elastic | 中国开发者大会

B站搜索自动化实践

管华新
BiliBili

B站搜索自动化实践

----管华新

目录

一 平台概况

- 搜索平台
  - SDK
    - Query
      - eq、in、range、not、combo
      - 模糊和精准like、高亮
      - group by、distinct、sum、count、cardinality等
      - order、random order、function_score
      - pn、ps
    - Upsert
      - 支持新增或修改索引
      - 支持批量多个索引
  - SVEN配置平台
    - 接入
      - 数据源配置：DB、ES、Databus、Table
      - 业务组配置：打包业务给容器、增量方式、增量开关
      - 单个业务配置
        - 数据源：单表、多表、分表、跨库、databus、kafka
        - 规则类：字段转换规则、索引创建和过期规则、数据归属规则、过滤规则等
    - 统计
      - 业务方接入统计
      - 各业务方数据量统计
  - 业务
    - 审核：稿件、评论、弹幕、工作流等各个业务方
    - 创作中心
    - 个人中心
    - 开放平台
    - 行为日志
    - 其他：电竞、pgc类、TV端、漫画等
  - ES集群
    - 独立集群
    - 公共集群
    - 内外集群
    - 日志集群
  - 监控
    - ES集群（Prometheus）：集群状态，GC，slow log
    - 业务监控：耗时、QPS
    - 物理机：负载，CPU，内存，磁盘，IO

平台概况

- **数据：**
  1. 15个集群，45个节点
  2. 19条业务线，覆盖90个小业务
  3. 总量50+TB
  4. 5kw+条/天，1kw+次查询/天（对外请求占95%）
- **以前的问题**：
  1. 接入和迭代较慢，job和接口代码冗余
  2. 集群混用，相互耦合
  3. 单节点负载高，gc频繁

