POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Crime reports and news in the U.S - some queries

Project Report
Systems and Methods for Big
and Unstructured Data

Authors:

Eleonora Giudici -
Luca Andrulli -
Luca Gerin -

Academic Year: 2023-24

Contents

1 Introduction

1.1. Objective

This project focuses on exploring compelling datasets to derive valuable insights applicable in real-world scenarios. The first dataset of interest is about the official crime records reported in Los Angeles from 2020 to 2023, while the second is about news headlines from 2012 to 2022 mainly from HuffPost.

1.2. Method

To obtain useful insights from the selected datasets, the first step is to define what are the questions to ask to the data and what are the expected results. This is crucial because the latter choice of the database technology to employ will be based on the requirements of the queries that we will need to run.
Then, two different technologies are picked according to their capabilities in terms of query solving and expression.
Upon importing the data, we initiated basic queries to assess its quality. Our goal isn't flawless data quality, nor do we aim to achieve it through extensive cleaning. Rather, we aim to utilize datasets with minimal inconsistencies, ensuring the reliability of our query outcomes.
The queries to address our research questions are then written and the results are obtained.

1.3. Technology choices

For analyzing the dataset about crimes, we've opted for MongoDB as our technology of choice. Its strength lies in facilitating the exploration of documents that reflect real business entities, offering a flexible schema accommodating varying structures—where certain attributes might be present in only a subset of the documents, like in our case.
To query the dataset about the news, instead, we chose to use Elasticsearch, that enables us to perform text search, that we believe useful for our purposes and for the kind of queries we want to run.

2 | Dataset

2.1. Crimes in LA dataset

The first dataset of choice contains the official crime records from Los Angeles City from January 2020 to December 2023.
It contains a total of 852250 datapoints, and presents the following 25 attributes:

- division_number: the division number of the crime incident. (type: int)
- date_reported: the date when the crime was reported. (type: datetime64[ns])
- date_occurred: the date when the crime occurred. (type: datetime64[ns])
- area: the area code where the crime incident occurred. (type: int)
- area_name: the name of the area where the crime incident occurred. (type: object)
- reporting_district: the reporting district of the crime incident. (type: int)
- part: the part number of the crime incident report, it can be either 1 or 2. (type: int)
- crime_code: the code corresponding to the crime type. (type: int)
- crime_description: the description of the crime. (type: object)
- modus_operandi: the modus operandi (method of operation) used in the crime. (type: object)
- victim_age: the age of the victim. (type: int)
- victim_sex: the gender of the victim. (type: object)
- victim_descent: the ethnic descent of the victim. (type: object)
- premise_code: the code representing the premises where the crime occurred. (type: object)
- premise_description: the description of the premises where the crime occurred. (type: object)
- weapon_code: the code representing the weapon used in the crime (if applicable). (type: object)
- weapon_description: the description of the weapon used in the crime (if applicable). (type: object)
- status: the status of the crime incident. (type: object)
- status_description: the description of the status of the crime incident. (type: object)
- crime_code_1: additional crime code 1 (if applicable). (type: object)
- crime_code_2: additional crime code 2 (if applicable). (type: object)
- location: the location of the crime incident. (type: object)
- cross_street: the cross street of the crime incident (if applicable). (type: object)
- latitude: the latitude coordinate of the crime incident location. (type: float64)
- longitude: the longitude coordinate of the crime incident location. (type: float64)

Not all the documents possess all the 25 attributes listed, but they have a varying structure. For example, the modus_operandi is absent in 14% of the documents, and also the weapon_code can be missing if a crime involves no weapon, while multiple crime codes may be present if there are more crimes to report together.
This is, as mentioned, one of the reasons why MongoDB was the choice, because it natively supports documents with heterogeneous structure and if in the future we decided to add a new document with a new field, like for example a "crime_code_3", if necessary, we would not need to perform any additional operation.

For the sake of our analysis, all the documents have been gathered inside the collection *LAcrimes_collection* on which queries will be executed.

An example of an instance of a document can be seen in image 2.1:


```
_id: ObjectId('658de95c363dd5e8b44a3624')
division_number: 10304468
date_reported: 2020-01-08T00:00:00.000+00:00
date_occurred: 2020-01-08T21:30:00.000+00:00
area: 3
area_name: "Southwest"
reporting_district: 377
part: 2
crime_code: 624
crime_description: "BATTERY - SIMPLE ASSAULT"
modus_operandi: "0444 0913"
victim_age: 36
victim_sex: "F"
victim_descent: "B"
premise_code: 501
premise_description: "SINGLE FAMILY DWELLING"
weapon_code: 400
weapon_description: "STRONG-ARM (HANDS, FIST, FEET OR BODILY FORCE)"
status: "AO"
status_description: "Adult Other"
crime_code_1: 624
location: "1100 W  39TH                          PL"
latitude: 34.0141
longitude: -118.2978
```


Figure 2.1: LAcrimes_collection instance document.

