Elasticsearch在数说全量库的应用实践

目 录

CONTENTS

数据中心架构概览

数说应用 | 数说聚合 | 数说工厂 | ... | 数说雷达

↑ 3. 定制化数据获取

数据仓库 (全量库) | API平台 | | | |
| 微博 | 微信 | 新闻论坛 | 电商 | 视频 |

↑ 2. 实时/定时数据抓取

数据采集 | 增量监控采集 | 在线异步采集 |
| 采集调度管理平台 | |

↑ 1. 原始数据

社交 | 电商 | 新闻 | 视频 | ...

1 多类型互联网数据源

交友社交 · 新闻门户 · 健康保健 · 娱乐时尚 · 金融财经 · DATASTORY 数说故事 · 在线视频 · 母婴网站 · 综合门户 · 社区博客 · 电商平台
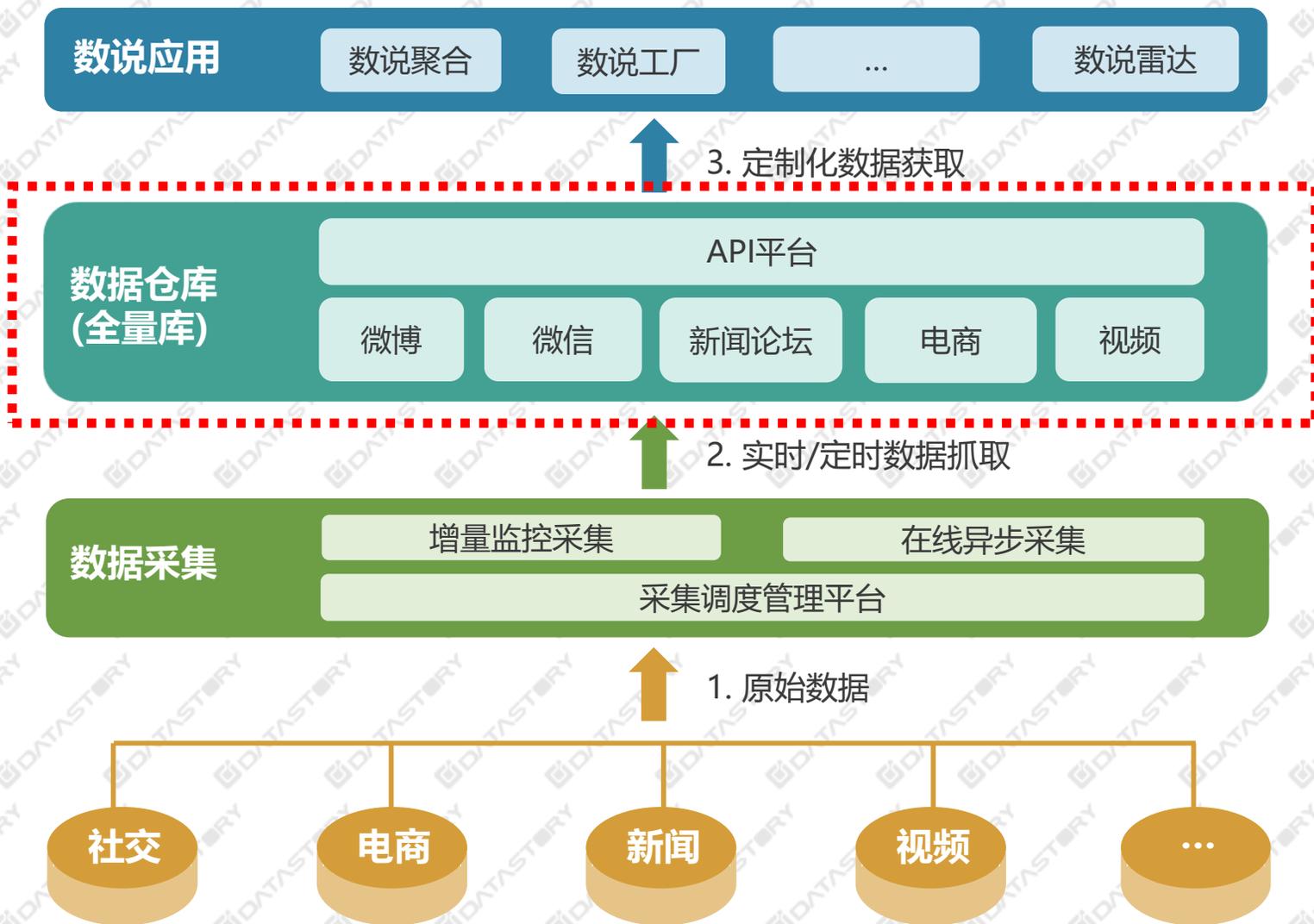
2 数据实时性

