SMBUD - PROJECT WORK
Delivery #3

Academic year 2021/2022
Professor: Brambilla Marco

Group number: 4

Group members:
Elisa Mariani - 10632876
Pietro Guglielmo Moroni - 10625198
Giacomo Palù - 10682209
Francesco Panebianco - 10632465
Silvio Scanzi - 10622285

Politecnico di Milano

TABLE OF CONTENTS

1.Introduction

The third assignment revolved around analyzing real-world data about covid-19 vaccine administration. In the assignment, the data was already provided and, with it, a predefined schema which had to be further analyzed and, if needed, changed through a mapping to better suit our needs for the queries.

1.1. Specifics and hypotheses

As it can be noticed in the later sections (mainly in the “Data schema” section), the data, as opposed to the previous assignments, has a “high-level view” of the pandemic, without focusing on the single individuals, but rather revolves around the whole population.
The granularity is, therefore, coarse and the data provide little to no details on the single individuals, besides some aggregation on personal data such as age group and sex.
This fact forced the analysis of the data, changing the focus from the individual to the entire population and, as you can see in the later sections, the queries that have been written are aimed at extracting useful statistics on the rate of vaccination of the whole country.
The data provided covers the time frame from 01-09-2021 to 22-12-2021 rather than the whole pandemic since a smaller amount of data was easier to transfer and handle.
Obviously, adding new data extending the time frame analyzed would result in more significant statistics, but the queries and the dashboards provided would be as effective as in a smaller time frame.

2.Data Schema

Field	Data Type	Description
index	Integer	Index of the record
area	String	Acronyms of the region of delivery
fornitore	String	Complete name of the supplier of the vaccine
data_somministrazione	Datetime	Administration date of the vaccines
fascia_anagrafica	String	Age group of the people administered with the vaccines
sesso_maschile	Integer	Number of vaccinations administered to males
sesso_femminile	Integer	Number of vaccinations administered to females
prima_dose	Integer	Number of people administered with the first dose
seconda_dose	Integer	Number of people administered with the second dose
pregressa_infezione	Integer	Number of people administered with a dose after they have been infected
dose_addizionale_booster	Integer	Number of people administered with an additional dose/recall
codice_NUTS1	String	European classification of territorial units: first level
codice_NUTS2	String	European classification of territorial units: second level
codice_regione_ISTAT	Integer	ISTAT code of a region
nome_regione	String	Name of the region (bilingual, when necessary)

As it can be seen from the previous table, the default mapping has been chosen for the scope of the project. In the next paragraph, it will be explained briefly.

Firstly, the index was left as an integer because Integers are the most efficient type for indexing, and converting them wouldn't have yielded any improvement.

For some fields, the choice of the mapping is self-explanatory: all the fields that represent data about the number of doses administrated to males and females and the fields that represent data about the distribution of the doses between first administration, second, and booster are obviously Integer. The upper mentioned fields are: "sesso_maschile", "sesso_femminile", "prima_dose", "seconda_dose", "dose_addizionale_booster", "pregressa_infezione".
Since these fields can't assume fractional values, the most suitable type was clearly Integer.

The same holds true for "data_somministrazione", which represents the date on which the vaccines were administrated. Being a date, Datetime was the most suitable type.

For all the fields which can assume alphanumeric values, namely "area", "fornitore", "codice_NUTS1","codice_NUTS2","nome_regione", "fascia_anagrafica" the only possible choice is to treat them as Strings if we wanted to import them "as-is", without changing their values. We initially considered changing the mapping of "fascia_anagrafica" (and therefore their values) to Integer, but, as we later noticed, the queries that we wanted to carry out on the provided data could easily be "translated" into ones that could match the field as a String. For example, as it can be seen in the "Queries" section, the second query is:
"*Extract the number of doses of AstraZeneca given to people under the age of 30 in each NUTS area*".
Instead of mapping each age group with the lower bound (e.g. 20-29 mapped to 20) and adding a filter such as "filter on fascia_anagrafica lower than 40", we decided to fully exploit the instruments of elastic search, with the should clause or in other ways, and, as it can be seen in the later sections, we managed to write useful queries based on the age groups without having to change the mapping.

2.1 Geographic data

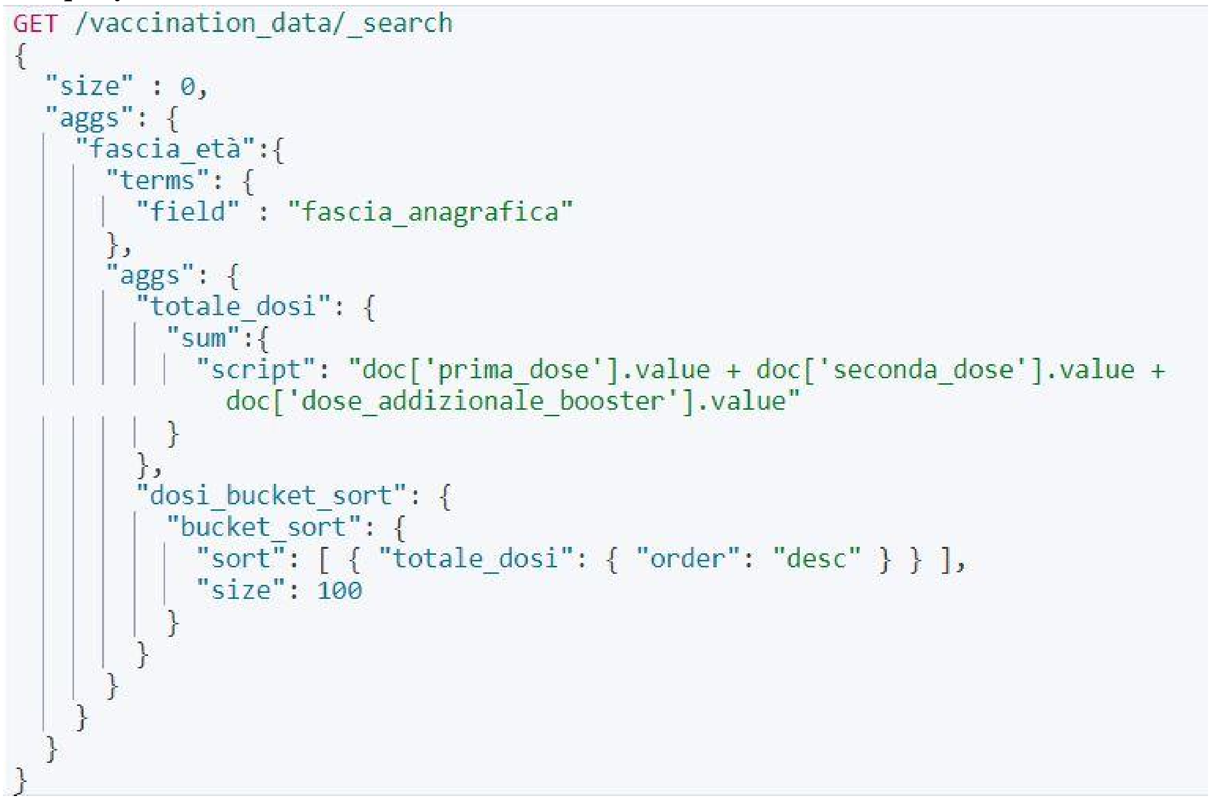
For use in the dashboard we have also used a geojson dataset containing the borders of each and every Italian region (view the *Dataset/regioni_ita.geojson* file).
Every document in this dataset has a `reg_istat_code_num` that we joined with the vaccination data's `codice_regione_ISTAT` field.
We used this source instead of downloading the *Elastic Map Service* because the latter wouldn't allow joining more than 10,000 elements and therefore being able to visualize the whole map.

3. Queries

Query #1. Extract the number of doses of vaccine given to each age group

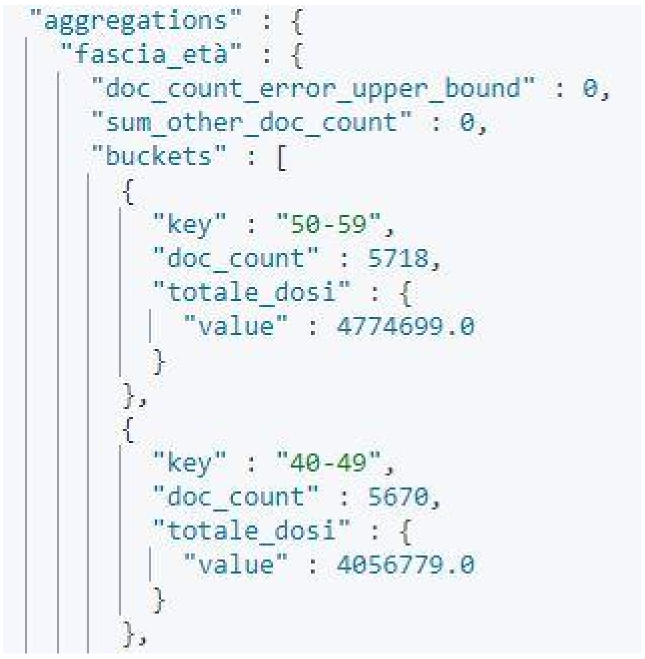
---

The query:



```
GET /vaccination_data/_search
{
  "size" : 0,
  "aggs": {
    "fascia_età":{
      "terms": {
        "field" : "fascia_anagrafica"
      },
      "aggs": {
        "totale_dosi": {
          "sum":{
            "script": "doc['prima_dose'].value + doc['seconda_dose'].value +
              doc['dose_addizionale_booster'].value"
          }
        },
        "dosi_bucket_sort": {
          "bucket_sort": {
            "sort": [ { "totale_dosi": { "order": "desc" } } ],
            "size": 100
          }
        }
      }
    }
  }
}
```

The outcome:



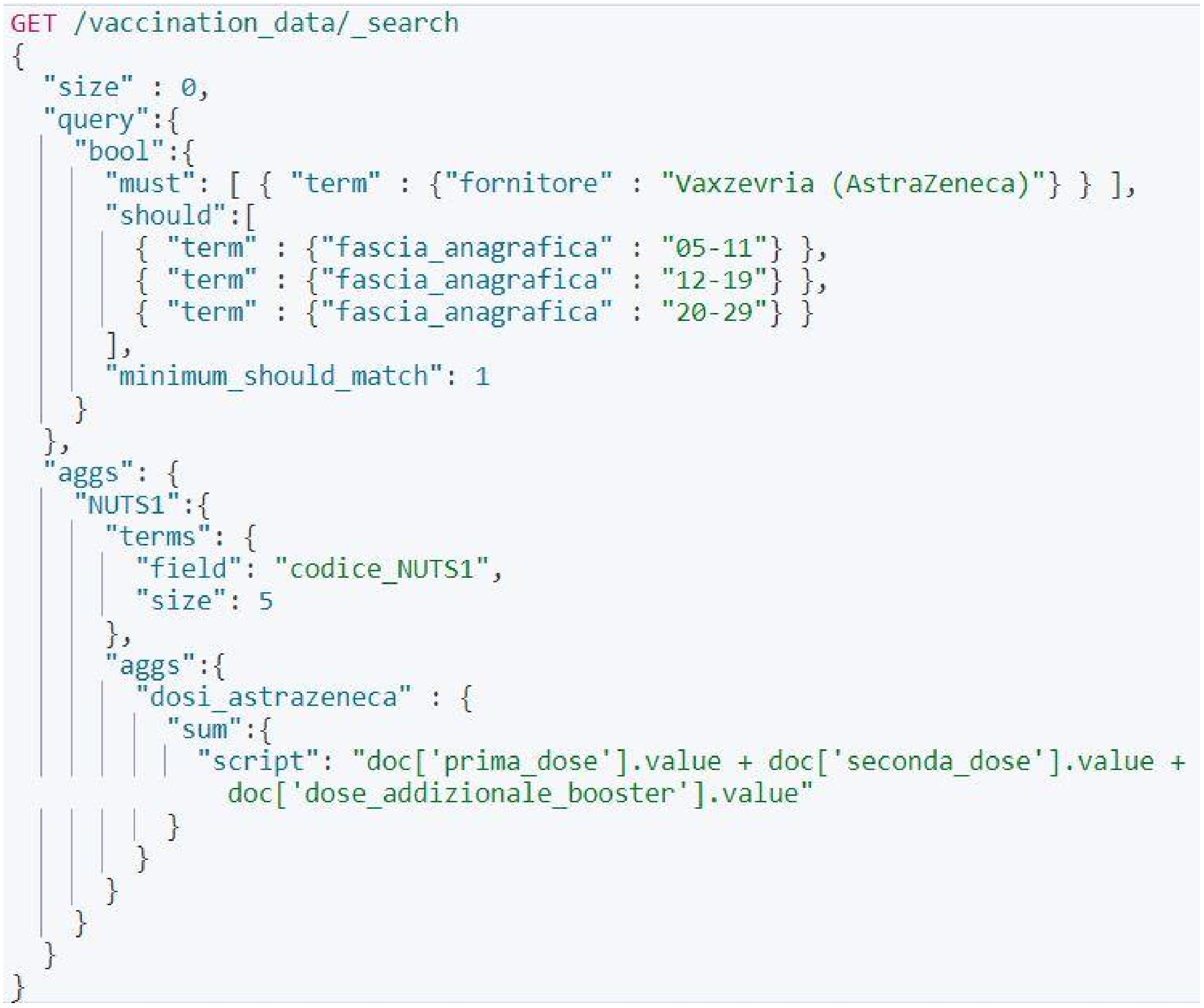
```
 "aggregations" : {
   "fascia_età" : {
     "doc_count_error_upper_bound" : 0,
     "sum_other_doc_count" : 0,
     "buckets" : [
       {
         "key" : "50-59",
         "doc_count" : 5718,
         "totale_dosi" : {
           "value" : 4774699.0
         }
       },
       {
         "key" : "40-49",
         "doc_count" : 5670,
         "totale_dosi" : {
           "value" : 4056779.0
         }
       },
```

The outcome has been reduced to only 2 buckets (age group 50-59, 40-49) just to demonstrate the result of the query without flooding the report with repeated data.
The query is aimed to extract useful statistics about the vaccination rate in the different age groups. Integrating the data of the vaccine with some simple data about age distribution in the country, we can compute, for each age group, the percentile of people who got vaccinated.
For example, the age distribution in Italy has been calculated (based on ISTAT data) here:
https://www.tuttitalia.it/statistiche/popolazione-eta-sesso-stato-civile-2021/

The query uses a simple script to compute the sum of all the doses, but we could get other useful statistics such as how many people in each group got the second dose or the booster dose, or other things by simply modifying the script.

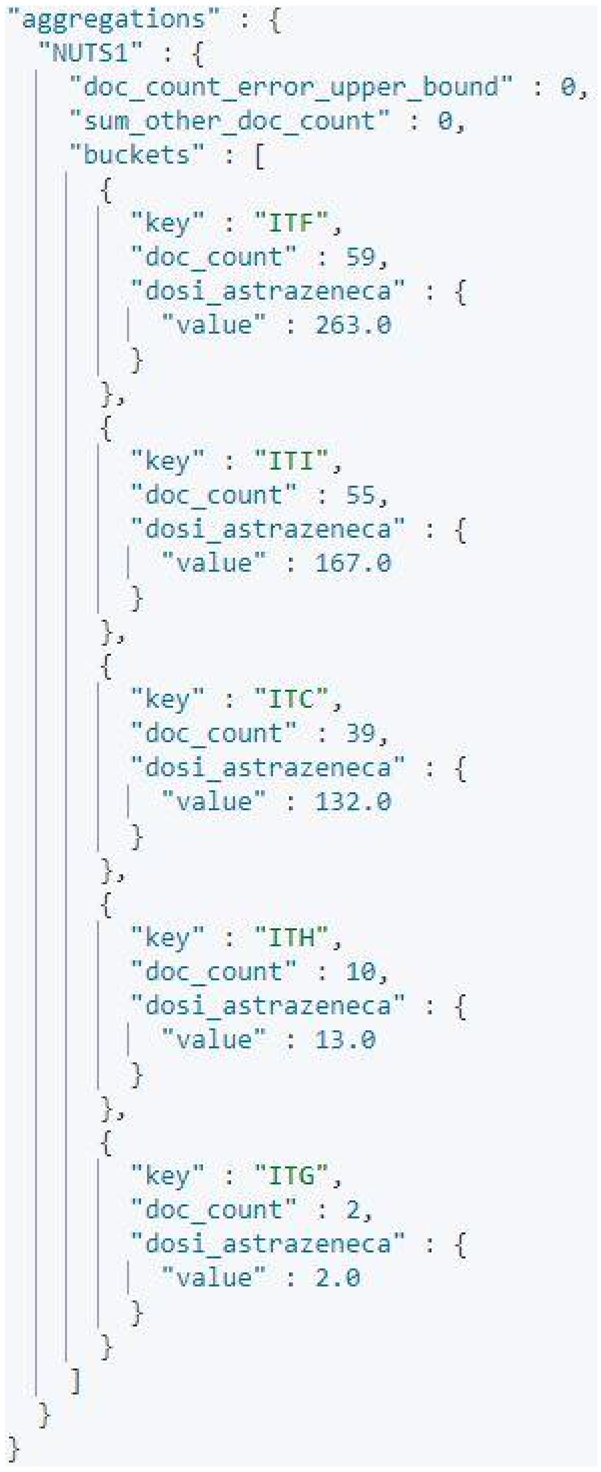
Query #2. Extract the number of doses of AstraZeneca given to people under the age of 30 in each NUTS area

The query:



```
GET /vaccination_data/_search
{
  "size" : 0,
  "query":{
    "bool":{
      "must": [ { "term" : {"fornitore" : "Vaxzevria (AstraZeneca)"} } ],
      "should":[
        { "term" : {"fascia_anagrafica" : "05-11"} },
        { "term" : {"fascia_anagrafica" : "12-19"} },
        { "term" : {"fascia_anagrafica" : "20-29"} }
      ],
      "minimum_should_match": 1
    }
  },
  "aggs": {
    "NUTS1":{
      "terms": {
        "field": "codice_NUTS1",
        "size": 5
      },
      "aggs":{
        "dosi_astrazeneca" : {
          "sum":{
            "script": "doc['prima_dose'].value + doc['seconda_dose'].value +
              doc['dose_addizionale_booster'].value"
          }
        }
      }
    }
  }
}
```

The outcome:

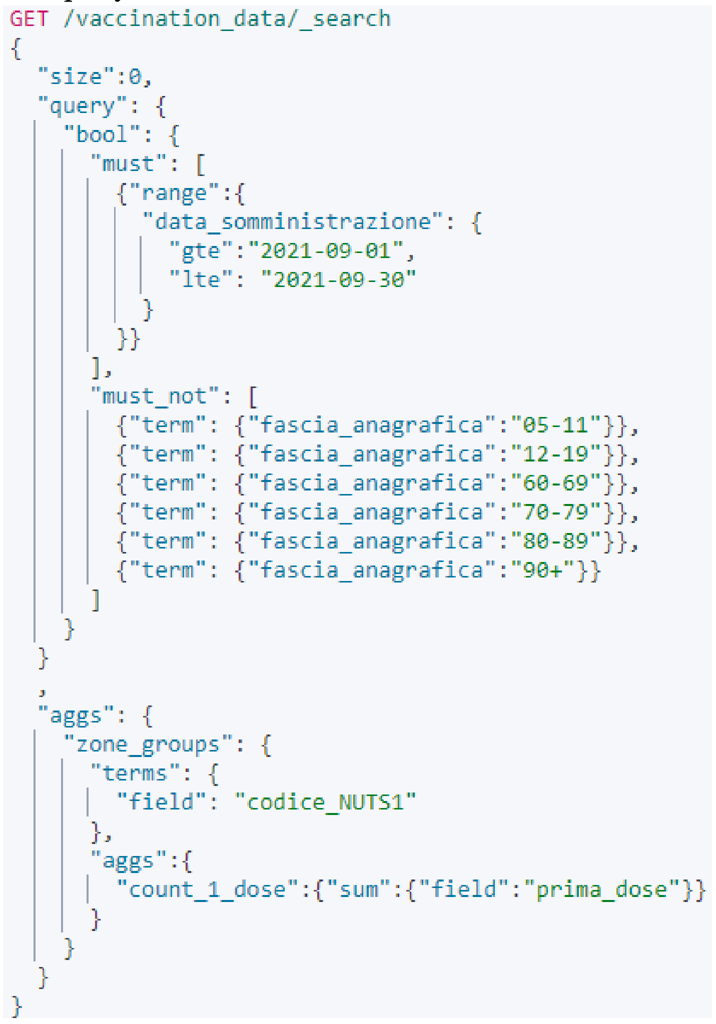


```
"aggregations" : {
  "NUTS1" : {
    "doc_count_error_upper_bound" : 0,
    "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "ITF",
        "doc_count" : 59,
        "dosi_astrazeneca" : {
          "value" : 263.0
        }
      },
      {
        "key" : "ITI",
        "doc_count" : 55,
        "dosi_astrazeneca" : {
          "value" : 167.0
        }
      },
      {
        "key" : "ITC",
        "doc_count" : 39,
        "dosi_astrazeneca" : {
          "value" : 132.0
        }
      },
      {
        "key" : "ITH",
        "doc_count" : 10,
        "dosi_astrazeneca" : {
          "value" : 13.0
        }
      },
      {
        "key" : "ITG",
        "doc_count" : 2,
        "dosi_astrazeneca" : {
          "value" : 2.0
        }
      }
    ]
  }
}
```

In the query, the should clause is used as an OR: by specifying in the should clause which values the field “fascia_anagrafica” is allowed to assume and adding the clause *"minimum_should_match": 1* we are ensuring that in every document found by the query, the value assumed by “fascia_anagrafica” is among the ones specified above.
The results are quite interesting: after the media case of AstraZeneca, we can see that in some areas, the administration of the vaccine has been practically reduced to zero (in particular the ITG region), while others kept vaccinating, probably with the last vials of the vaccine. Obviously, the results should then be weighted with the population count in each area to get some meaningful data.

