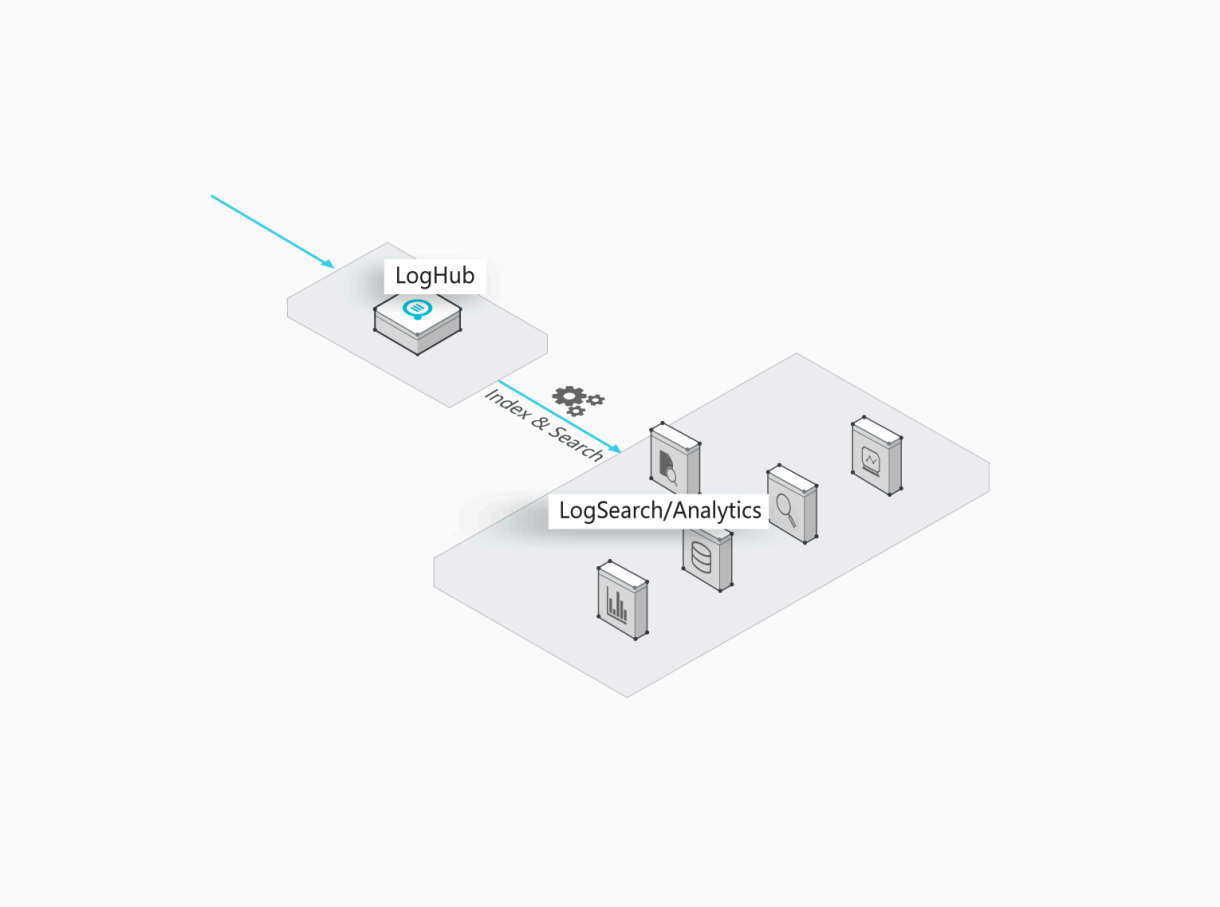
ログをリアルタイムに照会/分析

LogAnalytics は、LogHub のデータをリアルタイムにインデックスが作成され、キーワード検索、あいまい検索、コンテキスト、範囲指定、SQL 集約といったクエリが豊富に用意されています。

- 高性能にリアルタイム: データが書き込まれたと同時にクエリ可能
- 大容量で低コスト: PB/日でインデックスが作成され、独自開発する場合と比較して 15 % 割安