痛点优化

- 集群优化
- 查询性能优化
- 一致性和实时性优化
- 使用通用接口
- 使用通用配置

集群优化：解决单节点负载高（优化内存管理）

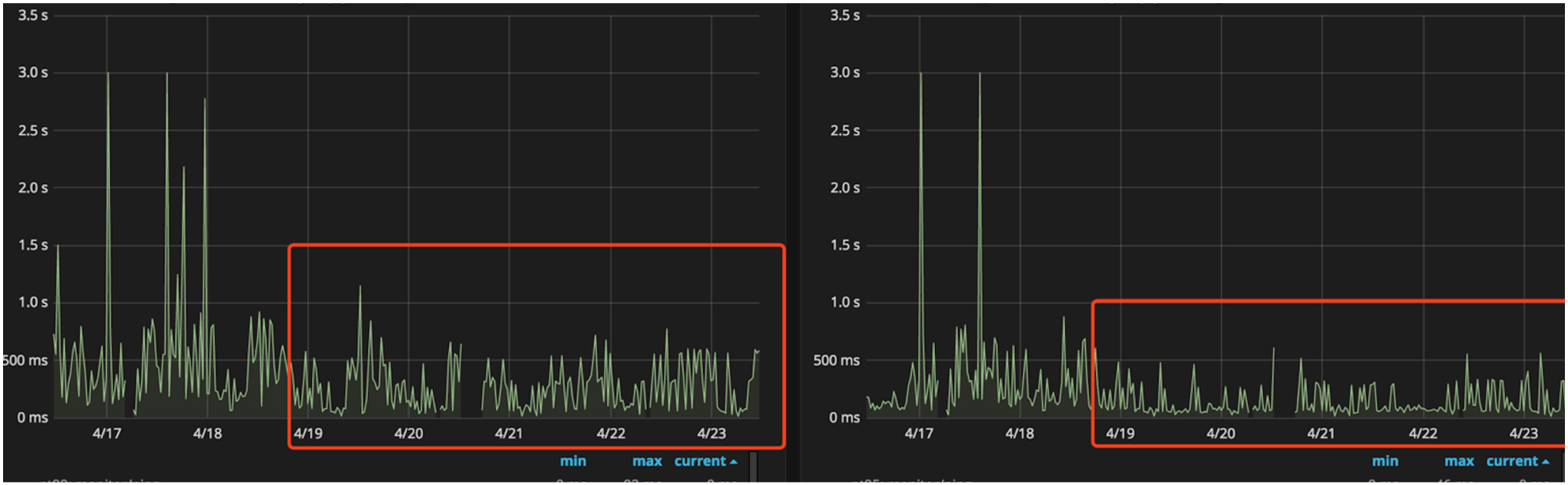
perf性能分析页回收频繁，而且free cache 低

```
Samples: 284K of event 'cycles', Event count (approx.): 61553373846
  49.51%  [kernel]                  [k] isolate_lru_pages.isra.46
   3.44%  [kernel]                  [k] __isolate_lru_page
   1.86%  [kernel]                  [k] shrink_inactive_list
   1.61%  [kernel]                  [k] __d_lookup_rcu
   0.78%  libc-2.19.so              [.] __lxstat64
   0.71%  libjvm.so                 [.] BlockOffsetArrayNonContigSpace::block_start_unsafe(void const*) const
   0.71%  libjvm.so                 [.] CompactibleFreeListSpace::block_size(HeapWord const*) const
   0.68%  [kernel]                  [k] _raw_spin_lock_irq
   0.68%  [kernel]                  [k] link_path_walk
   0.65%  libjvm.so                 [.] JVM_GetStackAccessControlContext
   0.56%  [kernel]                  [k] strncpy_from_user
   0.41%  libjvm.so                 [.] ClearNoncleanCardWrapper::do_MemRegion(MemRegion)
   0.29%  libjvm.so                 [.] ParNewGeneration::real_forwardee_slow(oopDesc*)
   0.28%  libjvm.so                 [.] ObjectMonitor::TrySpin_VaryDuration(Thread*)
   0.27%  libjvm.so                 [.] CodeHeap::find_start(void*) const
   0.27%  [kernel]                  [k] copy_user_enhanced_fast_string
   0.27%  libjvm.so                 [.] ParNewGeneration::copy_to_survivor_space_avoiding_promotion_undo(ParScanThreadState*, oopDesc*, unsigned long, markOopDesc*)
   0.25%  libjvm.so                 [.] vframeStreamCommon::fill_from_compiled_frame(int)
   0.25%  [kernel]                  [k] lookup_fast
   0.24%  libjvm.so                 [.] java_lang_Class::protection_domain(oopDesc*)
   0.24%  libjvm.so                 [.] SpinPause
   0.24%  libjvm.so                 [.] BacktraceBuilder::push(Method*, int, Thread*)
   0.23%  [kernel]                  [k] _raw_spin_lock
   0.22%  libjvm.so                 [.] vframeStreamCommon::next()
   0.22%  libjvm.so                 [.] java_lang_Throwable::fill_in_stack_trace(Handle, methodHandle, Thread*)
   0.22%  perf-162246.map           [.] 0x00007f9565f9a9c4
   0.21%  [kernel]                  [k] kmem_cache_alloc
   0.20%  [kernel]                  [k] kmem_cache_free
   0.20%  [kernel]                  [k] _raw_spin_lock_irqsave
   0.19%  [kernel]                  [k] __sk_run_filter
   0.19%  [kernel]                  [k] shrink_lruvec
