1、query string 分词

query string 必须以和 index 建立相同的 analyzer 进行分词
query string 对 exact value 和 full text的区别对待

data：exact value // 对于 data 类型的精确匹配
_all：full text   // 对于 全字符串 采用全文检索

例子：比如有一个 document， 其中有一个 field，包含的 value 是：hello you and me 建立倒排索引
　　我们搜索这个 document 对应的 index，搜索文本是 hello me， 这个搜索文本就是 query string
　　query string， 默认情况下，es 会使用它对应的 field 建立倒排索引时相同的分词器去进行分词，分词和normalization，只有这样，才能实现正确的搜索 例如：我们建立倒排索引的时候，将 dogs ---> dog, 结果你搜索的时候，还是一个 dogs， 那就搜索不到了，所以在搜索的时候，那个 dogs 也必须变成 dog 才行。

！！知识点： 不同类型的 field，可能有的就是 full text，有的就是exact value

post_data, data：exact value
_all：full text，分词，normalization

结论：在查询的时候回做一个分词，然后去倒排索引中去比对，但是如果搜索时候的分词和插入数据时候的分词器不一致的话，将无法进行正确的搜索；所以 搜索后的时候分词器与建立倒排缩影时候的分词器要一致

2、 因为 mapping 引入对搜索行为的影响

查询语句:
　GET /_search?q=2017
搜索的是 _all field, document 所有的 field 都会拼接成一个 大串，进行分词
例如：有三条数据

```
{
        "_index": "bookstore",
        "_type": "computer",
        "_id": "PbaqimEBmnUw7Gj9wAGp",
```

```
        "_score": 1,
        "_source": {
          "name": "daima jianjie zhidao ",
          "post_date": "2017-08-01",
          "author": "daiwei",
          "price": 30
        }
      },
      {
        "_index": "bookstore",
        "_type": "computer",
        "_id": "2",
        "_score": 1,
        "_source": {
          "name": "shengru lijie jvm",
          "post_date": "2017-10-01",
          "author": "zzq",
          "price": 24
        }
      },
      {
        "_index": "bookstore",
        "_type": "computer",
        "_id": "PLapimEBmnUw7Gj9EAHA",
        "_score": 1,
        "_source": {
          "name": "java 从入门到放弃",
          "post_date": "2017-10-10",
          "author": "zzw",
          "price": 23.5
        }
      }
```


如果进行，_all 进行搜索， 第二条数据可以得到以下 长串：

shengru lijie jvm 2017-10-01 zzq 24

因此，对段文本可以得到以下部分倒排索引：

	doc1	doc2	doc3
2017	*	*	*
10		*	*
08	*		
01	*	*	

因此 搜索 2017 能拿到三条数据

但是 如果 是以为 date 形式建立索引，2017 不会拆分 搜索 2017 将不会有结果

结论： 因为mapping 数据结构的不同对分词的结果会产生影响，因此造成搜索结果也会不同

3、测试分词器（可以调戏）

```
GET _analyze
{
  "analyzer": "standard",
  "text": " I love a girl who you understood well, I will lose my weight and show love to her when she is  single"
}
```

结果：

```
{
  "tokens": [
    {
      "token": "i",
      "start_offset": 1,
      "end_offset": 2,
      "type": "<ALPHANUM>",
      "position": 0
    },
    {
      "token": "love",
      "start_offset": 3,
      "end_offset": 7,
      "type": "<ALPHANUM>",
      "position": 1
    },
    {
      "token": "a",
      "start_offset": 8,
      "end_offset": 9,
      "type": "<ALPHANUM>",
      "position": 2
    },
    {
      "token": "girl",
      "start_offset": 10,
```

```
      "end_offset": 14,
      "type": "<ALPHANUM>",
      "position": 3
    },
    {
      "token": "who",
      "start_offset": 15,
      "end_offset": 18,
      "type": "<ALPHANUM>",
      "position": 4
    },
    {
      "token": "you",
      "start_offset": 19,
      "end_offset": 22,
      "type": "<ALPHANUM>",
      "position": 5
    },
    {
      "token": "understood",
      "start_offset": 23,
      "end_offset": 33,
      "type": "<ALPHANUM>",
      "position": 6
    },
    {
      "token": "well",
      "start_offset": 34,
      "end_offset": 38,
      "type": "<ALPHANUM>",
      "position": 7
    },
    {
      "token": "i",
      "start_offset": 40,
      "end_offset": 41,
      "type": "<ALPHANUM>",
      "position": 8
    },
    {
      "token": "will",
      "start_offset": 42,
      "end_offset": 46,
      "type": "<ALPHANUM>",
      "position": 9
    },
```

```
    {
      "token": "lose",
      "start_offset": 47,
      "end_offset": 51,
      "type": "<ALPHANUM>",
      "position": 10
    },
    {
      "token": "my",
      "start_offset": 52,
      "end_offset": 54,
      "type": "<ALPHANUM>",
      "position": 11
    },
    {
      "token": "weight",
      "start_offset": 55,
      "end_offset": 61,
      "type": "<ALPHANUM>",
      "position": 12
    },
    {
      "token": "and",
      "start_offset": 62,
      "end_offset": 65,
      "type": "<ALPHANUM>",
      "position": 13
    },
    {
      "token": "show",
      "start_offset": 66,
      "end_offset": 70,
      "type": "<ALPHANUM>",
      "position": 14
    },
    {
      "token": "love",
      "start_offset": 71,
      "end_offset": 75,
      "type": "<ALPHANUM>",
      "position": 15
    },
    {
      "token": "to",
      "start_offset": 76,
      "end_offset": 78,
```

```
      "type": "<ALPHANUM>",
      "position": 16
    },
    {
      "token": "her",
      "start_offset": 79,
      "end_offset": 82,
      "type": "<ALPHANUM>",
      "position": 17
    },
    {
      "token": "when",
      "start_offset": 83,
      "end_offset": 87,
      "type": "<ALPHANUM>",
      "position": 18
    },
    {
      "token": "she",
      "start_offset": 88,
      "end_offset": 91,
      "type": "<ALPHANUM>",
      "position": 19
    },
    {
      "token": "is",
      "start_offset": 92,
      "end_offset": 94,
      "type": "<ALPHANUM>",
      "position": 20
    },
    {
      "token": "single",
      "start_offset": 96,
      "end_offset": 102,
      "type": "<ALPHANUM>",
      "position": 21
    }
  ]
}
```