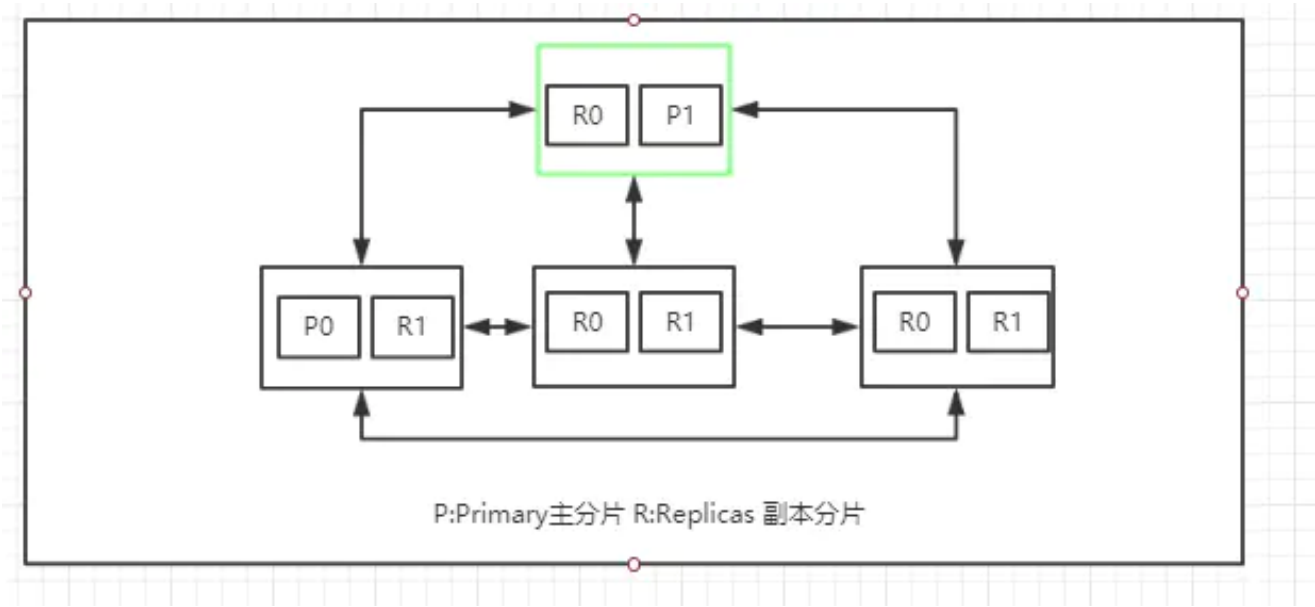
基本概念

Elasticsearch 是什么?

> 分布式、RESTful 风格的搜索和数据分析引擎

- Index索引：具有相似特性的文档集合，类比数据库表
- Document文档：相当于SQL里的一行记录
- Field字段：相当于SQL里的一个字段
- Shard分片：一个索引可切分为多个shard，分布在不同机器节点。每个shard都是一个lucene index
- Replica shard副本分片： shard故障备用，提升搜索吞吐量和性能
- Mapping映射：定义索引中字段的类型和索引设置【不设置的话，会保存但是不能用于搜索】



基本操作

kibana操作：http://10.248.33.26:30601/app/kibana#/dev_tools/console?_g=()

索引操作

增加索引：

```
PUT testq
{
    "mappings" : {
      "dynamic" : "false",  //未设置类型的字段是否自动设置
      "properties" : {
        "id" : {
          "type" : "keyword"
        },
```

```
        "qid" : {
          "type" : "long"
        },
        "question" : {
          "type" : "text",
          "fields" : {
            "keyword" : {
              "type" : "keyword",
              "ignore_above" : 256
            }
          }
        }
    },
    "settings": {
      "index": {
        "number_of_shards" : "3"  //分片数量
      }
    }
}
```

查询索引：

```
GET test_index/_mapping
GET test_index
```

```
GET _cat/indices/testq?v
health status index uuid                   pri rep docs.count docs.deleted store.size
pri.store.size
yellow open   testq SLCsRtLPQJSx6WG7SQ6zwg   3   1          0            0       690b
        690b
```

