به نام خدا

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده برق و کامپیوتر

آزمایشگاه پایگاه داده

دستور کار شماره ۷

نام و نام خانوادگی

معین شیردل ۸۱۰۱۹۷۵۳۵

آبان ماه ۱۴۰۰

بخش اول) آموزش های سایت bigdata.ir:

به کمک دستورات بالا تعدادی رکورد در دیتابیس محلی elasticsearch ذخیره می کنیم تا داده برای شروع داشته باشیم. خروجی، حاصل اجرای آخرین دستور است.

به کمک دستور GET به صورت بالا می توانیم اطلاعات رکورد (یا در این مثال فیلم) مورد نظر که ID آن را می دانیم را بدست آوریم. در حالت بالا اطلاعات فیلم اول را میبینیم.

```
POST _all/_search
{
    "query": {
        "query_string": {
            "query": "kill"
        }
    }
}
```


```
{
  "took" : 23,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 2,
      "relation" : "eq"
    },
    "max_score" : 1.2667978,
    "hits" : [
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "3",
        "_score" : 1.2667978,
        "_source" : {
          "title" : "To Kill a Mockingbird",
          "director" : "Robert Mulligan",
          "year" : 1962,
          "genres" : [
            "Crime",
            "Drama",
            "Mystery"
          ]
        }
      },
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "5",
        "_score" : 1.2667978,
        "_source" : {
          "title" : "Kill Bill: Vol. 1",
          "director" : "Quentin Tarantino",
          "year" : 2003
```


با جستجو در تمامی رکوردها، رکوردهایی که عبارت Kill در آن ها به کار رفته است نمایش داده شدند که دو رکورد مشاهده شده به واسطه نامشان که Kill Bill و To Kill a Mockingbird است در نتایج آمده اند.

```
GET /movies/_search
{
    "query": {
        "query_string": {
            "query" : "title:ford OR kill"
        }
    }
}
```


```
          "Drama",
          "Mystery"
        ]
      }
    },
    {
      "_index" : "movies",
      "_type" : "movie",
      "_id" : "5",
      "_score" : 1.2667978,
      "_source" : {
        "title" : "Kill Bill: Vol. 1",
        "director" : "Quentin Tarantino",
        "year" : 2003,
        "genres" : [
          "Action",
          "Crime",
          "Thriller"
        ]
      }
    },
    {
      "_index" : "movies",
      "_type" : "movie",
      "_id" : "6",
      "_score" : 0.82546186,
      "_source" : {
        "title" : "The Assassination of Jesse James by the Coward Robert Ford",
        "director" : "Andrew Dominik",
        "year" : 2007,
        "genres" : [
          "Biography",
          "Crime",
          "Drama"
        ]
      }
    }
  ]
}
}
```


```
GET /movies/_search
{
    "query": {
        "query_string": {
            "query": "(title:kill) AND (director:Tarantino)"
        }
    }
}
```


```
{
  "took" : 1,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 1,
      "relation" : "eq"
    },
    "max_score" : 3.0309887,
    "hits" : [
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "5",
        "_score" : 3.0309887,
        "_source" : {
          "title" : "Kill Bill: Vol. 1",
          "director" : "Quentin Tarantino",
          "year" : 2003,
          "genres" : [
            "Action",
            "Crime",
            "Thriller"
          ]
        }
      }
    ]
  }
}
```

به شکل های مختلف می توان شروط متفاوت روی مقادیر فیلد ها گذاشت. مثلا در تصویر اول نتایجی که در title خود کلمه Ford را دارند را نیز خواسته ایم که یک رکورد به رکورد های قبلی اضافه کرده است و در تصویر دوم، با شروط منطقی مشخص کرده ایم که رکورد هایی با title دارای عبارت Ford و director دارای کلمه Tarantino را می خواهیم.

