6. 전문 검색을 넘어서

이혜림

집계(Aggregation)

- 페이셋이 지원하지 않거나 페이셋으로 구현하기에 복잡한 기능을 더 쉽게 사용할 수 있게 개발된 모듈

- 크게 버킷 aggregation, 메트릭 aggregation 구분

  ➢ 메트릭 Aggregation : min, max, sum, avg, stats와 extended stats 집계 등이 있음

Aggregation 사용 형식

```
"aggregations" : {
    "<aggregation_name>" : {
        "<aggregation_type>" : {
            <aggregation_body>
        }
        [,"aggregations" : { [<sub_aggregation>]+ } ]?
    }
    [,"<aggregation_name_2>" : { ... } ]*
}
```

숫자집계 (Metric Aggregation)

1. min, max, sum, avg 집계

   ➢ 명시한 필드에 대해 최소값, 최대값, 모든 값의 합, 평균값을 반환

   ➢ 스크립트를 통해 입력 값을 생성할 수 있음

   - 예: year 값에서 1000을 뺀 값을 집계에 사용

```
{
  "aggs": {
    "min_year": {
      "min": {
        "field": "year"
      }
    }
  }
}
```

```
"min_year": {
  "value": 1886
}
```

```
  "min": {
    "script": "doc['year'].value - 1000"
}
```

```
 "min": {
   "field": "year",
   "script": "_value - 1000"
  }
```

```
 "min": {
   "field": "year",
   "script": "_value - mod",
   "params": {
     "mod" : 1000
   }
  }
```

숫자집계 (Metric Aggregation)

2. stats와 extended_stats 집계
   - ➢ stats: 해당 field에 대한 min, max, avg, sum 반환
   - ➢ Extended_stats: 해당 field에 대한 stats 기본정보 및 제곱의 합, 분산, 표준 편차 계산 값 반환

     (sum_of_squares, variance, std_deviation)

```
{
  "aggs": {
    "stats_year": {
      "stats": {
        "field": "year"
      }
    }
  }
}
```

```
"stats_year": {
    "count": 4,
    "min": 1886,
    "max": 1961,
    "avg": 1928,
    "sum": 7712
}
```

```
{
  "aggs": {
    "stats_year": {
      "extended_stats": {
        "field": "year"
      }
    }
  }
}
```

```
"stats_year": {
    "count": 4,
    "min": 1886,
    "max": 1961,
    "avg": 1928,
    "sum": 7712,
    "sum_of_squares": 14871654,
    "variance": 729.5,
    "std_deviation": 27.00925767213901
}
```

숫자집계 (Metric Aggregation)

3. value_count 집계
   - ➢ 해당 필드의 value 값을 토큰 기반으로 계산
   - ➢ 값의 형식을 잘 살펴 봐야 함

Library document의 character 필드 count

```
{
  "aggs": {
    "number_of_items": {
      "value_count": {
        "field": "characters"
      }
    }
  }
}
```

```
"number_of_items": {
  "value": 31
}
```

Library document의 일부


```
{ "title": "All Quiet on the Western Front","otitle": "Im Westen
  nichts Neues","author": "Erich Maria Remarque","year":
  1929,"characters": ["Paul Bäumer", "Albert Kropp", "Haie
  Westhus", "Fredrich Müller", "Stanislaus Katczinsky",
  "Tjaden"],"tags": ["novel"],"copies": 1, "available": true,
  "section" : 3}
```