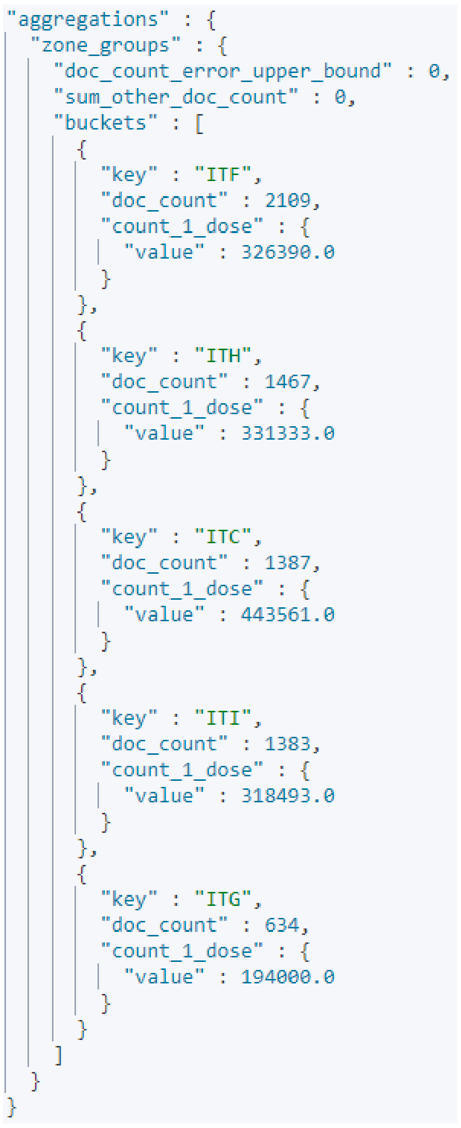
Query #3. Extract, for each geographical area (code NUTS1), the number of first doses of vaccine taken by the working population (20-60 years old) in the month of September 2021.

The query:



```
GET /vaccination_data/_search
{
  "size":0,
  "query": {
    "bool": {
      "must": [
        {"range":{
          "data_somministrazione": {
            "gte":"2021-09-01",
            "lte": "2021-09-30"
          }
        }}
      ],
      "must_not": [
        {"term": {"fascia_anagrafica":"05-11"}},
        {"term": {"fascia_anagrafica":"12-19"}},
        {"term": {"fascia_anagrafica":"60-69"}},
        {"term": {"fascia_anagrafica":"70-79"}},
        {"term": {"fascia_anagrafica":"80-89"}},
        {"term": {"fascia_anagrafica":"90+"}}
      ]
    }
  }
  ,
  "aggs": {
    "zone_groups": {
      "terms": {
        "field": "codice_NUTS1"
      },
      "aggs":{
        "count_1_dose":{"sum":{"field":"prima_dose"}}
      }
    }
  }
}
```

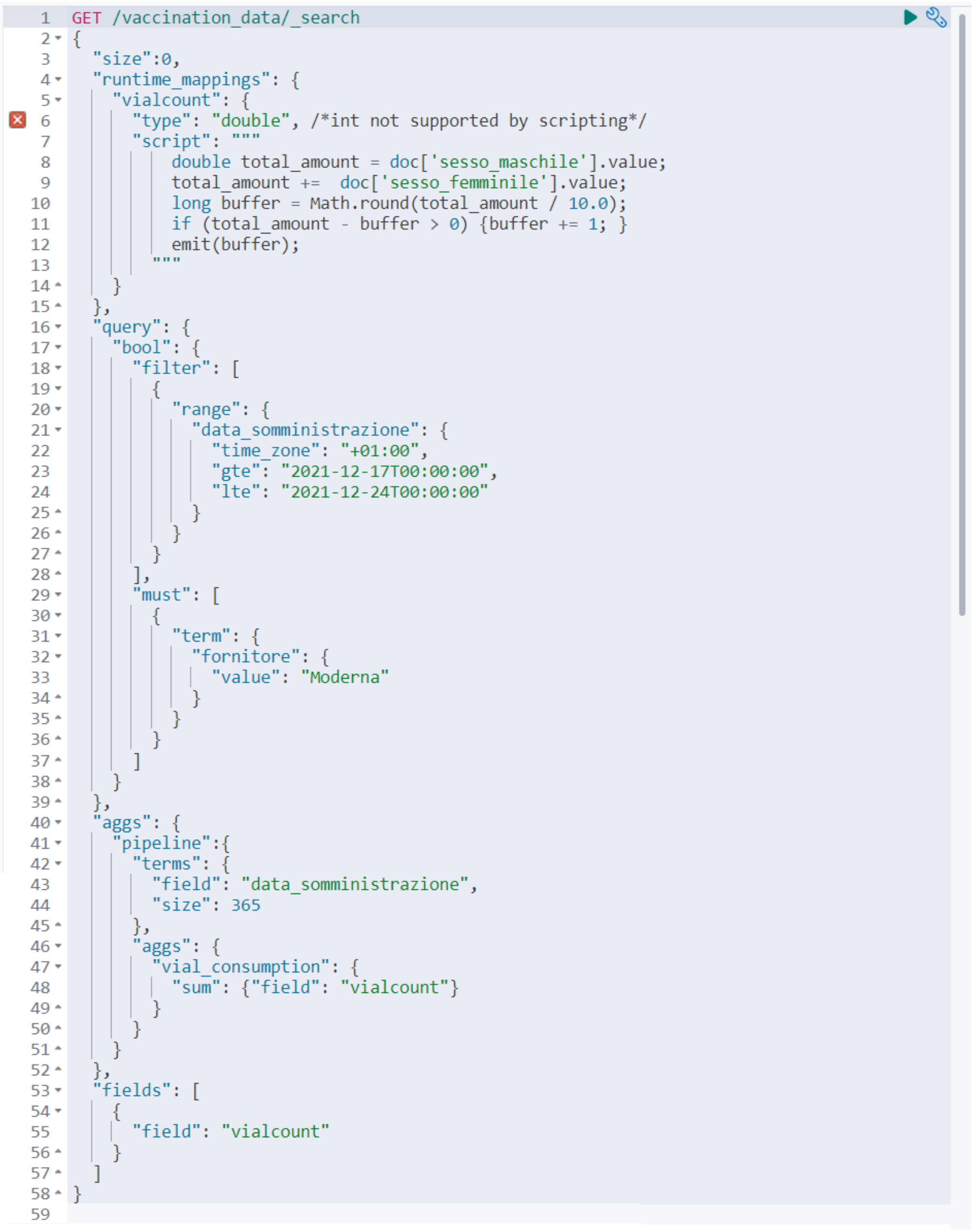
The outcome:



```
"aggregations" : {
  "zone_groups" : {
    "doc_count_error_upper_bound" : 0,
    "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "ITF",
        "doc_count" : 2109,
        "count_1_dose" : {
          "value" : 326390.0
        }
      },
      {
        "key" : "ITH",
        "doc_count" : 1467,
        "count_1_dose" : {
          "value" : 331333.0
        }
      },
      {
        "key" : "ITC",
        "doc_count" : 1387,
        "count_1_dose" : {
          "value" : 443561.0
        }
      },
      {
        "key" : "ITI",
        "doc_count" : 1383,
        "count_1_dose" : {
          "value" : 318493.0
        }
      },
      {
        "key" : "ITG",
        "doc_count" : 634,
        "count_1_dose" : {
          "value" : 194000.0
        }
      }
    ]
  }
}
```

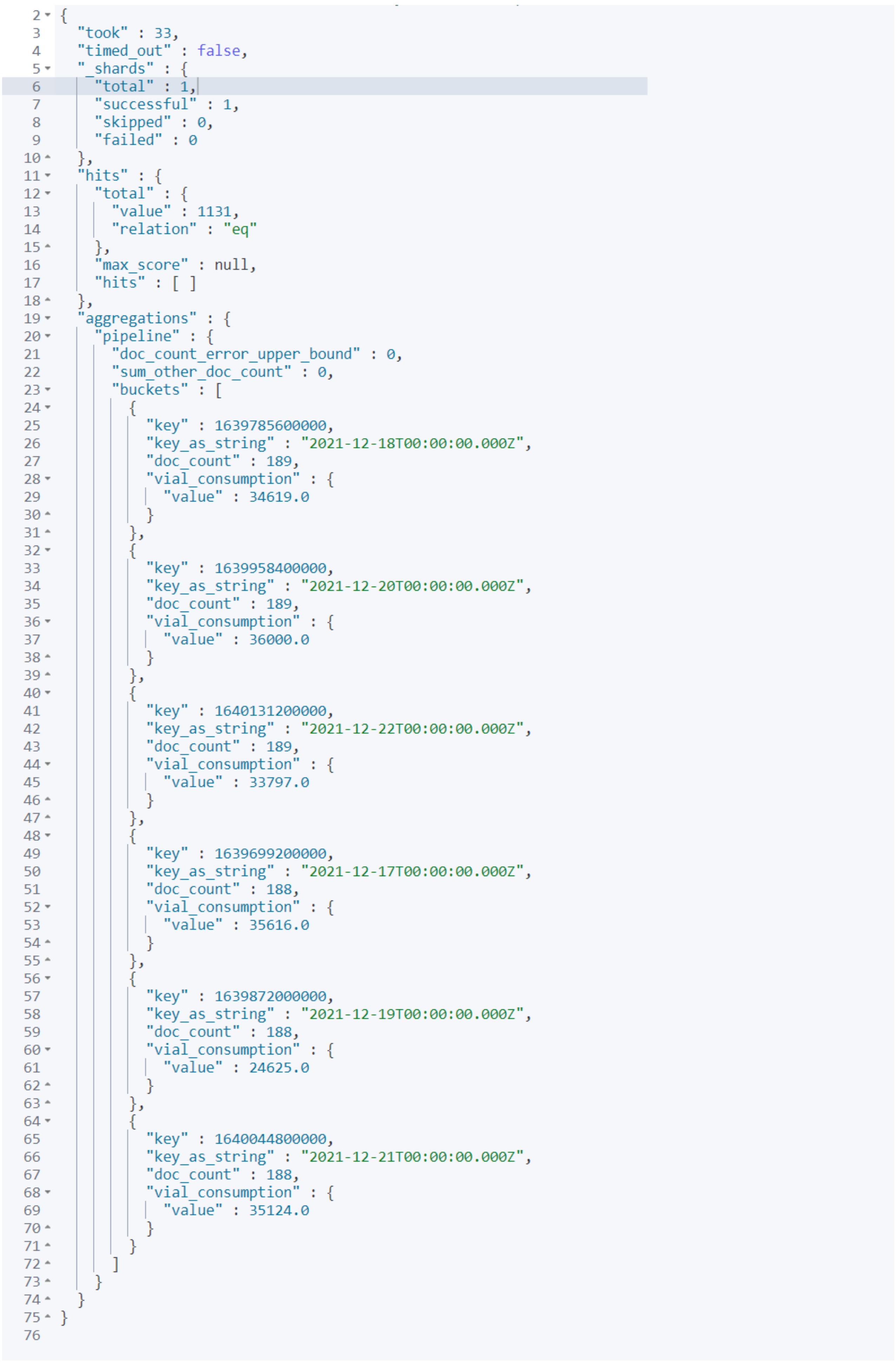
In this query, first, we impose the size equal to zero because we want to highlight the aggregated data, not single documents. Then, with the "query" parameter we want to filter only the documents related to September ("*must*" condition) and regarding the working population (we exclude non-working population sections in the "*must_not*" condition). Finally, we aggregate first by geographical area ("*codice_NUTS1*"), and then for each area, we sum the number of first doses.
Using this query, we could extract the same data also for other months (changing the "*range*" of the dates). This could be useful to make meaningful comparisons between the months, to highlight the effectiveness of the measures taken by the government to contain the pandemic.

Query #4. For each day in the week between December 17th and December 24th 2021, return the number of Moderna **vials** administered (10 doses per vial).



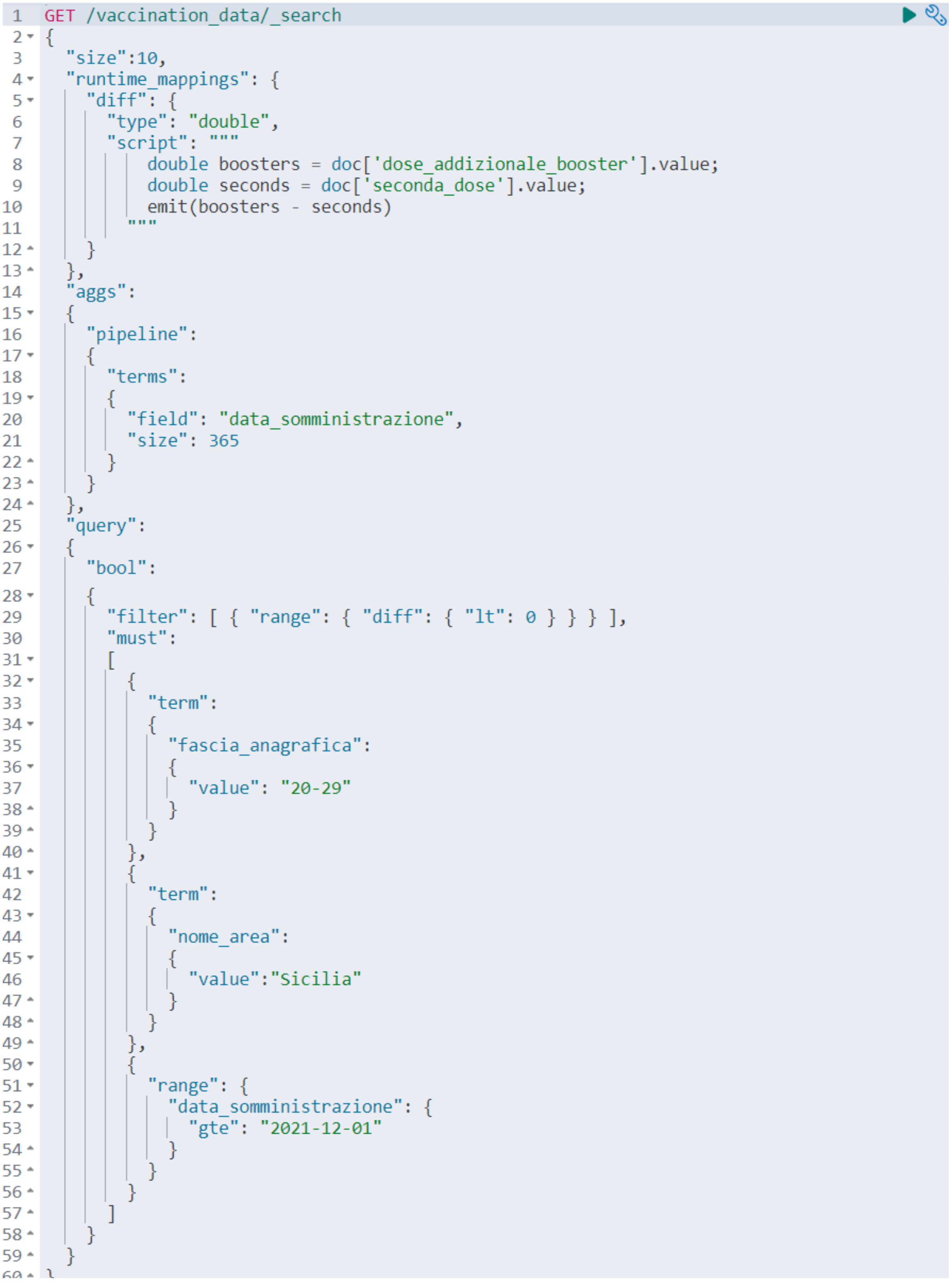
```
GET /vaccination_data/_search
{
  "size":0,
  "runtime_mappings": {
    "vialcount": {
      "type": "double", /*int not supported by scripting*/
      "script": """
        double total_amount = doc['sesso_maschile'].value;
        total_amount +=  doc['sesso_femminile'].value;
        long buffer = Math.round(total_amount / 10.0);
        if (total_amount - buffer > 0) {buffer += 1; }
        emit(buffer);
      """
    }
  },
  "query": {
    "bool": {
      "filter": [
        {
          "range": {
            "data_somministrazione": {
              "time_zone": "+01:00",
              "gte": "2021-12-17T00:00:00",
              "lte": "2021-12-24T00:00:00"
            }
          }
        }
      ],
      "must": [
        {
          "term": {
            "fornitore": {
              "value": "Moderna"
            }
          }
        }
      ]
    }
  },
  "aggs": {
    "pipeline":{
      "terms": {
        "field": "data_somministrazione",
        "size": 365
      },
      "aggs": {
        "vial_consumption": {
          "sum": {"field": "vialcount"}
        }
      }
    }
  },
  "fields": [
    {
      "field": "vialcount"
    }
  ]
}
```

The first part defines a scripted runtime mapping to get the total vial consumption as a field. The actual query filters the data within the range of the requested week and the requested vaccine (Moderna). Finally, a double aggregation is done (first on the date and then on the vial count, so as to get the total vial consumption for each date).