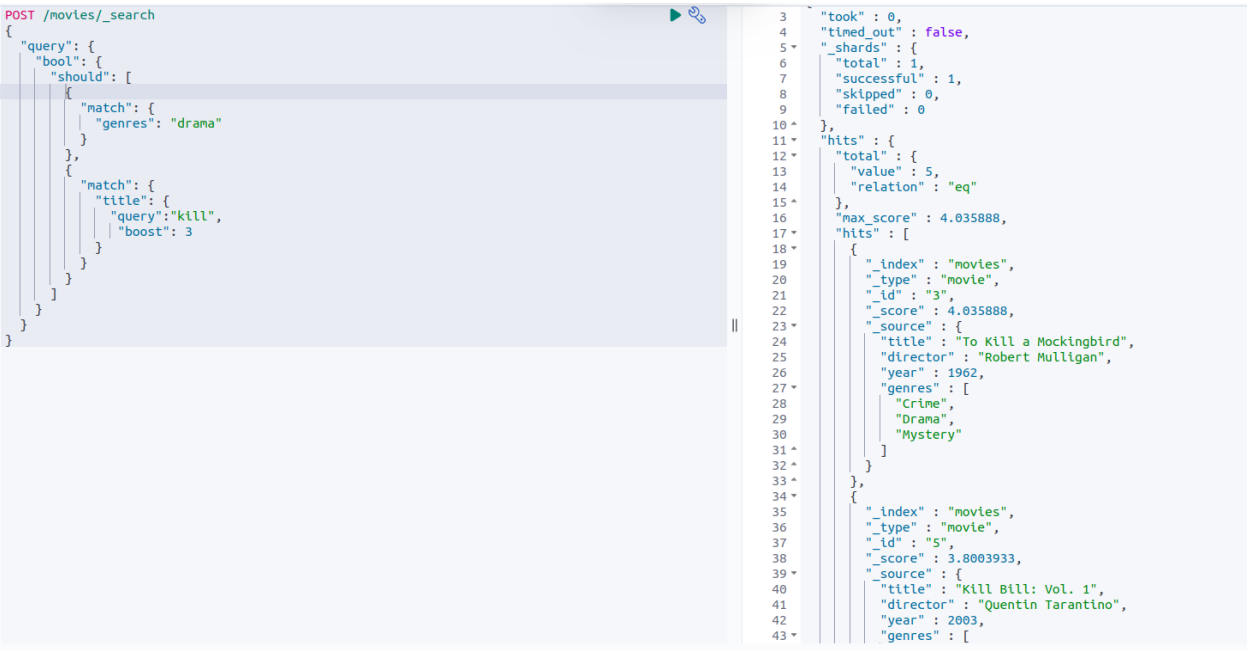
به کمک دستور بالا فیلم هایی با ژانر درام را پیدا میکنیم. این ساختار، ساختار عمومی تر برای اعمال فیلترها روی فیلدهاست.

هنگامی که کلمه query مانند Kill Bill باشد، الستیک آن را به مانند دو کلمه برای سرچ می بیند و به دنبال مچ کردن هر دو کلمه می گردد. Operator نشان دهنده نحوه قرار گرفتن نتایج هر دو مچ کردن در جواب نهاییست. می بینیم که حاصل این کوئری دو تا فیلم است که فیلم با عنوان Kill Bill Vol. 1 با هر دو کلمه مچ شده و score بالاتری هم گرفته است.

```
POST /movies/_search
{
  "query": {
    "multi_match": {
      "query": "ford",
      "fields": [
        "title^3",
        "director"
      ]
    }
  }
}
```

```
{
  "took" : 1,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 3,
      "relation" : "eq"
    },
    "max_score" : 2.4763856,
    "hits" : [
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "6",
        "_score" : 2.4763856,
        "_source" : {
          "title" : "The Assassination of Jesse James by the Coward Robert Ford",
          "director" : "Andrew Dominik",
          "year" : 2007,
          "genres" : [
            "Biography",
            "Crime",
            "Drama"
          ]
        }
      },
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "1",
        "_score" : 1.0313075,
        "_source" : {
          "title" : "The Godfather",
          "director" : "Francis Ford Coppola",
          "year" : 1972
```

