



从ElasticSearch到SelectDB 构建新一代日志平台

肖康

飞轮科技

2024.10

目录

01 日志存储分析场景需求

02 基于 ES 的日志平台痛点

03 基于 SelectDB 的新一代日志平台

04 实践案例

1 日志存储分析场景需求

日志存储分析的典型应用场景



可观测性

保障服务稳定 提升用户体验



网络安全

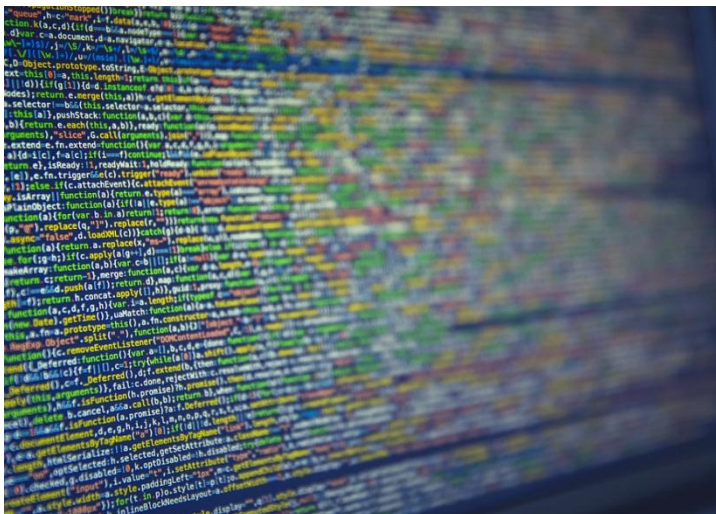
降低安全风险 提升系统安全性



业务分析

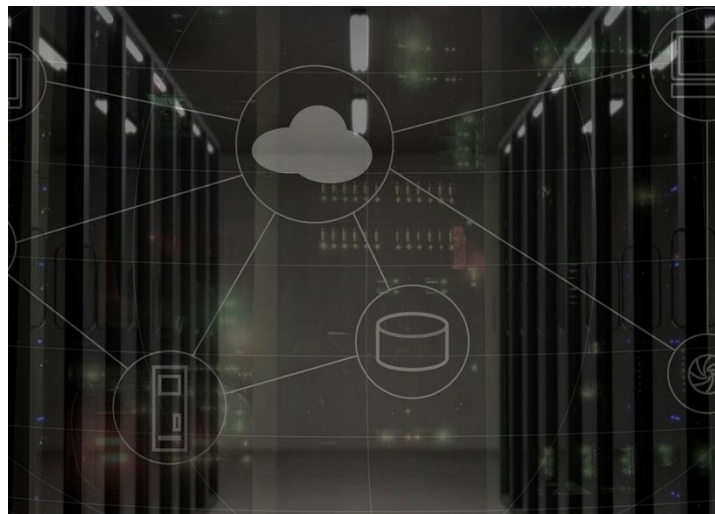
支持业务分析 加速业务增长

日志存储分析的 3V



Variety - Schema Free

数据类型多样, Text和JSON
Schema Evolution



Volume - 数据量大

存储规模大、存储周期长
对存储成本敏感

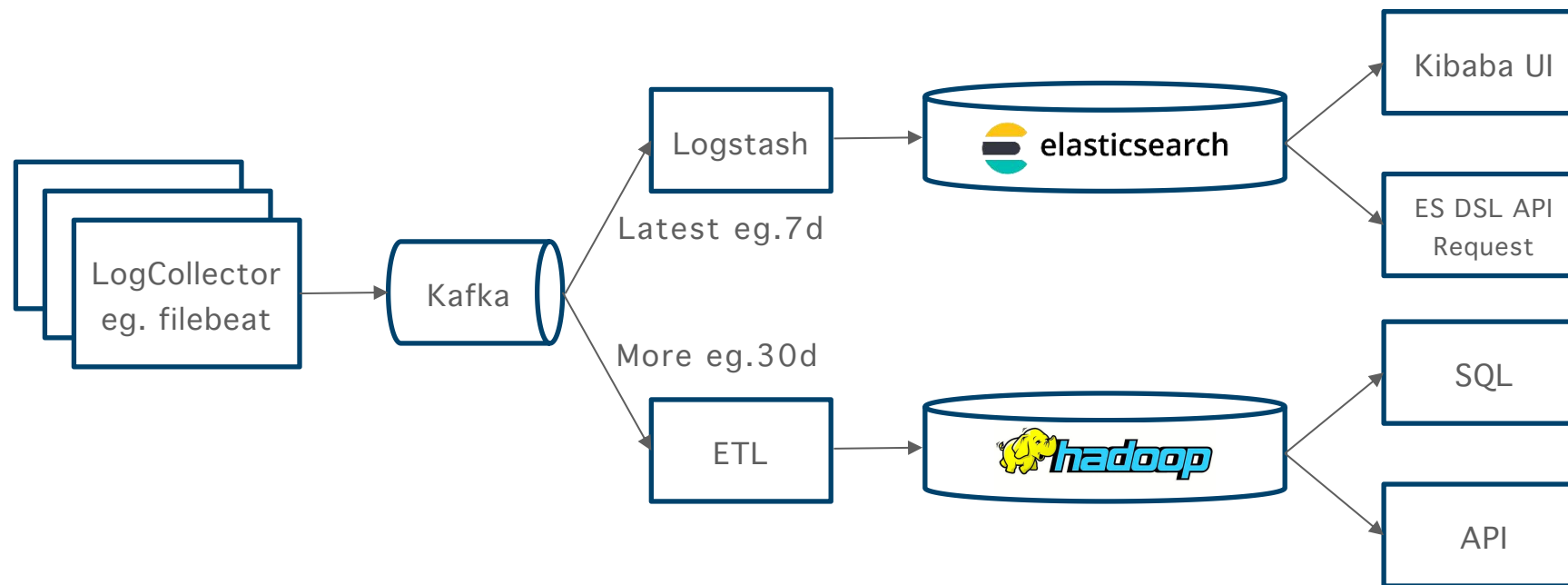


Velocity - 实时写入与检索

海量数据实时写入、低延迟可见
实时交互式分析

2 基于 ES 的日志系统痛点

基于 ES 的典型日志系统架构



每日增量：TB级

数据总量：PB级

查询并发：几个到百QPS

挑战 1 - 性价比低

写入性能低

- 数据写入需要构建索引、消耗大量 CPU 资源、写入效率低下
- 业务高峰期容易触发 reject，写入延迟升高

存储成本高

- 行存、倒排、列存等多份数据存储，高度冗余
- 整体数据压缩比约 1:1.5，远低于常见的 1:5

多套系统架构复杂

- 由于 ES 成本高，只存储最近几天的数据，更多或者全量数据在 Hadoop 或者数据湖
- 维护维护复杂：多套系统，多个数据流程
- 系统使用复杂：不同时间数据从不同系统查，查询方式差异大