- 高度に分析: あらゆるクエリが行えます。また、視覚化とアラーム通知機能が用意されています。

図 4-4 : リアルタイムなログ照会/分析



5 基本概念

5.1 概要

ログ

ログは、プロセス実行によってシステムに加えられた変更を要約したものです。ログの中身は、特定のオブジェクトに対して行なった処理を時系列に集まっています。LogFile、Event、BinLog、および Metric のデータは、それぞれ異なるログに格納されています。各ログファイルはログを 1 つ以上含み、各ログにはシステムイベントが 1 つ記述されています。Log Service で処理されるデータの最小単位がログです。

ロググループ

ロググループはログを集めたものであり、書き込みと読み取りの基本単位です。

ログトピック

Logstore 内のログは、ログトピック別に分類することができます。ログを書き込む際はトピックを指定することができます。照会の際は、トピックは必ず指定する必要があります。

プロジェクト

プロジェクトは、 Log Service におけるリソース管理の単位です。リソースをそれぞれ別々に管理することができます。プロジェクトにより、アプリケーションのログと関連するログソースをすべて管理できます。 プロジェクトで、Logstore 情報、ログ収集マシンの設定情報のすべてを管理し、Log Service リソースにアクセスするためのポータルとしての役割を果たします。

Logstore

Logstore は、Log Service におけるログデータの収集、保存、およびクエリの単位です。各 Logstore はプロジェクトに属し、各プロジェクトには複数の Logstore を作成することができます。

シャード

各 Logstore はいくつかのシャードに分割され、各シャードの範囲は MD5 左閉右開区間です。各区間は重複することがなく、全シャードの範囲は、すべての MD5 区間を合わせたものとなります。

5.2 ログ

半世紀前までは「ログ」と言えば、船長や操業者が書き込んだ分厚いノートが連想されたものです。コンピュータの出現に伴い、今日ではログはあらゆるところで生成され、利用されています。 サーバー、ルーター、センサー、GPS デバイス、発注、さまざまな IoT デバイスの書き出すログによって、さまざまな角度から我々の生きている世の中が写し出されます。 強力な計算処理能力により、ログを収集、処理、利用し、我々は絶えず変化する世界、システム全体に対する認識を新たにしています。

ログとは

タイムスタンプの記録に加え、ログには、テキスト、画像、気象条件、航路といったあらゆる情報が含まれます。数世紀が経ち、「船長のログ」は、発注、支払い記録、ユーザーアクセス、データベース操作といったさまざまな分野に広がりました。

ログが広く使われ、使い続けられているのは、最も端的に記録できるためです。 ログは、追加することのみ可能な時系列な記録の集まりです。ログ (時系列データ) を図示すると、下図のようになります。

図 5-1 : ログ

記録は、ログの最後尾に追加されていき、記録されたログは左から右に読み込まれます。各記録には、固有のログ記録番号が順に付番されます。

ログの順序は「時間」で決定されます。上図より、ログは右から左に時系列に並べられることがわかります。新たにイベントが記録されると、古いイベントは徐々に消えていきます。ログは、イベントを記録したものです。コンピュータ、人間、または世界全体の認識、意味づけの基礎となります。

ログサービスのログ

ログは、プロセス実行によってシステムに加えられた変更を要約したものです。ログの中身は、特定のオブジェクトに対して行なった処理を時系列に集まっています。LogFile、Event、BinLog、および Metric のデータは、それぞれ異なるログに格納されています。各ログファイルはログを 1 つ以上含み、各ログにはシステムイベントが 1 つ記述されています。Log Service で処理されるデータの最小単位がログです。

Log Service では、半構造化データモードでログは定義されます。このモードには、トピック、時間、コンテンツ、およびソースとデータ項目が 4 つあります。

また、Log Service では、ログのデータ項目ごとに形式が異なります。 詳細については、下表をご参照ください。

データ項目	意味	形式
Topic	ログに印を付けてログを分類するためのフィールド。たとえば、アクセスログをサイト別に印付けることができます。	128 バイト以 空文字列)。
Time	ログの予約フィールドで、ログの作成された時間を示すために使用されます。通常、ログのに基づいて直接生成されます。	UNIX の標準 秒数、単位:
Content	ログの内容を記録するために使用されます。キーと値を組み合わせた項目 1 つ以上で構成されます。	Key は UTF- (英数字およ ワードはキー ・ __time_ ・ __sourc ・ __topi ・ __part ・ _extra ・ __extra 値は、1024
Source	ソースログ。 たとえば、ログを生成するマシンの IP アドレス。	128 バイト以

