elasticsearch

加入尚学堂，一起进步！

- Elasticsearch中文分词器集成
- 详见文档

- Elasticsearch的java API
- 添加maven依赖
- 连接到es集
  - 1：通过TransportClient这个接口，我们可以不启动节点就可以和es集群进行通信，它需要指定es集群中其中一台或多台机的ip地址和端口
    - TransportClient client = new TransportClient().addTransportAddress(new InetSocketTransportAddress("host1", 9300)).addTransportAddress(new InetSocketTransportAddress("host2", 9300));
    - 如果需要使用其他名称的集群（默认是elasticsearch），需要如下设置
    - Settings settings = ImmutableSettings.settingsBuilder().put("cluster.name", "myClusterName").build();
    - TransportClientclient = new TransportClient(settings).addTransportAddress(new InetSocketTransportAddress("host1", 9300));
  - 2：通过TransportClient这个接口，自动嗅探整个集群的状态，es会自动把集群中其它机器的ip地址加到客户端中
    - Settings settings = ImmutableSettings.settingsBuilder().put("client.transport.sniff", true).build();
    - TransportClient client = new TransportClient(settings).addTransportAddress(new InetSocketTransportAddress("host1", 9300));

- Elasticsearch的java API
  - 索引index（四种json,map,bean,es helpers）
    - IndexResponse response = client.prepareIndex(“bjsxt", "emp", "1").setSource().execute().actionGet();
  - 查询get
    - GetResponse response = client.prepareGet(“bjsxt", "emp", "1").execute().actionGet();
  - 更新update
  - 更新或者插入upsert
  - 删除delete
    - DeleteResponse response = client.prepareDelete(“bjsxt", "emp", "1").execute().actionGet();
  - 总数count
    - long count = client.prepareCount(“bjsxt").execute().get().getCount();

- Elasticsearch的java API
- 批量操作bulk
- 查询search
  - SearchType

元素	含义
QUERY_THEN_FETCH	查询是针对所有的块执行的，但返回的是足够的信息，而不是文档内容（Document）。结果会被排序和分级，基于此，只有相关的块的文档对象会被返回。由于被取到的仅仅是这些，故而返回的hit的大小正好等于指定的size。这对于有许多块的index来说是很便利的（返回结果不会有重复的，因为块被分组了）。
QUERY_AND_FETCH	最原始（也可能是最快的）实现就是简单的在所有相关的shard上执行检索并返回结果。每个shard返回一定尺寸的结果。由于每个shard已经返回了一定尺寸的hit，这种类型实际上是返回多个shard的一定尺寸的结果给调用者。
DFS_QUERY_THEN_FETCH	与QUERY_THEN_FETCH相同，预期一个初始的散射相伴用来为更准确的score计算分配了的term频率。
DFS_QUERY_AND_FETCH	与QUERY_AND_FETCH相同，预期一个初始的散射相伴用来为更准确的score计算分配了的term频率。
SCAN	在执行了没有进行任何排序的检索时执行浏览。此时将会自动的开始滚动结果集。
COUNT	只计算结果的数量，也会执行facet。

- Elasticsearch的查询
- es的搜索类型有4种
  - query and fetch(速度最快)(返回N倍数据量)
  - query then fetch（默认的搜索方式）
  - DFS query and fetch(可以更精确控制搜索打分和排名。)
  - DFS query then fetch
- DFS解释：见备注
- 总结一下，从性能考虑QUERY_AND_FETCH是最快的，DFS_QUERY_THEN_FETCH是最慢的。从搜索的准确度来说，DFS要比非DFS的准确度更高。

- Elasticsearch的查询
- 查询:query
  - .setQuery(QueryBuilders.matchQuery("name", "test"))
- 分页:from/size
  - .setFrom(0).setSize(1)
- 排序:sort
  - .addSort("age", SortOrder.DESC)
- 过滤:filter
  - .setPostFilter(FilterBuilders.rangeFilter("age").from(1).to(19))
- 高亮:highlight

- 统计:facet(已废弃)使用aggregations 替代
  - 根据字段进行分组统计
  - 根据字段分组，统计其他字段的值
  - size设置为0，会获取所有数据，否则，只会返回10条

- Elasticsearch的分页
- 与SQL使用LIMIT来控制单“页”数量类似，Elasticsearch使用的是from以及size两个参数：
  - size：每次返回多少个结果，默认值为10
  - from：从哪条结果开始，默认值为0
- 假设每页显示5条结果，那么1至3页的请求就是：
  - GET /_search?size=5
  - GET /_search?size=5&from=5
  - GET /_search?size=5&from=10
- 注意：不要一次请求过多或者页码过大的结果，这么会对服务器造成很大的压力。因为它们会在返回前排序。一个请求会经过多个分片。每个分片都会生成自己的排序结果。然后再进行集中整理，以确保最终结果的正确性。

- Elasticsearch的分页
- timed_out告诉了我们查询是否超时
- curl -XGET http://localhost:9200/_search?timeout=10ms
  - es会在10ms之内返回查询内容
- 注意：timeout并不会终止查询，它只是会在你指定的时间内返回当时已经查询到的数据，然后关闭连接。在后台，其他的查询可能会依旧继续，尽管查询结果已经被返回了。

- Elasticsearch的分页
- timed_out告诉了我们查询是否超时
- curl -XGET http://localhost:9200/_search?timeout=10ms
  - es会在10ms之内返回查询内容
- 注意：timeout并不会终止查询，它只是会在你指定的时间内返回当时已经查询到的数据，然后关闭连接。在后台，其他的查询可能会依旧继续，尽管查询结果已经被返回了。