修改索引

```
//增加字段类型，允许
PUT testq/_mapping
{
  "properties": {
    "update_time": {
      "type": "date"
    }
  }
}

//修改字段类型，不允许
PUT test_index/_mapping
{
  "properties": {
    "create_time": {
```

```
      "type": "keyword"
    }
  }
}
```

删除索引：

```
DELETE test_index
```

注意点：

- 索引mapping设置：指定用于搜索的字段类型，不搜索可不设置
- 如果原来已有数据，新增索引字段后，根据此字段取索引老数据，是不会生效的【为什么？？】

文档操作

写入新文档：

```
POST testq/_doc/5639bf1b89bc4603d5c612d9
{
  "create_time" : "2015-11-04T08:17:31.343Z",
  "id" : "5639bf1b89bc4603d5c612d9",
  "qid" : 7,
  "question" : "是否有满足某条件的商品",
  "subcategory_id" : "5639bf1a89bc4603d5c61260",
  "update_time" : "2017-08-29T09:16:12.997Z"
}
```

查询新文档：

```
GET testq/_doc/5639bf1b89bc4603d5c612d9
{
  "_index" : "testq",
  "_type" : "_doc",
  "_id" : "5639bf1b89bc4603d5c612d9",
  "_version" : 1,
  "_seq_no" : 0,
  "_primary_term" : 1,
  "found" : true,
  "_source" : {
    "create_time" : "2015-11-04T08:17:31.343Z",    //上面索引设置中，没有此字段，但仍然有存储
    "id" : "5639bf1b89bc4603d5c612d9",
    "qid" : 7,
    "question" : "是否有满足某条件的商品",
    "subcategory_id" : "5639bf1a89bc4603d5c61260",
    "update_time" : "2017-08-29T09:16:12.997Z"
  }
```

```
}
```

修改文档：

```
POST testq/_doc/5639bf1b89bc4603d5c612d9
{
  "create_time" : "2015-11-04T08:17:31.343Z",
  "id" : "5639bf1b89bc4603d5c612d9",
  "qid" : 10,
  "question" : "是否有满足某条件的商品",
  "subcategory_id" : "5639bf1a89bc4603d5c61260",
  "update_time" : "2017-08-29T09:16:12.997Z"
}
```

删除文档：

```
DELETE testq/_doc/5639bf1b89bc4603d5c612d9
```

注意点：

- 修改文档：实际是分为删除文档和重新写入文档

搜索操作

```
GET testq/_search
{
  "query": {
    "match": {
      "question": {
        "query": "商品"
      }
    }
  }
}

{
  "took" : 0,
  "timed_out" : false,
  "_shards" : {
    "total" : 3,
    "successful" : 3,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 1,
```

```
      "relation" : "eq"
    },
    "max_score" : 0.26706278,
    "hits" : [
      {
        "_index" : "testq",
        "_type" : "_doc",
        "_id" : "5639bf1b89bc4603d5c612d9",
        "_score" : 0.26706278,
        "_source" : {
          "create_time" : "2015-11-04T08:17:31.343Z",
          "id" : "5639bf1b89bc4603d5c612d9",
          "qid" : 10,
          "question" : "是否有满足某条件的商品",
          "subcategory_id" : "5639bf1a89bc4603d5c61260",
          "update_time" : "2017-08-29T09:16:12.997Z"
        }
      }
    ]
  }
}
```

聚合操作

max、min、avg等

```
GET testq/_search
{
   "size" : 0,
   "aggs": {
      "avg_qid": {
         "avg": {
            "field": "qid"
         }
      }
   }
}

  "aggregations" : {
    "avg_qid" : {
      "value" : 15.0
    }
  }
```

分组操作

```
GET testq/_search
```

```
{
  "size":0,
  "aggs": {
    "qid_group": {
      "terms": { "field": "qid" }
    }
  }
}

  "aggregations" : {
    "qid_group" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : 10,
          "doc_count" : 1
        },
        {
          "key" : 20,
          "doc_count" : 1
        }
      ]
    }
  }
```