実際の利用場面においては、さまざまなログフォーマットがあります。Nginx アクセスログを Log Service のログデータモデルにマッピングする方法をご説明します。運用中の Nginx サーバーの IP アドレスが 10.249.201.117 であった場合、元のサーバーログは、次のようになります。

```
10 . 1 . 168 . 193  - - [ 01 / Mar / 2012 : 16 : 12 : 07  + 0800 ] "
GET  / Send ?  AccessKeyI  d = 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 "
```

次のようにソースログを Log Service のログデータモデルにマッピングします。

データフィールド	値	説明
Topic	“”	初期値（空文字列）が使用されています。
Time	1330589527	ログの生成された時刻(1970-1-1 00:00:00 UTC からの秒数)。 ソースログのタイムスタンプが変換されています。

データフィールド	値	説明
Content	キーと値の組み合わせ	ログ内容
Source	“10.249.201.117”	ログのソースに、サーバーの IP アドレスを使用します。

ソースログのコンテンツから、キーと値の組み合わせをどのように抽出するかを定義します。下表はその一例です。

キー	値
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”

ロググループ

ロググループは、ログの集まりであり、書き込みと読み取りの基本単位です。

ロググループの上限は、4096 ログまたは 10 MB です。

図 5-2 : ロググループ

5.3 プロジェクト

プロジェクトは、Log Service におけるリソース管理の単位です。各リソースをそれぞれ管理することができます。プロジェクトで、アプリケーションのログと関連するソースログをすべて管理できます。Logstore 情報やログ収集のためにマシンに設定した情報は、すべてプロジェクトで管理します。プロジェクトは、Log Service リソースにアクセスするためのポータルとしての役割を果たします。

プロジェクトの機能は、次のとおりです。

- プロジェクトは、複数の Logstore の整理および管理に役立ちます。Log Service を実際に使用する際、複数のプロジェクト、複数のプロダクト、また、さまざまな環境のログを一元管理する必要が出てきます。さまざまなログをプロジェクトごとに分類し、管理することができま

す。後々のログ読み込み、エクスポート、インデックス作成が容易になります。また、アクセスの権限管理もプロジェクトで行うことができます。

- プロジェクトは、Log Service のリソースにアクセスするためのポータルとしての役割を果たします。なお、プロジェクトごとに固有のアクセスポイントが割り当てられます。アクセスポイントが割り当てられることで、ネットワークを介してログを書き込み、読み込み、管理することができます。

5.4 Logstore

Logstore は、ログデータを収集、格納、および照会する際に使用する Log Service における単位です。各 Logstore はプロジェクトに属します。また、各プロジェクトには Logstore を複数作成することができるため、プロジェクトに必要な数だけ Logstore を作成することができます。通常は、アプリケーションの各ログごとに 1 つの Logstore を作成します。 たとえば、「big-game」というゲームアプリがあり、サーバーに operation_log、application_log、および access_log の 3 種類のログがあるとします。まず「big-game」という名のプロジェクトを作成します。作成したプロジェクト下には 3 種類のログそれぞれに Logstore を作成してログを収集、格納、および照会できるようにします。

ログに書き込む際、または、ログを照会する際は、Logstore を指定します。 オフライン分析にログデータを MaxCompute に転送する場合も、Logstore 単位でデータを同期して、Logstore 内のログデータが MaxCompute のテーブルに転送されるようにします。

Logstore の機能は、次のとおりです。

- ログの収集 - リアルタイムなロギング
- ログの格納 - リアルタイムなログ読み込み
- インデックスの作成 - リアルタイムにログを照会
- MaxCompute に送信するためのデータチャネルあり

5.5 シャード

Logstore 内の読み取り/書き込みログは、特定のシャードに格納される必要があります。各 Logstore はいくつかのシャードに分割され、各シャードは、MD5 左閉右開区間です。各区間が重複することはなく、すべての区間を合計すると、MD5 値すべてを合わせたものとなります。

範囲

Logstore を作成する際に、シャード数を指定します。指定したシャード数をもとに、全 MD5 値の合計範囲が均等に自動分割されます。各シャードの範囲には区間があり、MD5 モードで表記