   0.19%  libjvm.so                 [.] TypeArrayKlass::allocate_common(int, bool, Thread*)
   0.18%  perf-162246.map           [.] 0x00007f956923c086
   0.18%  libjvm.so                 [.] InstanceKlass::oop_oop_iterate_nv(oopDesc*, ParScanWithoutBarrierClosure*)
   0.18%  [kernel]                  [k] find_busiest_group
   0.17%  [kernel]                  [k] generic_permission
   0.17%  libjvm.so                 [.] FreeList<FreeChunk>::getFirstNChunksFromList(unsigned long, FreeList<FreeChunk>*)
   0.17%  perf-162246.map           [.] 0x00007f9566793dd0
   0.16%  libpthread-2.19.so        [.] __lll_lock_elision
   0.15%  [kernel]                  [k] path_init
   0.15%  libjvm.so                 [.] PromotionInfo::promoted_oops_iterate_nv(ParScanWithoutBarrierClosure*)
   0.15%  perf-162246.map           [.] 0x00007f956923bd33
   0.14%  [kernel]                  [k] lockref_get_not_dead
   0.13%  libc-2.19.so              [.] realpath
   0.13%  libjvm.so                 [.] CardTableModRefBS::process_chunk_boundaries(Space*, DirtyCardToOopClosure*, MemRegion, MemRegion, signed char**, unsigned long, unsigned long)
   0.13%  libjvm.so                 [.] InstanceKlass::oop_oop_iterate_nv(oopDesc*, ParScanWithBarrierClosure*)
   0.12%  libjvm.so                 [.] ParScanThreadState::trim_queues(int)
   0.12%  perf-162246.map           [.] 0x00007f95653427f4
   0.11%  libjvm.so                 [.] JavaCalls::call_helper(JavaValue*, methodHandle*, JavaCallArguments*, Thread*)
   0.11%  libjvm.so                 [.] ConcurrentMarkSweepGeneration::par_promote(int, oopDesc*, markOopDesc*, unsigned long)
   0.11%  [kernel]                  [k] security_inode_getattr
   0.11%  libjvm.so                 [.] ObjectMonitor::NotRunnable(Thread*, Thread*)
   0.11%  [kernel]                  [k] __inode_permission
   0.11%  [kernel]                  [k] system_call
   0.10%  [kernel]                  [k] __schedule
   0.10%  [kernel]                  [k] system_call_after_swapgs
   0.10%  perf-162246.map           [.] 0x00007f9565f9abfa
   0.10%  perf-162246.map           [.] 0x00007f9567c353c1
   0.10%  [kernel]                  [k] xfs_vn_getattr
   0.09%  libjvm.so                 [.] frame::sender_for_compiled_frame(RegisterMap*) const
   0.09%  perf-162246.map           [.] 0x00007f95653425bb
   0.09%  libjvm.so                 [.] ParallelTaskTerminator::offer_termination(TerminatorTerminator*)
   0.09%  [kernel]                  [k] lru_add_drain_cpu
```