业务使用实例

搜索高亮和排序

场景： 按关键词搜索，需要高亮

问题：关键词评分一样时，多次点击搜索，问题顺序会变


```
POST question_b/_search
{
  "highlight": {
    "fields": {
      "question": {}
    }
  },
  "sort": [
    {
      "_score": {
        "order": "desc"
      }
    }, {
      "update_time": {
        "order": "desc"
```

```
      }
    }
  ],
  "query": {
    "bool": {
      "must": [{
        "match_phrase": {
          "question": {
            "query": "商品"
          }
        }
      }, {
        "terms": {
          "id": ["5f9676ed9cb3f44f1829f463", "5c08d7aded33953ecbbd45a5",
"59eb2ab2369f99529e2aaf37", "59eb2ab2369f99529e2aaf54", "59eb2ab2369f99529e2aaf50",
"59eb2ab2369f99529e2aaf53", "5f90271ca5dd43298b59c798", "59eb2ab3369f99529e2aafb9",
"59eb2ab2369f99529e2aaf4c", "59eb2ab2369f99529e2aaf4f", "59eb2ab2369f99529e2aaf52",
"59eb2ab3369f99529e2aafc1", "59eb2ab2369f99529e2aaf39", "59eb2ab2369f99529e2aaf3b",
"59eb2ab2369f99529e2aaf51", "5db2c4249a560700128ac295", "59eb2ab2369f99529e2aaf56",
"59eb2ab2369f99529e2aaf3e", "59eb2ab2369f99529e2aaf4e", "5db2c4249a560700128ac292",
"59eb2ab2369f99529e2aaf3a", "59eb2ab3369f99529e2aaf91", "5c7fb6472afd1a3a4a71c484",
"59eb2ab2369f99529e2aaf44", "59eb2ab2369f99529e2aaf4b", "59eb2ab2369f99529e2aaf45",
"59eb2ab2369f99529e2aaf38", "59eb2ab3369f99529e2aafb5", "5df34c242752430018aa8b50",
"59eb2ab3369f99529e2aafc4", "59eb2ab2369f99529e2aaf55", "5c3eb7475fca0d7711bdf3e2",
"5f8ed169c9bbdce0a913b1ae", "5ba389481a6ab2424ce015b7", "59eb2ab2369f99529e2aaf3f",
"5be2cf7676251154bf6cd78d", "5be2cf7676251154bf6cd78c", "59eb2ab2369f99529e2aaf4d",
"5bcc687b369f9910638884d0", "5b62ec995e3773504aba4411", "5ddf73870a37b100146f76f1",
"5ddf73870a37b100146f76f3", "603706b7563150aae6083908", "5e4e77092dd8ca00181789ab"]
        }
      }]
    }
  }
}
```

返回结果：

```
      {
        "_index" : "orig1_question_b",
        "_type" : "_doc",
        "_id" : "59eb2ab3369f99529e2aafb5",
        "_score" : 5.640376,
        "_source" : {
          "create_time" : "2017-10-21T11:08:35.503Z",
          "id" : "59eb2ab3369f99529e2aafb5",
          "qid" : 251021,
          "question" : "看中某商品",
          "subcategory_id" : "59eb2aab1a6ab20c5c1bf256",
          "update_time" : "2021-02-25T02:08:32.311Z"
        },
```

```
        "highlight" : {
          "question" : [
            "看中某<em>商</em><em>品</em>"
          ]
        },
        "sort" : [
          5.640376,
          1614218912311
        ]
      }
```

字段组合查询

场景：商品关联回复页面导出，需要拿到商品所有回复，因为数据太多会超时。

```
question_id: id1, conds_md5: cond1
question_id: id1, conds_md5: cond2
question_id: id2, conds_md5: cond3
```

问题数目：最多1000个行业问题+ x个自定义问题

条件数目：精准意图条件、时效条件、售后阶段等条件，多达100+

一个商品的qid 和conds_md5 组合：1k+到1w+

优化前：

```
POST shop_condition_answer/_search
{
  "query": {
    "bool": {
      "minimum_should_match": "1",  //或者，should中任意满足一个
        "should": [
          {
              "bool": {
               "filter": [{"term": {"question_id": "5907f68b1a6ab2086eecc24d"}},
                  {"term": {"conds_md5": "4f9922bdc95c131342dada07e56aa5b7"}}]}
          },
          {
              "bool": {
               "filter": [{"term": {"question_id": "5907f7391a6ab2086eecc28c"}},
                  {"term": {"conds_md5": "7f979de66388f1fd173735695218caa9"}}]}
          }
          ]
      }
    }
}
```