挑战 2 - 维护和使用复杂

多套系统

- 由于 ES 成本高，只存储最近几天的数据
- 更多或者全量数据在 Hadoop 或者数据湖



实时数据，ES DSL 接口

维护复杂

- 维护不同技术栈的两套存储系统
- 维护两条数据处理的流程



历史数据，SQL或其他接口

使用复杂

- 最新数据从ES查，历史数据从Hadoop查
- 两个系统查询差异大，用户使用成本高

挑战3 - 分析能力弱

Query DSL 学习门槛高

- DSL (Domain Specific Language) 面向搜索场景设计
- 不符合使用习惯, 写查询经常需要查手册

DSL功能单一 不支持Join

- 只支持简单的单表分析
- 不支持多表 Join、子查询、视图等复杂分析

DSL生态封闭

- ES 生态自成体系, 与BI类系统或数据生态工具打通较为困难

```
{
  "size": 0,
  "query": {
    "bool": {
      "must": [
        {
          "range": {
            "timestamp": {
              "gte": "1998-05-01T00:00:00Z",
              "lt": "1998-05-02T00:00:00Z"
            }
          }
        },
        {
          "match": {
            "message": "error"
          }
        }
      ]
    }
  },
  "aggs": {
    "by_hour": {
      "date_histogram": {
        "field": "timestamp",
        "calendar_interval": "hour"
      }
    }
  }
}
```

挑战4 - Schema Evolution 支持有限

字段类型 固定不变

- 字段类型冲突不允许写入
- 字段类型不能更改 => reindex 重写数据

索引 固定不变

- 已有字段的索引不能增加或删除 => 全建索引
- 已有字段的索引不能调整分词等参数

```
{
  "mappings": {
    "properties": {
      "@timestamp": {
        "format": "strict_date_optional_time||epoch_second",
        "type": "date"
      },
      "message": {
        "type": "keyword",
        "index": false,
        "doc_values": false
      },
      "clientip": {
        "type": "ip"
      },
      "request": {
        "type": "match_only_text"
      },
      "status": {
        "type": "integer"
      },
      "size": {
        "type": "integer"
      }
    }
  }
}
```

3 基于SelectDB的新一代日志分析平台

基于 SelectDB 开放、高性能、低成本 日志系统架构

 Filebeat

 logstash

 fluentd


 fluentbit

 kafka

HTTP



SQL


Log Discover
(like Kibana Discover)


Grafana


Superset

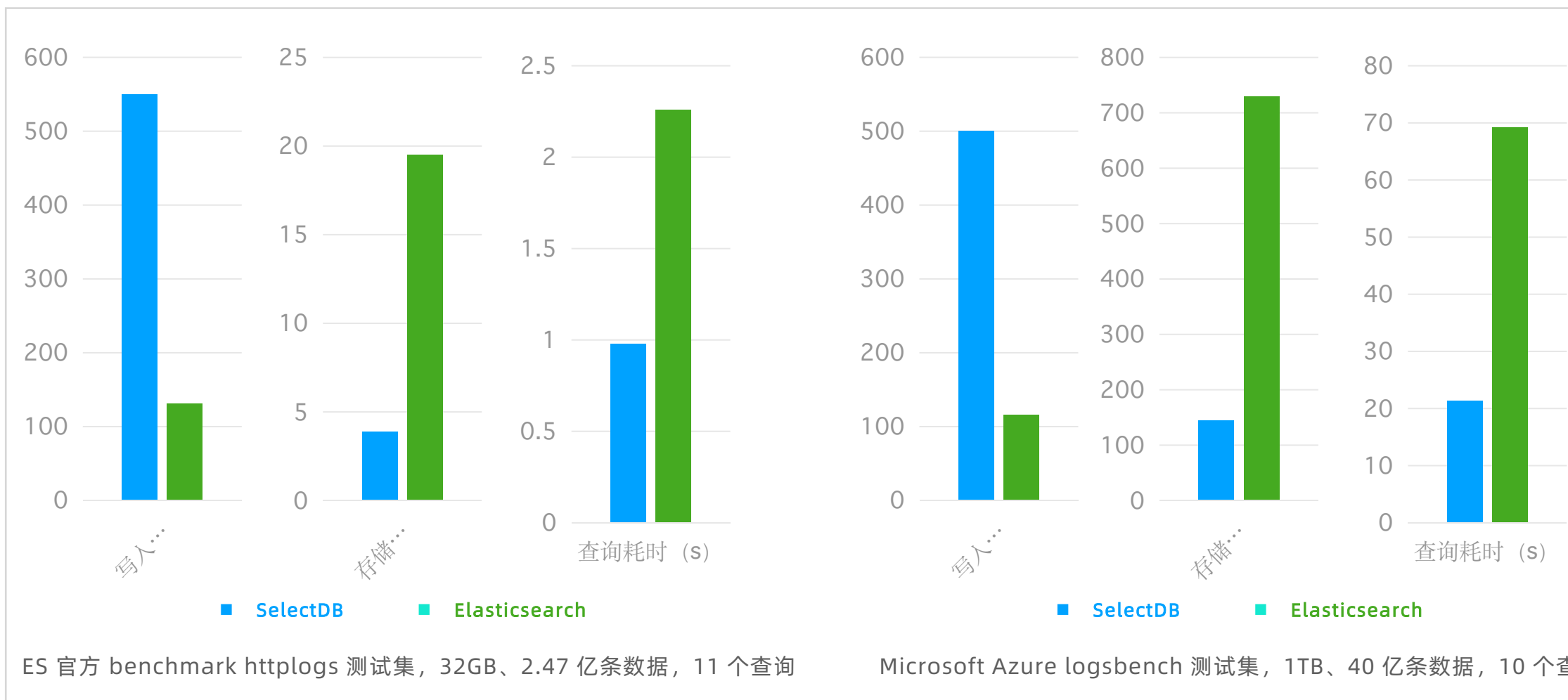
开放的日志接入方式

高性能、低成本
统一日志存储

兼容MySQL的开放生态

优势1 - 超高性价比

相对于ES **3~5倍** 写入吞吐提升, **80%** 存储空间降低, **2~3倍** 查询性能提升



优势 1 - 超高性价比

| | Elasticsearch | SelectDB | SelectDB 冷热分层 |
|---------------|-----------------------|-----------------------|----------------------|
| 日增数据 (TB) | 100 | 100 | 100 |
| 热数据天数 | 3 | 3 | 3 |
| 冷数据天数 | 27 | 27 | 27 |
| 数据压缩比 | 1.5 | 7.5 | 7.5 |
| 热数据存储空间 (TB) | 200 | 40 | 40 |
| 冷数据存储空间 (TB) | 1800 | 360 | 360 |
| 服务器配置 | 16C 64G 26.3TB | 16C 64G 26.3TB | 16C 64G 6.1TB |
| 服务器数量 | 95 | 19 | 19 |
| 计算资源成本 (万元/月) | 23.1 | 4.6 | 4.6 |
| 云盘存储成本 (万元/月) | 71.7 | 14.3 | 1.4 |
| 对象存储成本 (万元/月) | 0 | 0 | 3.8 |
| 云资源总成本 (万元/月) | 94.8 | 18.9 | 9.8 |
| 综合性价比 | 1 | 5倍 | 9.7倍 |