برای مچ کردن روی دو فیلد مختلف هم از multi_match استفاده می کنیم. در اینجا کلمه Ford را روی هر دو فیلد عنوان و کارگردان مچ کرده و به عنوان وزن بیشتری (سه برابر) داده ایم. به همین جهت score برای نتیجه ی اول بیشتر از دومی است.



```
POST /movies/_search
{
  "query": {
    "bool": {
      "should": [
        {
          "match": {
            "genres": "drama"
          }
        },
        {
          "match": {
            "title": {
              "query":"kill",
              "boost": 3
            }
          }
        }
      ]
    }
  }
}
```

```
  "took" : 0,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 5,
      "relation" : "eq"
    },
    "max_score" : 4.035888,
    "hits" : [
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "3",
        "_score" : 4.035888,
        "_source" : {
          "title" : "To Kill a Mockingbird",
          "director" : "Robert Mulligan",
          "year" : 1962,
          "genres" : [
            "Crime",
            "Drama",
            "Mystery"
          ]
        }
      },
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "5",
        "_score" : 3.8003933,
        "_source" : {
          "title" : "Kill Bill: Vol. 1",
          "director" : "Quentin Tarantino",
          "year" : 2003,
          "genres" : [
```

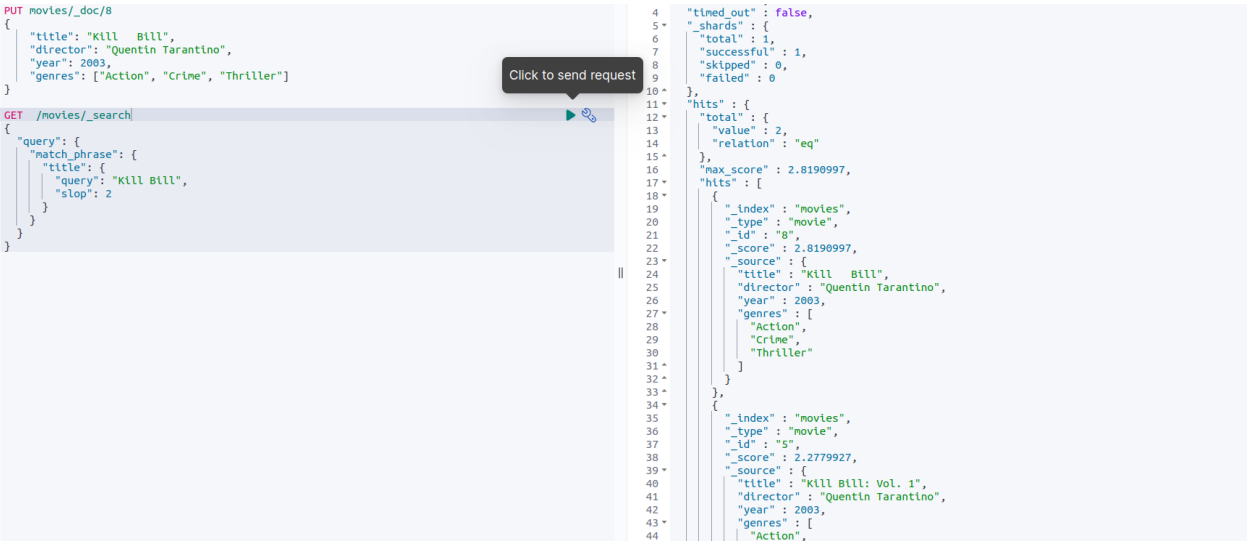
در کوئری فوق، چند تا مچینگ را با هم ترکیب کرده ایم. عبارت should به معنی Or شدن حاصل دو مچینگ است که مچ شدن عنوان ضریب ۳ دارد. ۵ تا فیلم به عنوان نتیجه آمده اند که اولی به علت داشتن هر دو مخصوصا مچ در عنوان، بیشترین امتیاز را دریافت کرده است.

درون یک عبارت bool هر کدام از عبارت های must_not ،must و should تنها یک بار باید ظاهر شود. (جمله مهم دستور کار)

برای سرچ غیر متنی مثلا فیلتر تصویر بالا روی سال ساخت، از term استفاده می کنیم. برای گشتن به دنبال مقدار دقیق برای فیلدهای متنی نیز می توان از term استفاده کرد. یعنی در حالاتی که مچ شدن به قسمتی از رشته کافی نیست و کل رشته باید برابری داشته باشد.