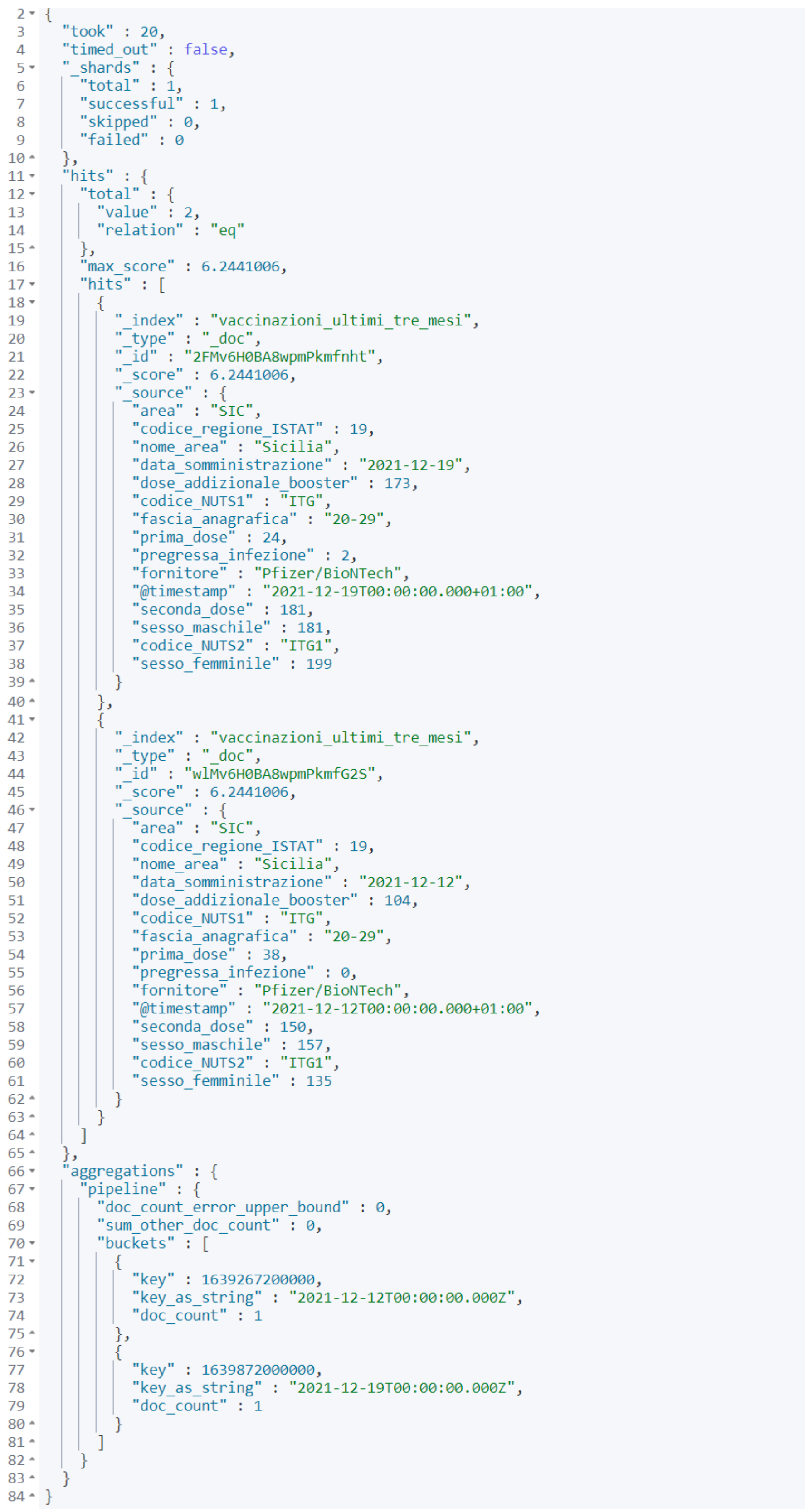
```
{
  "took" : 33,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 1131,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
  "aggregations" : {
    "pipeline" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : 1639785600000,
          "key_as_string" : "2021-12-18T00:00:00.000Z",
          "doc_count" : 189,
          "vial_consumption" : {
            "value" : 34619.0
          }
        },
        {
          "key" : 1639958400000,
          "key_as_string" : "2021-12-20T00:00:00.000Z",
          "doc_count" : 189,
          "vial_consumption" : {
            "value" : 36000.0
          }
        },
        {
          "key" : 1640131200000,
          "key_as_string" : "2021-12-22T00:00:00.000Z",
          "doc_count" : 189,
          "vial_consumption" : {
            "value" : 33797.0
          }
        },
        {
          "key" : 1639699200000,
          "key_as_string" : "2021-12-17T00:00:00.000Z",
          "doc_count" : 188,
          "vial_consumption" : {
            "value" : 35616.0
          }
        },
        {
          "key" : 1639872000000,
          "key_as_string" : "2021-12-19T00:00:00.000Z",
          "doc_count" : 188,
          "vial_consumption" : {
            "value" : 24625.0
          }
        },
        {
          "key" : 1640044800000,
          "key_as_string" : "2021-12-21T00:00:00.000Z",
          "doc_count" : 188,
          "vial_consumption" : {
            "value" : 35124.0
          }
        }
      ]
    }
  }
}
```

Query #5. Return the days in which second doses outnumbered boosters for the 20-29 age group in a given region (Lombardy has no hits, so Sicily was chosen instead).



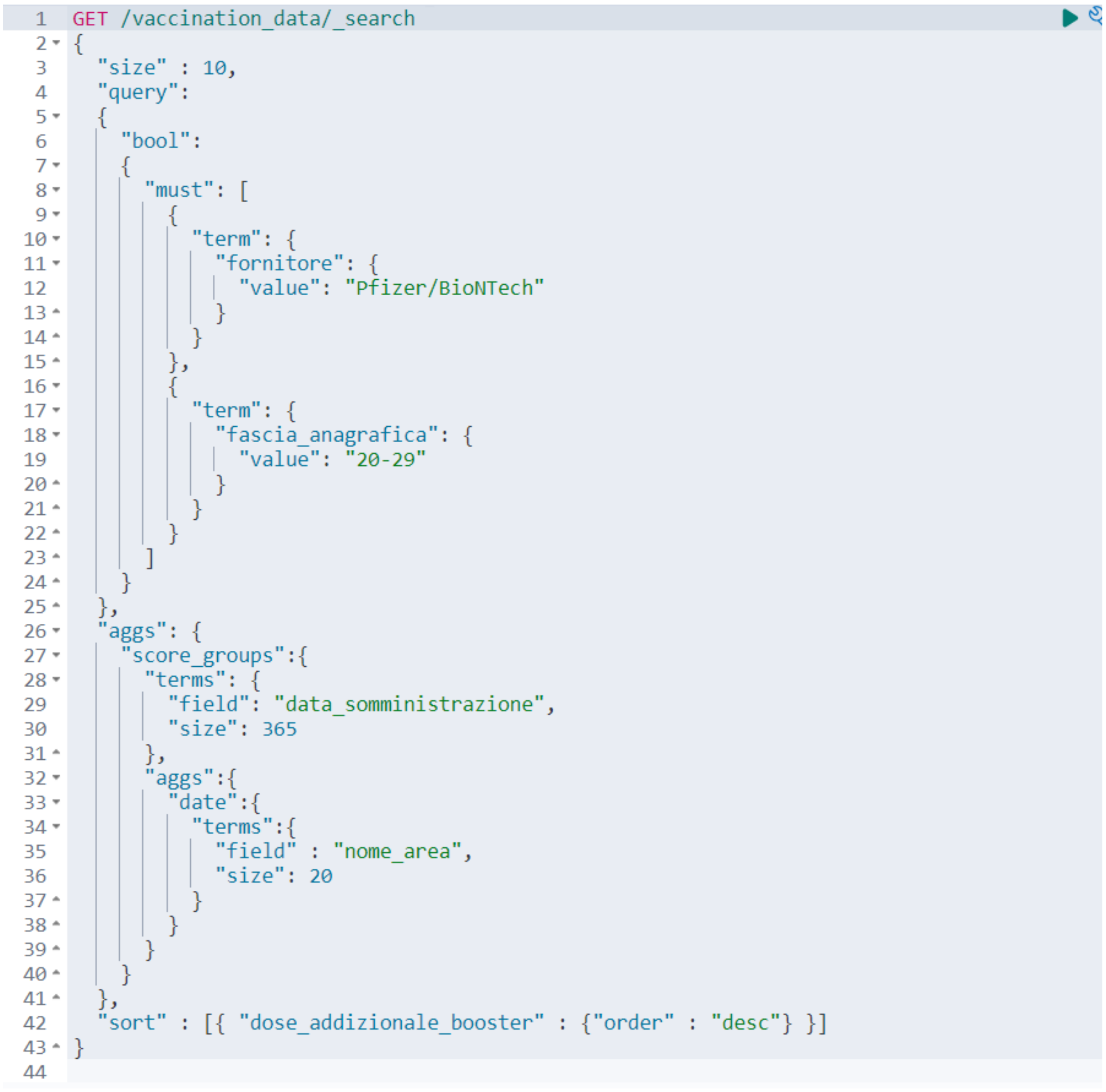
```
GET /vaccination_data/_search
{
  "size":10,
  "runtime_mappings": {
    "diff": {
      "type": "double",
      "script": """
        double boosters = doc['dose_addizionale_booster'].value;
        double seconds = doc['seconda_dose'].value;
        emit(boosters - seconds)
      """
    }
  },
  "aggs":
  {
    "pipeline":
    {
      "terms":
      {
        "field": "data_somministrazione",
        "size": 365
      }
    }
  },
  "query":
  {
    "bool":
    {
      "filter": [ { "range": { "diff": { "lt": 0 } } } ],
      "must":
      [
        {
          "term":
          {
            "fascia_anagrafica":
            {
              "value": "20-29"
            }
          }
        },
        {
          "term":
          {
            "nome_area":
            {
              "value":"Sicilia"
            }
          }
        },
        {
          "range": {
            "data_somministrazione": {
              "gte": "2021-12-01"
            }
          }
        }
      ]
    }
  }
}
```

In this query another script sets up a runtime mapping for the difference between booster doses and second doses (bool types not supported in scripting, otherwise a boolean return value would have been a wise choice). The query then aggregates on date. Finally, the *must* block picks the resulting documents with the given filters (diff < 0 (second doses greater than boosters), region restriction and age group.



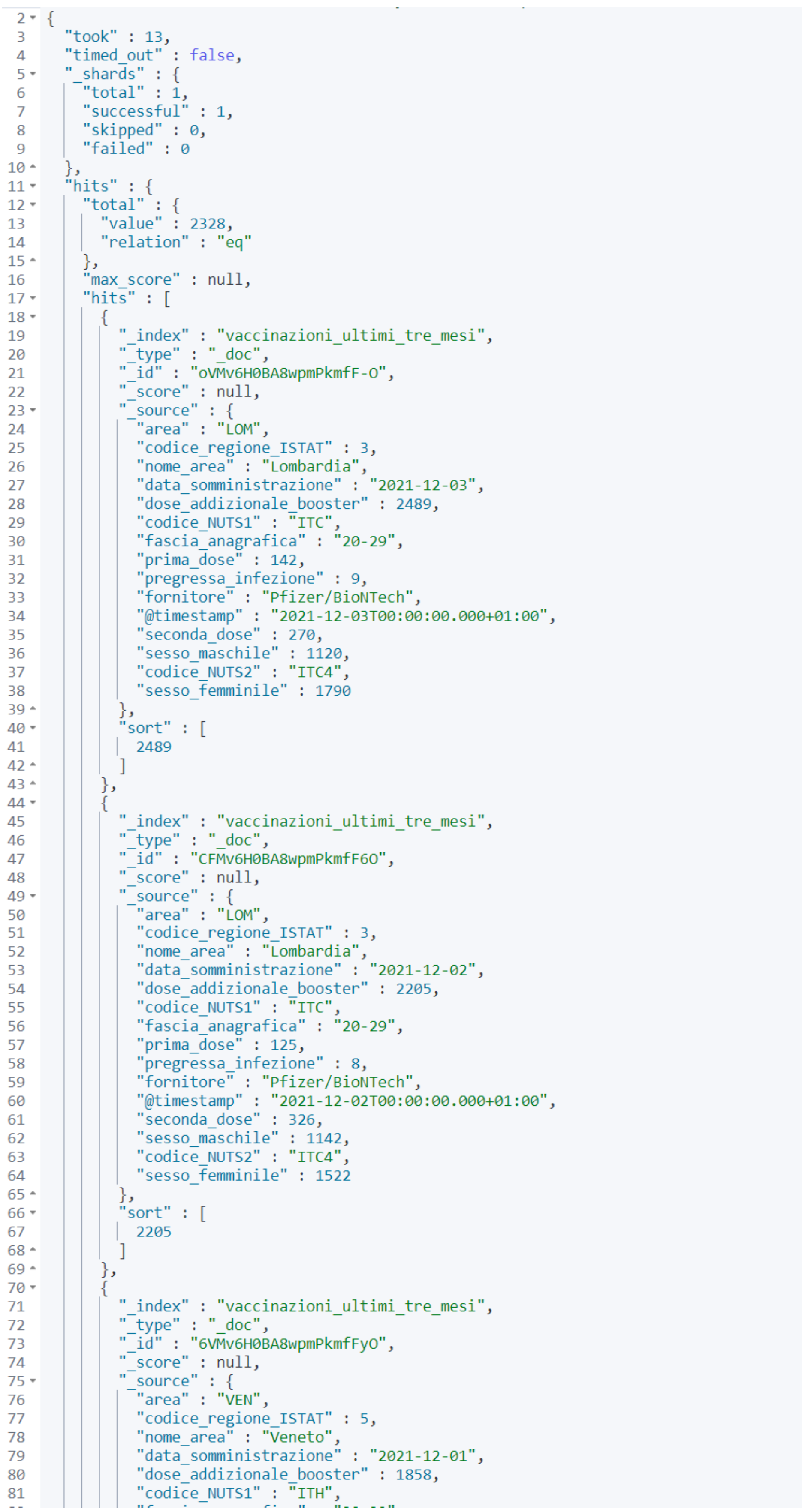
```
{
  "took" : 20,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 2,
      "relation" : "eq"
    },
    "max_score" : 6.2441006,
    "hits" : [
      {
        "_index" : "vaccinazioni_ultimi_tre_mesi",
        "_type" : "_doc",
        "_id" : "2FMv6H0BA8wpmPkmfnht",
        "_score" : 6.2441006,
        "_source" : {
          "area" : "SIC",
          "codice_regione_ISTAT" : 19,
          "nome_area" : "Sicilia",
          "data_somministrazione" : "2021-12-19",
          "dose_addizionale_booster" : 173,
          "codice_NUTS1" : "ITG",
          "fascia_anagrafica" : "20-29",
          "prima_dose" : 24,
          "pregressa_infezione" : 2,
          "fornitore" : "Pfizer/BioNTech",
          "@timestamp" : "2021-12-19T00:00:00.000+01:00",
          "seconda_dose" : 181,
          "sesso_maschile" : 181,
          "codice_NUTS2" : "ITG1",
          "sesso_femminile" : 199
        }
      },
      {
        "_index" : "vaccinazioni_ultimi_tre_mesi",
        "_type" : "_doc",
        "_id" : "wlMv6H0BA8wpmPkmfG2S",
        "_score" : 6.2441006,
        "_source" : {
          "area" : "SIC",
          "codice_regione_ISTAT" : 19,
          "nome_area" : "Sicilia",
          "data_somministrazione" : "2021-12-12",
          "dose_addizionale_booster" : 104,
          "codice_NUTS1" : "ITG",
          "fascia_anagrafica" : "20-29",
          "prima_dose" : 38,
          "pregressa_infezione" : 0,
          "fornitore" : "Pfizer/BioNTech",
          "@timestamp" : "2021-12-12T00:00:00.000+01:00",
          "seconda_dose" : 150,
          "sesso_maschile" : 157,
          "codice_NUTS2" : "ITG1",
          "sesso_femminile" : 135
        }
      }
    ]
  },
  "aggregations" : {
    "pipeline" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : 1639267200000,
          "key_as_string" : "2021-12-12T00:00:00.000Z",
          "doc_count" : 1
        },
        {
          "key" : 1639872000000,
          "key_as_string" : "2021-12-19T00:00:00.000Z",
          "doc_count" : 1
        }
      ]
    }
  }
}
```

Query #6. Best ten days for booster doses injections (aggregated by date and region) of *Pfizer-BioNTech* in the age group 20-29.



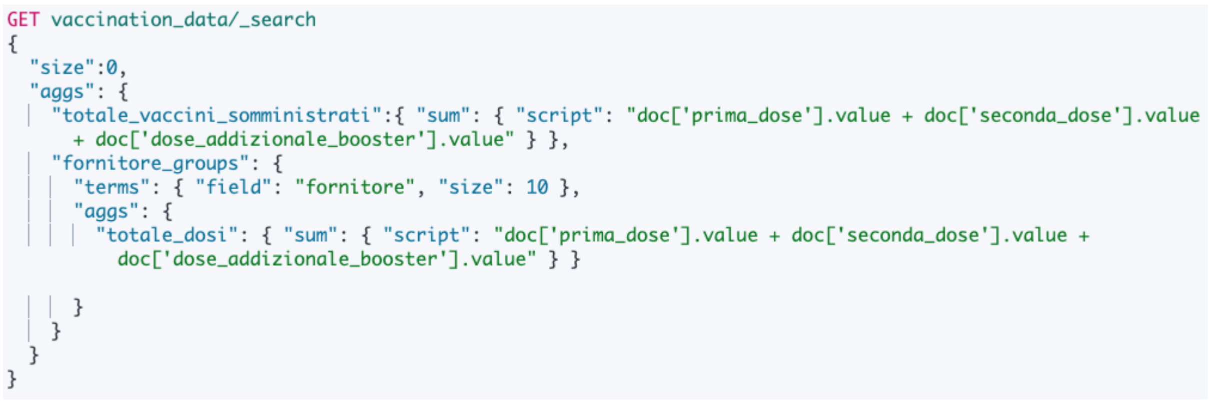
```
GET /vaccination_data/_search
{
  "size" : 10,
  "query":
  {
    "bool":
    {
      "must": [
        {
          "term": {
            "fornitore": {
              "value": "Pfizer/BioNTech"
            }
          }
        },
        {
          "term": {
            "fascia_anagrafica": {
              "value": "20-29"
            }
          }
        }
      ]
    }
  },
  "aggs": {
    "score_groups":{
      "terms": {
        "field": "data_somministrazione",
        "size": 365
      },
      "aggs":{
        "date":{
          "terms":{
            "field" : "nome_area",
            "size": 20
          }
        }
      }
    }
  },
  "sort" : [{ "dose_addizionale_booster" : {"order" : "desc"} }]
}

```

In this query, we first select the age group and the vaccine manufacturer, then we aggregate on the date and the region name (chosen over the ISTAT code for easier readability of the result), finally we sort on the amount of third doses. The complete result of the query is omitted for brevity, but two of the 10 hits are shown below.



```
{
  "took" : 13,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 2328,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [
      {
        "_index" : "vaccinazioni_ultimi_tre_mesi",
        "_type" : "_doc",
        "_id" : "oVMv6H0BA8wpmPkmfF-O",
        "_score" : null,
        "_source" : {
          "area" : "LOM",
          "codice_regione_ISTAT" : 3,
          "nome_area" : "Lombardia",
          "data_somministrazione" : "2021-12-03",
          "dose_addizionale_booster" : 2489,
          "codice_NUTS1" : "ITC",
          "fascia_anagrafica" : "20-29",
          "prima_dose" : 142,
          "pregressa_infezione" : 9,
          "fornitore" : "Pfizer/BioNTech",
          "@timestamp" : "2021-12-03T00:00:00.000+01:00",
          "seconda_dose" : 270,
          "sesso_maschile" : 1120,
          "codice_NUTS2" : "ITC4",
          "sesso_femminile" : 1790
        },
        "sort" : [
          2489
        ]
      },
      {
        "_index" : "vaccinazioni_ultimi_tre_mesi",
        "_type" : "_doc",
        "_id" : "CFMv6H0BA8wpmPkmfF6O",
        "_score" : null,
        "_source" : {
          "area" : "LOM",
          "codice_regione_ISTAT" : 3,
          "nome_area" : "Lombardia",
          "data_somministrazione" : "2021-12-02",
          "dose_addizionale_booster" : 2205,
          "codice_NUTS1" : "ITC",
          "fascia_anagrafica" : "20-29",
          "prima_dose" : 125,
          "pregressa_infezione" : 8,
          "fornitore" : "Pfizer/BioNTech",
          "@timestamp" : "2021-12-02T00:00:00.000+01:00",
          "seconda_dose" : 326,
          "sesso_maschile" : 1142,
          "codice_NUTS2" : "ITC4",
          "sesso_femminile" : 1522
        },
        "sort" : [
          2205
        ]
      },
      {
        "_index" : "vaccinazioni_ultimi_tre_mesi",
        "_type" : "_doc",
        "_id" : "6VMv6H0BA8wpmPkmfFyO",
        "_score" : null,
        "_source" : {
          "area" : "VEN",
          "codice_regione_ISTAT" : 5,
          "nome_area" : "Veneto",
          "data_somministrazione" : "2021-12-01",
          "dose_addizionale_booster" : 1858,
          "codice_NUTS1" : "ITH",
```

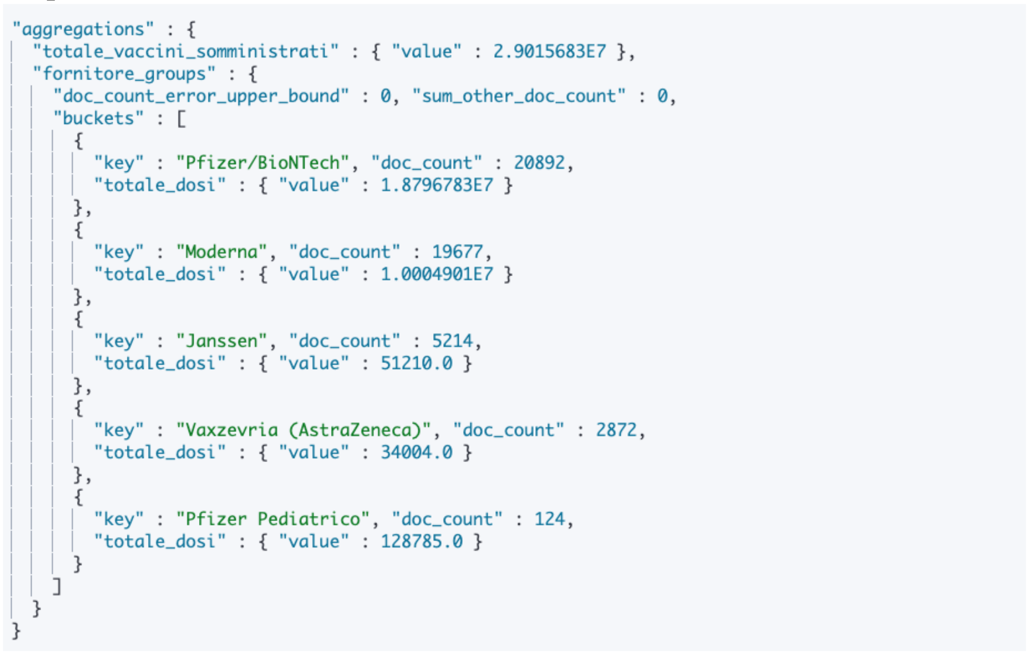
Query #7. Get the total sum of vaccination shots given out per vaccine manufacturer



```
GET vaccination_data/_search
{
  "size":0,
  "aggs": {
    "totale_vaccini_somministrati":{ "sum": { "script": "doc['prima_dose'].value + doc['seconda_dose'].value
      + doc['dose_addizionale_booster'].value" } },
    "fornitore_groups": {
      "terms": { "field": "fornitore", "size": 10 },
      "aggs": {
        "totale_dosi": { "sum": { "script": "doc['prima_dose'].value + doc['seconda_dose'].value +
          doc['dose_addizionale_booster'].value" } }

      }
    }
  }
}
```

This query is based on aggregate values. It returns the total sum of vaccine shots that have been given first, and then broken down by the manufacturer. For each bucket it returns the manufacturer's name (the vaccine's brand), the count of documents for that manufacturer and the total sum of shots that have been given.