➢ 最快采集频率（分钟级别）
➢ 最慢采集评率（小时级别）

3 日均消费

➢ 总量（千万量级）
➢ 微博（千万量级）
➢ 新闻（千万量级）
➢ 消费峰值（百万量级/5min）

4 数据量级

➢ 百亿量级

全量库架构

应用层	**Application**

数据服务层

ES-Spark | SDK | Gaia
API+Router
Elasticsearch | Hive

→ 请求监控

数据存储层

HBase

微博 | 微信 | ... | 新闻论坛

→ 入库监控

流式计算层

SparkStreaming

情感分析 | 数据清洗 | ... | 水军算法

→ 消费监控

消息队列层

Kafka

微博topic | 微信topic | ... | 新闻论坛topic

数据输入 | **基于消息订阅的采集数据流**

日志收集

目 录
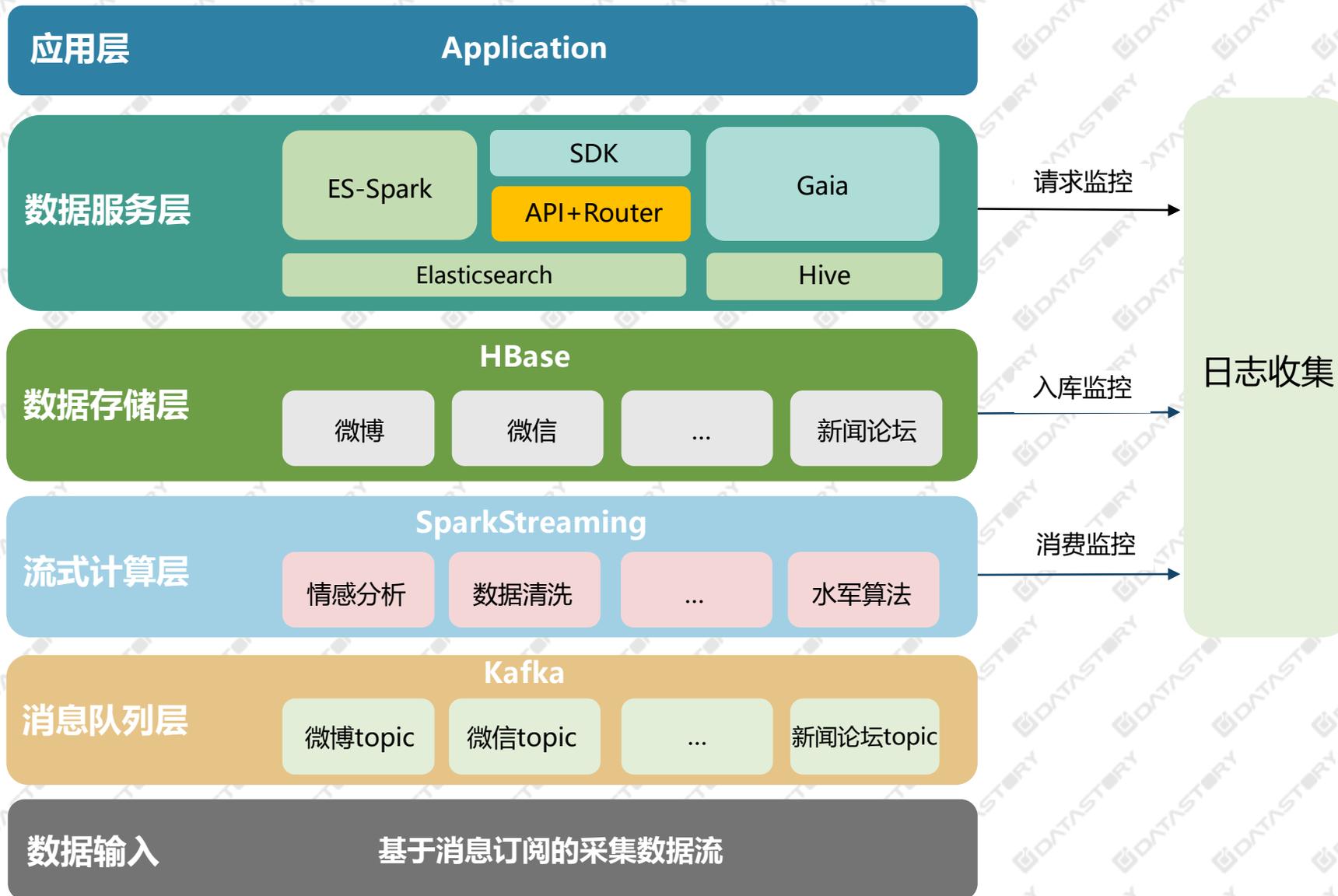
CONTENTS

索引拆分

原因剖析

- 索引大
- 分片少
- 分片大
- 节点负载高
- 大量读写

诱因

超大索引 → 14t 48shard 300g/shard

consume 消费 → 大量写请求

sdk gaia es-spark → 大量读请求

原因

shard大 请求多
- shard命中概率高
- 响应慢

节点 负载高
- 请求容易集中到个别节点
- 容易集群crash

线程池 打满
- search线程池打满
- index线程池受影响

现象

消费积压

查询慢 → 父子查询 聚合查询

索引拆分

拆分前

index
- parent_all
  - child_1月
  - ...