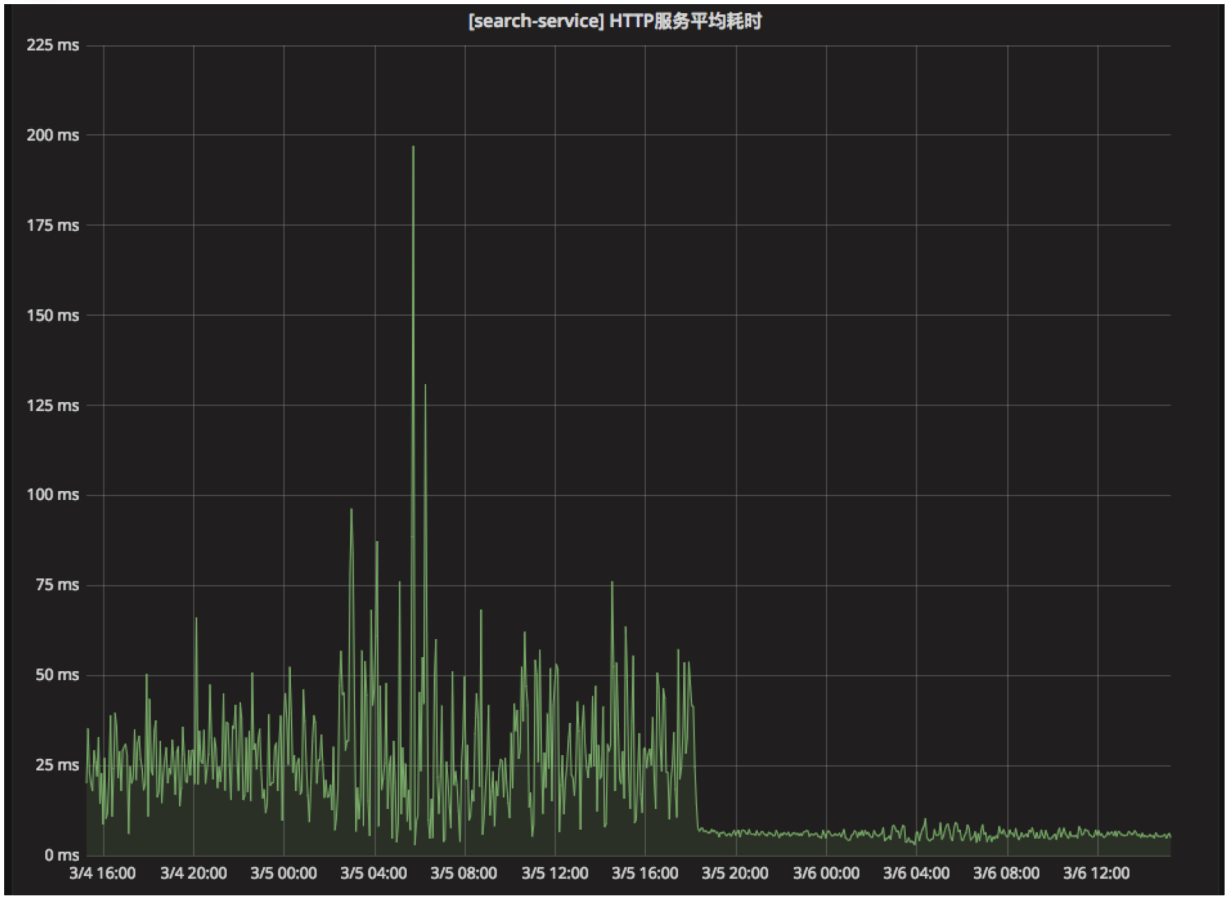
解决办法：

1. vm.swappiness = 1
2. vm.min_free_kbytes=655350
3. vm.zone_reclaim_mode = 0

查询优化：Mapping优化（int 转 keyword）



二 查询优化：SSD -> PCIE（接口响应平滑）



二 查询优化：CMS -> G1 GC（减少old gc和yong gc ）

二 查询优化：CMS -> G1 GC（减少old gc和yong gc ）

danielmitterdorfer Elastic Team Member 27d

Hi,

we will be supporting G1 with the upcoming version of Elasticsearch 6.5.0 but only if you run Elasticsearch on JDK 10 or later. For more context: the support has been added in #33685 17.

Daniel

1. ES6.5正式支持G1
2. 官方建议jdk10或以上版本使用G1

Add commented out JVM options for G1GC #33685

Merged dakrone merged 3 commits into elastic:master from dakrone:add-g1gc-for-jdk-10 on 22 Sep

Conversation 10 Commits 3 Checks 0 Files changed 1

Changes from all commits Jump to... +8 −0 Diff settings Review changes

8 distribution/src/config/jvm.options View

```
@@ -37,6 +37,14 @@
 -XX:CMSInitiatingOccupancyFraction=75
 -XX:+UseCMSInitiatingOccupancyOnly

+## G1GC Configuration
+# NOTE: G1GC is only supported on JDK version 10 or later.
+# To use G1GC uncomment the lines below.
+# 10-:-XX:-UseConcMarkSweepGC
+# 10-:-XX:-UseCMSInitiatingOccupancyOnly
+# 10-:-XX:+UseG1GC
+# 10-:-XX:InitiatingHeapOccupancyPercent=75
+
 ## optimizations

 # pre-touch memory pages used by the JVM during initialization
```

优化：一致性和实时性优化

- 消息队列替代扫表增量
- 统一使用Bulk和doc_as_upsert，对版本conflict做retry
- 当消费慢的时候，业务方对实时性又比较敏感，建议业务方使用update接口阻塞式更新，或是业务方查缓存和db对数据剔除
- 增量和全量交给业务方，一致性交给他们，我们保障集群的稳定性