- 测试环境：1000个左右时，耗时6s-10s，如果几千个会超过20s，导致失败
- 解决办法：新增一个字段qid_md5，字段值为“question_id.conds_md5”，将或查询优化为terms查询
- 优化后耗时：几千个回复，只需要0.8s-1s

优化后：

```
POST shop_condition_answer/_search
{
  "query": {
    "bool": {
      "filter": {
            "terms": {
              "qid_md5": [
                "5907f68b1a6ab2086eecc24d.4f9922bdc95c131342dada07e56aa5b7",
                "5639bf1b89bc4603d5c612d6.09ce0fc5b50de2cd93dc35a895a9fc85",
                "5907f7391a6ab2086eecc28c.5e479b954b0c4c0c301958bf5c2e49ab",
                "5907f7391a6ab2086eecc28c.7f979de66388f1fd173735695218caa9"]
            }
      }
    }
  }
}
```

分组聚合排序

功能：全局搜索下，按条件聚合展示

- 展示按问题分组，每个问题下面按conds_md5字段（即同样的条件）分组
- 条件的更新时间：取值于条件下所有回复的最新更新时间
- 问题下只展示最新的一个条件，并且需展示总条件数
- 条件下只展示最新的两条回复，并且需展示总回复数

以前方式：

1. 从es中获取店铺这一批问题下，所有的回复记录
2. 内存中，使用分组和排序，分别计算每个问题条件数、回复数
3. 如果店铺回复数过多，会导致响应慢超时，上万回复数时会耗时20s左右。

现在方式：使用es来进行分组排序，大数据量下响应稳定在400ms以下

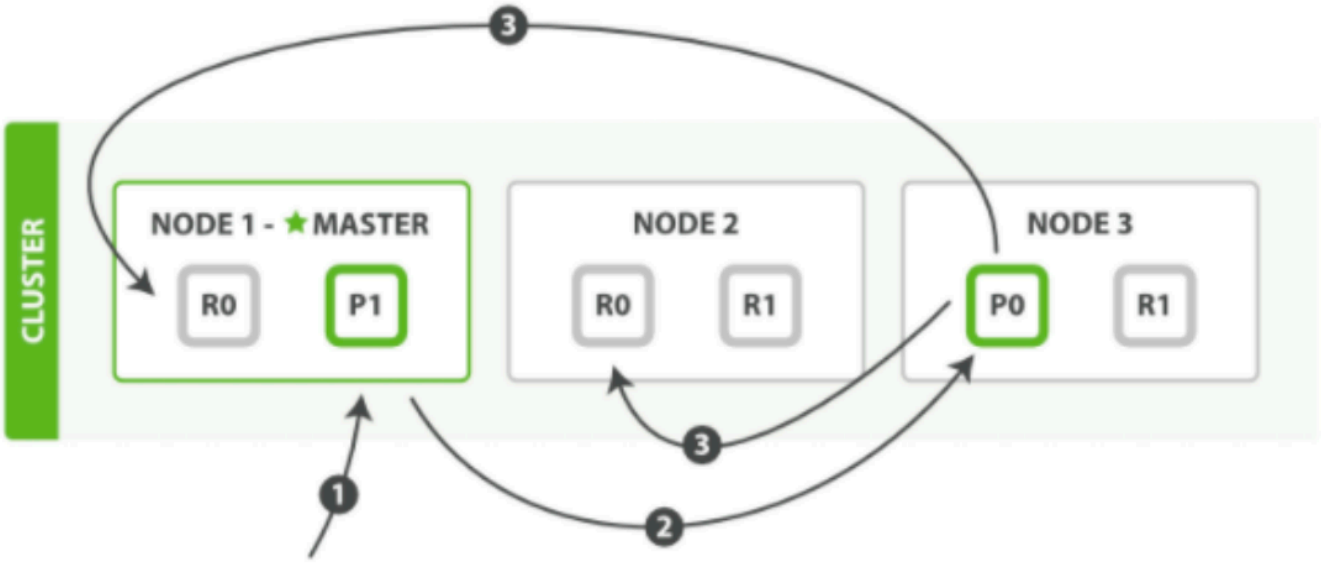
```
{
  "query": {},
  "aggregations": {
    "question_id_agg": { //按问题聚合
      "terms": {  //指定按 question_id 分组，返回20个问题的数据
        "field": "question_id",
        "size": 20
      },
      "aggregations": { //子聚合，即条件聚合
        "conds_md5_agg": {  //子聚合1，  取到conds_md5组内中最大的update_time，来对所有
conds_md5排序，取前一个。
          "aggregations": {
            "max_update_time": {
              "max": {
                "field": "update_time"
              }
            }
          },
          "terms": {
            "field": "conds_md5",
            "order": [{
              "max_update_time": "desc"
            }],
            "size": 1
          }
        },
        "distinct_md5": {  //子聚合2，获取一个问题下不同的conds_md5个数（即基数）
          "cardinality": {
            "field": "conds_md5",
            "precision_threshold": 40000
          }
        }
      }
    }
  }
}
```