다큐먼트의 characters 필드에서 space로 구분한 토큰을 계산함

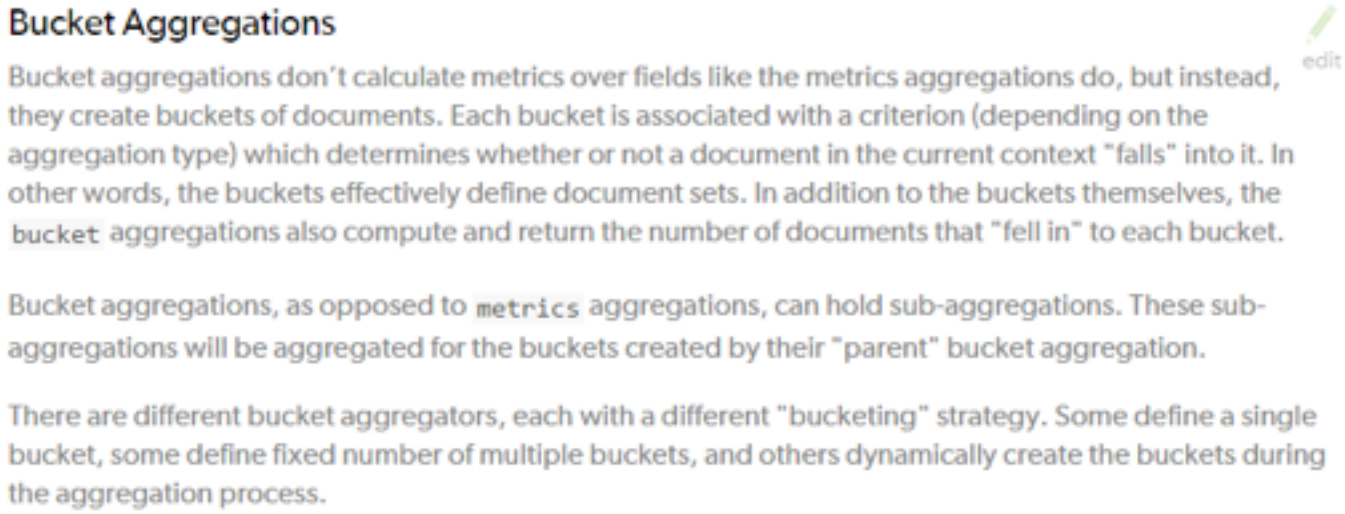
따라서 “Erich Maria” -> “Erich”, “Maria”로 계산됨

위와 같은 형식에서 원하는 값을 얻기 위해서는 not_analyzed된

Character 필드를 사용해 집계해야 함

버킷 집계 (Bucketing aggregation)

- 조건에 맞는 각 집합의 모음을 이용해 버킷을 생성함
- 여러 부분 집합을 반환
- Terms aggregation, Global aggregation, Range aggregation, nested aggregation 등이 있음



Bucket Aggregations

edit

Bucket aggregations don't calculate metrics over fields like the metrics aggregations do, but instead, they create buckets of documents. Each bucket is associated with a criterion (depending on the aggregation type) which determines whether or not a document in the current context "falls" into it. In other words, the buckets effectively define document sets. In addition to the buckets themselves, the `bucket` aggregations also compute and return the number of documents that "fell in" to each bucket.

Bucket aggregations, as opposed to `metrics` aggregations, can hold sub-aggregations. These sub-aggregations will be aggregated for the buckets created by their "parent" bucket aggregation.

There are different bucket aggregators, each with a different "bucketing" strategy. Some define a single bucket, some define fixed number of multiple buckets, and others dynamically create the buckets during the aggregation process.

버킷 집계 (Bucketing aggregation)

1. terms 집계

   ➢ filed에 존재하는 각 키워드에 대해 단일 버킷을 반환

   ➢ 질의 예

   - 가장 대출 가능한 책은 몇 권인가?
   - 가장 사본이 많은 책은 몇 권인가?

```
{
  "aggs": {
    "availability": {
      "terms": {
        "field": "copies"
      }
    }
  }
}
```

```
"availability": {
  "buckets": [
    {
      "key": 0,
      "doc_count": 2
    },
      "doc_count": 1
    },
    {
      "key": 6,
      "doc_count": 1
    }
  ]
}
```

질의

반환 값

버킷 집계 (Bucketing aggregation)

