链滴

深入理解 Query String Query

作者：felayman

原文链接：https://ld246.com/article/1512989390999

来源网站：链滴

Query String Query

国内对于Elasticsearch深入的人并不多,或者大多数大牛不屑于分享关于Elasticsearch的知识,这里讲讲Elasticsearch中的Constant Score Query

关于

- 官方文档: 官方文档
- ApacheCN : 中文文档

概念

Query String Query提供了无需指定某字段而对文档全文进行匹配查询的一个高级查询,同时可以指在哪些字段上进行匹配.

Query String Query是真正意义上的全文检索

- 可以在多个索引下进行无指定字段检索
- 可以在指定的索引/类型下进行无指定字段检索

这两个特性导致Query String Query是真正意义上的全文检索,可以让你在浩瀚的ES集群中检索出你要的数据.

语法

```
GET /_search
{
   "query": {
      "query_string" : {
         "default_field" : "content",
         "query" : "this AND that OR thus"
      }
   }
}
```

query_string作为一个高级的全文查询,有很多参数来控制其检索行为

- query 查询匹配的内容
- default_field 如果未指定前缀字段，则为查询字词的默认字段。 默认为index.query.default_fiel索引设置，默认为_all。
- default_operator 如果未指定显式运算符，则使用默认运算符。 例如，使用OR的默认运算符，匈利的查询资本将转换为OR匈牙利的资本OR，如果使用默认运算符AND，则相同的查询将转换为AN匈牙利的资本AND。 默认值为OR。
- analyzer 用于分析查询字符串的分析器
- allow_leading_wildcard 设置为 *或? 被允许作为第一个字符。 默认为true。
- lowercase_expanded_terms 通配符，前缀，模糊和范围查询的条件是否自动降低或不降低（因为们没有被分析）。 默认为true。

- enable_position_increments 设置为true可在结果查询中启用位置增量。 默认为true。
- fuzzy_max_expansions 控制模糊查询将扩展到的术语数。 默认值为50
- fuzziness 设置模糊查询的模糊性。 默认为AUTO。 有关允许的设置，请参阅“Fuzzinessedit”一节
- fuzzy_prefix_length 设置模糊查询的前缀长度。 默认值为0。
- phrase_slop 设置短语的默认斜率。 如果为零，则需要精确的短语匹配。 默认值为0。
- boost 设置查询的提升值。 默认为1.0。
- analyze_wildcard 默认情况下，不分析查询字符串中的通配符术语。 将此值设置为true，将尽力析这些值。
- auto_generate_phrase_queries 是否自动生成短语查询,默认False
- max_determinized_states 限制允许创建多少个自动机状态regexp查询。 这防止了太难的（例如数级的）正则表达式。 默认为10000。
- minimum_should_match 一个值，用于控制在生成的布尔查询中应该匹配多少个“应该”子句。可以是绝对值（2），百分比（30%）或两者的组合。
- lenient 如果设置为true将导致基于格式的失败（例如向数字字段提供文本）被忽略。
- locale 应用于字符串转换的区域设置。 默认为ROOT。
- time_zone 要应用于与日期相关的任何范围查询的时区。 另请参阅JODA时区

常用参数

default_field

用于指定默认查询的字段

Multi Field

Java API

```
  @Test
     public void testForClient() throws Exception {
         MultiMatchQueryBuilder multiMatchQueryBuilder = QueryBuilders.multiMatchQuery("e
asticsearch match query","title", "descrption");

         multiMatchQueryBuilder.analyzer("standard");
         multiMatchQueryBuilder.cutoffFrequency(0.001f);
         multiMatchQueryBuilder.field("title",20);
         multiMatchQueryBuilder.fuzziness(Fuzziness.TWO);
         multiMatchQueryBuilder.maxExpansions(100);
         multiMatchQueryBuilder.prefixLength(10);
         multiMatchQueryBuilder.tieBreaker(20);
         multiMatchQueryBuilder.type(MultiMatchQueryBuilder.Type.BEST_FIELDS);
         multiMatchQueryBuilder.boost(20);



        SearchResponse searchResponse =  client.prepareSearch()
                .setIndices("blogs")
```

```
                .setTypes("blog")
                .setQuery(multiMatchQueryBuilder)
                .execute()
                .actionGet();

        System.out.println(ResponseUtil.parse(searchResponse));
    }
```

更多关于Java API,请参考:MultiMatchQueryDemo

参考

- Match Phrase Query
- Common Terms Query