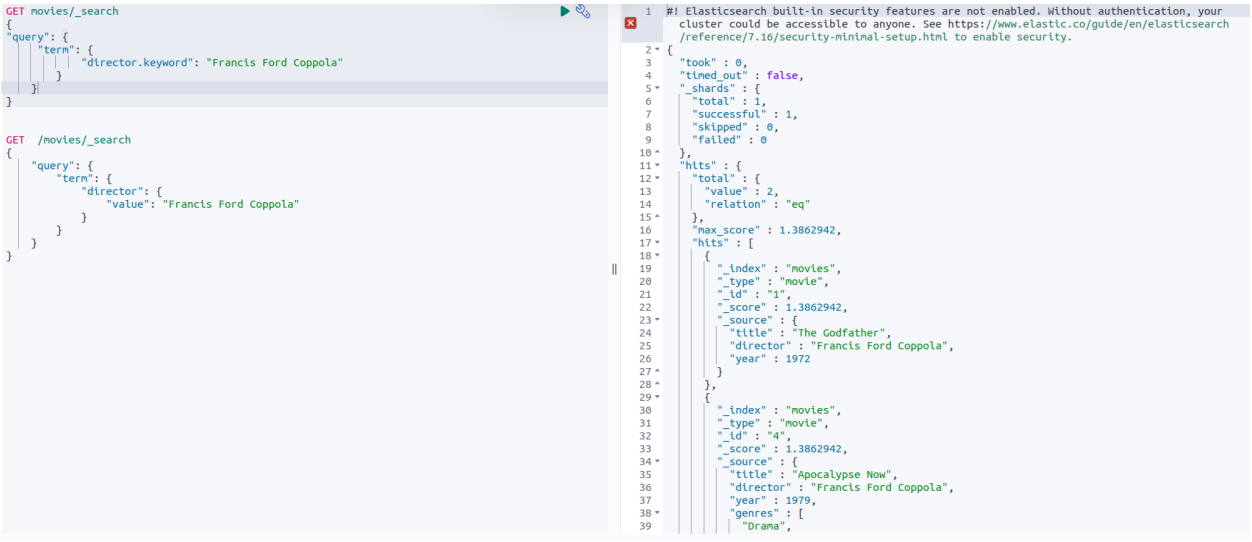
در کوئری فوق، فیلترینگ به همراه مچینگ اعمال کرده ایم به این صورت که داشتن ژانر درام در ژانر ها و سال ساخت بعد از ۱۹۶۰ را به عنوان فیلتر ها در نظر گرفته ایم و سپس مچ شدن عنوان با کلمه Kill را اعمال می کنیم. در مجموع کوئری ها به دو بخش متنی (matching) و غیر متنی (filtering) تقسیم می شوند که می توان این دو را ترکیب هم کرد یا حتی اصلا یکی را اعمال نکرد. مثلا با match_all می توانیم سرچ متنی انجام ندهیم.

در این قسمت، برای داشتن حداقل دو نتیجه، فیلم Kill Your Darlings به رکوردها اضافه شد و در نتیجه نیز دیده می شود.



در کوئری فوق، نیاز به سرچ متنی دقیق را برطرف کرده ایم که با استفاده از match_phrase گفته ایم که مقادیری انتخاب شوند که عنوانشان دقیقا عبارت Kill Bill را دارد ولی می توانند نهایتا ۳ تا space نیز داشته باشند. (slop=2)

برای این مرحله هم یک رکورد با عنوان Kill Bill با ۳ تا Space اضافه شده که میبینیم در نتایج نیز نمایش داده می شود.