优势 2 - 实时和历史数据统一

一套系统

- 一套系统部署和数据处理流程
- 一套查询接口和查询界面

维护简单

- 维护一套存储系统和数据处理流程
- 支持SSD、HDD、S3 三种存储介质
- 定义好数据生命周期，自动热转冷、过期删除

使用简单

- 一个SQL查询接口，冷热数据查询方式一致
- 可以跨冷热分界混合查询



优势 3 - 基于 SQL 的分析引擎

简单易用

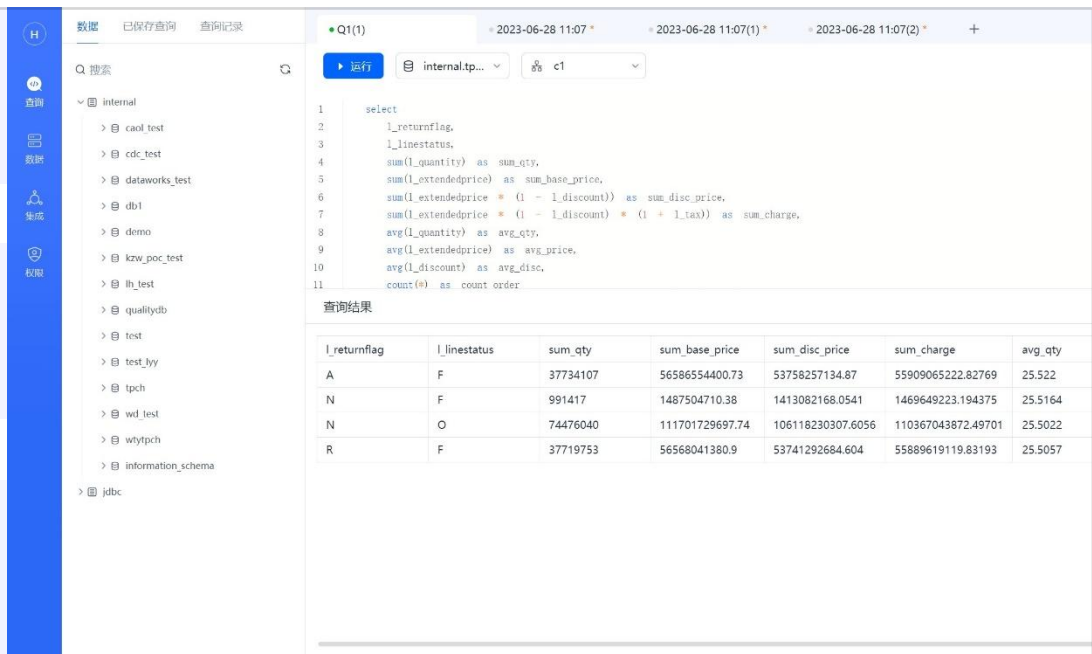
- 支持标准SQL，无额外学习成本
- SQL语法与MySQL高度兼容

丰富的数据生态

- MySQL协议兼容，可直接使用MySQL CLI
- 无缝对接各类BI工具以及大数据生态组件

强大的分析能力

- 支持检索、聚合、多表JOIN、子查询、窗口函数、UDF、视图/物化视图等功能



The screenshot displays a SQL query editor interface. On the left, a sidebar shows a tree view of databases and schemas, including 'internal', 'caol_test', 'cdc_test', 'dataworks_test', 'db1', 'demo', 'kzw_poc_test', 'lh_test', 'qualitydb', 'test', 'test_ly', 'tpch', 'wd_test', 'wytpch', 'information_schema', and 'jdbc'. The main area shows a SQL query being executed. The query is as follows:

```
1 select
2   l_returnflag,
3   l_linestatus,
4   sum(l_quantity) as sum_qty,
5   sum(l_extendedprice) as sum_base_price,
6   sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
7   sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
8   avg(l_quantity) as avg_qty,
9   avg(l_extendedprice) as avg_price,
10  avg(l_discount) as avg_disc,
11  count(*) as count_order
```

Below the query, the results are displayed in a table format:

| l_returnflag | l_linestatus | sum_qty | sum_base_price | sum_disc_price | sum_charge | avg_qty |
|--------------|--------------|----------|-----------------|-------------------|--------------------|---------|
| A | F | 37734107 | 56586554400.73 | 53758257134.87 | 55909065222.82769 | 25.522 |
| N | F | 991417 | 1487504710.38 | 1413082168.0541 | 1469649223.194375 | 25.5164 |
| N | O | 74476040 | 111701729697.74 | 106118230307.6056 | 110367043872.49701 | 25.5022 |
| R | F | 37719753 | 56568041380.9 | 53741292684.604 | 55889619119.83193 | 25.5057 |

优势 4 - 原生的半结构化数据支持

丰富的数据类型

- Text, JSON, Array, Map
- **Variant**, 允许一个字段多种类型

```
{  
  "id": 134567,  
  "name": "name1"  
} → {  
  "id": "vip_48679",  
  "name": "name2"  
}
```

Schema Evolution

- 在线**毫秒级增减字段**
- **在线按需增减索引**, 指定分区**构建索引**
- 在线按需更改类型

```
ALTER TABLE t ADD COLUMN c;
```

```
ALTER TABLE t ADD INDEX idx_a(a) USING INVERTED;
```

```
BUILD INDEX idx_a ON t PARTITION(p20230808);
```