し、[00000000000000000000000000000000、ffffffffffffffffffffffffffffffff] の範囲内である必要があります。

シャードの範囲はすべて左閉右開区間であり、次のキーで構成されます。

- BeginKey: シャードの開始を示します。 キーはシャードの範囲内です。
- EndKey: シャードの終わりを示します。 キーはシャードの範囲外です。

シャードに範囲があることで、ハッシュキーを指定してログを書き込み、シャードを分割または結合することができます。なお、データの読み取りには、対応するシャードを指定する必要があります。 ロードバランシングモードまたは指定されたハッシュキーモードを使用してシャードにデータを書き込むことができます。 ロードバランシングモードでは、データパケットは利用可能なシャードにランダムに書き込まれます。 指定されたハッシュキーモードでは、指定されたキーを含む範囲のデータがシャードに書き込まれます。シャードからデータを読み取るには、該当するシャードを指定します。シャードにデータを書き込むには、Server Load Balancer を使用するかハッシュキーを指定します。Server Load Balancer を使用すると、各データパケットは利用可能なシャードにランダムに書き込みます。ハッシュキーを指定することにより、データは指定されたキーの範囲内にあるシャードに書き込まれます。

Logstore にシャードが 4 つあり、この Logstore の MD5 値が [00、ff] である場合、各シャードの範囲は次のようになります。

シャード番号	範囲
Shard0	[00,40)
Shard1	[40,80)
Shard2	[80,C0)
Shard3	[C0,FF)

ログに書き込む際に、ハッシュキーの MD5 のキーに 5F を指定すると、5F を含む shard1 にログデータが書き込まれます。 MD5 キーの値に 8C を指定すると、8C を含む shard2 にログデータが書き込まれます。

読み取り/書き込み容量

各シャードには容量に上限があります。

- 書き込み: 5 MB/秒、2000回/秒
- 読み取り: 10 MB/秒、100回/秒

シャード数を計画する際は、実際のデータトラフィックをもとにすることを推奨します。トラフィックが読み取り上限や書き込み上限を超える場合は、都度、シャードを分割し、シャード数

を増やしてシャードの読み取り/書き込み領域を増やします。シャードの読み取り書き込み上限よりもはるかに少ないトラフィックであれば、シャードを結合し、シャード数を減らすことを推奨します。レンタル料の削減が図れます。

たとえば、readwrite ステータスのシャードが 2 つあり、最大 10 MB/秒でデータを書き込むことができる場合、14 MB/秒 でリアルタイムにデータを書き込むには、その内のシャード 1 つを分割して readwrite シャード数を 3 に増やすことが推奨されます。リアルタイムにデータを 3 MB/秒で書き込むのであれば、シャードは 1 つで充分なため、2 つのシャードを 1 つに結合することを推奨します。

注 :

- 書き込みの際に、API が 403 または 500 のエラーを通知し続ける場合、Log Service のモニタリング指標を確認し、シャード数を増やす必要があるかどうかを検討します。
- シャードの上限を超える読み取り/書き込み処理となった場合は、ベストエフォートとなります。

ステータス

シャードのステータスは次のとおりです。

- readwrite: 読み取り/書き込み可能
- readonly: 読み取り専用データ

シャードのステータスは、生成時は readwrite です。シャードを分割または結合すると、シャードのステータスは readonly に変わり、新たに生成されるシャードのステータスは readwrite です。シャードのステータスは、データの読み込みパフォーマンスとは関係ありません。readwrite ステータスのシャードへのデータ書き込みパフォーマンスは変わりませんが、readonly ステータスのシャードにデータを書き込むことはできません。

シャードを分割する際は、readwrite ステータスで MD5 で ShardId を指定します。MD5 は、シャードの BeginKey より大きく、EndKey より小さくなければなりません。分割すると、1 つのシャードは 2 つのシャードに分割されます。つまり、シャードは 1 つから 2 つに増えます。指定した元のシャードのステータスは、分割が完了すると、readwrite から readonly に変わります。データを読み取ることはできますが、新たにデータを書き込むことはできません。新たに生成された 2 つのシャードは、readwrite ステータスで、元のシャードの後ろに追加されます。2 つのシャードの MD5 の範囲を合わせたものは、元のシャードの範囲に含まれます。