Response:



```
"aggregations" : {
  "totale_vaccini_somministrati" : { "value" : 2.9015683E7 },
  "fornitore_groups" : {
    "doc_count_error_upper_bound" : 0, "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "Pfizer/BioNTech", "doc_count" : 20892,
        "totale_dosi" : { "value" : 1.8796783E7 }
      },
      {
        "key" : "Moderna", "doc_count" : 19677,
        "totale_dosi" : { "value" : 1.0004901E7 }
      },
      {
        "key" : "Janssen", "doc_count" : 5214,
        "totale_dosi" : { "value" : 51210.0 }
      },
      {
        "key" : "Vaxzevria (AstraZeneca)", "doc_count" : 2872,
        "totale_dosi" : { "value" : 34004.0 }
      },
      {
        "key" : "Pfizer Pediatrico", "doc_count" : 124,
        "totale_dosi" : { "value" : 128785.0 }
      }
    ]
  }
}
```

(the json structure of the response has been slightly altered to make it fit in the page)

Query #8. Get information about child vaccinations.

```
GET vaccination_data/_search
{
  "query": { "bool": { "must": [ { "match": { "fornitore": "Pfizer Pediatrico" } } ] } },
  "size":0,
  "aggs": {
    "regioni":{
      "terms": { "field": "nome_area", "size": 20 },
      "aggs": {
        "totale_pregresse_infezioni": { "sum": { "field": "pregressa_infezione" } },
        "totale_prime_dosi" : { "sum" : { "field": "prima_dose" } },
        "totale_seconde_dosi" : { "sum" : { "field": "seconda_dose" } },
        "totale_dosi_booster" : {"sum" : {"field": "dose_addizionale_booster" } },
        "totale_dosi": { "sum": { "script": "doc['prima_dose'].value + doc['seconda_dose'].value +
          doc['dose_addizionale_booster'].value" } }
      }
    }
  }
}
```

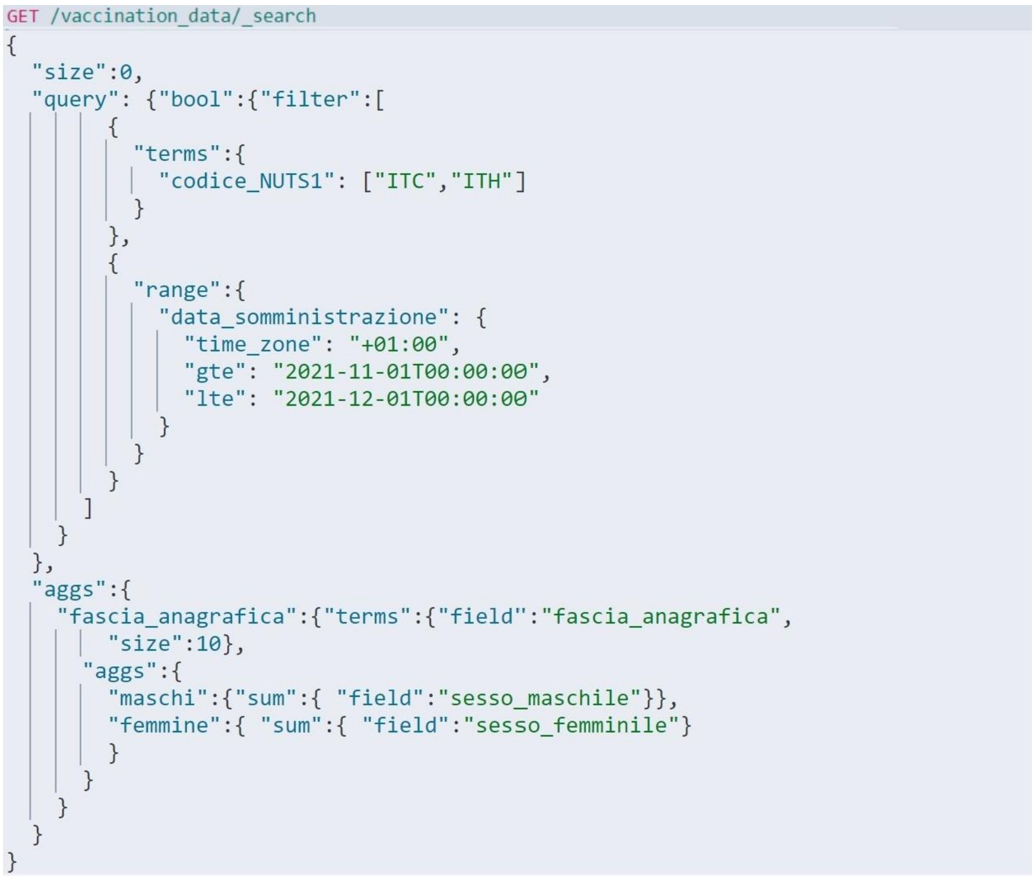
This query relies on aggregate values. It returns a bucket for each Italian region containing information about the Pfizer Pediatrico vaccinations: count of first, second and booster shots, count of previous infections and total sum of vaccine shots.
Please note that, as of writing, people in the age range of 5-11 years old have only been given first shots, so the counts for second and booster shots will result zero, and the overall sum will coincide with the count of first shots.

Response:

```
{
    "hits" : {
    "total" : { "value" : 124, "relation" : "eq" },
    "max_score" : null, "hits" : [ ]
  },
  "aggregations" : {
    "regioni" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 1,
      "buckets" : [
        {
          "key" : "Lazio", "doc_count" : 8,
          "totale_prime_dosi" : { "value" : 19029.0 },
          "totale_pregresse_infezioni" : { "value" : 9.0 },
          "totale_dosi" : { "value" : 19029.0 },
          "totale_dosi_booster" : { "value" : 0.0 },
          "totale_seconde_dosi" : { "value" : 0.0 }
        },
        {
          "key" : "Calabria",
          "doc_count" : 7,
          "totale_prime_dosi" : { "value" : 2055.0 },
          "totale_pregresse_infezioni" : { "value" : 4.0 },
          "totale_dosi" : { "value" : 2055.0 },
          "totale_dosi_booster" : { "value" : 0.0 },
          "totale_seconde_dosi" : { "value" : 0.0  }
        },
        {
          "key" : "Campania",
          "doc_count" : 7,
```

(the json structure of the response has been slightly altered to make it fit in the page)

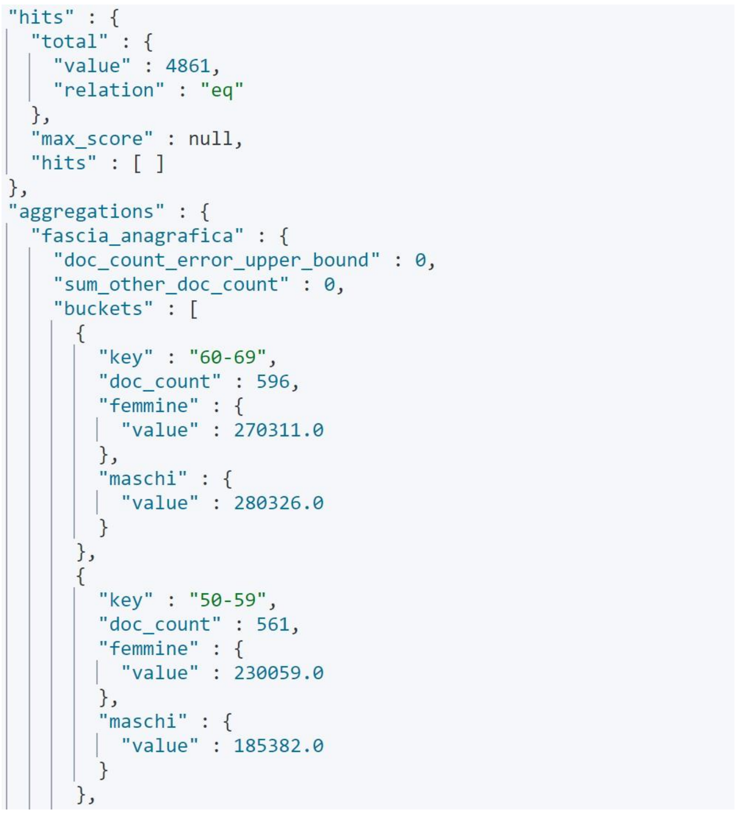
Query #9. Find how many doses were administered in North Italy in November 2021, group by age group and sex.



```
GET /vaccination_data/_search
{
  "size":0,
  "query": {"bool":{"filter":[
        {
          "terms":{
            "codice_NUTS1": ["ITC","ITH"]
          }
        },
        {
          "range":{
            "data_somministrazione": {
              "time_zone": "+01:00",
              "gte": "2021-11-01T00:00:00",
              "lte": "2021-12-01T00:00:00"
            }
          }
        }
      ]
    }
  },
  "aggs":{
    "fascia_anagrafica":{"terms":{"field":"fascia_anagrafica",
        "size":10},
      "aggs":{
        "maschi":{"sum":{ "field":"sesso_maschile"}},
        "femmine":{ "sum":{ "field":"sesso_femminile"}
        }
      }
    }
  }
}
```

This query is based on aggregate values. We first filter only the vaccinations that were made in North Italy, using the appropriate NUTS codes, during November 2021, using the “range” operator. Then we aggregate by age, and for each bucket we report the number of vaccinations administered to men and to women.

Response:

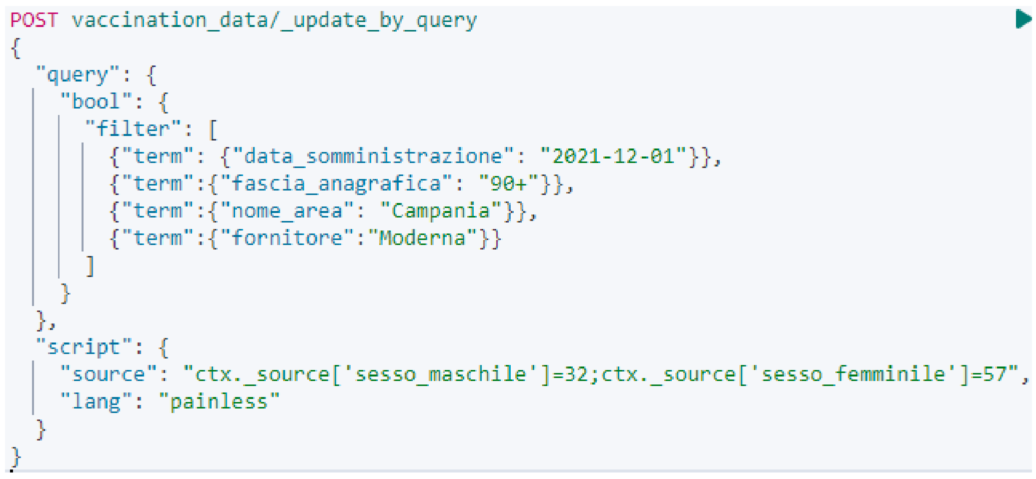


```
"hits" : {
  "total" : {
    "value" : 4861,
    "relation" : "eq"
  },
  "max_score" : null,
  "hits" : [ ]
},
"aggregations" : {
  "fascia_anagrafica" : {
    "doc_count_error_upper_bound" : 0,
    "sum_other_doc_count" : 0,
    "buckets" : [
      {
        "key" : "60-69",
        "doc_count" : 596,
        "femmine" : {
          "value" : 270311.0
        },
        "maschi" : {
          "value" : 280326.0
        }
      },
      {
        "key" : "50-59",
        "doc_count" : 561,
        "femmine" : {
          "value" : 230059.0
        },
        "maschi" : {
          "value" : 185382.0
        }
      },
```

4. Commands

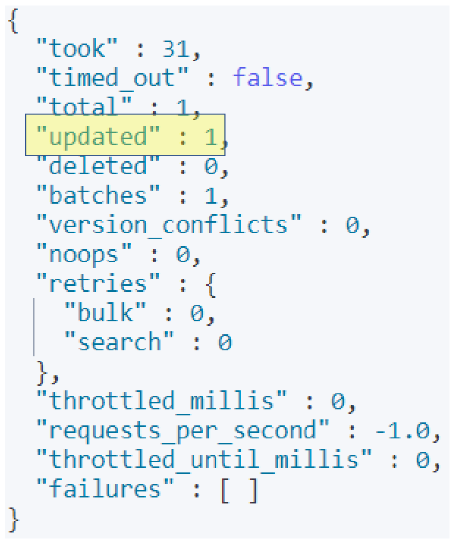
Command #1. Assuming there was an error in the recording of data, change the number of 'Moderna' vaccines performed on males and those performed on females on 01/12/2021, in the region of Campania, in the age group '90+'. In particular, men should be 32 and women 57.
N.B. The total amount of vaccines is supposed to be correct (89 people) : the only wrong data was the division of that amount between males and females.

The command:



```
POST vaccination_data/_update_by_query
{
  "query": {
    "bool": {
      "filter": [
        {"term": {"data_somministrazione": "2021-12-01"}},
        {"term":{"fascia_anagrafica": "90+"}},
        {"term":{"nome_area": "Campania"}},
        {"term":{"fornitore":"Moderna"}}
      ]
    }
  },
  "script": {
    "source": "ctx._source['sesso_maschile']=32;ctx._source['sesso_femminile']=57",
    "lang": "painless"
  }
}
```

The outcome:

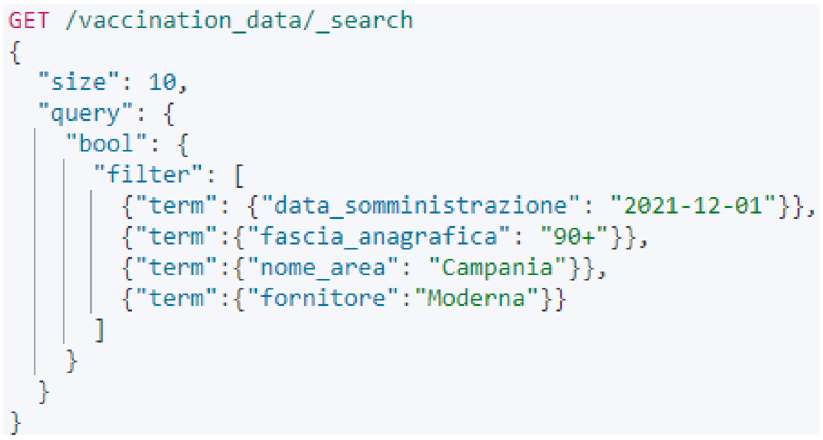


```
{
  "took" : 31,
  "timed_out" : false,
  "total" : 1,
  "updated" : 1,
  "deleted" : 0,
  "batches" : 1,
  "version_conflicts" : 0,
  "noops" : 0,
  "retries" : {
    "bulk" : 0,
    "search" : 0
  },
  "throttled_millis" : 0,
  "requests_per_second" : -1.0,
  "throttled_until_millis" : 0,
  "failures" : [ ]
}
```

In this command, first we select the document relating to the desired day through the "query" operation, filtering the wanted day ('*data_somministrazione*'), age group ('*fascia_anagrafica*'), the area ('*nome area*') and the brand of the vaccine ('*fornitore*'). Then, with the 'script' operation we change the wanted fields.

We can easily see the effectiveness of the command by running this simple query before and after the command.