集群读写流程

几个基本概念

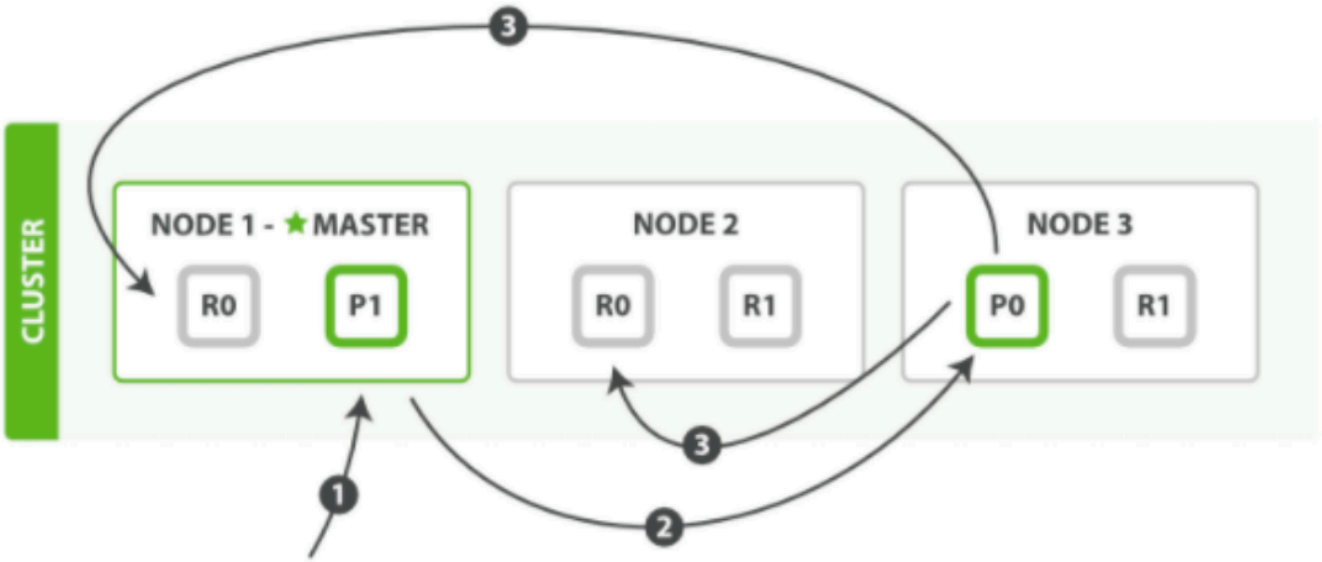
- 集群Master节点：管理集群范围的变更，例如创建或删除索引、添加或删除节点
- 请求可发到集群任一节点，根据路由转发到实际节点
- 如何负载均衡：轮询所有节点【客户端是否可像redis缓存路由信息，减少转发？？？】
- 如何路由：shard = hash(routing)%index_primary_shard_number
- routing是啥：默认是_id，可根据业务情况制定，比如shop_id