シャードを結合する際は、readwrite ステータスのシャードを指定します。なお、readwrite ステータスの最後のシャードを指定することはできません。サーバーは指定したシャードとその右隣りのシャードを自動的に結合します。指定したシャードとその右隣りのシャードのステータス

は、結合完了時は readonly です。データを読み取ることはできますが、新たにデータを書き込むことはできません。readwrite ステータスのシャードが新たに生成され、MD5 の範囲は、元の 2 つのシャードの全範囲を含んだものになります。

5.6 ログトピック

Logstore 内のログは、ログトピックで分類することができます。トピックは、ログに書き込む際、また、ログを照会する際に指定します。 たとえば、Log Service にログを書き込む際、ログトピックに Log Service ユーザー ID を指定することができます。そうすることで、ログを照会する際に、特定のログトピックのログのみを表示させることができます。Logstore でログを分類する必要がない場合は、すべてのログに同じトピックを使用します。

注 :

ログトピックに、空の文字列を使えます。ログの書き込みや照会の際、空の文字列がデフォルトのログトピックとなります。したがって、ログの書き込みや照会に、ログトピックの必要がない場合、デフォルトのログトピックである空の文字列を使用するのが、最も簡単な方法です。

Logstore、ログトピック、およびログの関係は、下図のとおりです。

6 制約事項

6.1 基本リソース

リソース	制限	注意事項
プロジェクト	各アカウントで最大 50 のプロジェクトを作成可能	さらに作成する必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。
Logstore	各プロジェクトに最大 200 の Logstore を作成可能	さらに作成する必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。
シャード	・ 各プロジェクトに最大 200 のシャードを作成可能 ・ 各 Logstore に最大 10 のシャードを作成可能 (シャードを分割するとシャード数は増えます)	さらに作成する必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。
LogtailConfig	各プロジェクトに最大 100 の LogtailConfig を作成可能	さらに作成する必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。
ログの保存期間	無期限に保存可 保存時間の指定も可能（指定可能な値: 1~3000）	-
マシングループ	各プロジェクトに最大 100 のマシングループを作成可能	さらに作成する必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。
コンシューマグループ	各Logstore に最大 10 のコンシューマグループを作成可能	不要になったコンシューマグループは削除できます。
クイッククエリ	各プロジェクトに最大 100 のクイッククエリを作成可能	-

リソース	制限	注意事項
ダッシュボード	・各プロジェクトに最大 50 のダッシュボードを作成可能 ・各ダッシュボードに最大 50 の分析グラフを作成可能	-
LogItem	各 LogItem の上限: 1 MB	上限 1 MB は、API パラメータの場合のみであり、Logtail によるログ収集の場合の上限は 512 KB です。
LogItem (Key)	上限: 128 Byte	-
LogItem (Value)	上限: 1 MB	-
ロググループ	各ロググループに最大 4096 のログを作成可能 (各ログは 10 MB 以内)	-

6.2 データの読み取りと書き込み

リソース	制限	説明	ご注意
プロジェクト	書き込みトラフィック保護	書き込みトラフィックは最大 30 GB/秒。	上限を超える場合は、ステータスコード 403 が返され、「Inflow Quota Exceed」と表示されます。 上限を上げる必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。
	書き込み回数保護	書き込み回数は最大 600000 回/分。	上限を超える場合、ステータスコード 403 が返され、「Write QPS Exceed」と表示されます。 上限を上げる必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。

リソース	制限	説明	ご注意
	読み取り回数保護	読み取り回数は最大600000 回/分。	上限を超える場合、ステータスコード 403 が返され、「Read QPS Exceed」と表示されます。 上限を上げる必要のある場合は、チケットを起票し、サポートセンターにお問い合わせください。
シャード	書き込みトラフィック	書き込みトラフィックは最大 5 MB/秒。	絶対的な制限ではありません。5 MB/秒を超える場合は、ベストエフォートとなります。
	書き込み回数	書き込み回数は最大500 回/秒。	絶対的な制限ではありません。 500 回/秒を超える場合、ベストエフォートとなります。
	読み取りトラフィック	読み取りトラフィックは最大10 MB/S。	絶対的な制限ではありません。 10 MB/秒を超えると、ベストエフォートとなります。
	読み取り回数	読み取り回数は最大100 回/秒	絶対的な制限ではありません。100 回/秒を超えると、ベストエフォートとなります。