The query:



```
GET /vaccination_data/_search
{
  "size": 10,
  "query": {
    "bool": {
      "filter": [
        {"term": {"data_somministrazione": "2021-12-01"}},
        {"term":{"fascia_anagrafica": "90+"}},
        {"term":{"nome_area": "Campania"}},
        {"term":{"fornitore":"Moderna"}}
      ]
    }
  }
}
```

Output before the command:

```
"hits" : [
    {
      "_index" : "vaccination_data",
      "_type" : "_doc",
      "_id" : "eWKz7XoBA78eEdoSHUgT",
      "_score" : 0.0,
      "_source" : {
       "area" : "CAM",
       "codice_regione_ISTAT" : 15,
       "nome_area" : "Campania",
       "data_somministrazione" : "2021-12-01",
       "dose_addizionale_booster" : 89,
       "codice_NUTS1" : "ITF",
       "fascia_anagrafica" : "90+",
       "prima_dose" : 0,
       "pregressa_infezione" : 0,
       "fornitore" : "Moderna",
       "@timestamp" :
"2021-12-01T00:00:00.000+01:00",
       "seconda_dose" : 0,
       "sesso_maschile" : 28,
       "codice_NUTS2" : "ITF3",
       "sesso_femminile" : 61
      }
    }
  ]
```

Output after the command:

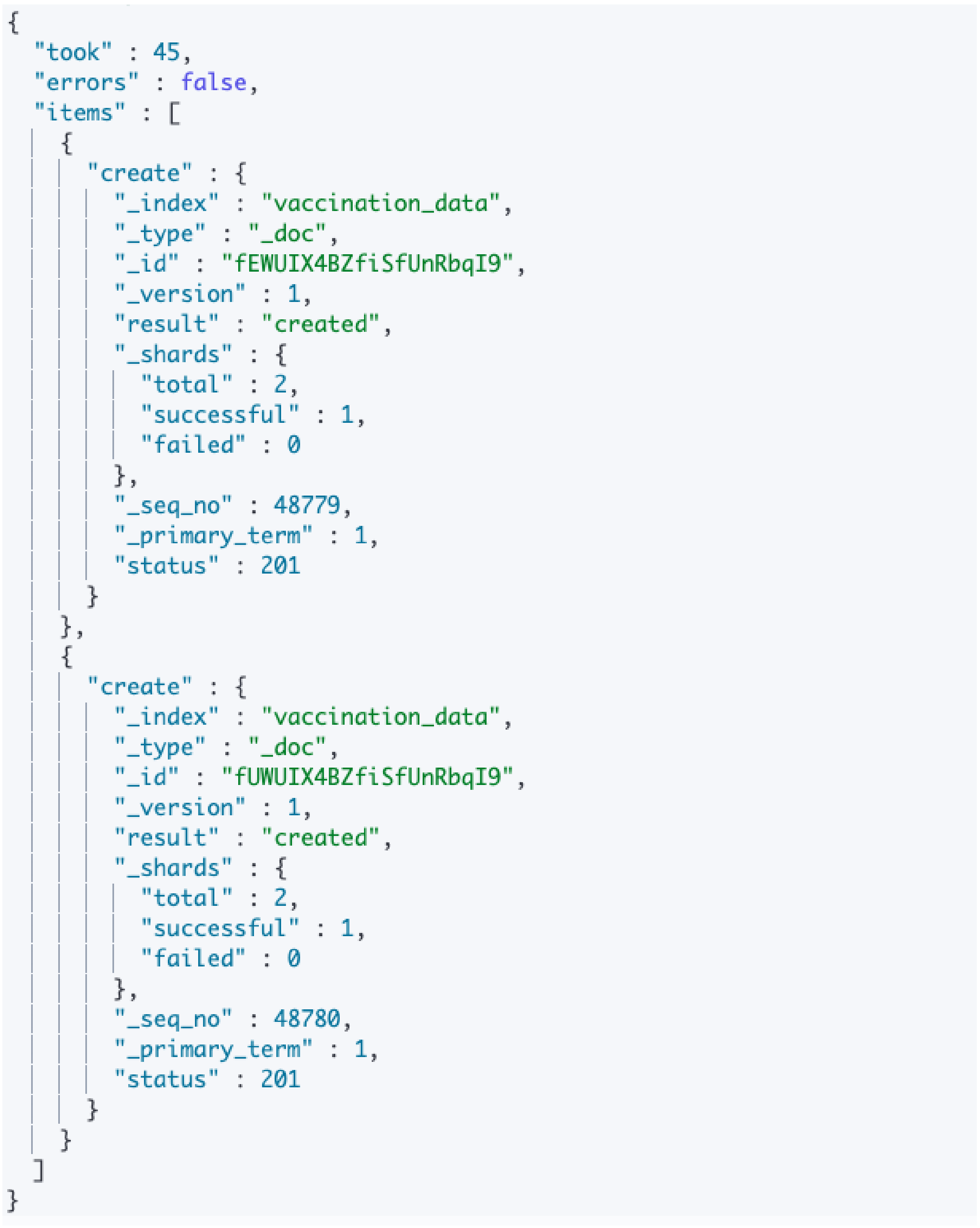
```
"hits" : [
    {
      "_index" : "vaccination_data",
      "_type" : "_doc",
      "_id" : "eWKz7XoBA78eEdoSHUgT",
      "_score" : 0.0,
      "_source" : {
       "area" : "CAM",
       "codice_regione_ISTAT" : 15,
       "nome_area" : "Campania",
       "data_somministrazione" : "2021-12-01",
       "dose_addizionale_booster" : 89,
       "codice_NUTS1" : "ITF",
       "fascia_anagrafica" : "90+",
       "prima_dose" : 0,
       "pregressa_infezione" : 0,
       "fornitore" : "Moderna",
       "@timestamp" :
"2021-12-01T00:00:00.000+01:00",
       "seconda_dose" : 0,
       "sesso_maschile" : 32,
       "codice_NUTS2" : "ITF3",
       "sesso_femminile" : 57
      }
    }
  ]
```

Command #2. Bulk insert of new documents into the database.

```
POST vaccination_data/_bulk
{"create":{"_index" : "vaccination_data"}}
{ "area" : "PIE", "codice_regione_ISTAT" : 1, "nome_area" : "Piemonte",
  "data_somministrazione" : "2021-12-29", "dose_addizionale_booster" : 8, "codice_NUTS1"
  : "ITC", "fascia_anagrafica" : "12-19", "prima_dose" : 94, "pregressa_infezione" : 0,
  "fornitore" : "Pfizer/BioNTech", "@timestamp" : "2021-12-29T00:00:00.000+01:00",
  "seconda_dose" : 106, "sesso_maschile" : 129, "codice_NUTS2" : "ITC1",
  "sesso_femminile" : 79 }
{"create":{"_index" : "vaccination_data"}}
{ "area" : "LOM", "codice_regione_ISTAT" : 1, "nome_area" : "Piemonte",
  "data_somministrazione" : "2021-12-29", "dose_addizionale_booster" : 8, "codice_NUTS1"
  : "ITC", "fascia_anagrafica" : "12-19","prima_dose" : 94,  "pregressa_infezione" : 0
  ,"fornitore" : "Pfizer/BioNTech","@timestamp" : "2021-12-29T00:00:00.000+01:00",
  "seconda_dose" : 106, "sesso_maschile" : 129, "codice_NUTS2" : "ITC1","sesso_femminile"
  : 79 }
```

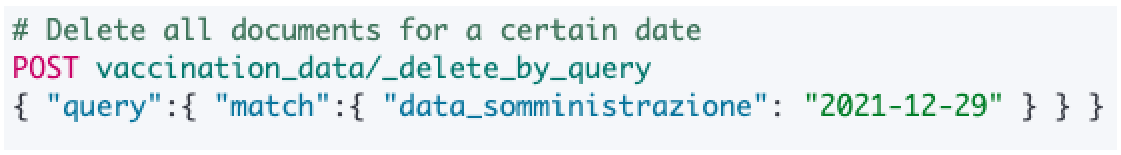
This command adds multiple documents to the database in bulk using *Elasticsearch*'s *bulk API*. This has been thought of as a tool to add multiple documents for the same date and is complemented by command #3.
Each document is to be written on a single line in order for the *bulk API* to accept it and it also has to be preceded by a {“create”:{“_index”}} command, since the API can perform a number of different operations.

Response:



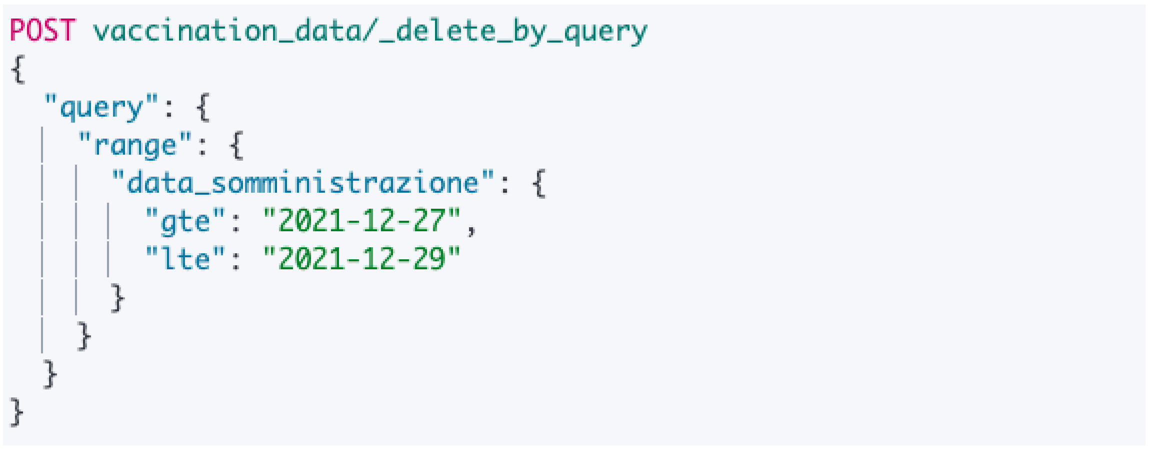
```
{
  "took" : 45,
  "errors" : false,
  "items" : [
    {
      "create" : {
        "_index" : "vaccination_data",
        "_type" : "_doc",
        "_id" : "fEWUIX4BZfiSfUnRbqI9",
        "_version" : 1,
        "result" : "created",
        "_shards" : {
          "total" : 2,
          "successful" : 1,
          "failed" : 0
        },
        "_seq_no" : 48779,
        "_primary_term" : 1,
        "status" : 201
      }
    },
    {
      "create" : {
        "_index" : "vaccination_data",
        "_type" : "_doc",
        "_id" : "fUWUIX4BZfiSfUnRbqI9",
        "_version" : 1,
        "result" : "created",
        "_shards" : {
          "total" : 2,
          "successful" : 1,
          "failed" : 0
        },
        "_seq_no" : 48780,
        "_primary_term" : 1,
        "status" : 201
      }
    }
  ]
}
```

Command #3. Delete all documents for a certain date from the database



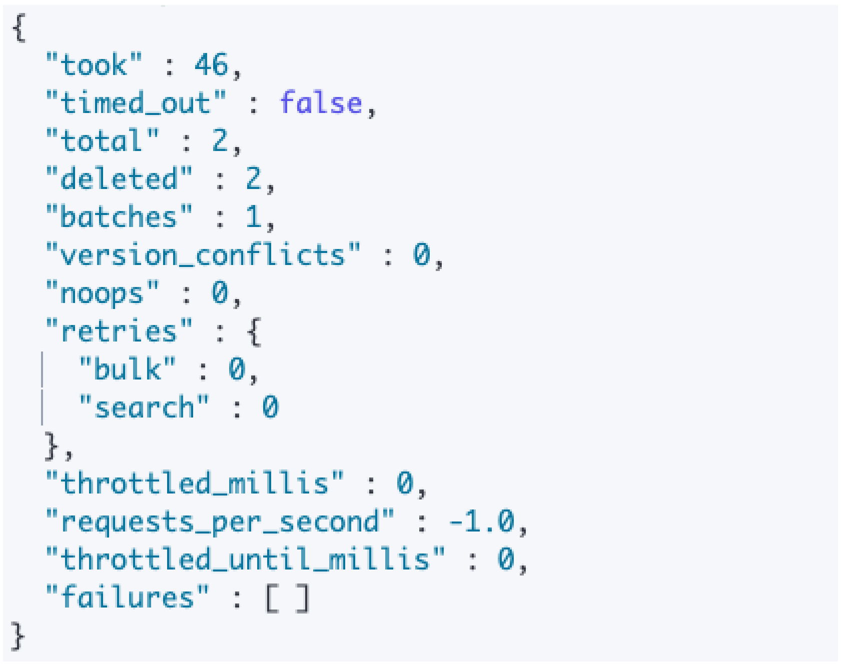
```
# Delete all documents for a certain date
POST vaccination_data/_delete_by_query
{ "query":{ "match":{ "data_somministrazione": "2021-12-29" } } }
```

This command uses the *_delete_by_query* API to remove all documents matching the query parameters from the database. The exact date *match* can also be swapped out and replaced by the *range* operator in order to delete all documents within the desired time range.



```
POST vaccination_data/_delete_by_query
{
  "query": {
    "range": {
      "data_somministrazione": {
        "gte": "2021-12-27",
        "lte": "2021-12-29"
      }
    }
  }
}
```

Response:



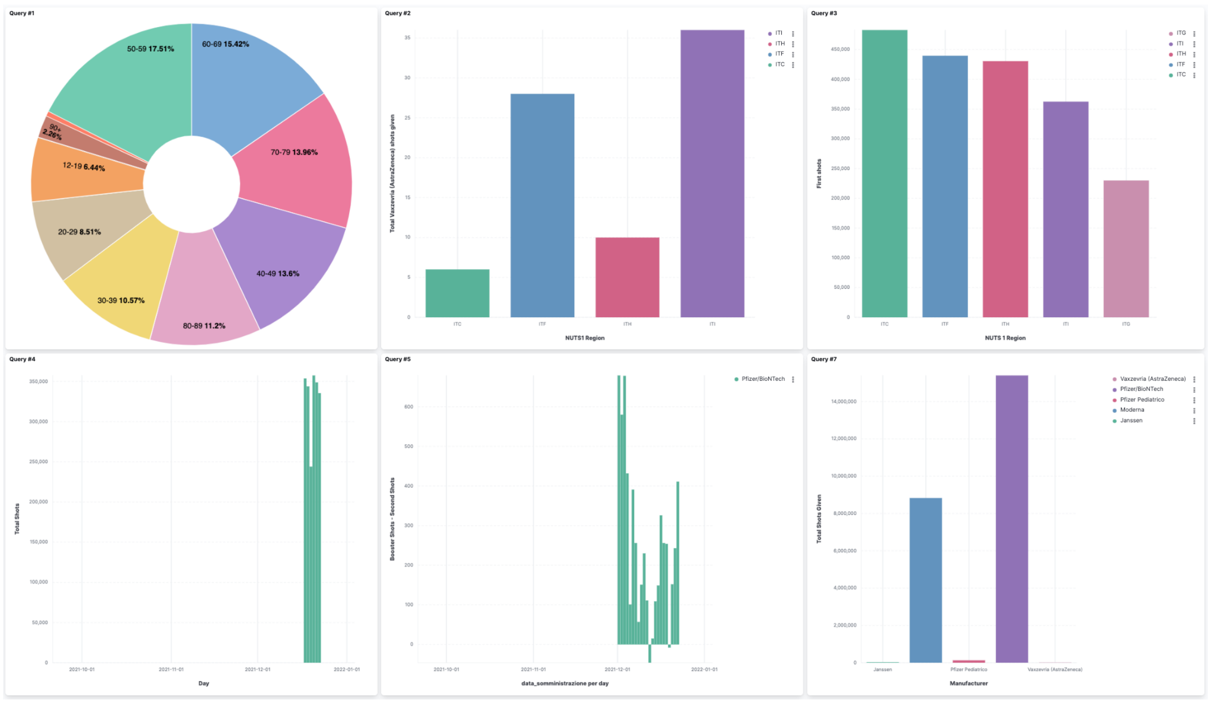
```
{
  "took" : 46,
  "timed_out" : false,
  "total" : 2,
  "deleted" : 2,
  "batches" : 1,
  "version_conflicts" : 0,
  "noops" : 0,
  "retries" : {
    "bulk" : 0,
    "search" : 0
  },
  "throttled_millis" : 0,
  "requests_per_second" : -1.0,
  "throttled_until_millis" : 0,
  "failures" : [ ]
}
```

Note that the deleted count is 2 due to the only documents that match the desired dates in the database at the time of writing were the ones added with command #2.

5. Dashboard

Using the Kibana software we were able to not only query the data stored with elasticsearch, but also create visualization dashboards in order to extract useful information from such data.
We have produced two distinct dashboards, a basic one and a more interactive one.
Please note that the screenshots have been rendered on the data for the months of september-december.

5.1 Basic dashboard



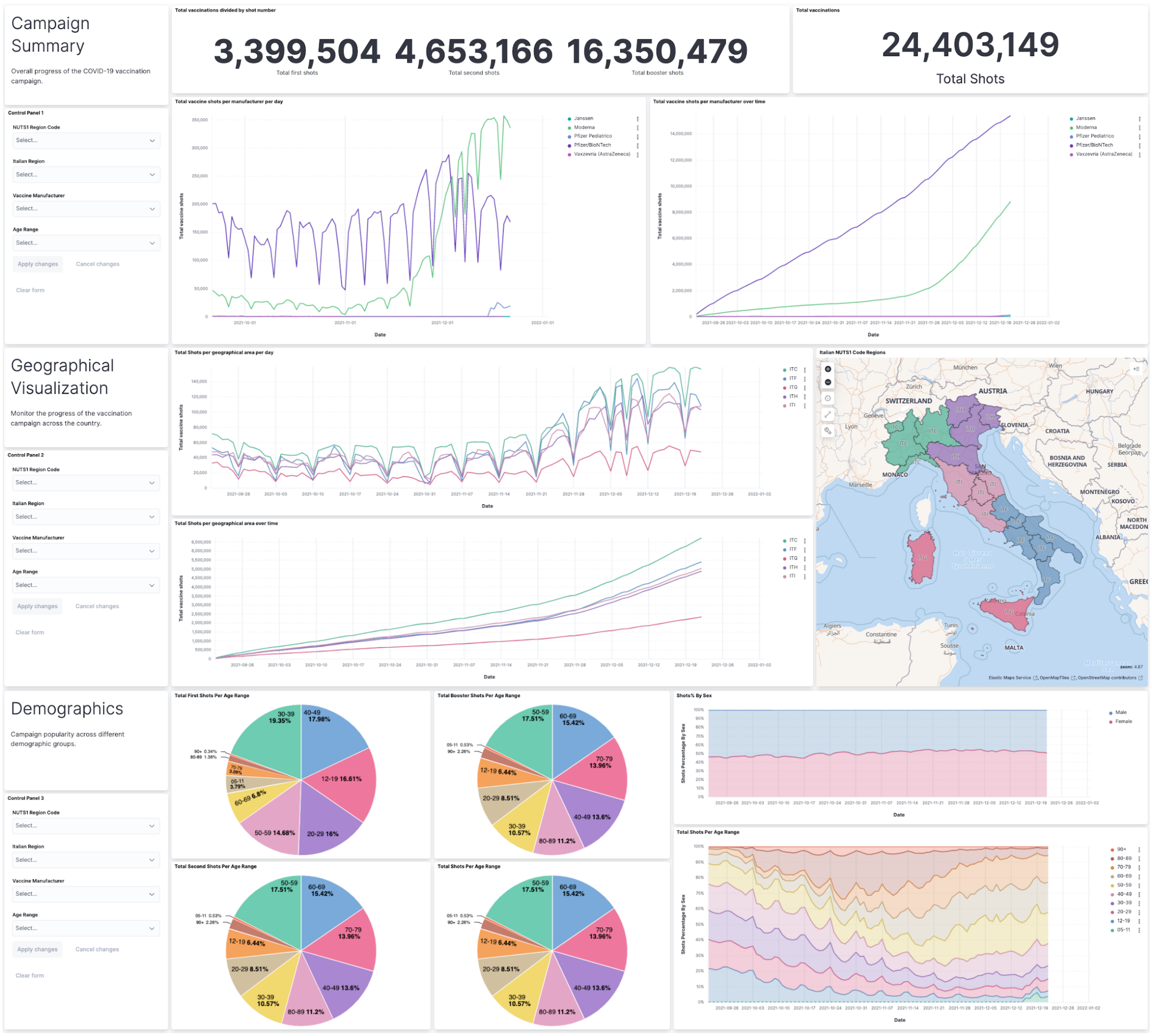
(stripped down screenshot of the dashboard without any markdown elements)

The first dashboard has been developed to visualize some of the queries from section 3.
Of course, not all the tools and functionalities that are available with *Elasticsearch*'s API are also available with the Kibana visualization engine, so some compromises have been made.

For example, in **query #5**, without using runtime mappings it's not possible to extract the only documents that have a higher number of booster shots than they have second shots. For this reason, a bar graph has been chosen to represent the difference between the two metrics and the days that have a negative value correspond to the query's hits. We've also filtered the documents in this lens to only include the Pfizer vaccine to improve readability.

The dashboard is also provided with markdown text elements to explain what every single graph illustrates and make the dashboard itself appear less crowded.

5.2 Interactive dashboard



Campaign Summary
Overall progress of the COVID-19 vaccination campaign.
Total vaccinations divided by shot number
3,399,504
Total first shots
4,653,166
Total second shots
16,350,479
Total booster shots
Total vaccinations
24,403,149
Total Shots
Control Panel 1
NUTS1 Region Code
Italian Region
Vaccine Manufacturer
Age Range
Apply changes
Cancel changes
Clear form
Total vaccine shots per manufacturer per day
Total vaccine shots per manufacturer over time
Janssen
Moderna
Pfizer Pediatrico
Pfizer/BioNTech
Vaxzevria (AstraZeneca)
Geographical Visualization
Monitor the progress of the vaccination campaign across the country.
Control Panel 2
Total Shots per geographical area per day
Total Shots per geographical area over time
Italian NUTS1 Code Regions
Demographics
Campaign popularity across different demographic groups.
Control Panel 3
Total First Shots Per Age Range
Total Booster Shots Per Age Range
Total Second Shots Per Age Range
Total Shots Per Age Range
Shots% By Sex
Male
Female
Total Shots Per Age Range
Date

(zoomed out view of the interactive dashboard)

The second dashboard has been thought to combine the data-query logic with the power of Kibana's visualization tools.
The dashboard is subdivided in three sections that show the overall progress of the vaccination campaign, some geographical visualizations and some demographic metrics respectively.
Values have not been filtered in the single lenses: each section is provided with a control panel that makes it easier to apply filters and make more in depth queries and analyses.

The map visualization in section 2 has been provided to easily interpret the NUTS1 region codes without the need to look them up. It uses a secondary index in the *Elasticsearch* database which contains the *geojson* data used to find the region borders. We could have also used the internal EMS data for Italian Regions, but it was not possible to join more than 10000 documents with it and the map appeared incomplete.