写流程



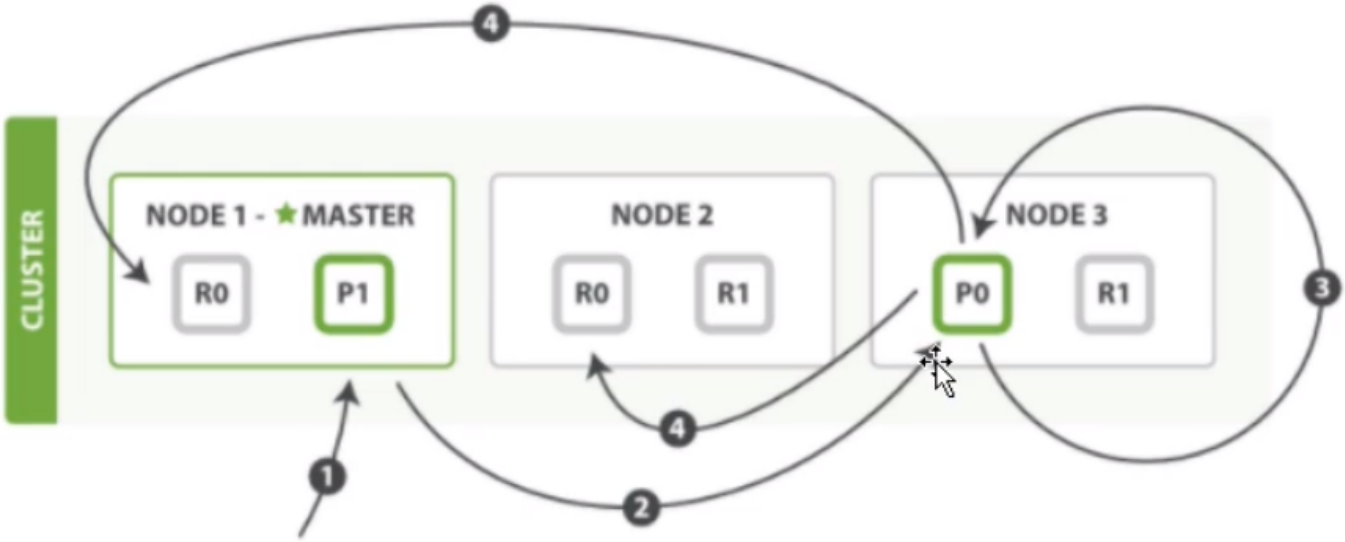
1. 客户端向 Node 1 发送插入一条文档
2. Node 1根据 _id 计算路由应该写到分片 0 ，转发Node 3处理
3. Node 3 在主分片上面执行请求，并转发到 Node 1 和 Node 2 的副本分片上
4. 返回给Node 1，Node 1返回给客户端

读流程



1. 客户端向 Node 1 发送获取请求
2. Node 1根据 _id 计算路由应该读分片 0， 分片 0 的存在于三个节点上，转发到任意节点，这里是Node 2
3. Node 2 将文档返回给 Node 1 ，Node 1返回给客户端

更新流程



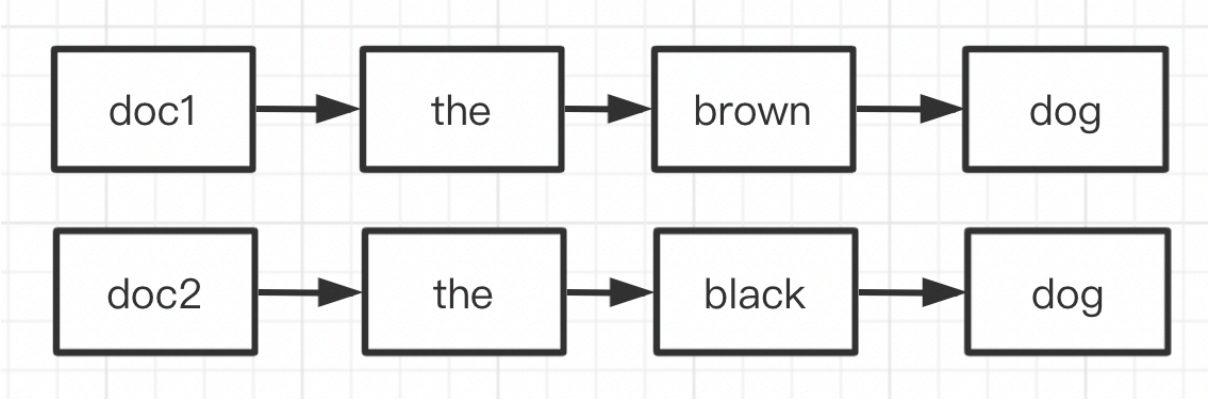
1. 客户端向 Node 1 发送更新请求
2. 路由转发到主分片所在的 Node 3
3. Node 3 从主分片检索文档，修改后重新写入一条文档，将老文档标记为删除
4. 更新文档后，同步到其他副本分片
5. Node 3返回给Node 1，Node 1返回到客户端

