深入理解 Match Phrase Query

作者：felayman
原文链接：https://ld246.com/article/1512989203733
来源网站：链滴

Match Phrase 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

(Elasticsearch6.0会移除该查询,建议使用Common Terms Query)

关于

- 官方文档: 官方文档
- ApacheCN : 中文文档

概念

match_phrase查询分析文本,并从分析的文本中创建短语查询,match_phrase的主要作用是用于匹配含当前短语的文档

语法

```
GET /_search
{
    "query": {
        "match_phrase" : {
            "message" : "this is a test",
            "analyzer":"english",
            "slop":0
        }
    }
}
```

参数说明

- analyzer 指定何种分析器来对该短语进行分词处理
- slop 用于指定查询短语间的词项(term)间的距离
- boost 用于设置该查询的权重

slop参数说明

我们先看下测试数据

```
POST /my_index/my_type/_search
{
  "query": {
    "match_all": {}
  }
}
```

结果为:


```
{
  "took": 0,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 1,
    "hits": [
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "6",
        "_score": 1,
        "_source": {
          "title": "this is not a test"
        }
      },
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "7",
        "_score": 1,
        "_source": {
          "title": "this is really a test"
        }
      },
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "8",
        "_score": 1,
        "_source": {
          "title": "this is really not a test"
        }
      }
    ]
  }
}
```


我们使用match_phrase来进行普通(无参)查询

```
POST /my_index/my_type/_search
{
  "query": {
    "match_phrase": {
      "title": "this is"
    }
  }
```

```
}
```

结果会匹配所有文档,因为所有文档中都包含了this is,

```
{
  "took": 0,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 1,
    "hits": [
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "6",
        "_score": 1,
        "_source": {
          "title": "this is not a test"
        }
      },
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "7",
        "_score": 1,
        "_source": {
          "title": "this is really a test"
        }
      },
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "8",
        "_score": 1,
        "_source": {
          "title": "this is really not a test"
        }
      }
    ]
  }
}
```

我们改写查询为:

```
POST /my_index/my_type/_search
{
  "query": {
    "match_phrase": {
      "title": "this is a"
```

```
    }
  }
}
```

我们发现没有结果能匹配上,

```
{
  "took": 0,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "hits": {
    "total": 0,
    "max_score": null,
    "hits": []
  }
}
```

因为没有文档中包含"this is a",有人会说 不是都包含吗,只是中间隔了一个单词,对的这就是match_phrase的作用,匹配到的文档中必须

包含"this is a"而且顺序间隔必须跟输入的内容保持一致.

那如果我想文档中只要包含"this is a",但是他们之间能隔着一些单词呢?这个时候就需要靠参数来控制,这个参数就是slop,slop的数值意味着

你输入的短语中每个词项(term)之间允许隔着几个词项(term),我们改写上面的查询:

```
POST /my_index/my_type/_search
{
  "query": {
    "match_phrase": {
      "title": {
        "query": "this is a",
        "slop": 1
      }
    }
  }
}
```

我们发现会匹配到两条结果:

```
{
  "took": 1,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "hits": {
    "total": 2,
    "max_score": 0.26203912,
```

```
    "hits": [
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "6",
        "_score": 0.26203912,
        "_source": {
          "title": "this is not a test"
        }
      },
      {
        "_index": "my_index",
        "_type": "my_type",
        "_id": "7",
        "_score": 0.26203912,
        "_source": {
          "title": "this is really a test"
        }
      }
    ]
  }
}
```

发现在"is"和"a"之间都隔了一个单词(term),我们再次改写查询为:

```
POST /my_index/my_type/_search
{
  "query": {
    "match_phrase": {
      "title": {
        "query": "this is a",
        "slop": 2
      }
    }
  }
}
```

我们发现隔了两个term,也能搜出来了

```
{
  "took": 0,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 0.26203912,
    "hits": [
      {
        "_index": "my_index",
        "_type": "my_type",
```

```
      "_id": "6",
      "_score": 0.26203912,
      "_source": {
        "title": "this is not a test"
      }
    },
    {
      "_index": "my_index",
      "_type": "my_type",
      "_id": "7",
      "_score": 0.26203912,
      "_source": {
        "title": "this is really a test"
      }
    },
    {
      "_index": "my_index",
      "_type": "my_type",
      "_id": "8",
      "_score": 0.16023767,
      "_source": {
        "title": "this is really not a test"
      }
    }
  ]
 }
}
```

这就是slop的作用,在很多业务设计中都是需要用到这个参数来控制更好的全文检索的.

Java API

```
    @Test
    public void test() throws Exception {
        String key = "this is a";
        MatchPhraseQueryBuilder matchPhraseQueryBuilder = QueryBuilders.matchPhraseQuery
"title",key);

        matchPhraseQueryBuilder.boost(10);
        matchPhraseQueryBuilder.analyzer("standard");
        matchPhraseQueryBuilder.slop(2);

          SearchResponse searchResponse = client.prepareSearch()
                  .setIndices("my_index")
                  .setTypes("my_type")
                  .setQuery(matchPhraseQueryBuilder)
                  .execute()
                  .actionGet();
           System.out.println(ResponseUtil.parse(searchResponse));
    }
```

更多关于Java API,请参考:MatchPhraseQueryDemo

参考

- Match Phrase Query
- Common Terms Query