6. Optional 1: Cassandra

Loading the tuples in *Cassandra* is procedurally straightforward but certainly time-consuming, considering it is a columnar database. In fact, columnar databases are probably not the best storage model for the dataset. For this reason, the data insertion was limited to the month of December 2021 (8438 rows). The keyspace was created with the following command.

```
CREATE KEYSPACE vaccini
WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 3};
```

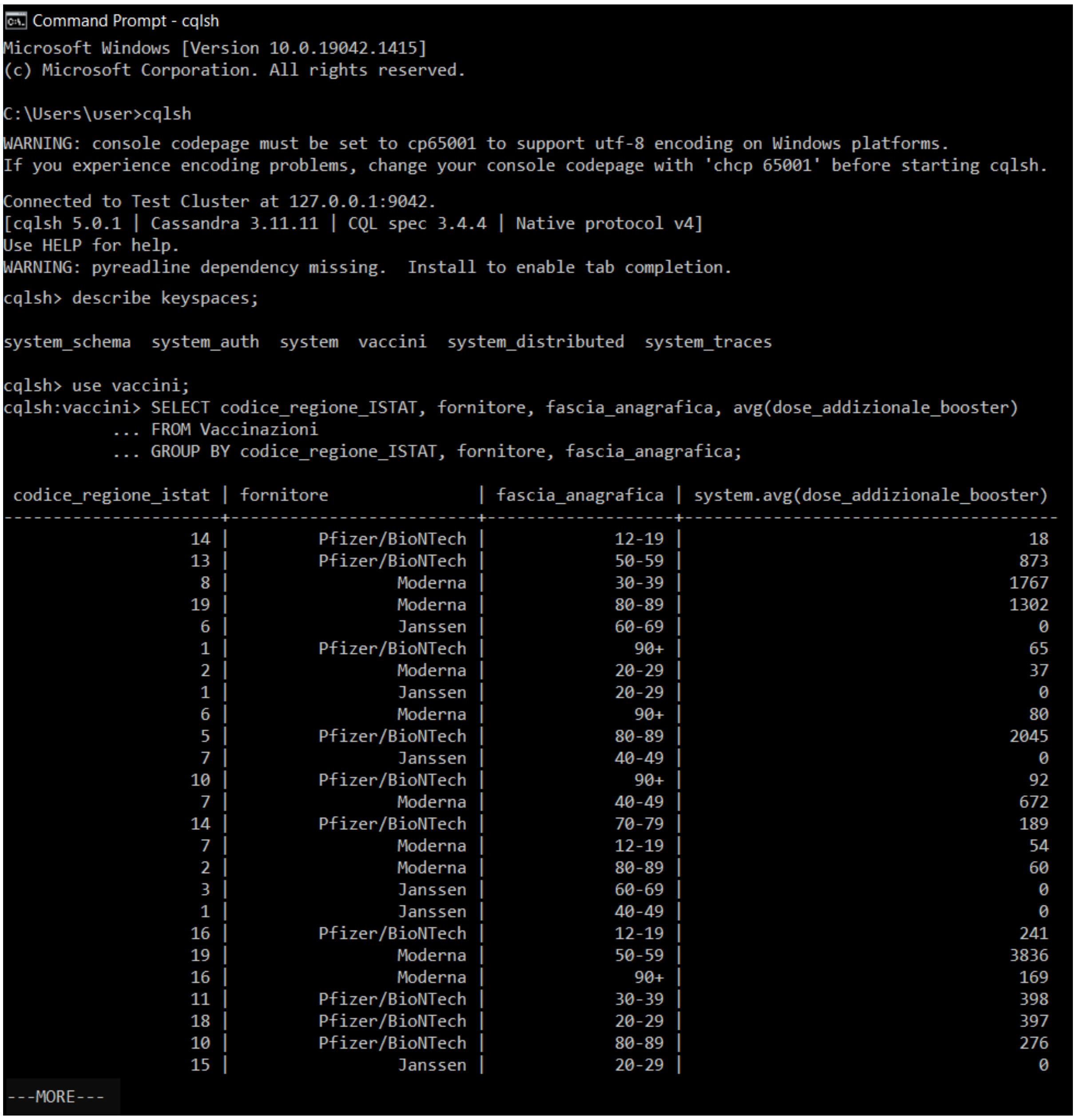
A row is uniquely identified by `data_somministrazione`, `fornitore`, `fascia_anagrafica` and `codice_regione_ISTAT`. It was chosen to make `data_somministrazione` the partition key to improve performance on date range queries.

```
CREATE TABLE Vaccinazioni (
data_somministrazione DATE,
fornitore TEXT,
area TEXT,
fascia_anagrafica TEXT,
sesso_maschile INT,
sesso_femminile INT,
prima_dose INT,
seconda_dose INT,
pregressa_infezione INT,
dose_addizionale_booster INT,
codice_NUTS1 TEXT,
codice_NUTS2 TEXT,
codice_regione_ISTAT SMALLINT,
nome_area TEXT,
PRIMARY KEY((codice_regione_ISTAT, fornitore, fascia_anagrafica),
data_somministrazione))
WITH CLUSTERING ORDER BY (data_somministrazione ASC);
```

Once the keyspace and the table have been manually created in *cqlsh*, a python script called `cassandra_interface.py` handles the insertion of all December records through a direct database connection with the *DataStax Cassandra Python Driver*. Batches were not implemented considering the performance gain in testing was not significant. As for queries, given *CQL* is a very simple query language, often replaced by *Hive* or *SparkSQL* for complex querying, not many interesting queries can be shown. In fact, as of version 3 of *Cassandra* (the last version compatible with *Windows*), nested queries or even aggregations which don't take on the ENTIRE partition key are not allowed.

```
SELECT codice_regione_ISTAT, fornitore, fascia_anagrafica,
avg(dose_addizionale_booster)
FROM Vaccinazioni
GROUP BY codice_regione_ISTAT, fornitore, fascia_anagrafica
```

This simple query above aggregates by region, vaccine manufacturer and age group, basically giving the average of their category. Ordering by the average is not an option, given the ORDER BY command only supports clustering keys;



Command Prompt - cqlsh

```
Microsoft Windows [Version 10.0.19042.1415]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user>cqlsh
WARNING: console codepage must be set to cp65001 to support utf-8 encoding on Windows platforms.
If you experience encoding problems, change your console codepage with 'chcp 65001' before starting cqlsh.

Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.11 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing.  Install to enable tab completion.
cqlsh> describe keyspaces;

system_schema  system_auth  system  vaccini  system_distributed  system_traces

cqlsh> use vaccini;
cqlsh:vaccini> SELECT codice_regione_ISTAT, fornitore, fascia_anagrafica, avg(dose_addizionale_booster)
           ... FROM Vaccinazioni
           ... GROUP BY codice_regione_ISTAT, fornitore, fascia_anagrafica;

 codice_regione_istat | fornitore               | fascia_anagrafica | system.avg(dose_addizionale_booster)
----------------------+-------------------------+-------------------+-------------------------------------
                   14 |         Pfizer/BioNTech |             12-19 |                                   18
                   13 |         Pfizer/BioNTech |             50-59 |                                  873
                    8 |                 Moderna |             30-39 |                                 1767
                   19 |                 Moderna |             80-89 |                                 1302
                    6 |                 Janssen |             60-69 |                                    0
                    1 |         Pfizer/BioNTech |               90+ |                                   65
                    2 |                 Moderna |             20-29 |                                   37
                    1 |                 Janssen |             20-29 |                                    0
                    6 |                 Moderna |               90+ |                                   80
                    5 |         Pfizer/BioNTech |             80-89 |                                 2045
                    7 |                 Janssen |             40-49 |                                    0
                   10 |         Pfizer/BioNTech |               90+ |                                   92
                    7 |                 Moderna |             40-49 |                                  672
                   14 |         Pfizer/BioNTech |             70-79 |                                  189
                    7 |                 Moderna |             12-19 |                                   54
                    2 |                 Moderna |             80-89 |                                   60
                    3 |                 Janssen |             60-69 |                                    0
                    1 |                 Janssen |             40-49 |                                    0
                   16 |         Pfizer/BioNTech |             12-19 |                                  241
                   19 |                 Moderna |             50-59 |                                 3836
                   16 |                 Moderna |               90+ |                                  169
                   11 |         Pfizer/BioNTech |             30-39 |                                  398
                   18 |         Pfizer/BioNTech |             20-29 |                                  397
                   10 |         Pfizer/BioNTech |             80-89 |                                  276
                   15 |                 Janssen |             20-29 |                                    0
---MORE---
```

7. Optional 2: Integration of an additional DB

7.1. The Database

The additional Database we have chosen is taken from the following GitHub folder by the Italian Civil Protection Department:
https://github.com/pcm-dpc/COVID-19/tree/master/dati-andamento-nazionale
This database was chosen because it provides national information on swabs, hospitals, positive people, and those healed. Therefore we thought it was a good integration to the first database analyzed: the two indices make it possible to make statistics of different types and analyze different aspects of the pandemic at the national level.

In particular, the fields reported in the new database are:

Name of the field	Description
data	Date of notification
stato	Country of reference
ricoverati_con_sintomi	Hospitalised patients with symptoms
terapia_intensiva	Intensive Care
totale_ospedalizzati	Total hospitalised patients
isolamento_domiciliare	Home confinement
totale_positivi	Total amount of current positive cases (Hospitalised patients + Home confinement)
variazione_totale_positivi	New amount of current positive cases (totale_positivi current day - totale_positivi previous day)
nuovi_positivi	New amount of current positive cases (totale_casi current day - totale_casi previous day)
dimessi_guariti	Recovered
deceduti	Deceased
casi_da_sospetto_diagnostico	Positive cases emerged from clinical activity
casi_da_screening	Positive cases emerging from surveys and tests, planned at national or regional level
totale_casi	Total amount of positive cases
tamponi	Tests performed
casi_testati	Total number of people tested (processed with molecular tests)
note	Notes
ingressi_terapia_intensiva	Daily admissions to intensive care
note_test	Notes on the tests carried out
note_casi	Notes on the cases tested
totale_positivi_test_molecolare	Total number of people resulted positive with a molecular test
totale_positivi_test_antigenico_rapido	Total number of people resulted positive with an antigenic test
tamponi_test_molecolare	Total number of molecular tests
tamponi_test_antigenico_rapido	Total number of antigenic tests

As regards the types of the fields mentioned above, they are all numerical except '*data*' (YYYY-MM-DD HH:MM:SS- ISO 8601 format), '*stato*' (XYZ -ISO 3166-1 alpha-3 format) and the 'note' fields (*textual*).
We decided not to change them for the same considerations that were made for the first database (paragraph 2). In particular, the *'data'* field is convenient to have it in the format 'date' to be able to use operations like 'date_histogram' and to be able to use it as 'meeting

point', a common field between the two databases: "*data_somministrazione*" in the first database is also in the YYYY-MM-DD HH:MM:SS format.
All the other fields have been left numerical as it is very convenient to use them to do operations and statistics

We decided to delete some fields as they are not relevant for the construction of queries and the elaboration of statistics and reasoning on the data. These fields are *'note', 'note_test', 'note_casi', 'stato'* (all the data cover the same state, Italy). We deleted them using the following simple command:

```
POST /tests/_update_by_query
{
  "script": {
    "source":
"ctx._source.remove('note');ctx._source.remove('stato');ctx._source.remove('note_test');ctx._sour
ce.remove('note_casi');",
    "lang": "painless"
  },
  "query" : {
    "bool": {
      "should": [
        {"exists": { "field": "note" }},
        {"exists": { "field": "stato" }},
        {"exists": { "field": "note_test"}},
        {"exists": { "field": "note_casi"}}
      ]
    }
  }
}
```

This command extracted and deleted (with the 'script' operator) all the wanted fields from all the documents that contained them (selected with the 'query' operator).
Another change we made on the initial database was to match the time span of the first database to the one covered by this additional one: therefore we have deleted all documents which do not relate to days between 1 September 2021 and 22 December 2021. To do this we used the following command:

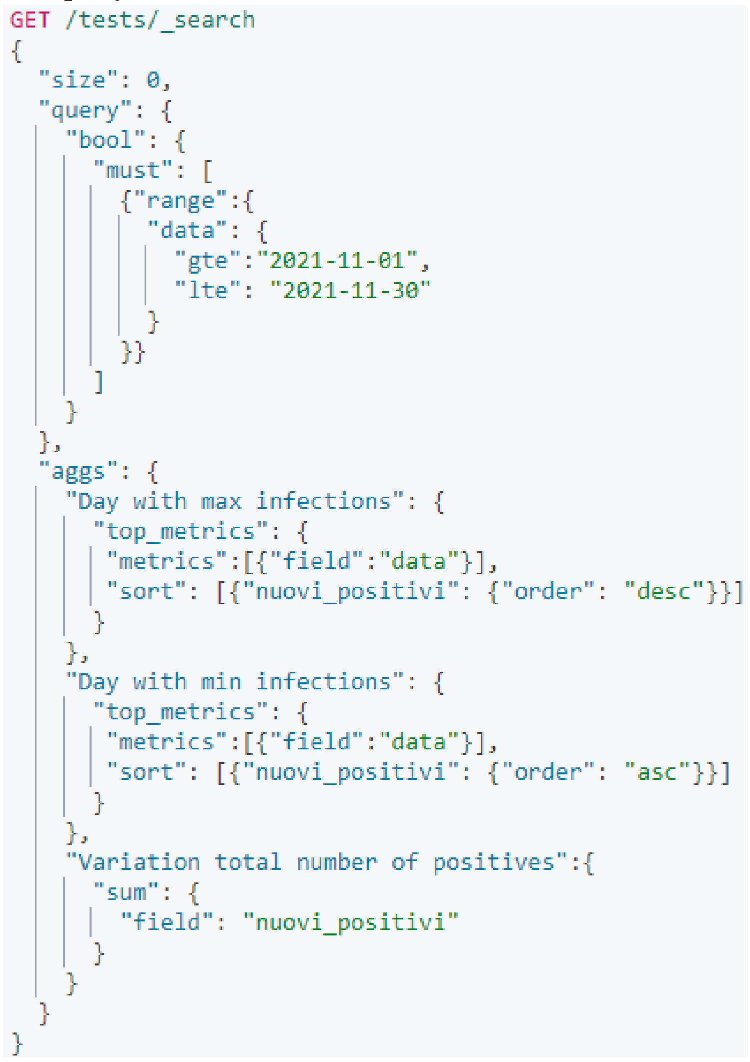
```
POST /tests/_delete_by_query
{
   "query": {
    "bool": {
      "should": [
        {"range":{"data": {"gt":"2021-12-22"}}},
        {"range":{"data": {"lt": "2021-09-01"}}}
      ]
    }
  }
}
```

This command extracted and deleted all the documents of days before the 1st of September and after the 22nd of December 2021 with the 'should' operator used as the logic operator 'OR'.

7.2. Queries

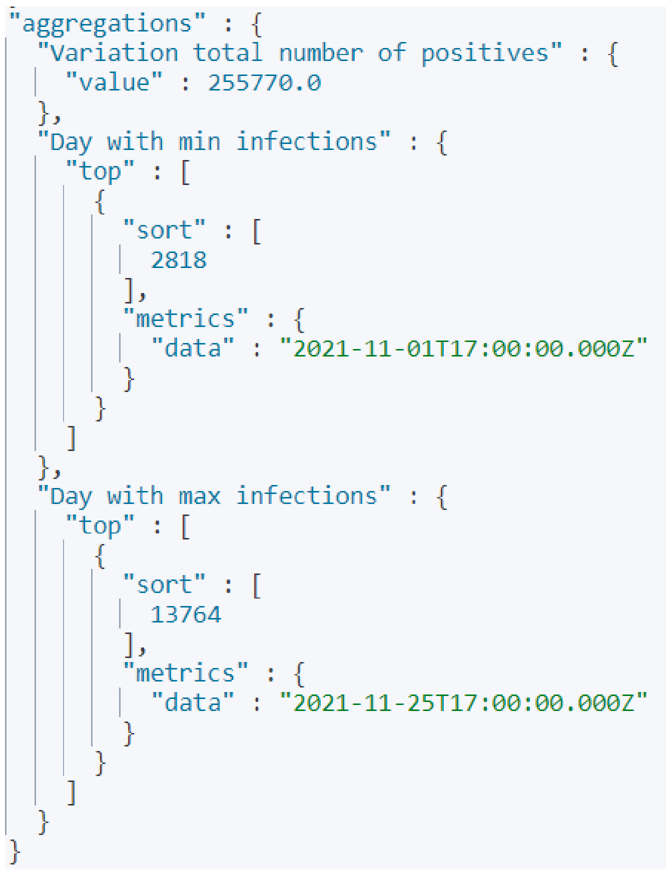
Query #1. For the month of November, extract the increase in positive cases (the total) and the days when there were the largest and smaller number of new infections.

The query:



```
GET /tests/_search
{
  "size": 0,
  "query": {
    "bool": {
      "must": [
        {"range":{
          "data": {
            "gte":"2021-11-01",
            "lte": "2021-11-30"
          }
        }}
      ]
    }
  },
  "aggs": {
    "Day with max infections": {
      "top_metrics": {
       "metrics":[{"field":"data"}],
       "sort": [{"nuovi_positivi": {"order": "desc"}}]
      }
    },
    "Day with min infections": {
      "top_metrics": {
       "metrics":[{"field":"data"}],
       "sort": [{"nuovi_positivi": {"order": "asc"}}]
      }
    },
    "Variation total number of positives":{
      "sum": {
        "field": "nuovi_positivi"
      }
    }
  }
}
```

The outcome:

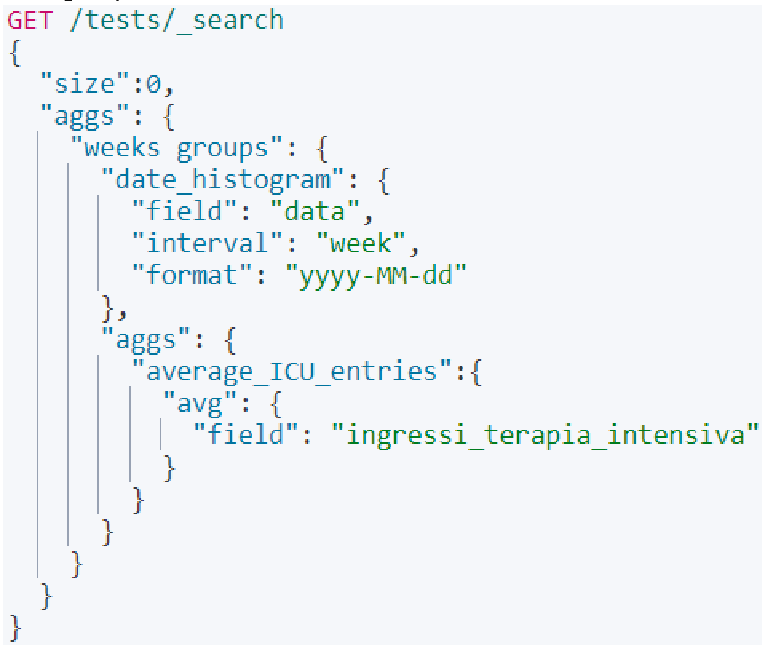


```
"aggregations" : {
  "Variation total number of positives" : {
    "value" : 255770.0
  },
  "Day with min infections" : {
    "top" : [
      {
        "sort" : [
          2818
        ],
        "metrics" : {
          "data" : "2021-11-01T17:00:00.000Z"
        }
      }
    ]
  },
  "Day with max infections" : {
    "top" : [
      {
        "sort" : [
          13764
        ],
        "metrics" : {
          "data" : "2021-11-25T17:00:00.000Z"
        }
      }
    ]
  }
}
```

In this query first we filter the documents with the query operation, selecting only those which have a date in the range 01/11/21-30-11-21. In the 'aggs' operation then we order the documents according to the number of new positives (first descending and then ascending) and we take the field 'data' of the first entry. This is done using the 'top_metrics' operation, respectively to find the day with the most infections and the day with the least infections. At the end we calculate the change in the total number of positives in the month by adding the new positives of every day.