SDK：通用查询体特点

1. 加快接入和迭代

业务方按照自己的查询场景，自助写查询语句，现在90%业务做到0发版。

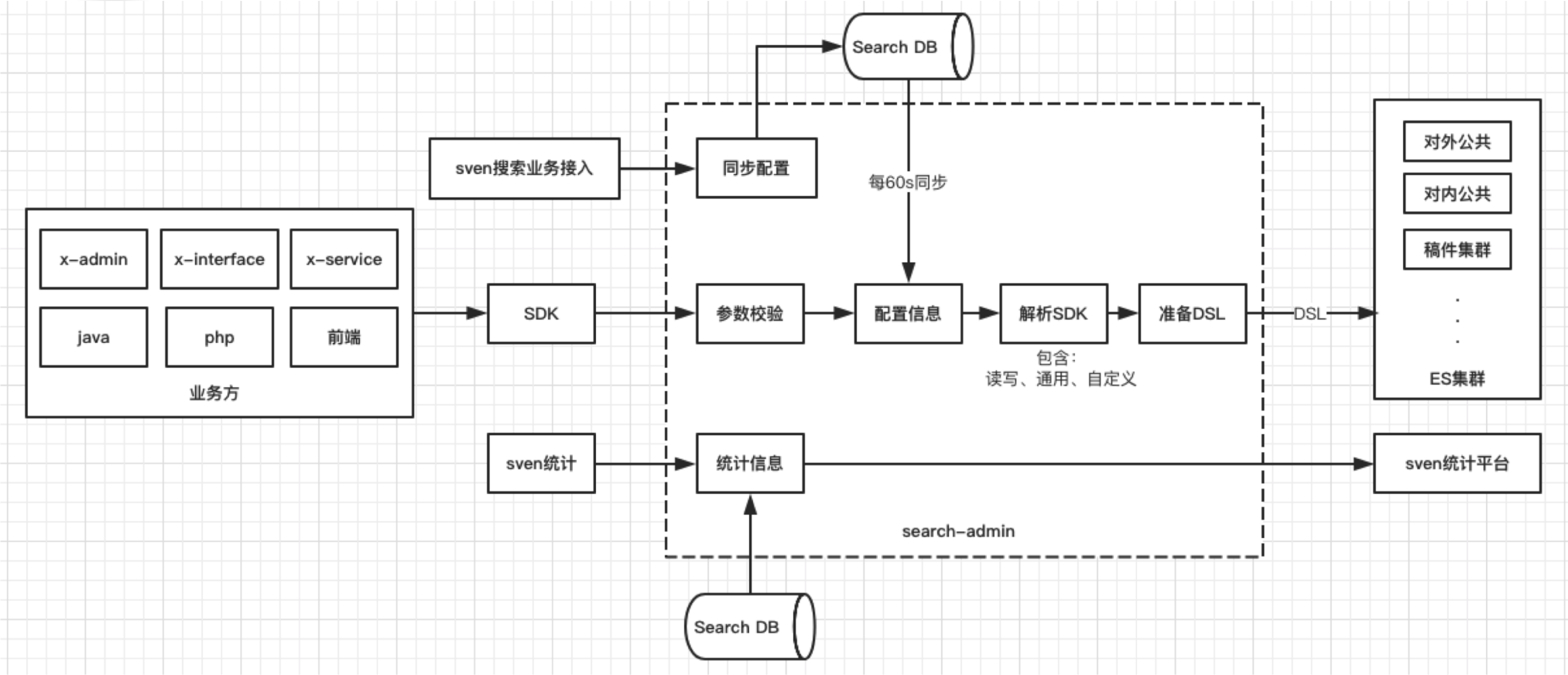
2. 简化复杂度

官方的DSL对业务方比较绕，而且容易采坑。简化nested，group by， cardinality，混合逻辑查询等的使用。

3. 安全、规范、效率

业务方直接使用DSL查询容易对集群有隐患，给不同的业务分配：最多查询索引个数、max_result_window、scroll、是否集群嗅探等权限点。

三 SDK: 接口服务设计


Search DB
sven搜索业务接入
同步配置
每60s同步
对外公共
对内公共
稿件集群
ES集群
x-admin
x-interface
x-service
java
php
前端
业务方
SDK
参数校验
配置信息
解析SDK
准备DSL
DSL
包含:
读写、通用、自定义
sven统计
统计信息
sven统计平台
search-admin
Search DB

SDK：查询功能

1. 过滤：Eq、in、range、not、combo
2. 关键词：模糊、精准、高亮
3. 统计：group by、distinct、sum、count、cardinality等
4. 排序：order、random order、function_score
5. 分页：pn、ps、scroll
6. 定制：继续使用通用SDK传参，只需要很少量的硬编码append到对应业务的通用查询中