2. Range 집계

- 설정한 값의 범위 별로 버킷을 생성할 수 있음
- 버킷마다 레이블(label)을 생성할 수 있음 (“keyed”: true)
    - 레이블 임의 지정 형식: {“key” : “Before 18th century”, “to”: 1799},

```
{
  "aggs": {
    "years": {
      "range": {
        "field": "year",
        "keyed": true,
        "ranges": [
          { "to" : 1850 },
          { "from": 1851, "to": 1900 },
          { "from": 1901, "to": 1950 },
          { "from": 1951, "to": 2000 },
          { "from": 2001 }
        ]
      }
    }
  }
}
```

질의

```
"years": {
  "buckets": {
    "*-1850.0": {
      "to": 1850,
      "doc_count": 0
    },
    "1851.0-1900.0": {
      "from": 1851,
      "to": 1900,
      "doc_count": 1
    },
    "1901.0-1950.0": {
      "from": 1901,
      "to": 1950,
      "doc_count": 2
    },
    "1951.0-2000.0": {
      "from": 1951,
      "to": 2000,
      "doc_count": 1
    },
    "2001.0-*": {
      "from": 2001,
      "doc_count": 0
    }
  }
}
```

반환 값

버킷 집계 (Bucketing aggregation)

3. date_range 집계

- 날짜 타입을 사용하는 필드를 위해 설계
- “format” 속성을 통해 표시될 날짜 형태도 정의 가능
- /M 수식을 통해 월로 반올림한 달 단위로 계산 가능

```
"aggs": {
  "years": {
    "date_range": {
      "field": "published",
      "ranges": [
        { "to" : "2009/12/31" },
        { "from": "2010/01/01", "to": "2010/12/31" },
        { "from": "2011/01/01" }
        ]
    }
  }
}
```

```
{
  "from": 1262304000000,
  "from_as_string": "2010/01/01 00:00:00",
  "to": 1293753600000,
  "to_as_string": "2010/12/31 00:00:00",
  "doc_count": 2
},
```

```
"date_range": {
    "field": "date",
    "format": "MM-yyy",
    "ranges": [
        { "to": "now-10M/M" }, ❶
        { "from": "now-10M/M" } ❷
    ]
}
```

```
"range": {
    "buckets": [
        {
            "to": 1.3437792E+12,
            "to_as_string": "08-2012",
            "doc_count": 7
        },
        {
            "from": 1.3437792E+12,
            "from_as_string": "08-2012",
            "doc_count": 2
        }
```

버킷 집계 (Bucketing aggregation)

3. Nested 집계

- 중첩된 다큐먼트에 대한 aggregation

```
{
    ...

    "product" : {
        "properties" : {
            "resellers" : { 1
                "type" : "nested",
                "properties" : {
                    "name" : { "type" : "string" },
                    "price" : { "type" : "double" }
                }
            }
        }
    }
}
```

```
{
    "query" : {
        "match" : { "name" : "led tv" }
    },
    "aggs" : {
        "resellers" : {
            "nested" : {
                "path" : "resellers"
            },
            "aggs" : {
                "min_price" : { "min" : { "field" : "resellers.price" } }
            }
        }
    }
```

```
{
    "aggregations": {
        "resellers": {
            "min_price": {
                "value" : 350
            }
        }
    }
}
```

버킷 집계 (Bucketing aggregation)

4. geo_distance 집계 (지도, 공간 검색 관련 집계)

- origin으로 부터의 거리를 기준으로 버킷 생성
- unit: 거리 단위 설정(km, yd, m ,cm, mi등 설정 가능. 기본값은 km)
- distance_type: elasticsearch가 거리를 계산하는 방식 설정.