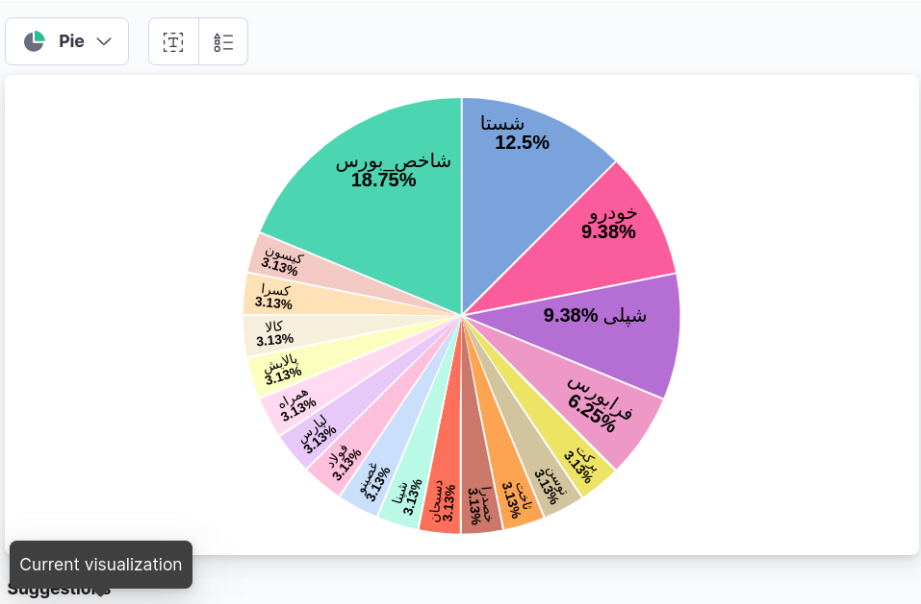


مشکلی که وجود دارد این است که با اجرای کوئری دوم در تصویر بالا نتیجه ای نمایش داده نمی شود. آن هم به این دلیل است که الستیک به طور خودکار فیلد های متنی را پردازش می کند و به کلمات و بخش های تشکیل دهنده شان تقسیمشان می کند. به همین علت، باید مشخص کنیم که دقیقا به دنبال همین عبارت هستیم و این کار را با جایگزین کردن کوئری دوم با کوئری اول انجام می دهیم. یعنی می گوییم که مقدار Francis Ford Coppola یک نوع keyword در جستجوی ماست و دقیقا به دنبال آن عبارت هستیم. نتیجه، دیدن هر دو فیلم به کارگردانی این فرد در حاصل کوئری است.

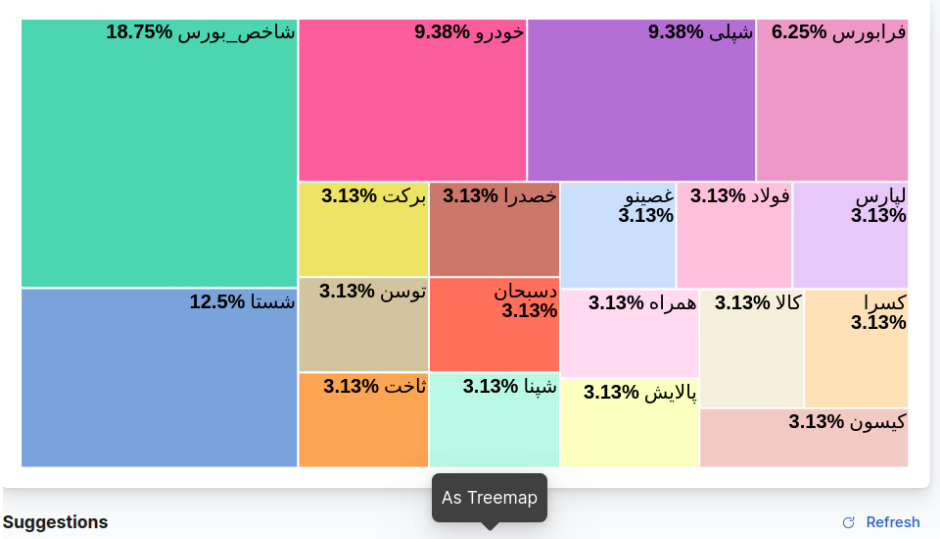
بخش دوم) جمع آوری توییت ها:

در این بخش، فایل توییت های قرار داده شده در گروه را وارد الستیک سرچ کردیم. بدین وسیله دیتای ۵۰۰ توییت وارد شد. سپس با ساختن Index pattern برای داده ورودی و ساختن یک dashboard برای visualization، داده ها و هشتگ های موجود در توییت ها و نحوه توزیع آن ها به دو روش Tree Chart (تقریبا مشابه تابلو های بورسی) و Pie Chart بصری سازی شد.

- Pie Chart:



- Tree Chart:



- احتمالا به علت ورژن متفاوت کیبانا، برای من امکان رسم Tag Cloud نبود و کیبانا آپشن رسم آن را در اختیار من نمی گذاشت. به همین علت آن را با Tree Chart جایگزین کردم.
- متاسفانه به علت نوشتن گزارش در Google Docs امکان رعایت چارچوب گزارش وجود نداشت. از این بابت عذرخواهی می کنم.