优势 4 - 原生的半结构化数据类型 variant

JSON数据 自适应

- 自动识别JSON数据中的**字段名和类型**
- 自动将频繁出现的字段**采用列式存储**
- 自动将不频繁的字段合并存储，避免类似ES的mapping爆炸

支持一个字段 多个类型

- **允许一个字段有多种类型**
- 更好满足业务发展中字段类型变化的需求

支持倒排索引

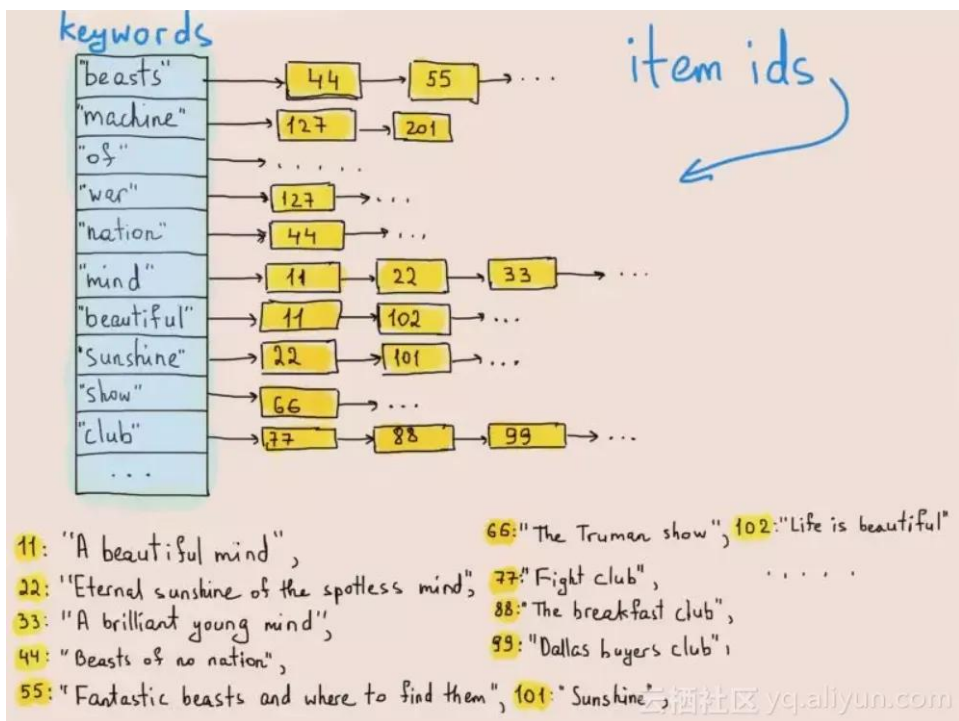
- 为variant字段创建**倒排索引**，子字段自动创建
- 可指定文本字段是否分词、分词类型等参数
- 后续版本将支持各个子字段索引灵活定义

```
-- 创建了三个VARIANT类型的列, actor, repo和payload
-- 创建表的同时创建了payload列的倒排索引idx_payload
-- USING INVERTED 指定索引类型是倒排索引, 用于加速子列的条件过滤
-- PROPERTIES("parser" = "english") 指定对子列采用english分词
CREATE TABLE IF NOT EXISTS github_events (
  id BIGINT NOT NULL,
  type VARCHAR(30) NULL,
  actor VARIANT NULL,
  repo VARIANT NULL,
  payload VARIANT NULL,
  public BOOLEAN NULL,
  created_at DATETIME NULL,
  INDEX idx_payload (`payload`) USING INVERTED PROPERTIES("parser" = "english")
)
```

```
mysql> desc github_events;
```

| Field | Type | Null | Key | Default | Extra |
|------------------------------------|------------|------|-------|---------|-------|
| id | BIGINT | No | true | NULL | |
| type | VARCHAR(*) | Yes | false | NULL | NONE |
| actor | VARIANT | Yes | false | NULL | NONE |
| actor.avatar_url | TEXT | Yes | false | NULL | NONE |
| actor.display_login | TEXT | Yes | false | NULL | NONE |
| actor.id | INT | Yes | false | NULL | NONE |
| actor.login | TEXT | Yes | false | NULL | NONE |
| actor.url | TEXT | Yes | false | NULL | NONE |
| created_at | DATETIME | Yes | false | NULL | NONE |
| payload | VARIANT | Yes | false | NULL | NONE |
| payload.action | TEXT | Yes | false | NULL | NONE |
| payload.before | TEXT | Yes | false | NULL | NONE |
| payload.comment.author_association | TEXT | Yes | false | NULL | NONE |
| payload.comment.body | TEXT | Yes | false | NULL | NONE |
| | | | | | |

关键技术 - 倒排索引



```
CREATE TABLE httplog
(
  `ts` DATETIME,
  `clientip` VARCHAR(20),
  `request` TEXT,
  INDEX idx_clientip (`clientip`) USING INVERTED,
  INDEX idx_request (`request`) USING INVERTED PROPERTIES("parser" = "unicode")
)
DUPLICATE KEY(`ts`)
...

-- 查看最新的10条数据
SELECT * FROM httplog ORDER BY ts DESC LIMIT 10;

-- 查询clientip为'8.8.8.8'的最新10条数据
SELECT * FROM httplog WHERE clientip = '8.8.8.8' ORDER BY ts DESC LIMIT 10;

-- 检索request字段中有error或者404的最新10条数据
SELECT * FROM httplog WHERE request MATCH_ANY 'error 404' ORDER BY ts DESC LIMIT 10;

-- 检索request字段中有image和faq的最新10条数据
SELECT * FROM httplog WHERE request MATCH_ALL 'image faq' ORDER BY ts DESC LIMIT 10;
```

关键技术 - 日志检索查询优化

```
SELECT * FROM log
WHERE ts >= t1 AND ts <= t2 AND message MATCH 'error'
ORDER BY ts DESC LIMIT 100
```