(가장 빠르지만 덜 정확한 순에서 가장 느리지만 정확한 순. Plane, sloop_arc(기본값), arc)

```
{
  "aggs": {
    "neighborhood": {
      "geo_distance": {
        "field": "location",
        "origin": [-0.1275, 51.507222],
        "ranges": [
          { "to": 1200 },
          { "from": 1201 }
        ]
      }
    }
  }
}
```



```
"neighborhood": {
  "buckets": [
    {
      "key": "*-1200.0",
      "from": 0,
      "to": 1200,
      "doc_count": 1
    },
    {
      "key": "1201.0-*",
      "from": 1201,
      "doc_count": 4
    }
  ]
}
```

Origin(여기서는 런던)을 기준으로 1200Km 이내,

1200Km 밖의 도시로 나눈다.

버킷 집계 (Bucketing aggregation)

4. geohash_grid 집계 (지도, 공간 검색 관련 집계)

- 모든 위치를 적절한 셀에 대입
- 셀의 크기가 작을 수록 정확도가 높아지나 메모리를 사용률도 높아짐

```
{
  "aggs": {
    "neighborhood": {
      "geohash_grid": {
        "field": "location",
        "precision": 5
      }
    }
  }
}
```

https://www.devmynd.com/blog/2014-2-geohash-grid-aggregation-with-elasticsearch/

Precision 값에 따른 셀 크기:
https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-bucket-geohashgrid-aggregation.html

버킷 집계 (Bucketing aggregation)

1. 집계 중첩

   ➢ 복잡한 질의를 만드는데 필요한 기능

   ➢ 상위 집계 내부에 중첩된 또 다른 집계를 추가

   ➢ 상위 집계(부모 집계)의 결과를 가져와 하위에서 집계를 다시 한번 거침

   ➢ 이론상으로는 무한히 중첩 가능함

```
{
  "aggs": {
    "variations": {
    "nested": {
      "path": "variation"
    },
    "aggs": {
      "sizes": {
        "terms": {
          "field": "variation.size"
        }
      }
    }
  }
}
```

```
"variations": {
  "doc_count": 2,
  "sizes": {
    "buckets": [
      {
        "key": "XL",
        "doc_count": 1
      },
      {
        "key": "XXL",
        "doc_count": 1
      }
    ]
  }
}
```

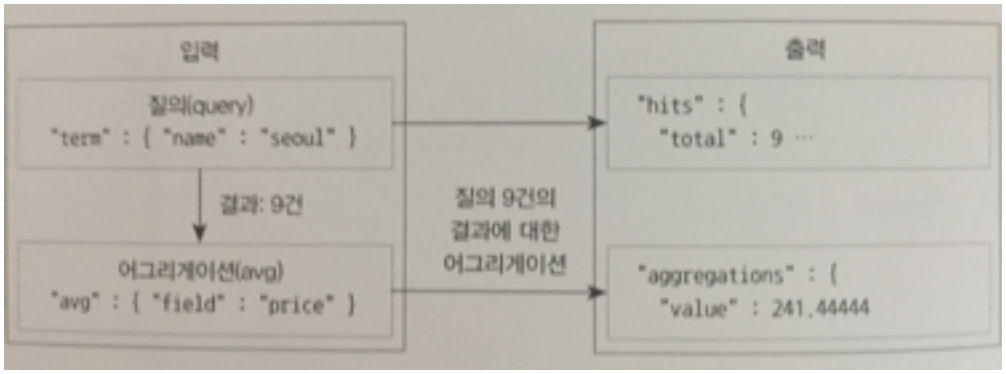
버킷 집계 (Bucketing aggregation)