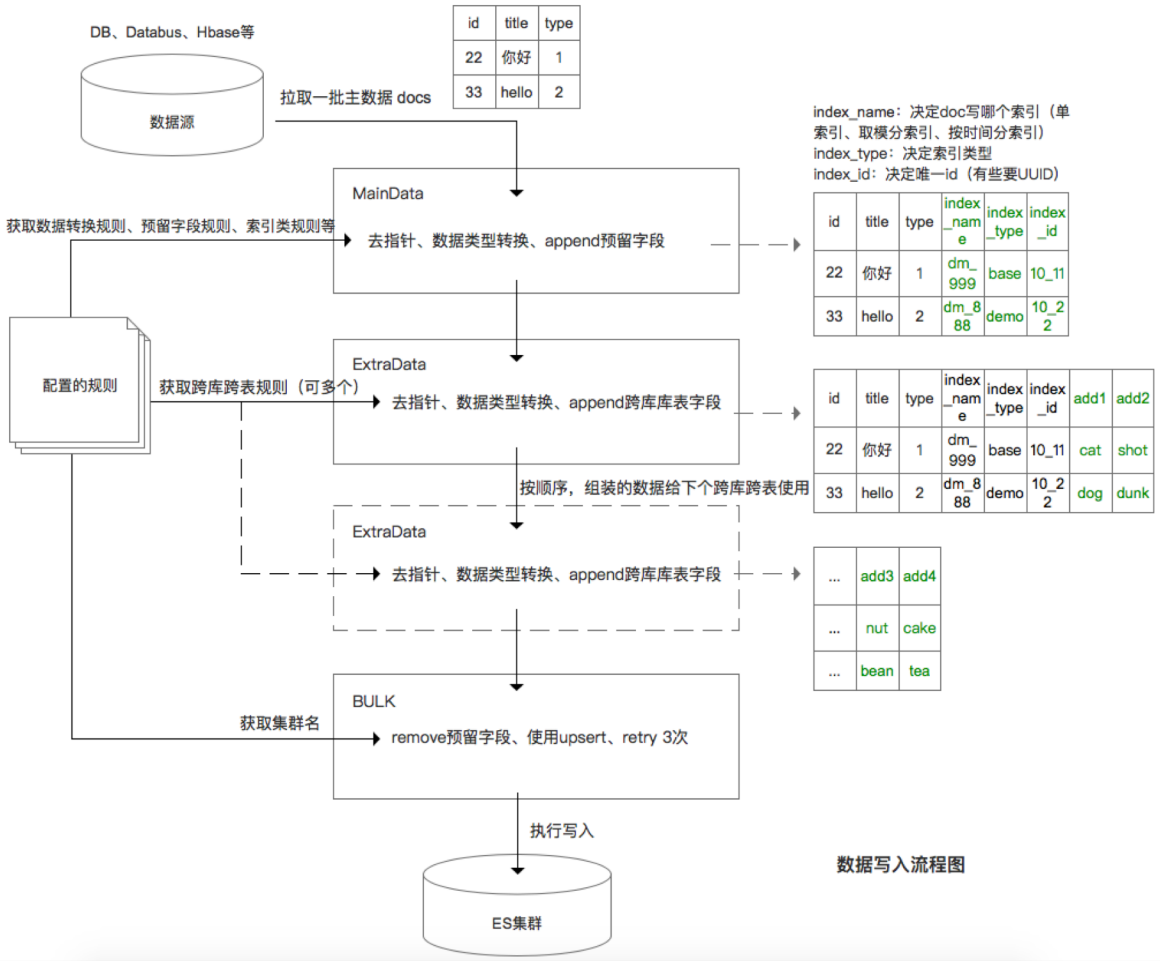
SDK：关键词查询

1. wildcard
2. ngram(2,2)
3. match

SDK: 通用 Update & Insert

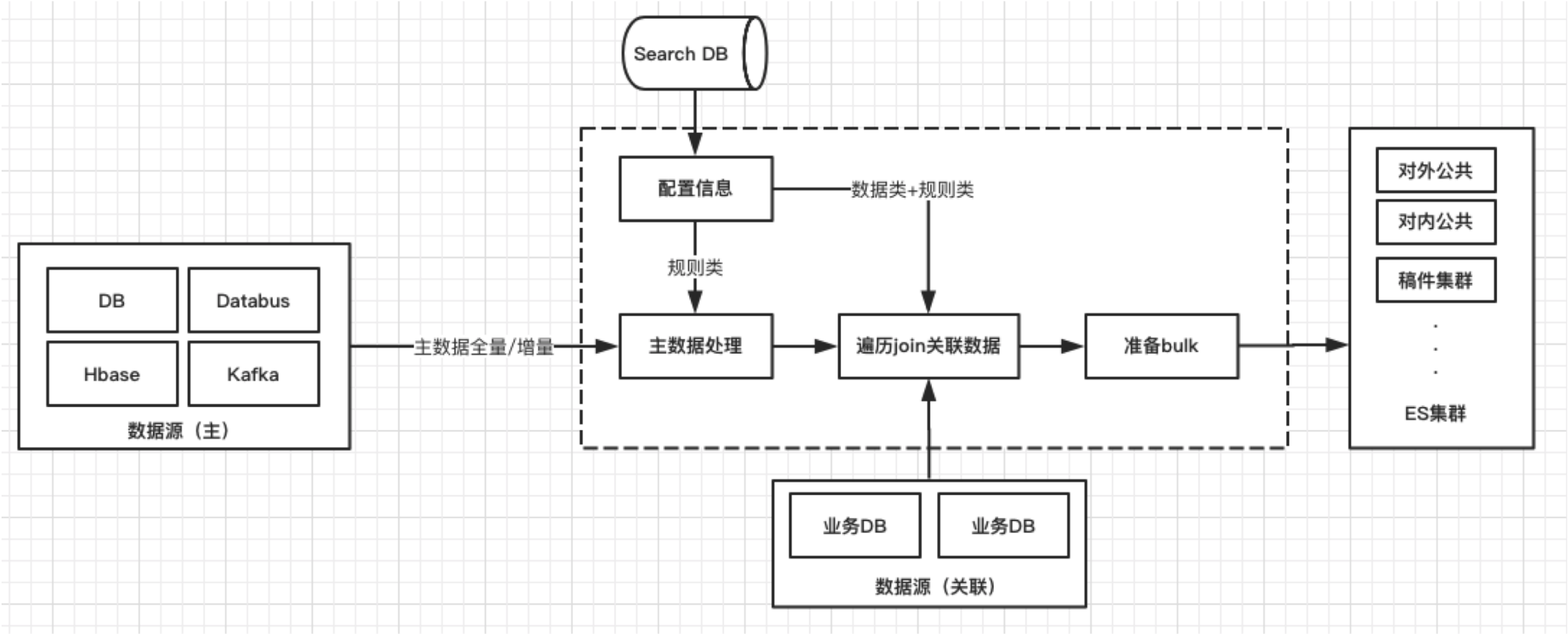
简化BULK，面向多个索引批量更新、新增数据

四 JOB: 数据聚合设计

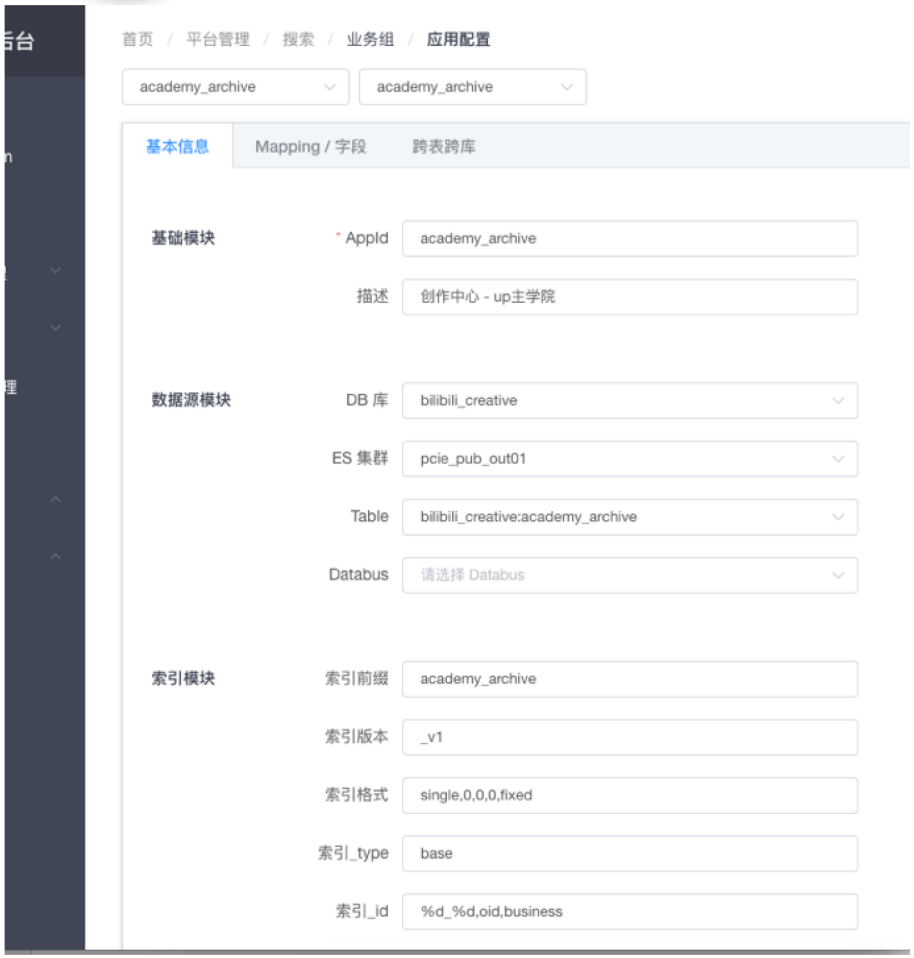


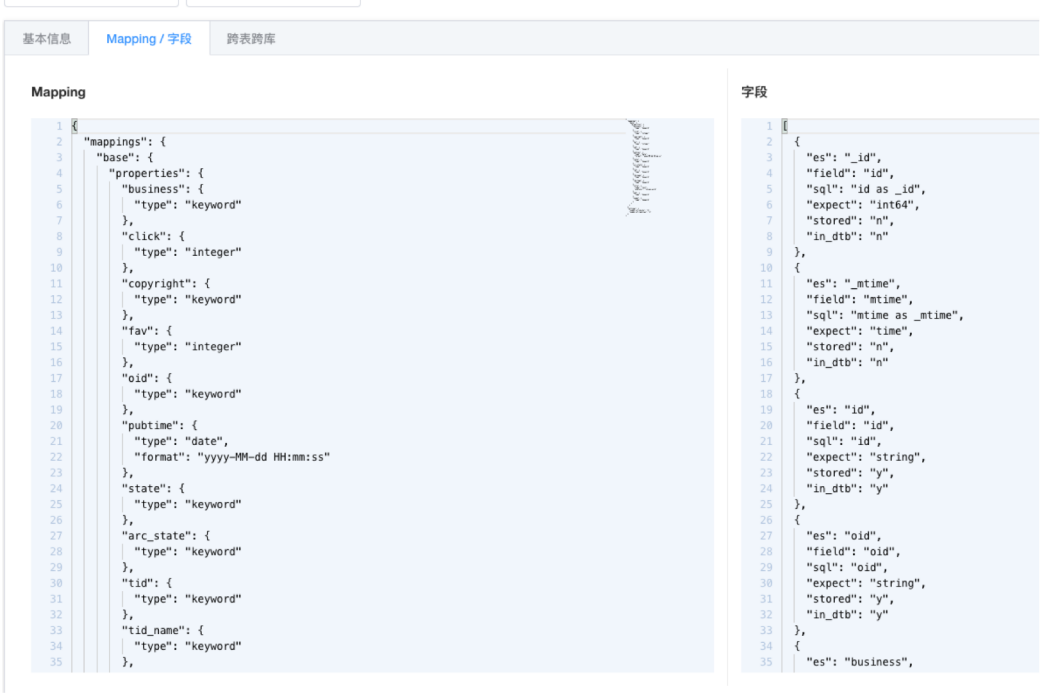
数据写入流程图

四 JOB: 通用job服务设计



四 JOB: SEVN配置平台





四 JOB：跨库跨表（实时join）

基本信息　Mapping / 字段　跨表跨库

跨表跨库

```
{
  "condition": {
    "in_field": "oid",
    "include": "business=1"
  },