  - child_6月

拆分后

index_201801
- parent_1月
  - child_1月
  - child_6月

index_201806
- parent_6月
  - child_6月

拆分后

p_index_201801
- parent_1月

p_index_201806
- parent_6月

c_index_201801
- child_1月

c_index_201806
- child_6月

拆分方案1

- 保留父子文档
- 按父文档时间拆分

缺点

- 路由优化难

拆分方案2

- 拆分父子文档
- 按父、子文档时间分别拆分
- shard大小控制
- 制定索引划分标准（季度、半年、年）

入库优化

- 增加回压机制
- 守护进程
- 增加retry机制
- 提升写性能

- 合理划分Kafka的topic
- 合理配置Spark参数
- 算法接口改为RPC调用

入库优化

V1

- 消费速率跟不上
- 消费进程假死
- 丢数据

SparkStreaming

外部数据 → Kafka → ETL Analyze → HBase → ES

V2

- 增加回压机制
- 守护消费进程
- 增加重试机制

backpressure

SparkStreaming

外部数据 → Kafka → ETL Analyze → HBase → ES

hook

Deamon → hook

hook → HDFS

hook → ES
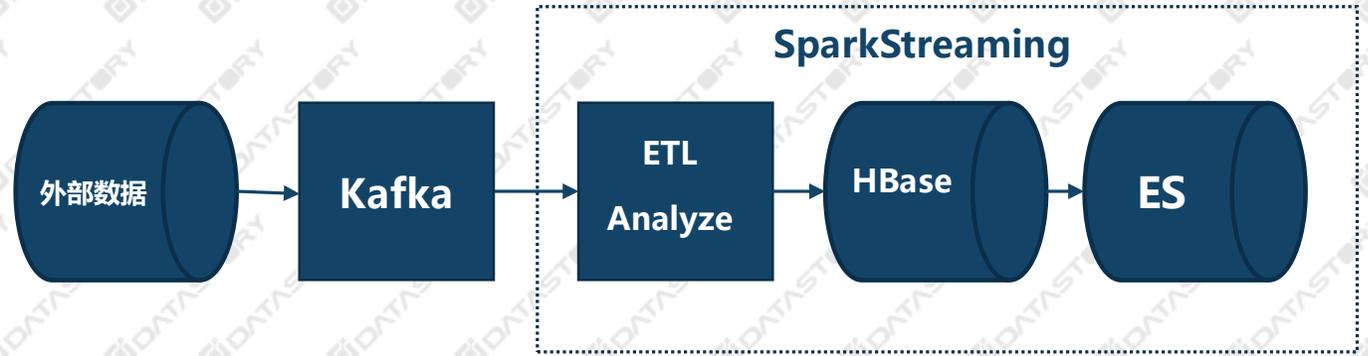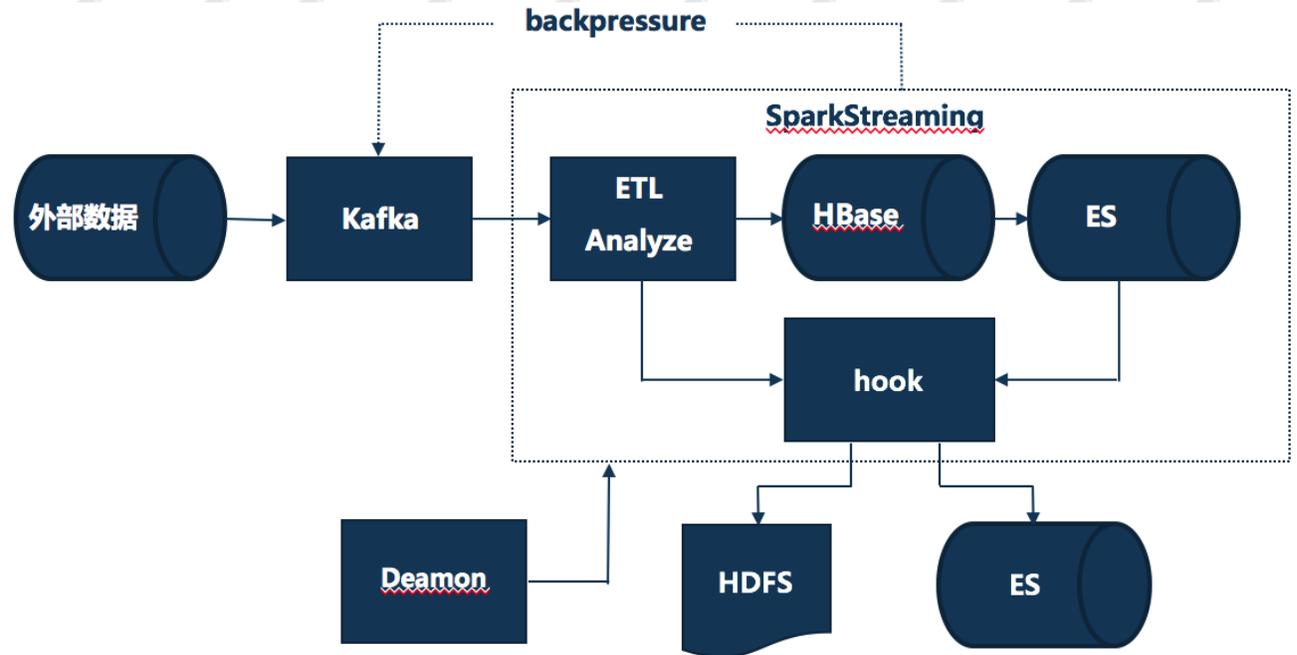
提升写性能