6.3 照会 (検索)/分析/可視化

機能	項目	制限	備考
照会 (検索)	キーワード数	キーワード検索に指定可能なブール論理演算子以外の条件は、1 クエリにつき、最大 30 のキーワード	例:「a and b or c and d...」

機能	項目	制限	備考
	各値の長さ	10 KB 以内 (超過した部分はクエリ対象外)	値の長さが 10 KB を超えた場合、キーワードでログが見つからない可能性がありますが、データは完全です。
	各プロジェクトの並行クエリ実行数	最大 100	-
	クエリ結果の応答エントリ数	最大 100 件 (指定のない場合)	次のページに進むことにより、完全なクエリ結果を閲覧できます。
	各ログ内容の表示	10,000 字 (上限を超える場合、Web ブラウザのパフォーマンスを考慮し、Log Service は DOM ワードセグメンテーションを採用し、最初の 10,000 字のみを表示)	-
SQL 分析	各値の長さの上限	2 KB 以内 (上限を超える部分はクエリ対象外)	上限を超える場合、照会結果の精度は落ちますが、データに問題はありません。
	各プロジェクトの並行クエリ実行数	最大 30	-
	各分析結果のエントリ数	最大 100 MB、100,000 エントリの結果を応答可能	-

6.4 予約フィールド

Log Service では、一部のフィールドは予約フィールドです。 API を使用してデータをログに書き込むときや Logtail 構成を追加するときは、必須フィールドの名前を予約済みフィールドの名前と同じにすることはできません。

注意事項

ログを収集するとき、または他のクラウド製品にデータを配信するとき、Log Service はログソースやタイムスタンプなどの情報をキーバリュー形式でログに追加できます。 `__source__` など、固定名のフィールドは予約フィールドです。

- API を使用してデータをログに書き込むとき、または Logtail 構成を追加するときは、必須フィールドの名前を予約済みフィールドの名前に使用しないようご注意ください。 そうしなければ、クエリの結果が不正確になる可能性があります。
- プレフィックスが `__tag__` であるフィールドを配信することはできません。

予約フィールド

次の表では、予約フィールドについて説明します。

表 6-1 : 予約フィールド

予約フィールド	タイプ	インデックスと統計の設定	説明
`__time__`	標準の Unix 時間フォーマットの整数 例： `__time__ : 1523868463`	• インデックス設定：このフィールドは API の `from` および `to` パラメータを通じて設定できるため、このフィールドにインデックスを追加する必要はありません。 • 統計設定：デフォルトでは、他の列の統計機能が有効にされると、このフィールドの統計は自動的に有効化されます。	このフィールドは、API または SDK を使用してデータをログに書き込むときのログ生成時間を指定します。 ログ配信、クエリ、および分析に使用できます。

予約フィールド	タイプ	インデックスと統計の設定	説明
`__source__`	String	・インデックス設定：インデックス機能を有効にすると、Log Service はデフォルトでこのフィールドのインデックスを作成します。インデックスは text 型であり、区切り文字は指定されません。このフィールドをクエリする場合は、`source : 127 . 0 . 0 . 1` または `__source__ : 127 . 0 . 0 . 1` を入力します。 ・統計設定：デフォルトでは、他の列の統計機能が有効にされると、このフィールドの統計は自動的に有効化されます。	このフィールドは、ログ収集元のデバイスを指定します。ログ配信、クエリ、分析、およびカスタム消費に使用できます。
`__topic__`	String	・インデックス設定：インデックス機能を有効にすると、Log Service はデフォルトでこのフィールドのインデックスを作成します。インデックスは text 型であり、区切り文字は指定されません。このフィールドをクエリする場合は、`__topic__ : XXX` を入力します。 ・統計設定：デフォルトでは、他の列の統計機能が有効にされると、このフィールドの統計は自動的に有効化されます。	このフィールドはログトピックを指定します。ログトピックを設定した場合、Log Service は自動的にキーを `__topic__` に設定し、値を指定したトピックコンテンツとして設定したフィールドをログに追加します。このフィールドは、ログ配信、クエリ、分析、およびカスタム消費に使用できます。