  "dbname": "[redacted]",
  "in_field": "id",
  "fields_str": "id:id:int64:y,mid:mid:int64:y,title:title:string:y,pubtime:pubtime:string:y,copyright:copyright:string:y,arc_state:s
  "remove_fields": [
    "id"
  ],
  "fields": [
    "id",
    "mid",
    "title",
    "pubtime",
    "copyright",
    "arc_state"
  ],
  "sql": "select id,mid, title, pubtime,copyright,state as arc_state from [redacted] where id in (%s)"
},
```

四 JOB: 索引平滑升级

1. job初始化索引时，实体索引统一加版本号，加别名。默认使用别名读写。
2. 修改索引mapping时，新建带有新版本号的实体索引，双写两个索引，通过控制切换别名的方式，达到切换mapping的目的

四 JOB: 支持索引管理

1. 单个索引、按表名一致方式建索引（单表或分表都可以）、取模拆分、按时间拆分（年月日周）
2. 不同业务方，配置单次请求最多索引个数，默认为1

运维：部署和监控

- Ansible统一部署，es版本一致
- ES-Exporter接入Prometheus，集群监控、GC、线程池等增加阈值告警
- 物理机监控，内存、CPU、磁盘等阈值告警
- 业务方接口监控，耗时告警
- 消息队列监控，延迟告警

运维：索引管理

- 基于官方curator
- 面向按时间拆分的索引，提前创建、别名、template（年月日周）
- 定期挪动冷索引到冷节点
- 定期删除过期索引

谢谢

elastic
中文社区

专业、垂直、纯粹的 Elastic 开源技术交流社区
https://elasticsearch.cn/

elastic