入库核心逻辑

- Connect
- Bulk

需求点

- connect和bulk互斥
- 尽量避免connect

优点

- 提升性能

缺点

- Socket异常可能多批次bulk失败
- 补救：hook（retry）

伪代码

```
read().lock();
if (socket异常) {
    read().unlock();
    write().lock();
    if (socket异常) {// 双重校验
        connect();// connect操作
    }
    read().lock();// 降级锁
    write().unlock();
}
bulk(batchRequest);// bulk操作
read().unlock();
```

搜索优化

- 隔离+高可用+高并发
- 路由中间件
- 优化字段使用
- 统一使用规范
- 取数流程优化
- 复杂查询优化
- es版本升级

- 上线微服务
- 读写分离
- 统一分词器

全量库服务中间件（V1-1版）

小批量、小规模
部门隔离
物理部署
扩展困难
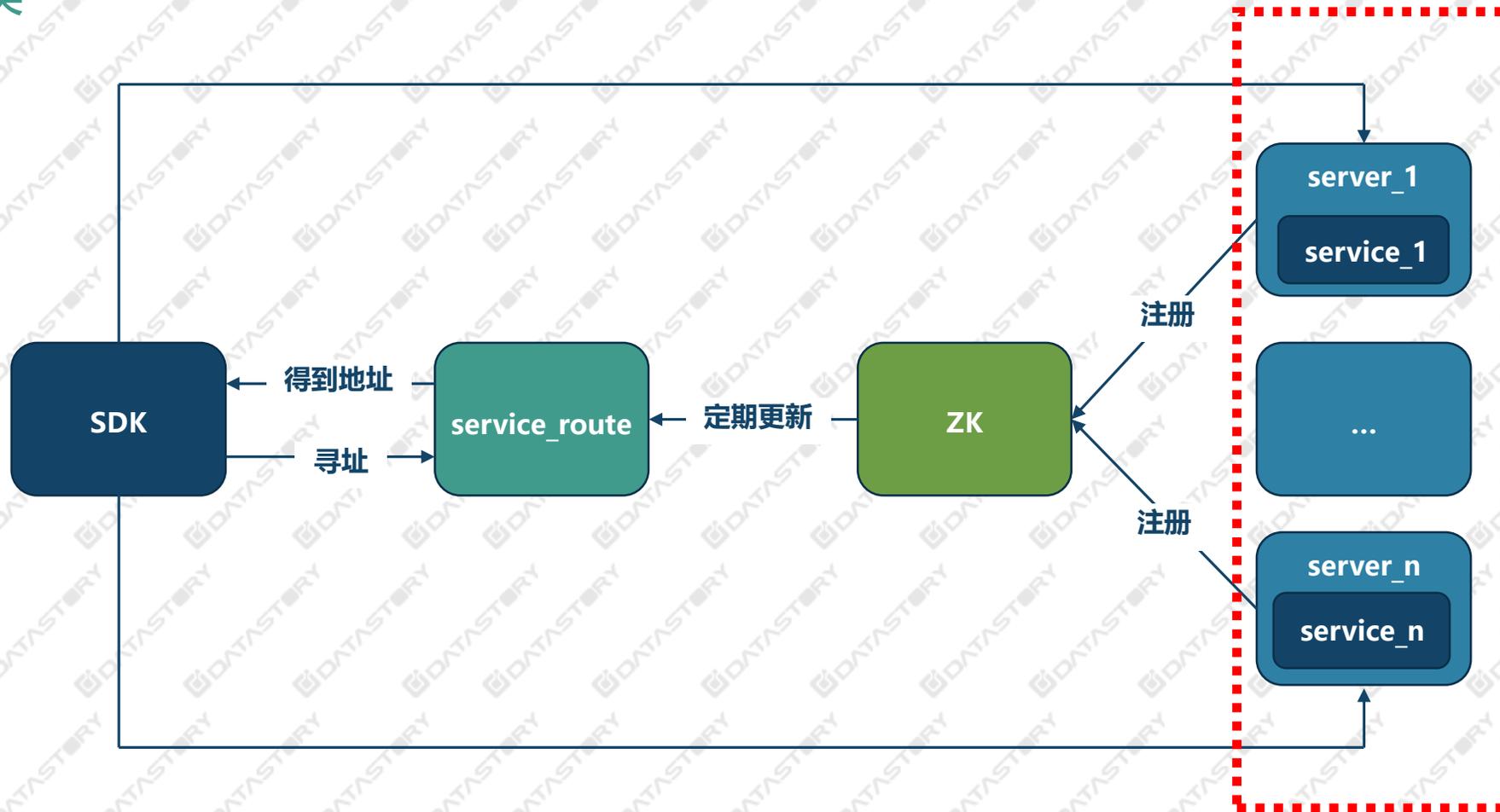
# 全量库服务中间件（V1-2版）