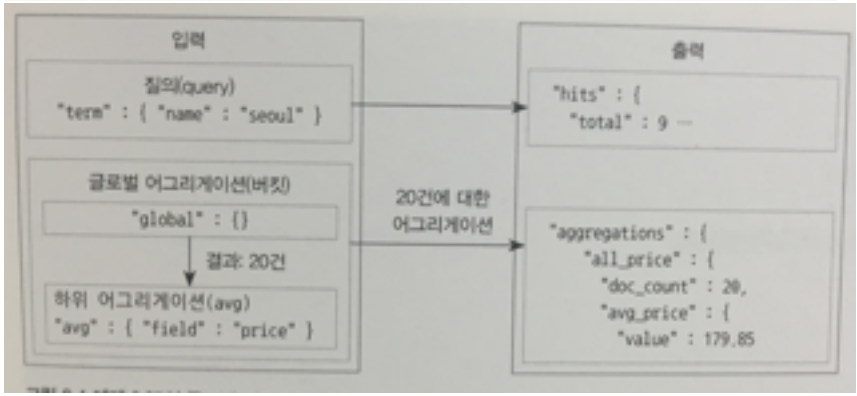
2. 전역과 부분집합

➢ 검색 범위의 인덱스나 타입에 해당하는 모든 도큐먼트를 하나의 버킷에 모두 담으며 질의에 영향을 받지 않음

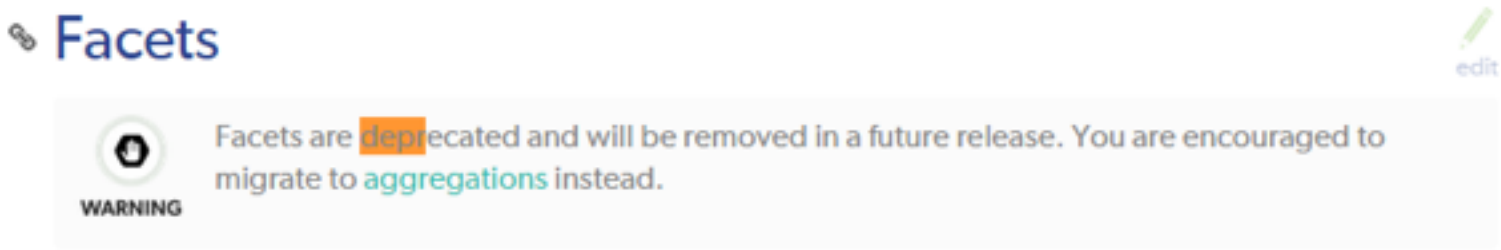
```
curl 'localhost:9200/hotels/_search?pretty' -d '
{
  "query" : {
    "term" : { "name" : "seoul" }
  },
  "aggs" : {
    "avg_price" : {
      "avg" : { "field" : "price" }
    }
  }
}'
```

```
curl 'localhost:9200/hotels/_search?pretty' -d '
{
  "query" : {
    "term" : { "name" : "seoul" }
  },
  "aggs" : {
    "all_price" : {
      "global" : {},
      "aggs" : {
        "avg_price" : {
          "avg" : { "field" : "price" }
        }
      }
    }
```





Facets


Facets
edit
WARNING
Facets are deprecated and will be removed in a future release. You are encouraged to migrate to aggregations instead.

Facets

Facets에서 Aggregation으로의 migration

https://www.elastic.co/guide/en/elasticsearch/reference/current/search-facets-migrating-to-aggs.html

```
{
    "facets" : {
        "tag_price_stats" : {
            "terms_stats" : {
                "key_field" : "tag",
                "value_field" : "price"
            }
        }
    }
}
```

```
{
    "facets" : {
        "facet1" : {
            "terms" : {
                "field" : "name"
            },
            "nested" : "obj1"
        }
    }
}
```

```
{
    "aggs" : {
        "tags" : {
            "terms" : {
                "field" : "tag"
            },
            "aggs" : {
                "price_stats" : {
                    "stats" : {
                        "field" : "price"
                    }
                }
            }
        }
    }
}
```

```
{
    "aggs" : {
        "agg1" : {
            "nested" : {
                "path" : "obj1"
            },
            "aggs" : {
                "agg1": {
                    "terms": {
                        "field" : "obj1.name"
                    }
                }
            }
        }
    }
}
```