予約フィールド	タイプ	インデックスと統計の設定	説明
`_extract_others_`	JSON マップに逆シリアル化できる文字列	このフィールドはどのログにも存在しないため、インデックスを追加する必要はありません。	このフィールドは `__extract__others__` と同じように機能します。`__extract__others__` を使用することを推奨します。
`__tag__:__client_ip__`	String	・インデックス設定：インデックス機能を有効にすると、Log Serviceはデフォルトですべての tags のインデックスを作成します。インデックスは text 型であり、区切り文字は指定されません。 完全一致検索とあいまい検索の両方がサポートされています。 ・統計設定：デフォルトでは、このフィールドで示される列の統計機能は無効になっています。 このフィールドの統計を有効にする場合は、フィールドのインデックスを追加してから統計機能を有効にします。	このフィールドはシステムタグで、ログ収集元のデバイスのインターネット IP アドレスを指定します。 インターネット IP アドレスの記録機能が有効になった後、サーバーはログを受信後生ログにこのフィールドを追加します。 このフィールドは、ログのクエリ、分析、およびカスタム消費に使用できます。

予約フィールド	タイプ	インデックスと統計の設定	説明
`__tag__:__receive_time__`	標準の Unix 時間形式で整数に変換できる文字列	・インデックス設定：インデックス機能を有効にすると、Log Serviceはデフォルトですべての tags のインデックスを作成します。インデックスは text 型であり、区切り文字は指定されません。 完全一致検索とあいまい検索の両方がサポートされています。 ・統計設定：デフォルトでは、この列の統計機能は無効になっています。 このフィールドの統計を有効にする場合は、フィールドのインデックスを追加してから統計機能を有効にします。	このフィールドはシステム tag で、サーバーがログを受信する時刻を指定します。 インターネット IP アドレスの記録機能が有効になった後、サーバーはログを受信後に生ログにこのフィールドを追加します。 このフィールドは、ログのクエリ、分析、およびカスタム消費に使用できます。
`__tag__:__path__`	String	・インデックス設定：インデックス機能を有効にすると、Log Service はデフォルトでこのフィールドのインデックスを作成します。インデックスは text 型であり、区切り文字は指定されません。 このフィールドをクエリする場合は、`__tag__:__path__:XXX` を入力します。 ・統計設定：デフォルトでは、他の列の統計機能が有効にされると、このフィールドの統計は自動的に有効化されます。	このフィールドは、Logtail によって収集されたログファイルパスを指定します。 Logtail は自動的にこのフィールドをログに追加します。 ログクエリ、分析、およびカスタム消費に使用できます。

予約フィールド	タイプ	インデックスと統計の設定	説明
`__tag__ : __hostname __`	String	・インデックス設定：インデックス機能を有効にすると、Log Service はデフォルトでこのフィールドのインデックスを作成します。インデックスは text 型であり、区切り文字は指定されません。このフィールドをクエリする場合は、`__tag__ : __hostname __ : XXX` を入力します。 ・統計設定：デフォルトでは、他の列の統計機能が有効にされると、このフィールドの統計は自動的に有効化されます。	このフィールドは、Logtail がデータを収集するホストの名前を指定します。Logtail は自動的にこのフィールドをログに追加します。ログクエリ、分析、およびカスタム消費に使用できます。
`__raw_log_ _`	String	このフィールドに text タイプのインデックスを追加して設定し、必要に応じて統計機能を有効にする必要があります。	このフィールドは、解析に失敗した生ログを指定します。解析失敗機能付きログの破棄を無効にした後、Logtail はログ解析が失敗すると生ログをアップロードします。このフィールドでは、Key は `__raw_log_ _` で、Valueはログの内容です。このフィールドは、ログ配信、クエリ、分析、およびカスタム消費に使用できます。

予約フィールド	タイプ	インデックスと統計の設定	説明
`__raw__`	String	このフィールドに text タイプのインデックスを追加して設定し、必要に応じて統計機能を有効にする必要があります。	このフィールドは、正常に解析された生ログを示します。生ログのアップロード 機能が有効になった後、Logtail は生ログをこのフィールドと見なし、正常に解析されたログとともにログをアップロードします。 通常、このフィールドはログ監査とコンプライアンスチェックに使用されます。ログ配信、クエリ、分析、およびカスタム消費にも使用できます。