挑战 从海量日志中全文检索关键词



基于分区、主键的时间范围快速跳过
基于倒排索引的全文检索精准定位

挑战 按时间排序取满足条件的最新N条日志



按时间排序的时序存储模型
基于动态剪枝的TopN查询算法

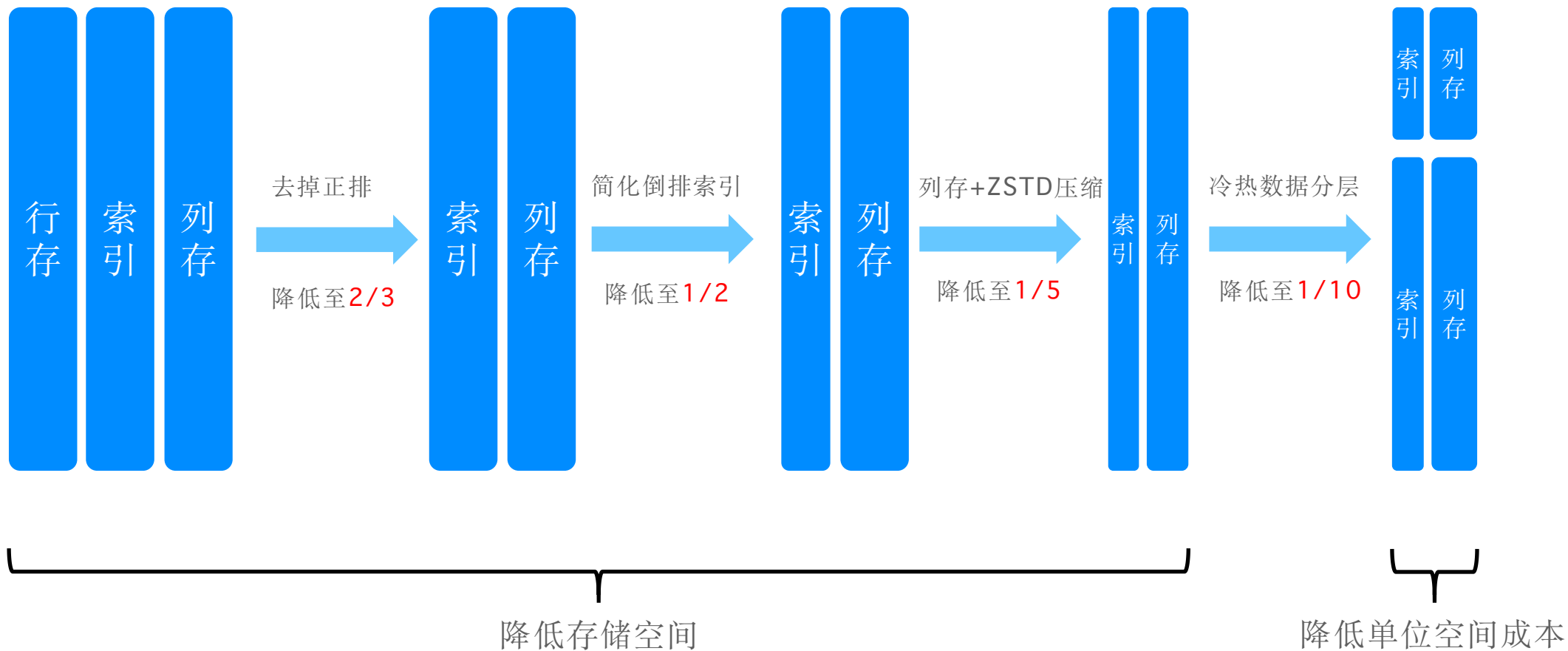


效果：百亿日志检索秒级响应

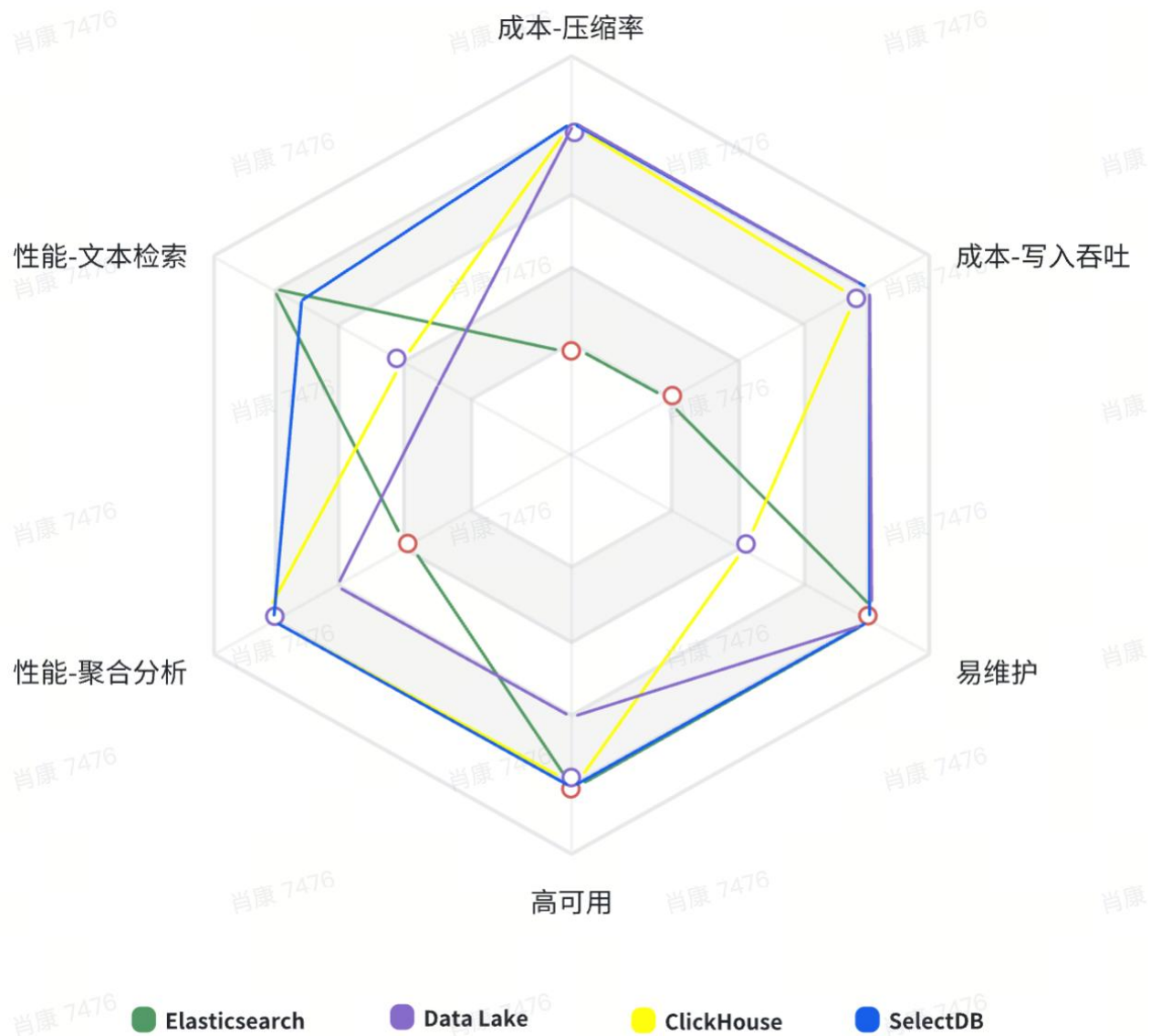
关键技术 - 导入性能优化



关键技术 - 存储成本优化



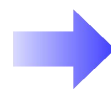
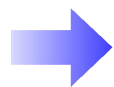
多种方案对比



4 实践案例

实践案例1 - 可观测性 & 汽车制造

“SelectDB提供了灵活半结构化类型variant，成本相比云上ES节省**70%**，全文检索提升**2-3倍**”