小批量、小规模
项目隔离
容器部署
扩展方便

SDK ←—得到地址—— service_route ←—定期更新—— ZK

SDK ——寻址—→ service_route

ZK —注册→ pod_1 → service_1

ZK —注册→ pod_n → service_n

...
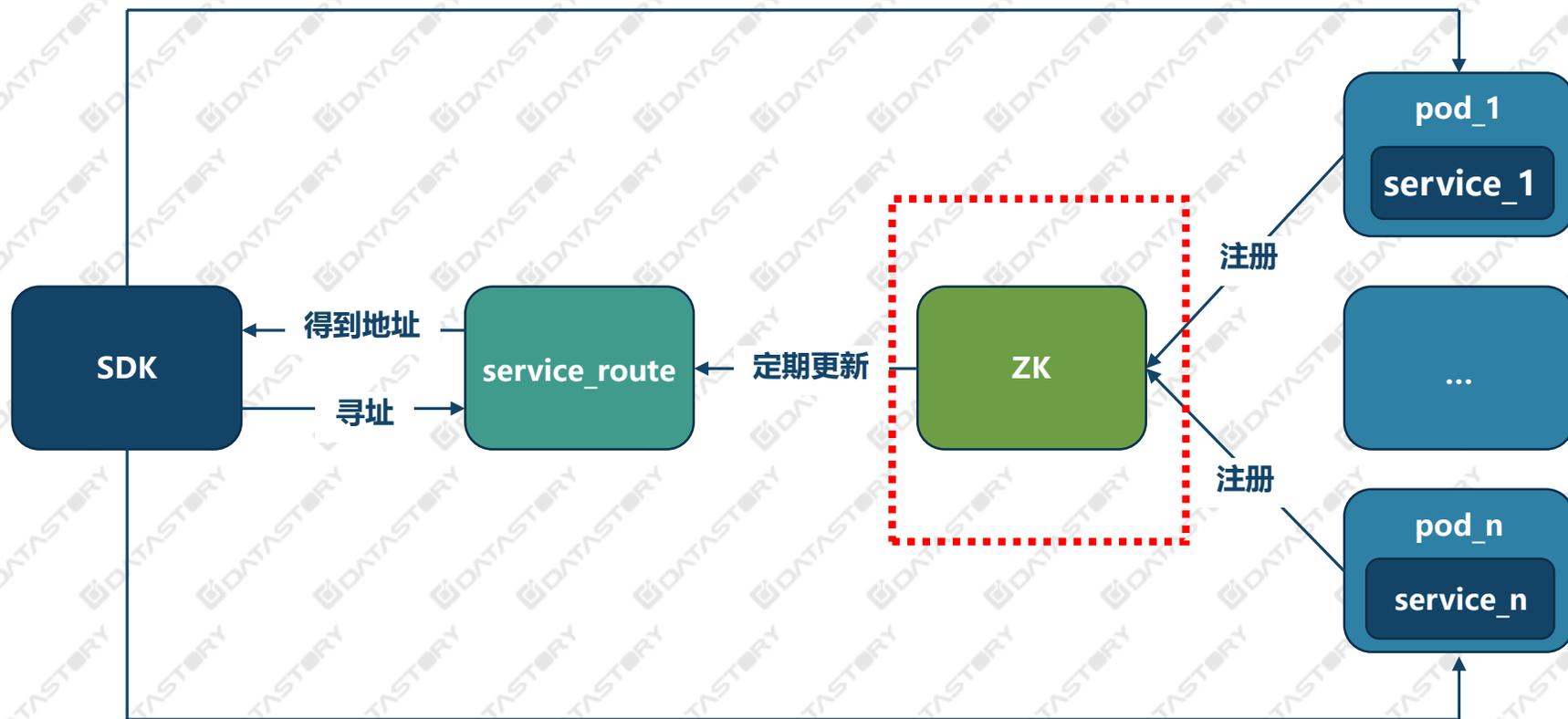
全量库服务中间件（V2）

高并发、高可用
业务＆数据源隔离
容器部署
扩展方便

- 横向扩展（业务）
- 纵向扩展（数据源）

客户端	**SDK**

| 业务隔离层 | **业务1** | **...** | **业务2** |

数据源/
查询类型
隔离层

wb_user	**wb_weibo**	**...**	**nf_post**	**nf_comment**
count agg search	count agg search		count agg search	count agg search