搜索原理

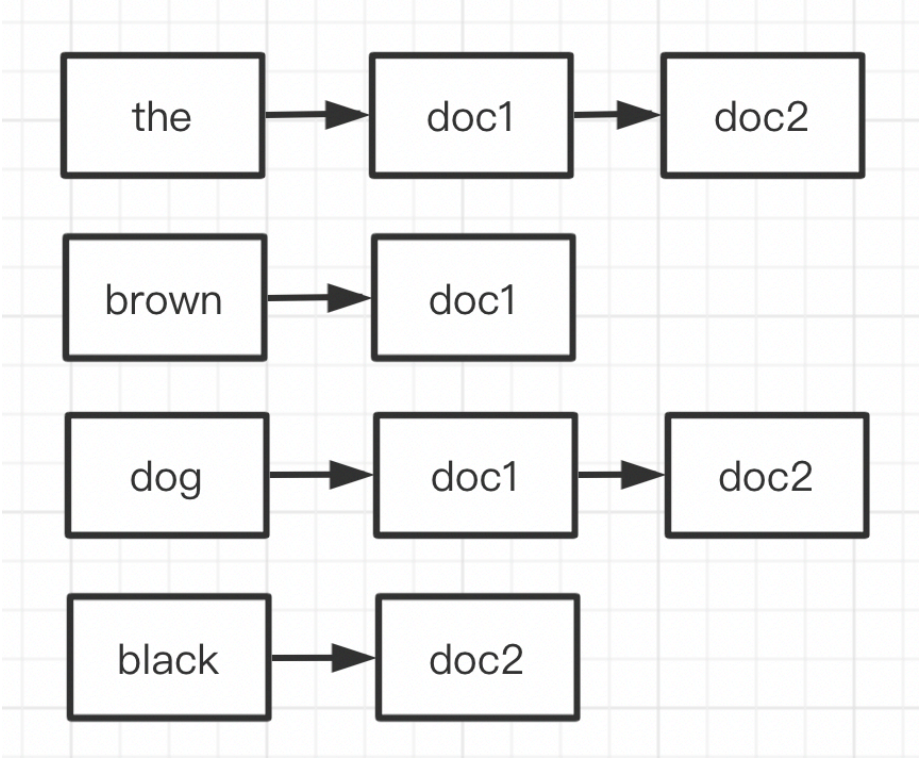
倒排索引

- doc1：the brown dog
- doc2：the black dog

正向索引：



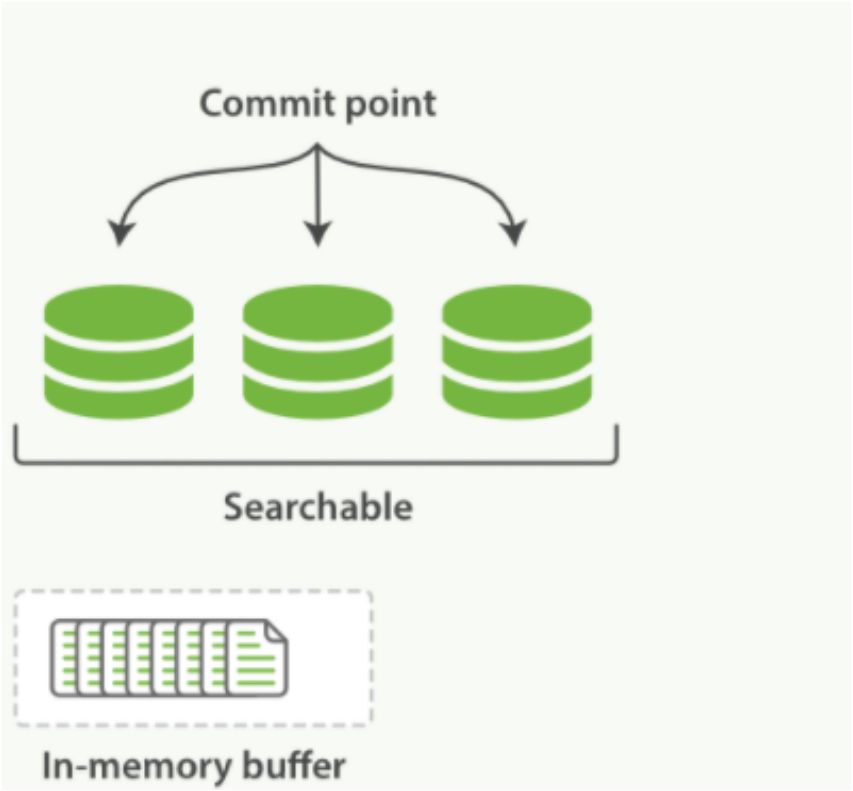
倒排索引:



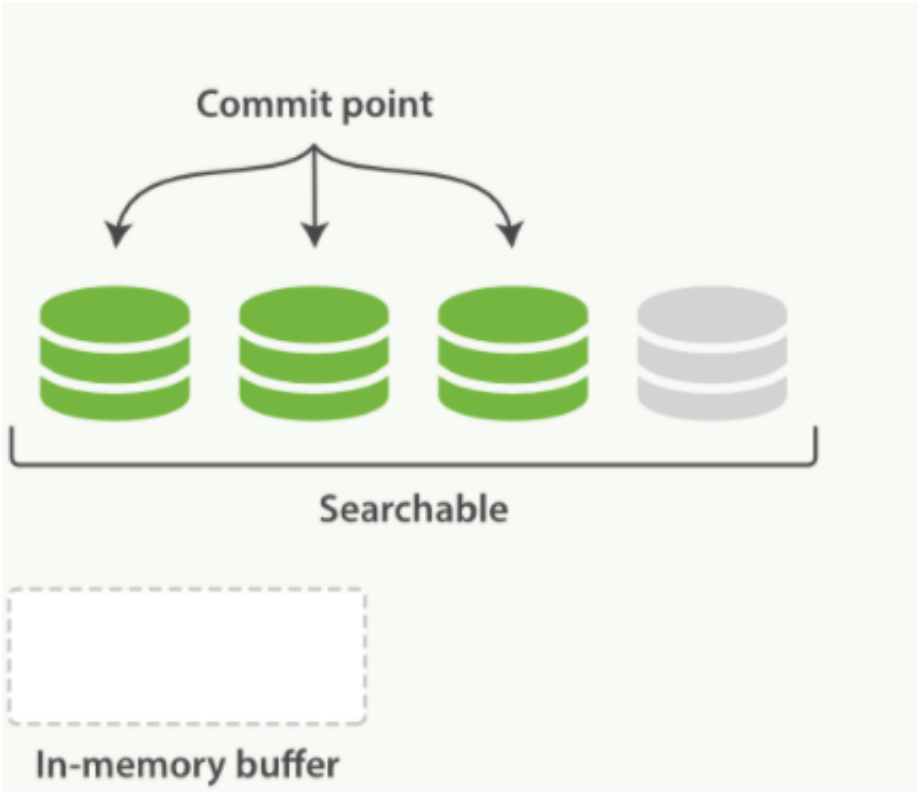
如何设置分词：官网分词器介绍

文档刷新

在内存缓冲区中包含了新文档的Lucene 索引



缓冲区的内容已经被写入一个可被搜索的段中，但还没有进行提交



- 文档首先写入内存中，此时可获取但不可搜索
- 默认每隔1s 从内存refresh到新段segment中，然后可搜索
- 刷新有性能消耗，不需要高实时性，时间可以设置长一点

```
PUT question_b/_settings
{
  "index": {
    //"refresh_interval": "30s" //全力同步数据时
    "refresh_interval": "1s" //默认
  }
}
```

文档刷新和刷写