可观测性数据采集

Log Trace 统一存储平台

可观测性可视化分析

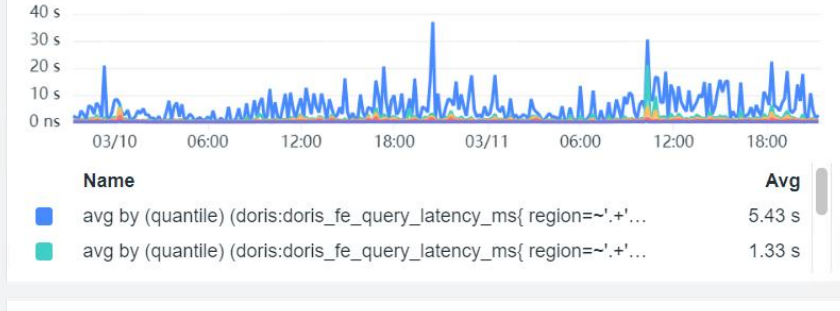
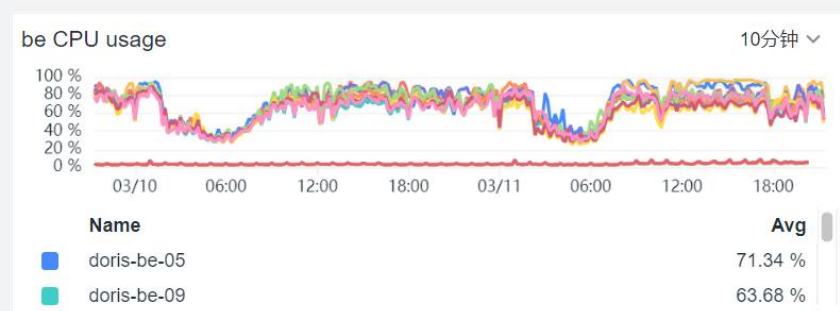
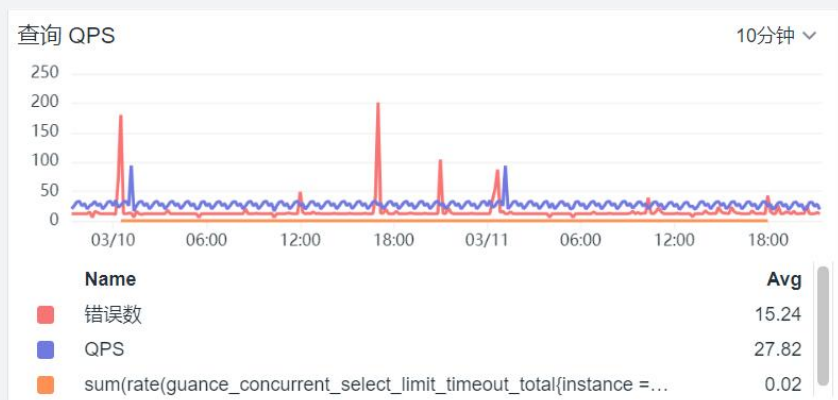
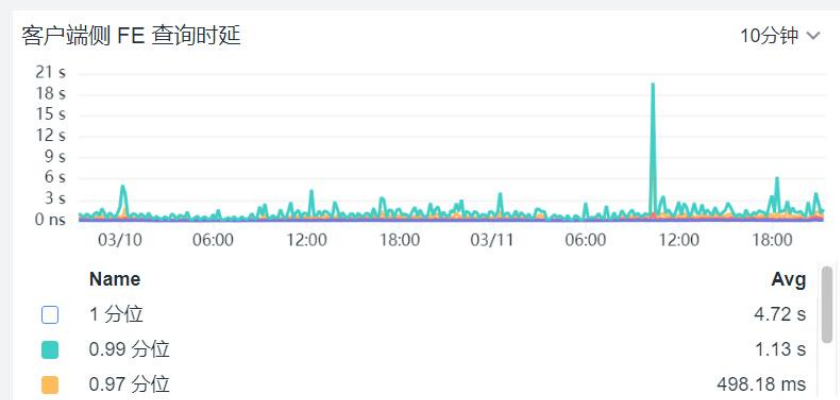
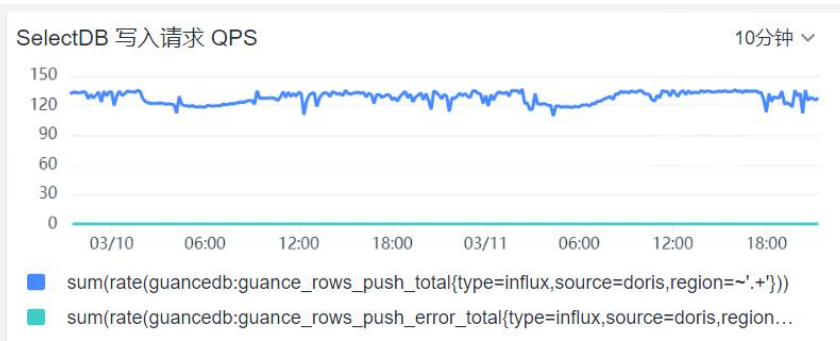
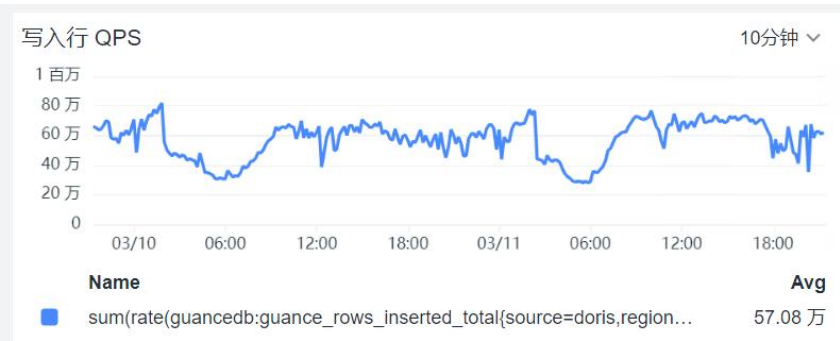
集群规模：10 台虚拟机

数据增量：每天新增 400 亿条数据、80TB，SelectDB 压缩后 16TB（包括倒排索引，压缩率1:5）

数据总量：1 副本保存 30 天，总共 480TB、1.2 万亿条

写入性能：线上平均 50w/s、1GB/s，峰值 **100w/s, 2GB/s**，秒级实时写入

实践案例 1 - 可观测性 & 汽车制造

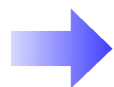


实践案例 2 - 网络安全

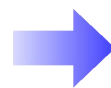
“SelectDB只用原来**1/5**的服务器，承载了**1GB/s**的写入流量，安全分析查询响应速度更快”