数据层	**全量库**

Eureka Server
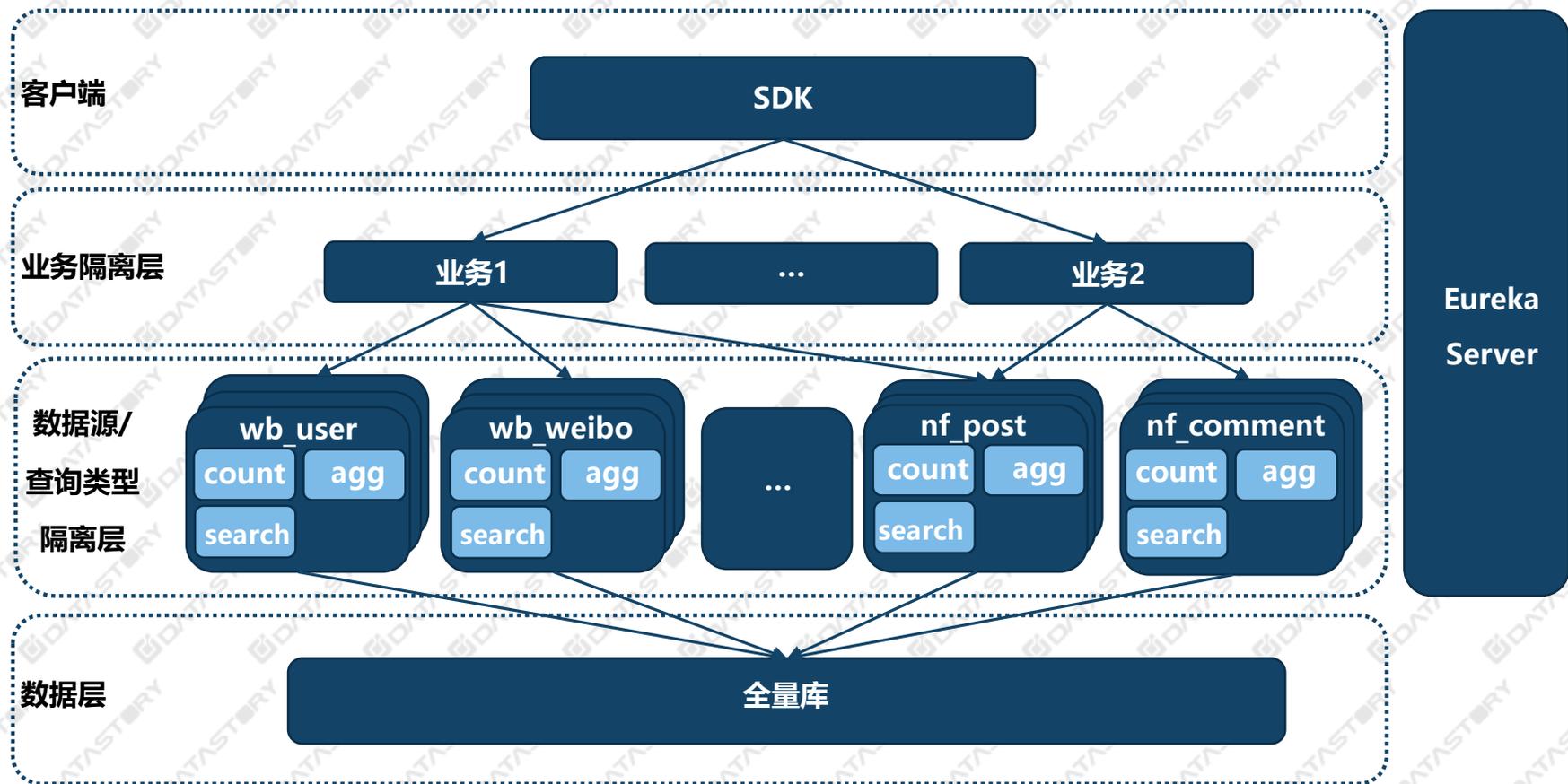
搜索优化

路由中间件

- 业务封装
- 适合的路由策略
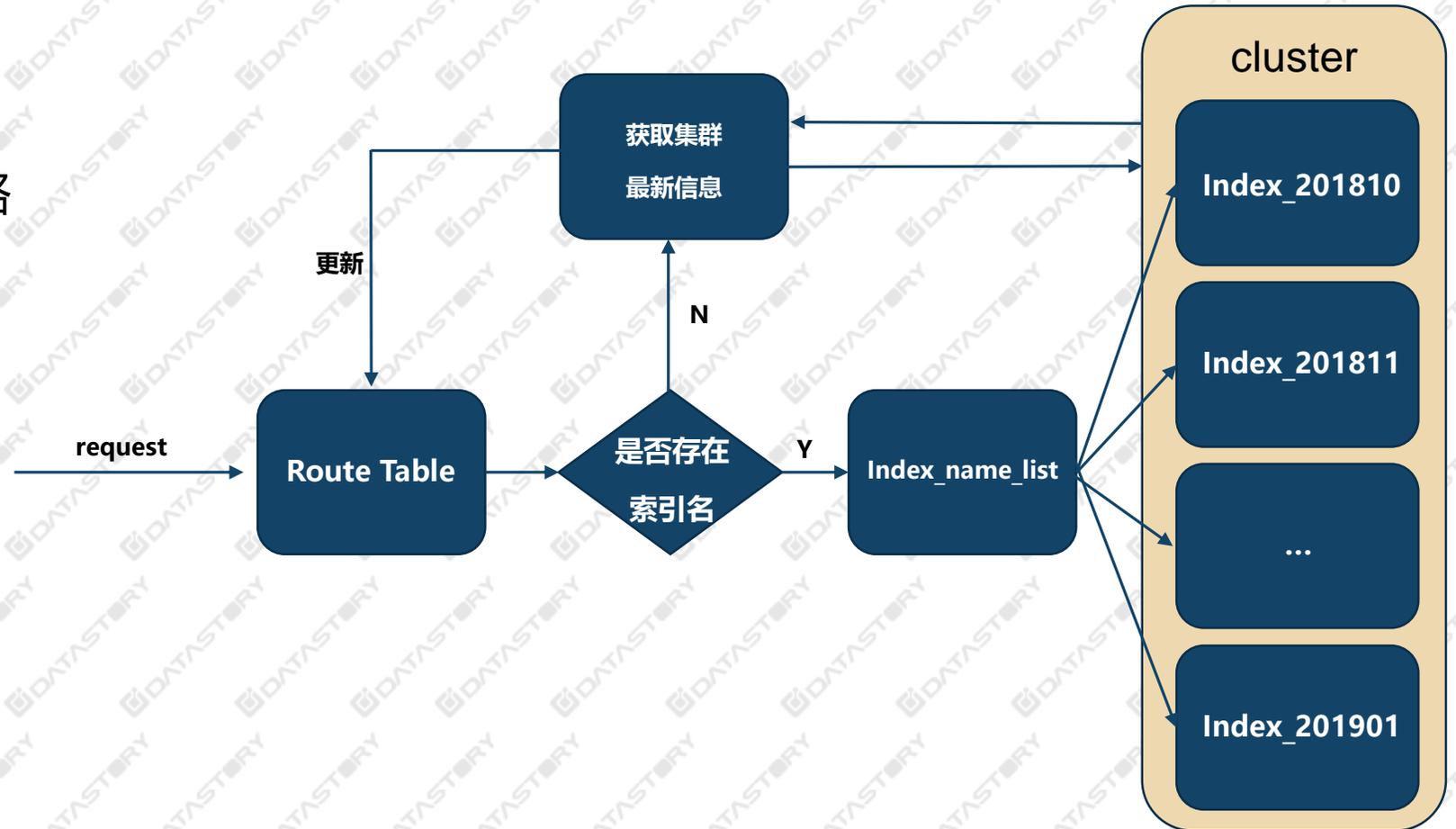
搜索优化

优化字段使用

- 真实环境