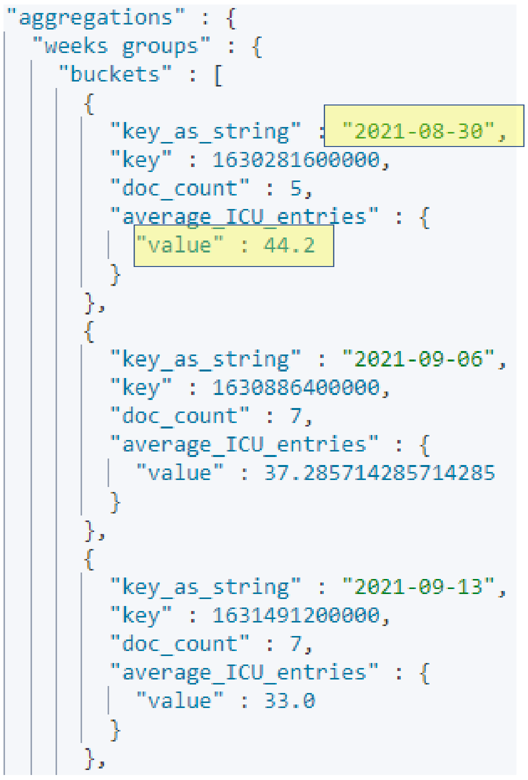
Query #2. Extract for every week the average number of ICU entries per day.

The query:



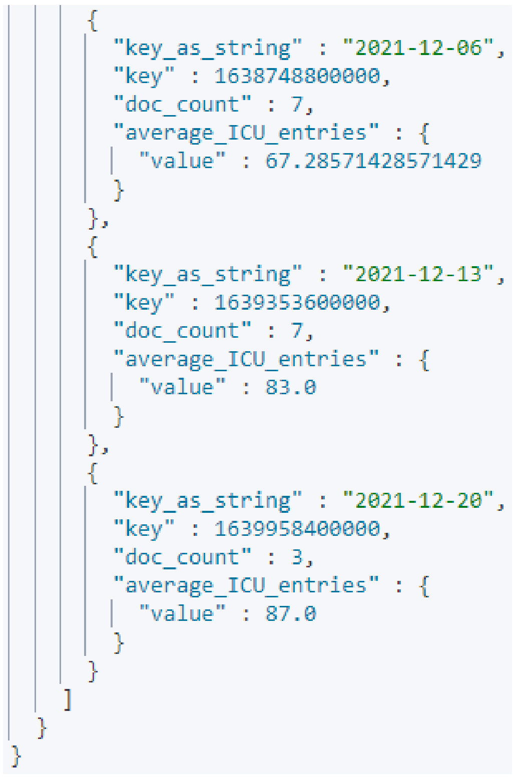
```
GET /tests/_search
{
  "size":0,
  "aggs": {
    "weeks groups": {
      "date_histogram": {
        "field": "data",
        "interval": "week",
        "format": "yyyy-MM-dd"
      },
      "aggs": {
        "average_ICU_entries":{
          "avg": {
            "field": "ingressi_terapia_intensiva"
          }
        }
      }
    }
  }
}
```

The outcome:



```
"aggregations" : {
  "weeks groups" : {
    "buckets" : [
      {
        "key_as_string" : "2021-08-30",
        "key" : 1630281600000,
        "doc_count" : 5,
        "average_ICU_entries" : {
          "value" : 44.2
        }
      },
      {
        "key_as_string" : "2021-09-06",
        "key" : 1630886400000,
        "doc_count" : 7,
        "average_ICU_entries" : {
          "value" : 37.285714285714285
        }
      },
      {
        "key_as_string" : "2021-09-13",
        "key" : 1631491200000,
        "doc_count" : 7,
        "average_ICU_entries" : {
          "value" : 33.0
        }
      },
```

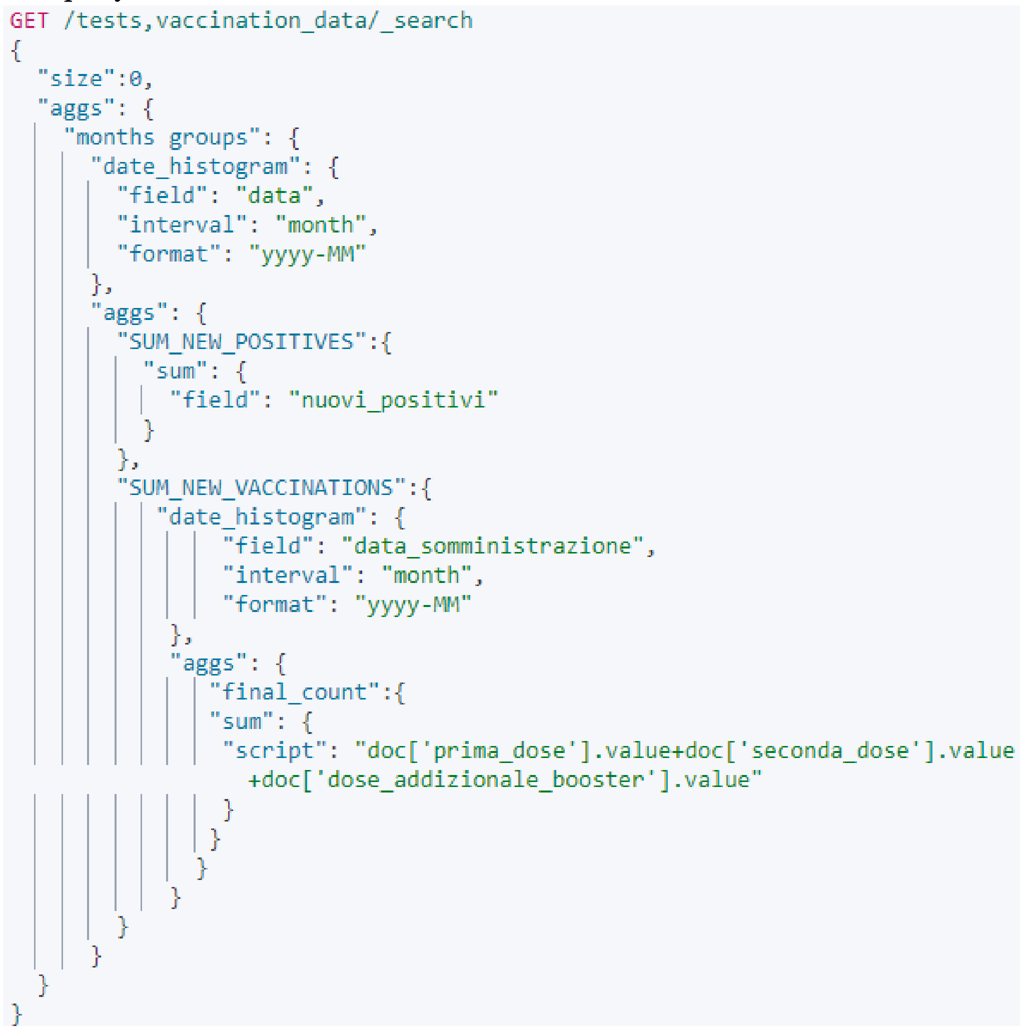
....


```
      {
        "key_as_string" : "2021-12-06",
        "key" : 1638748800000,
        "doc_count" : 7,
        "average_ICU_entries" : {
          "value" : 67.28571428571429
        }
      },
      {
        "key_as_string" : "2021-12-13",
        "key" : 1639353600000,
        "doc_count" : 7,
        "average_ICU_entries" : {
          "value" : 83.0
        }
      },
      {
        "key_as_string" : "2021-12-20",
        "key" : 1639958400000,
        "doc_count" : 3,
        "average_ICU_entries" : {
          "value" : 87.0
        }
      }
    ]
  }
}
```

We impose size =0 because we want only the final numbers to be visualized and not single documents. The first aggregation we do is to partition the documents in weeks. This is done automatically using the operator ‘ date_histogram’. As we can see by the results some week sections (the first one of September and the last one of December) will not be composed of 7 documents, one for each day. Those two sections will have a smaller amount of documents because our database covers the dates between 01/09/21 and 22/12/21 and those dates do not correspond exactly with Mondays(start of the week). In our case this is not a problem for the number we want to compute because we are looking for an average between the days.
At the end, to compute that average we use the sub-aggregation with operator ‘avg’.

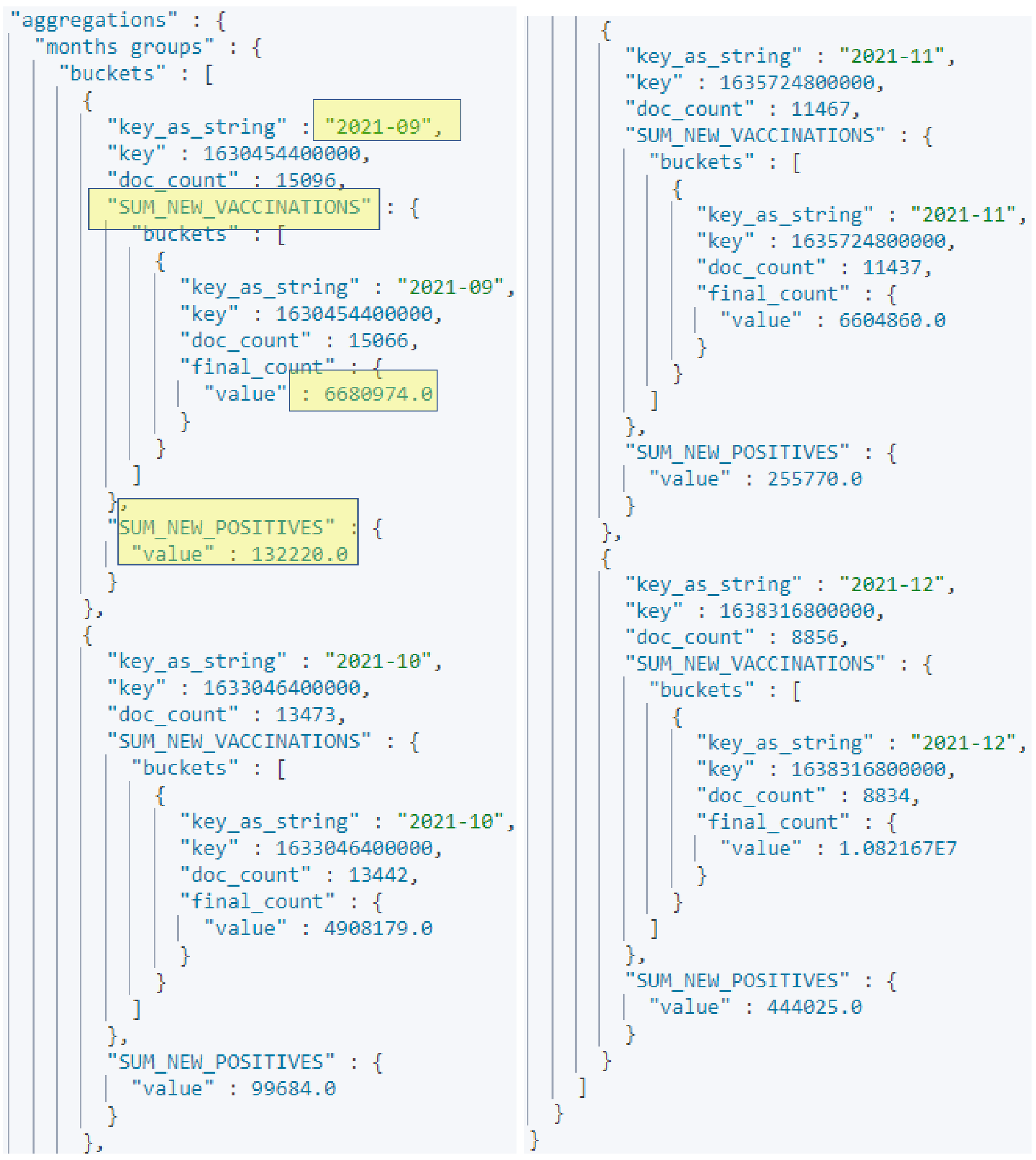
Query #3 (Multiple index). Extract the total number of vaccines made and the number of people found positive in each month (throughout Italy).

The query:



```
GET /tests,vaccination_data/_search
{
  "size":0,
  "aggs": {
    "months groups": {
      "date_histogram": {
        "field": "data",
        "interval": "month",
        "format": "yyyy-MM"
      },
      "aggs": {
        "SUM_NEW_POSITIVES":{
          "sum": {
            "field": "nuovi_positivi"
          }
        },
        "SUM_NEW_VACCINATIONS":{
           "date_histogram": {
               "field": "data_somministrazione",
               "interval": "month",
               "format": "yyyy-MM"
            },
            "aggs": {
              "final_count":{
              "sum": {
                "script": "doc['prima_dose'].value+doc['seconda_dose'].value
                  +doc['dose_addizionale_booster'].value"
                }
              }
            }
          }
        }
      }
    }
  }
}
```

The outcome:

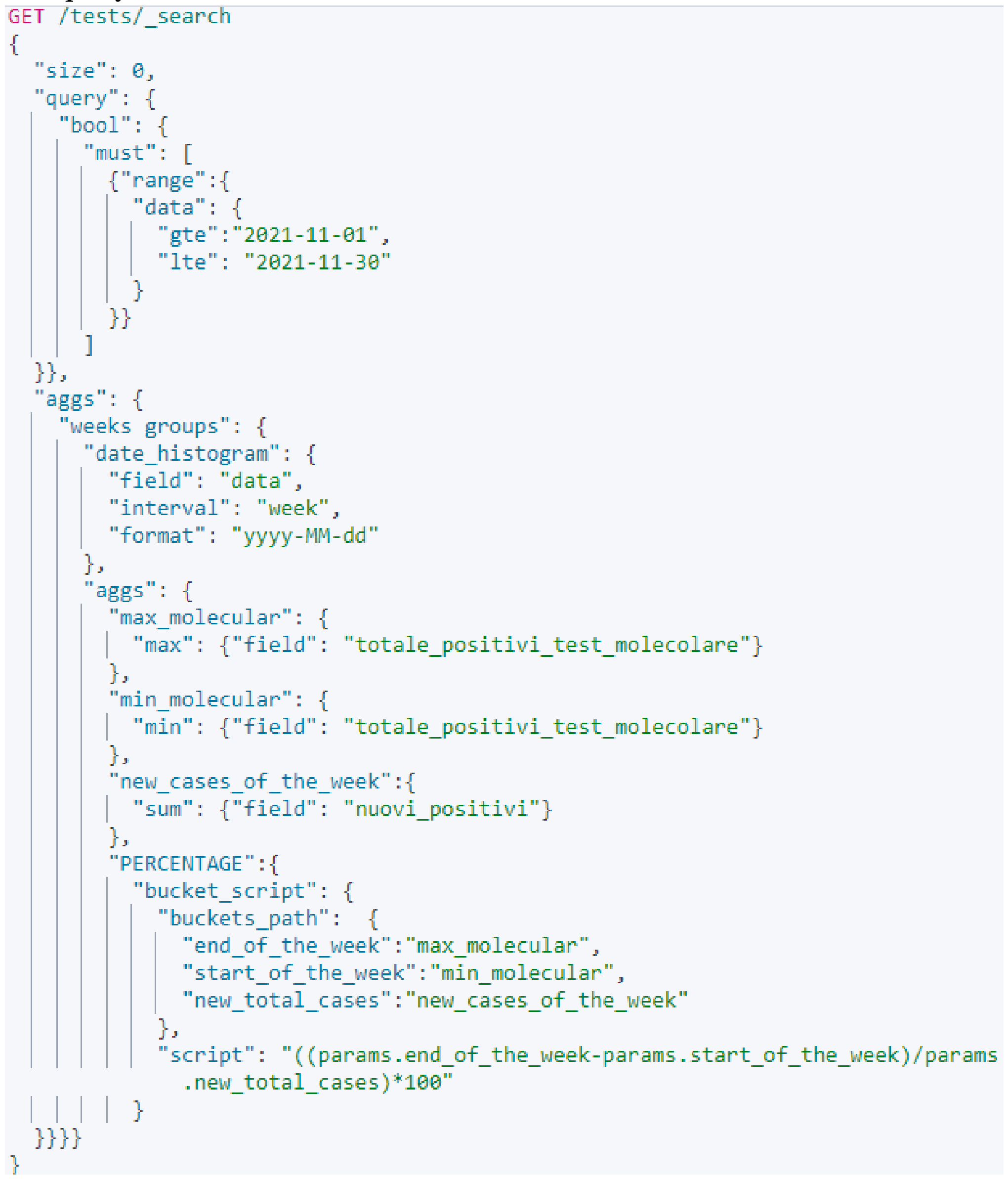


```
"aggregations" : {
  "months groups" : {
    "buckets" : [
      {
        "key_as_string" : "2021-09",
        "key" : 1630454400000,
        "doc_count" : 15096,
        "SUM_NEW_VACCINATIONS" : {
          "buckets" : [
            {
              "key_as_string" : "2021-09",
              "key" : 1630454400000,
              "doc_count" : 15066,
              "final_count" : {
                "value" : 6680974.0
              }
            }
          ]
        },
        "SUM_NEW_POSITIVES" : {
          "value" : 132220.0
        }
      },
      {
        "key_as_string" : "2021-10",
        "key" : 1633046400000,
        "doc_count" : 13473,
        "SUM_NEW_VACCINATIONS" : {
          "buckets" : [
            {
              "key_as_string" : "2021-10",
              "key" : 1633046400000,
              "doc_count" : 13442,
              "final_count" : {
                "value" : 4908179.0
              }
            }
          ]
        },
        "SUM_NEW_POSITIVES" : {
          "value" : 99684.0
        }
      },
      {
        "key_as_string" : "2021-11",
        "key" : 1635724800000,
        "doc_count" : 11467,
        "SUM_NEW_VACCINATIONS" : {
          "buckets" : [
            {
              "key_as_string" : "2021-11",
              "key" : 1635724800000,
              "doc_count" : 11437,
              "final_count" : {
                "value" : 6604860.0
              }
            }
          ]
        },
        "SUM_NEW_POSITIVES" : {
          "value" : 255770.0
        }
      },
      {
        "key_as_string" : "2021-12",
        "key" : 1638316800000,
        "doc_count" : 8856,
        "SUM_NEW_VACCINATIONS" : {
          "buckets" : [
            {
              "key_as_string" : "2021-12",
              "key" : 1638316800000,
              "doc_count" : 8834,
              "final_count" : {
                "value" : 1.082167E7
              }
            }
          ]
        },
        "SUM_NEW_POSITIVES" : {
          "value" : 444025.0
        }
      }
    ]
  }
}
```

First we partition the documents of the index ‘tests’ into months according to the parameter ‘data’ with the operator ‘date_histogram’. With a sub-aggregation, then we can compute the total number of new positives in each month (with the operator ‘sum’). At this point, to partition also the documents of the index ‘vaccination_data’ we have to use again the ‘date_histogram’ operator, this time using the ‘*data_somministrazione*’ field, setting the interval to ‘month’ again. After that, we can use a sub-aggregation to sum for each of the selected documents the number of first doses, second doses, and booster doses (through the ‘script’) and sum the results with the operator ‘sum’.

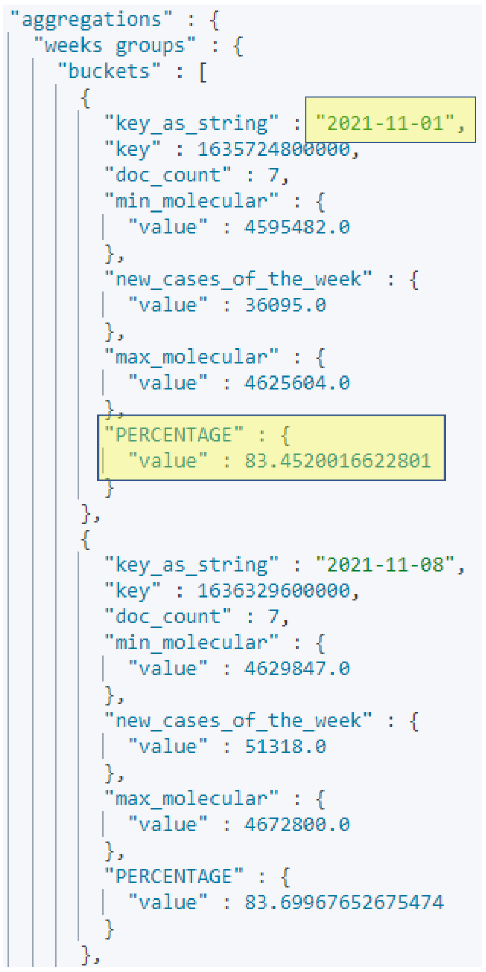
Query #4. For each week of the month of November, find the percentage of new cases detected by molecular tests.