消息系统数据导入



统一日志存储分析平台



安全数据分析

集群规模：10 台物理机

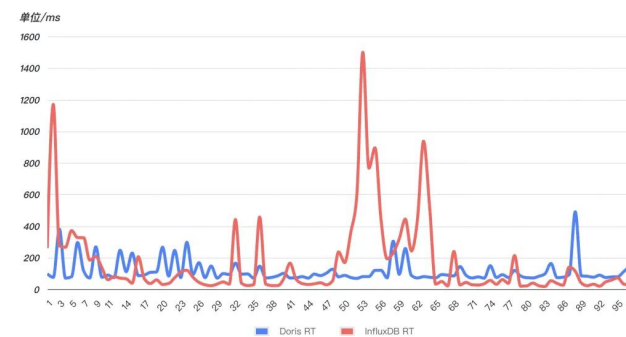
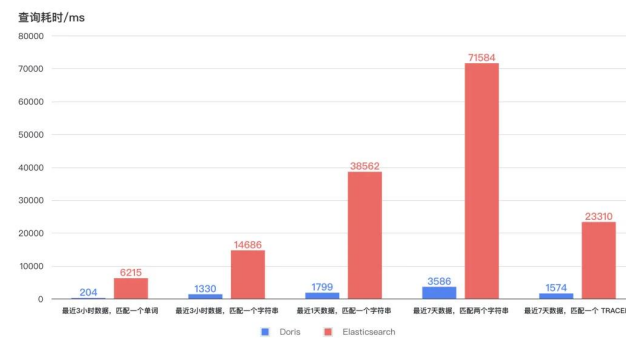
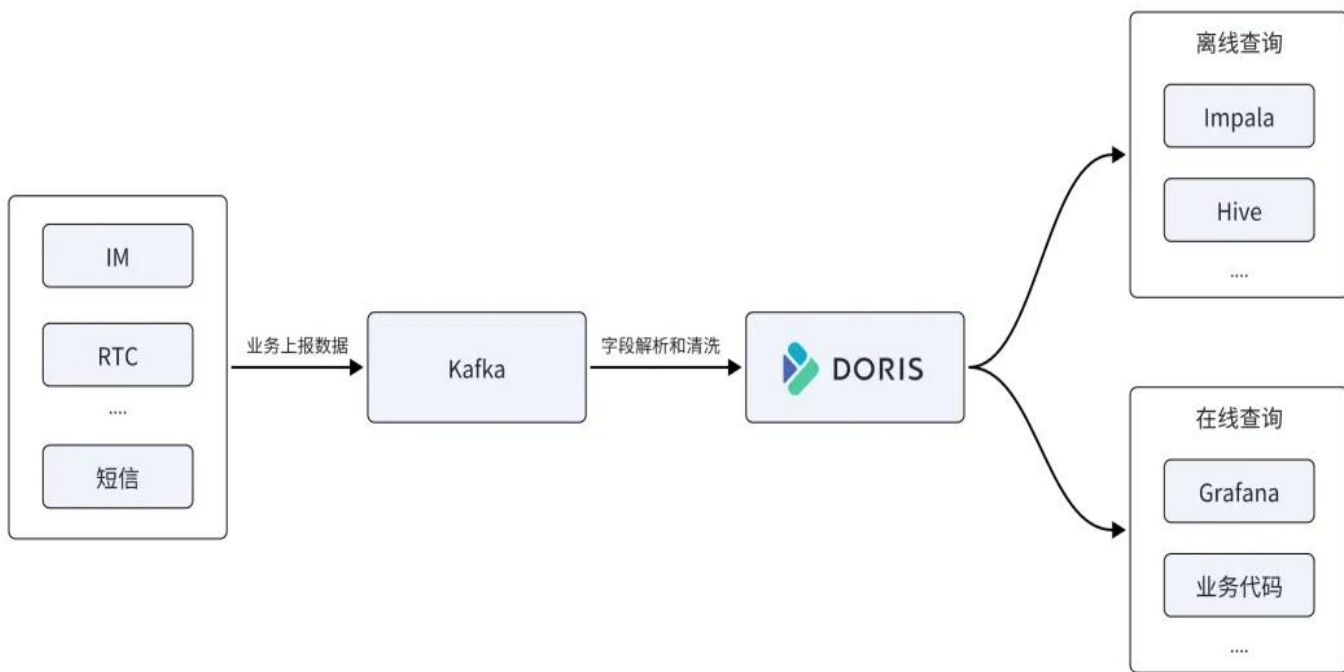
数据增量：每天新增 150 亿条日志、8.3TB，SelectDB 压缩后 1.4TB（包括倒排索引，压缩率 5.9）

数据总量：3 副本保存 60 天，总共 252TB、9 千亿条

写入性能：线上平均 20w/s、100MB/s，峰值 100w/s、500MB/s，压测 **3 台机器 200w/s、1GB/s**

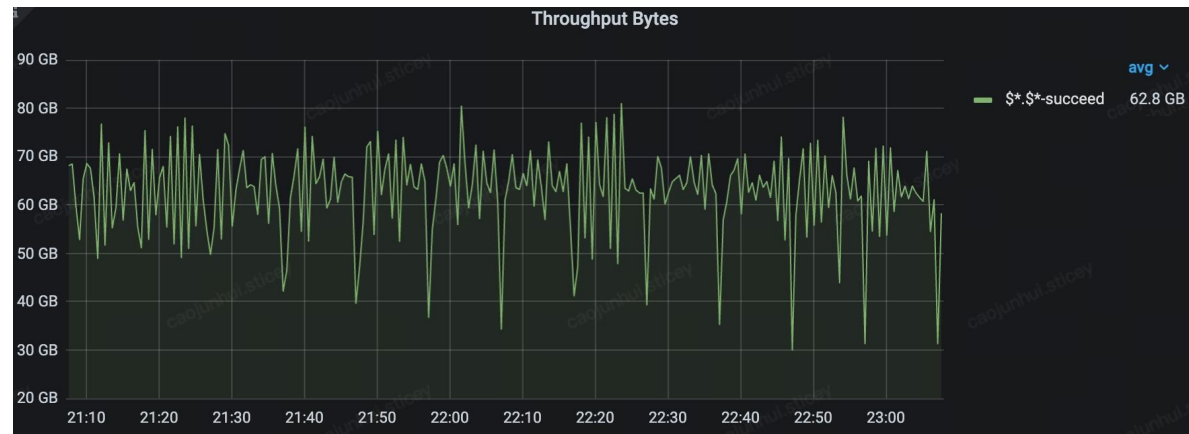
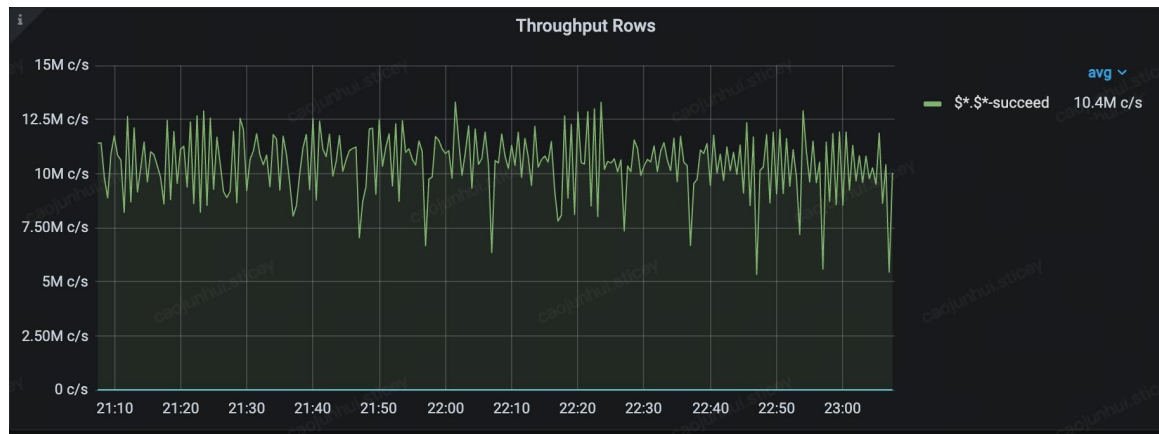
实践案例 3 - 互联网