  - 字段类型
  - hit数据量
 - 索引大小
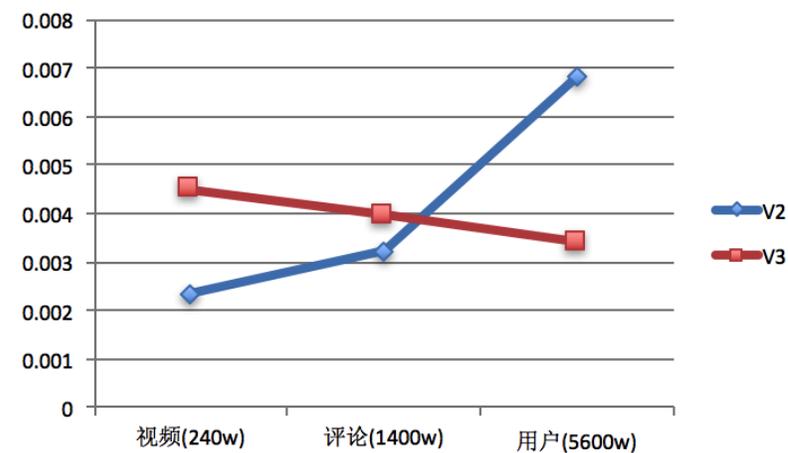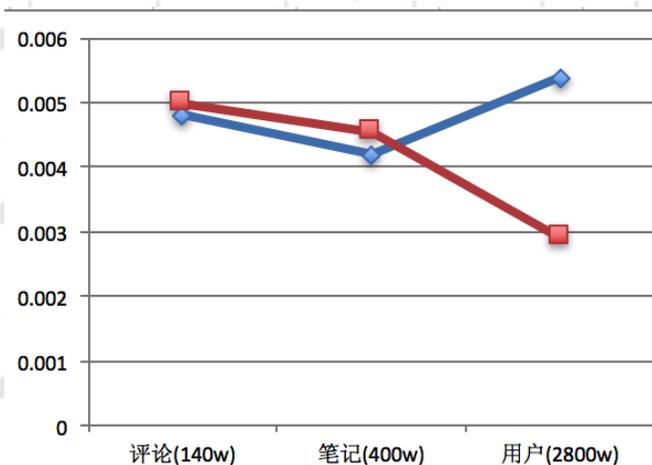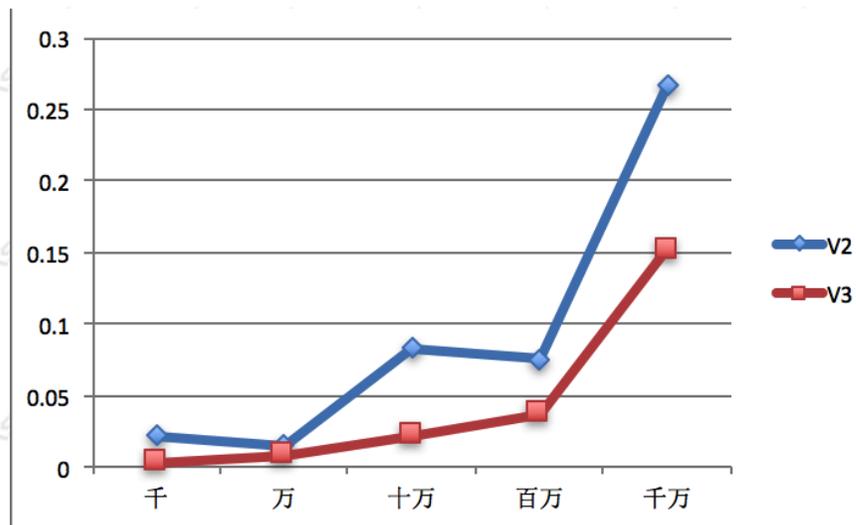
搜索优化

