ES底层原理

1. 正排索引（doc values）VS倒排索引：

(1) **概念**：从广义来说，doc values 本质上是一个序列化的 列式存储 。列式存储 适用于聚合、排序、脚本等操作，所有的数字、地理坐标、日期、IP 和不分析（not_analyzed）字符类型都会默认开启。

(2) **特点**：倒排索引的优势 在于查找包含某个项的文档，相反，如果用它确定哪些项是否存在单个文档里。

(3) **优化**：es官方是建议，es大量是基于os cache来进行缓存和提升性能的，不建议用jvm内存来进行缓存，那样会导致一定的gc开销和oom问题，给jvm更少的内存，给os cache更大的内存。比如64g服务器，给jvm最多4~16g（1/16~1/4），os cache可以提升doc value和倒排索引的缓存和查询效率。

2. fielddata：查询时内存数据结构

```
GET /product/_search
{
  "aggs": {
    "tag_agg_group": {
      "terms": {
        "field": "tags.keyword"
      }
    }
  },
  "size":0
}
```

3. 基于mget批量查询以及基于bulk的批量增删改

(1) mget：批量查询

GET /_mget

GET /<index>/_mget

(2) bulk：批量增删改 no-query

语法格式：

POST /_bulk

POST /<index>/_bulk

{"action": {"metadata"}}

{"data"}

Operate：

1. create：PUT /index/_create/id/，强制创建（是否制定id）
2. delete：删除（lazy delete原理）
3. index：可以是创建，也可以是全量替换
4. update：执行partial update（全量替换，部分替换）

4. ES并发冲突问题（悲观锁和乐观锁）

(1) 悲观锁：各种情况，都加锁，读写锁、行级锁、表级锁。使用简单，但是并发能力很低

(2) 乐观锁：并发能力高，操作麻烦，每次no-query操作都需要比对version