“**网易**日志数据存储空间降低到ES的**1/3**，查询效率获得**10倍**提升，查询性能更加平稳
时序场景替代 InfluxDB, 服务器节省**50%**，存储空间降低**67%**”



实践案例 4 - 移动互联网

“Log Trace 场景能够完全支持，标志着 Doris 几乎能扛住**抖音集团**绝大部分场景的导入性能需求”



集群规模：3000+ core

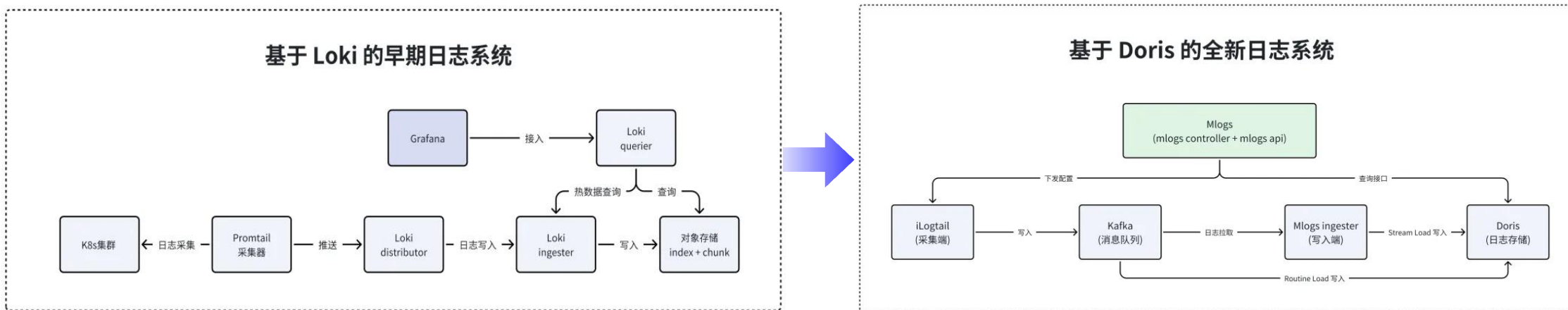
数据增量：每天新增 8000 亿条日志、500TB

数据总量：总共 7PB、24 万亿条

写入性能：线上平均 1000w/s、60GB/s，峰值 1500w/s，90GB/s

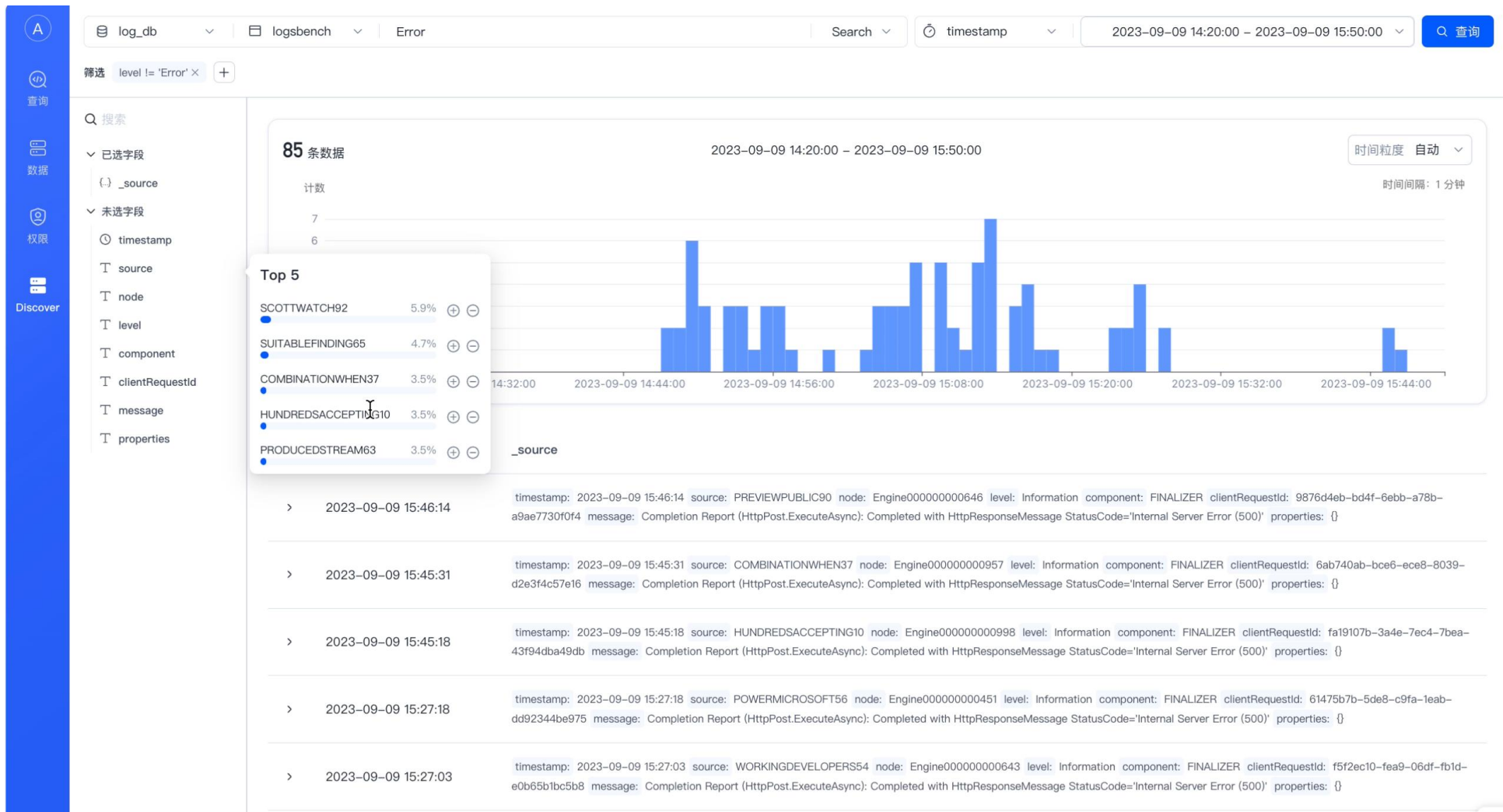
实践案例 5 - 大模型

“基于 Doris 的新系统已接入 MiniMax 内部所有业务线日志数据，满足实时写入和查询的需求”



数据规模：**PB级** 写入性能：**10GB/s** 查询：**秒级响应** 冷热分层：7 天热 30 天冷

可视化日志检索



联系我们



欢迎关注SelectDB微信公众号

获取最新活动资讯、技术解析、社区动态

公司邮箱: support@selectdb.com

SelectDB 官网: www.selectdb.com

Apache Doris 官网: <https://doris.apache.org/>

Apache Doris GitHub: <https://github.com/apache/doris>

飞轮科技 - 让数据分析快速简单

THANKS



公众号



免费体验