统一使用规范

- 基于RESTful风格api
- 增加业务权限管控
- 统一接入方式

取数优化

- 指定列簇读取
- 批量读
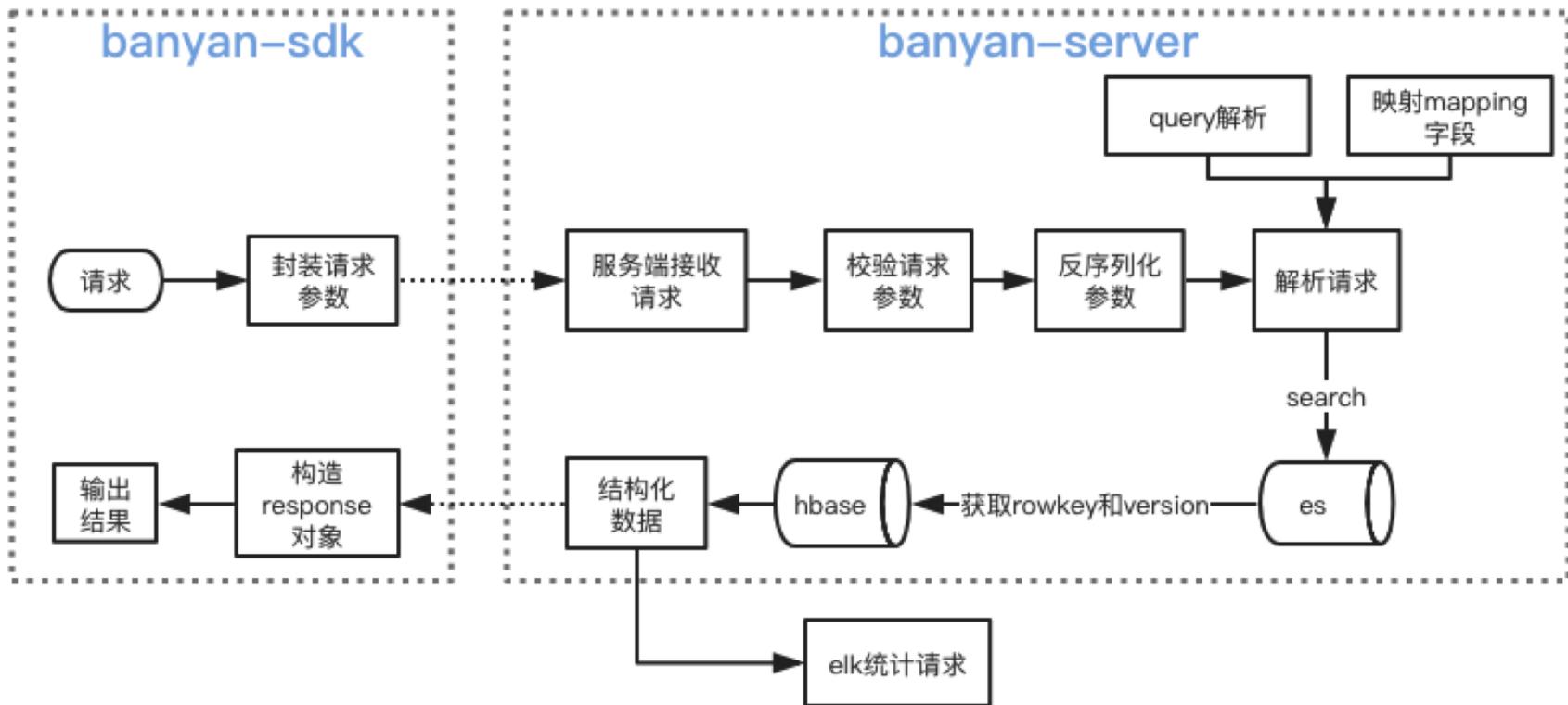
搜索优化

复杂查询优化

- 限制查询时间范围
    - 时间粒度最好年、季度
    - 尽量少跨索引查询
- 控制关键词和过滤词数
    - 避免单独使用must_not
- 默认不排序
- 限制查询字段
    - SDK限制字段查询方式
- 业务侧缓存
- 数据预处理
    - 降低查询复杂度
        - date_histogram->terms

搜索优化

es版本升级（2.3->5.6）

- 稳定性
- 支持Lucene 6.x
  - 索引优化（数值等类型字段底层索引结构Block K-d Tree）
  - 性能提升
- Aggregation能力大幅提高
- Scroll性能提升（并行）
- Profile API
- 后续版本升级方便
- 配套工具多

其他优化

HBase脚本

- 定期自动Major Compaction

Es脚本

- 定期自动Force Merge
  - max_num_segments=index_size/shards/2g
- 定期自动新建索引
- 定期自动索引迁移，根据index的tag标签进行集群域（zone）切换

Es配置

- index.refresh_interval=10s（控制segment数量）
- allocated_processors最大分配32核
- thread_pool.search（allocated_processors*3/2+1=49)
- thread_pool.index（allocated_processors=32)
- thread_pool.bulk（allocated_processors=32)
- ...

Thank you!