The query:

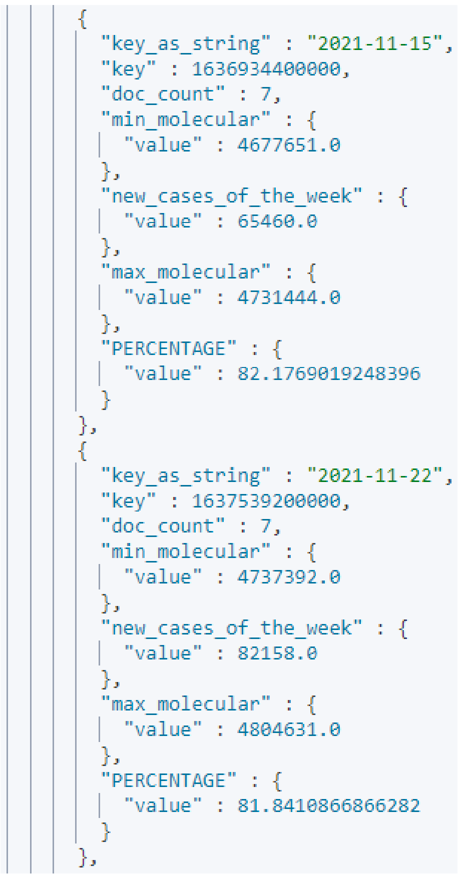


```
GET /tests/_search
{
  "size": 0,
  "query": {
    "bool": {
      "must": [
        {"range":{
          "data": {
            "gte":"2021-11-01",
            "lte": "2021-11-30"
          }
        }}
      ]
  }},
  "aggs": {
    "weeks groups": {
      "date_histogram": {
        "field": "data",
        "interval": "week",
        "format": "yyyy-MM-dd"
      },
      "aggs": {
        "max_molecular": {
          "max": {"field": "totale_positivi_test_molecolare"}
        },
        "min_molecular": {
          "min": {"field": "totale_positivi_test_molecolare"}
        },
        "new_cases_of_the_week":{
          "sum": {"field": "nuovi_positivi"}
        },
        "PERCENTAGE":{
          "bucket_script": {
            "buckets_path":  {
              "end_of_the_week":"max_molecular",
              "start_of_the_week":"min_molecular",
              "new_total_cases":"new_cases_of_the_week"
            },
            "script": "((params.end_of_the_week-params.start_of_the_week)/params
              .new_total_cases)*100"
          }
  }}}}
}
```

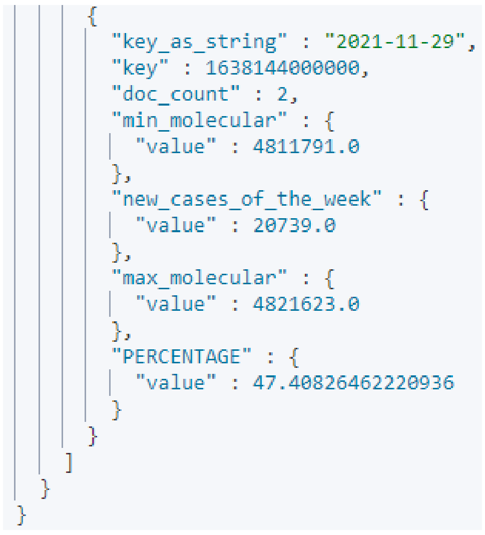
The outcome:



```
"aggregations" : {
  "weeks groups" : {
    "buckets" : [
      {
        "key_as_string" : "2021-11-01",
        "key" : 1635724800000,
        "doc_count" : 7,
        "min_molecular" : {
          "value" : 4595482.0
        },
        "new_cases_of_the_week" : {
          "value" : 36095.0
        },
        "max_molecular" : {
          "value" : 4625604.0
        },
        "PERCENTAGE" : {
          "value" : 83.4520016622801
        }
      },
      {
        "key_as_string" : "2021-11-08",
        "key" : 1636329600000,
        "doc_count" : 7,
        "min_molecular" : {
          "value" : 4629847.0
        },
        "new_cases_of_the_week" : {
          "value" : 51318.0
        },
        "max_molecular" : {
          "value" : 4672800.0
        },
        "PERCENTAGE" : {
          "value" : 83.69967652675474
        }
      },
```



```
      {
        "key_as_string" : "2021-11-15",
        "key" : 1636934400000,
        "doc_count" : 7,
        "min_molecular" : {
          "value" : 4677651.0
        },
        "new_cases_of_the_week" : {
          "value" : 65460.0
        },
        "max_molecular" : {
          "value" : 4731444.0
        },
        "PERCENTAGE" : {
          "value" : 82.1769019248396
        }
      },
      {
        "key_as_string" : "2021-11-22",
        "key" : 1637539200000,
        "doc_count" : 7,
        "min_molecular" : {
          "value" : 4737392.0
        },
        "new_cases_of_the_week" : {
          "value" : 82158.0
        },
        "max_molecular" : {
          "value" : 4804631.0
        },
        "PERCENTAGE" : {
          "value" : 81.8410866866282
        }
      },
```



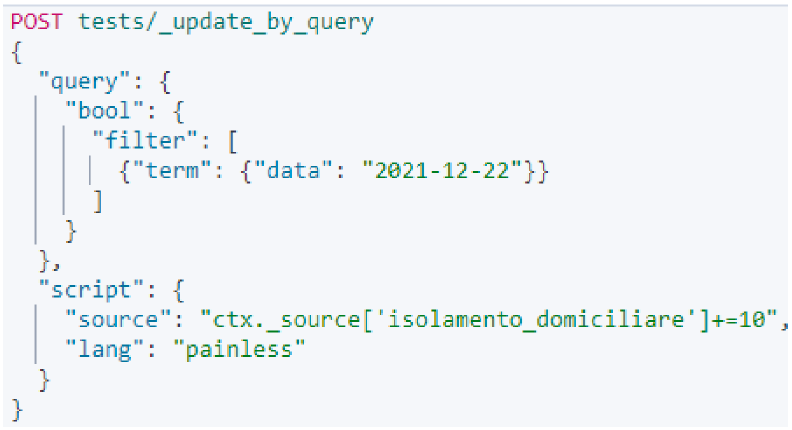
```
      {
        "key_as_string" : "2021-11-29",
        "key" : 1638144000000,
        "doc_count" : 2,
        "min_molecular" : {
          "value" : 4811791.0
        },
        "new_cases_of_the_week" : {
          "value" : 20739.0
        },
        "max_molecular" : {
          "value" : 4821623.0
        },
        "PERCENTAGE" : {
          "value" : 47.40826462220936
        }
      }
    ]
  }
}
```

In this query, first we filter the documents with the query operation, selecting only those which have a date in the range 01/11/21-30-11-21. Then we partition the documents in weeks with the operator ‘date_histogram’, selecting ‘week’ as the interval. Then we compute the parameters that we will need to calculate the wanted percentage: the maximum/minimum numbers of molecular tests (because they are cumulative fields, that track the total numbers from the beginning of the pandemic and we will need to do maximum-minimum to obtain the test executed in that week) and the new positive cases of the week (using a simple ‘sum’ operator). Then we can compute the wanted percentage with a ‘bucket_script’ operator that allows us to perform the desired calculation: *(maximum number of molecular tests-minimum number of molecular tests/total number of new cases of the week)*100*.

7.3. Commands

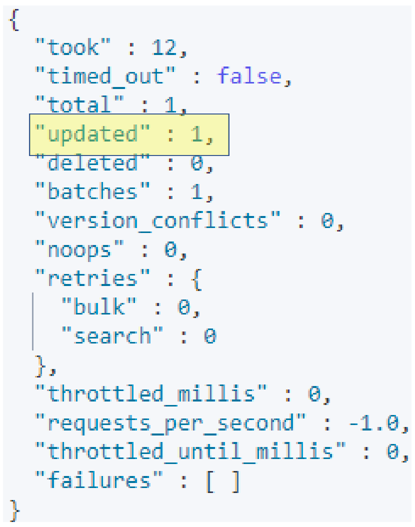
Command #1. Assuming there was an error in the registration of the document of 22-12-2021, add 10 people in home isolation for that date.

The command:



```
POST tests/_update_by_query
{
  "query": {
    "bool": {
      "filter": [
        {"term": {"data": "2021-12-22"}}
      ]
    }
  },
  "script": {
    "source": "ctx._source['isolamento_domiciliare']+=10",
    "lang": "painless"
  }
}
```

The outcome:

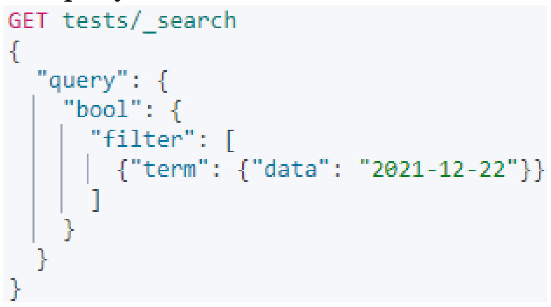


```
{
  "took" : 12,
  "timed_out" : false,
  "total" : 1,
  "updated" : 1,
  "deleted" : 0,
  "batches" : 1,
  "version_conflicts" : 0,
  "noops" : 0,
  "retries" : {
    "bulk" : 0,
    "search" : 0
  },
  "throttled_millis" : 0,
  "requests_per_second" : -1.0,
  "throttled_until_millis" : 0,
  "failures" : [ ]
}
```

In this command, first we select the document relating to the desired day through the “query” operation, filtering by the wanted day (‘*data*’). Then, with the ‘script’ operation we change the wanted fields.

We can easily see the effectiveness of the command by running this simple query before and after the command.

The query:



```
GET tests/_search
{
  "query": {
    "bool": {
      "filter": [
        {"term": {"data": "2021-12-22"}}
      ]
    }
  }
}
```

Output before the command:

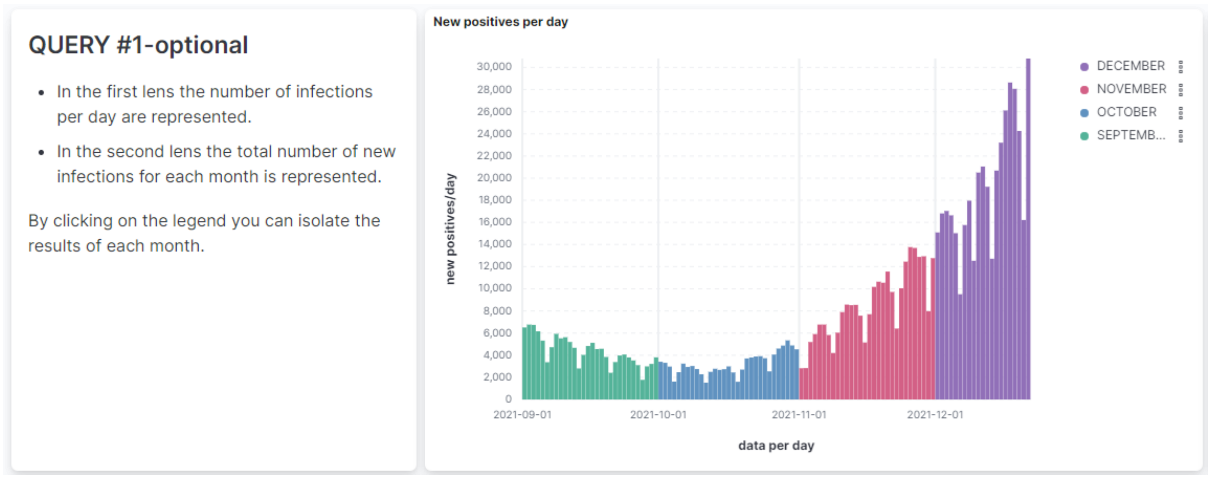
```
"hits" : [
    {
      "_index" : "tests",
      "_type" : "_doc",
      "_id" : "IhpFDH4BsFwopzhhIkUj",
      "_score" : 0.0,
      "_source" : {
        "totale_ospedalizzati" : 9554,
        "totale_positivi_test_antigenico_rapido" :
293155,
        "totale_casi" : 5472469,
        "data" : "2021-12-22T17:00:00",
        "variazione_totale_positivi" : 18585,
        "casi_testati" : 39766135,
        "totale_positivi" : 402729,
        "ricoverati_con_sintomi" : 8544,
        "tamponi_test_molecolare" : 71055692,
        "isolamento_domiciliare" : 393175,
        "deceduti" : 136077,
        "dimessi_guariti" : 4933663,
        "totale_positivi_test_molecolare" : 5179314,
        "@timestamp" :
"2021-12-22T17:00:00.000+01:00",
        "tamponi" : 132337126,
        "nuovi_positivi" : 36293,
        "tamponi_test_antigenico_rapido" : 61281434,
        "ingressi_terapia_intensiva" : 92,
        "terapia_intensiva" : 1010
      }
    }
  ]
```

Output after the command:

```
"hits" : [
    {
      "_index" : "tests",
      "_type" : "_doc",
      "_id" : "IhpFDH4BsFwopzhhIkUj",
      "_score" : 0.0,
      "_source" : {
        "totale_ospedalizzati" : 9554,
        "totale_positivi_test_antigenico_rapido" :
293155,
        "totale_casi" : 5472469,
        "data" : "2021-12-22T17:00:00",
        "variazione_totale_positivi" : 18585,
        "casi_testati" : 39766135,
        "totale_positivi" : 402729,
        "ricoverati_con_sintomi" : 8544,
        "tamponi_test_molecolare" : 71055692,
        "isolamento_domiciliare" : 393185,
        "deceduti" : 136077,
        "dimessi_guariti" : 4933663,
        "totale_positivi_test_molecolare" : 5179314,
        "@timestamp" :
"2021-12-22T17:00:00.000+01:00",
        "tamponi" : 132337126,
        "nuovi_positivi" : 36293,
        "tamponi_test_antigenico_rapido" : 61281434,
        "ingressi_terapia_intensiva" : 92,
        "terapia_intensiva" : 1010
      }
    }
  ]
```

7.4. Dashboard

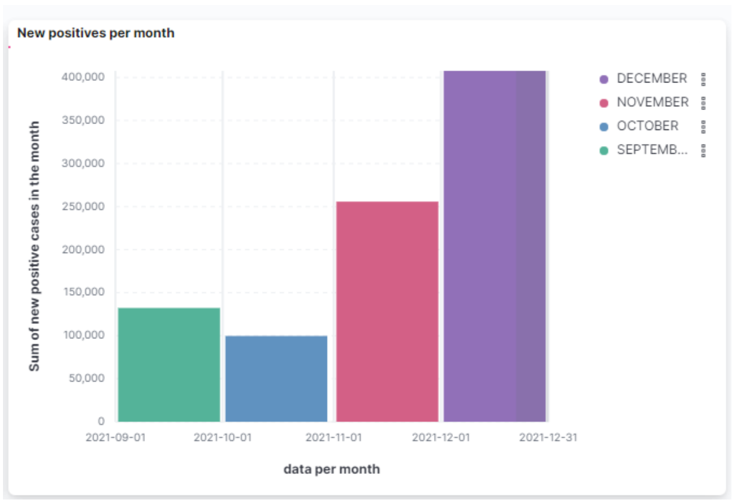
A small dashboard was also made using this database. We focused mainly on the 4 queries in paragraph 7.2.
Below we report a screenshot of the four main parts of this small dashboard. The selected period taken into consideration in these images is 01/09/2021-22/12/2021, which is the period covered in the database.
Note that for each panel you can change the time range by selecting Options>Customize time range if you are interested in a particular period.

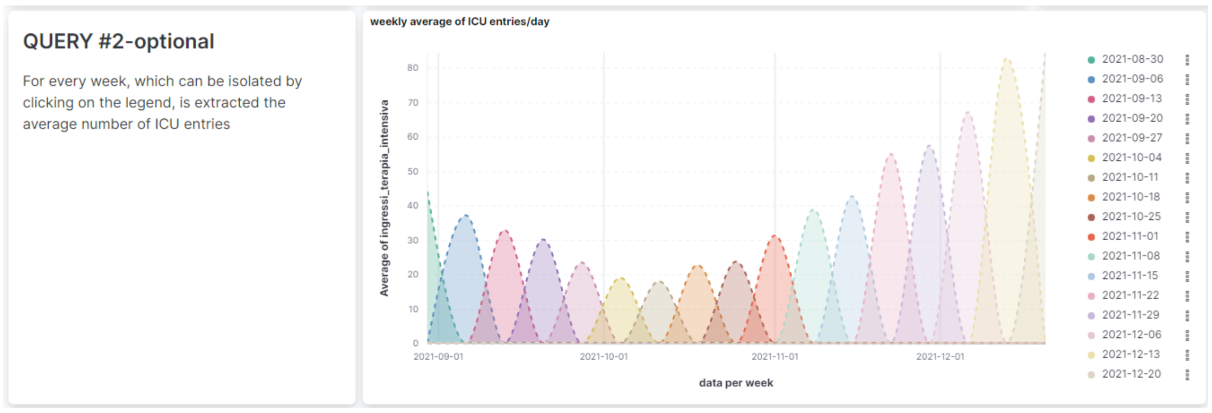


QUERY #1-optional

- In the first lens the number of infections per day are represented.
- In the second lens the total number of new infections for each month is represented.

By clicking on the legend you can isolate the results of each month.



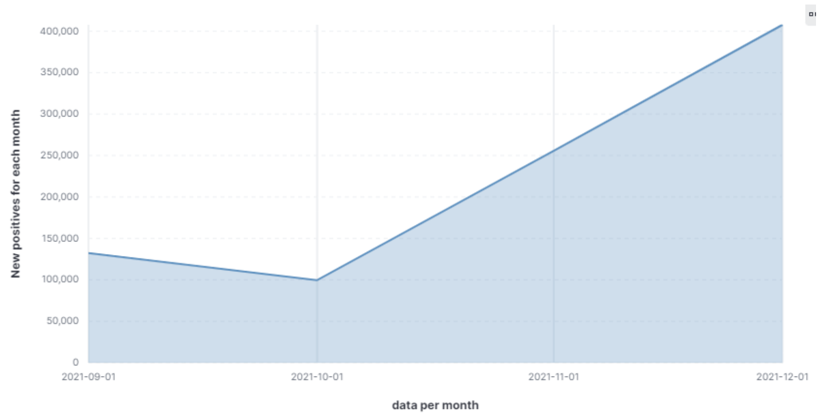


QUERY #2-optional

For every week, which can be isolated by clicking on the legend, is extracted the average number of ICU entries

QUERY #3-optional

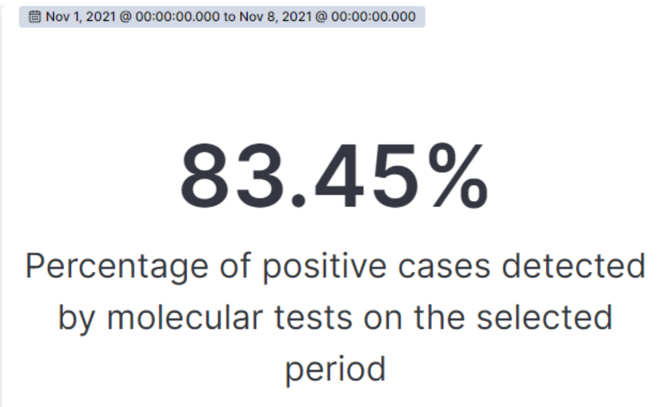
For every month, the total number of vaccines done and the number of people who have been infected during that month are extracted.



QUERY #4-optional

This lens extract the percentage of new cases detected by molecular tests.

To change the selected period click on the actual period (in grey) and customize.

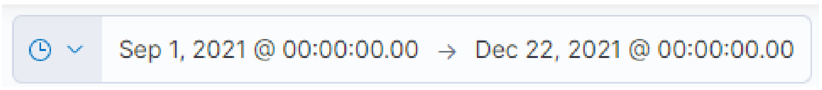


Nov 1, 2021 @ 00:00:00.000 to Nov 8, 2021 @ 00:00:00.000

83.45%

Percentage of positive cases detected by molecular tests on the selected period

If you want to select a period for the whole dashboard without having to change all the individual panels you can use the following slot to set the desired period. Since all the graphs have a period of time on the abscissa we thought that the setting of dynamic filters was not significant in this case: the most significant statistics can be done by changing the selected period.



Sep 1, 2021 @ 00:00:00.00 → Dec 22, 2021 @ 00:00:00.00