2.2. News headlines dataset

The second dataset picked for our scope is about news headlines from 2012 to 2022, mainly from HuffPost. It contains around 210k data points, most of which are dated before 2018.
The fields of each entry are the followings:

- category: category in which the article was published.
- headline: the headline of the news article.
- authors: list of authors who contributed to the article.
- link: link to the original news article.
- short_description: abstract of the news article.
- date: publication date of the article.

As mentioned, we chose to exploit Elasticsearch to run queries on this dataset, because of its text search capabilities that will come in handy when querying fields such as the *authors* in which more than one person's name is contained and most importantly the *headline* and *description* of the articles.

The mapping we decided to employ is the following:

```
{
  "properties": {
    "@timestamp": {
      "type": "date"
    },
    "authors": {
      "type": "text",
      "fields": {
          "keyword": {
            "type": "keyword"
          }
      }
    },
    "category": {
      "type": "keyword"
    },
    "date": {
      "type": "date",
      "format": "iso8601"
    },
    "headline": {
      "type": "text"
    },
    "link": {
      "type": "text"
    },
    "short_description": {
      "type": "text"
    }
  }
}
```

The choices we made for the mapping are the followings:

- category: is a keyword; there are 42 categories in total.
- headline: is a text field intended for text search and non-exact matching due to its nature.
- authors: is a text field also intended for text search and non-exact matching. It may contain multiple authors separated by "and" with various abbreviations used, leading to different representations. This field includes a sub-field called *keyword* of type keyword, utilized for grouping purposes.
- link: is a text field suitable for text search, containing references to different journals at times.
- short_description: is a text field aimed at text search and non-exact matching.
- date: is a field storing dates in the ISO8601 format.

An example of a document is in image 2.2.


```
"_source": {
  "date": "2022-09-23",
  "short_description": "Health experts said it is too early to predict whether demand would match up with the
    171 million doses of the new boosters the U.S. ordered for the fall.",
  "@timestamp": "2022-09-23T00:00:00.000+02:00",
  "link": "https://www.huffpost.com/entry/covid-boosters-uptake-us_n_632d719ee4b087fae6feaac9",
  "category": "U.S. NEWS",
  "headline": "Over 4 Million Americans Roll Up Sleeves For Omicron-Targeted COVID Boosters",
  "authors": "Carla K. Johnson, AP"
}
```


Figure 2.2: news_headlines instance document.

3 Queries

3.1. Crimes in LA - queries

3.1.1. Last 10 crimes against male children

This query selects the most recent 10 crime reports where the victim is a male children, with age between 3 and 10.

```
LAcrimes.LAcrimes_collection.find(
    {
        "victim_sex": "M",
        "victim_age": { "$gte": 3, "$lte": 10 }
    }
).sort({"date_occurred": -1}).limit(10)
```

The query is searching for documents where the victim sex is "M" (Male) and the victim age is between 3 and 10 years. Documents are sorted in descending order by "date_occurred" to retrieve the most recent crimes, and the output is limited to the first 10 results. The results are shown in figure 3.1

Crimes_collection

	crime_code Int32	crime_description String	victim_age Int32	victim_sex String	premise_code Double	premise_description String
1	627	"CHILD ABUSE (PHYSICAL) - SIM…	9	"M"	502	"MULTI-UNIT DWELLING (APARTME…
2	627	"CHILD ABUSE (PHYSICAL) - SIM…	7	"M"	501	"SINGLE FAMILY DWELLING"
3	627	"CHILD ABUSE (PHYSICAL) - SIM…	4	"M"	501	"SINGLE FAMILY DWELLING"
4	624	"BATTERY - SIMPLE ASSAULT"	7	"M"	704	"ELEMENTARY SCHOOL"
5	627	"CHILD ABUSE (PHYSICAL) - SIM…	7	"M"	101	"STREET"
6	237	"CHILD NEGLECT (SEE 300 W.I.C…	4	"M"	502	"MULTI-UNIT DWELLING (APARTME…
7	627	"CHILD ABUSE (PHYSICAL) - SIM…	8	"M"	501	"SINGLE FAMILY DWELLING"
8	930	"CRIMINAL THREATS - NO WEAPON…	4	"M"	501	"SINGLE FAMILY DWELLING"

Figure 3.1: query 3.1.1 results.

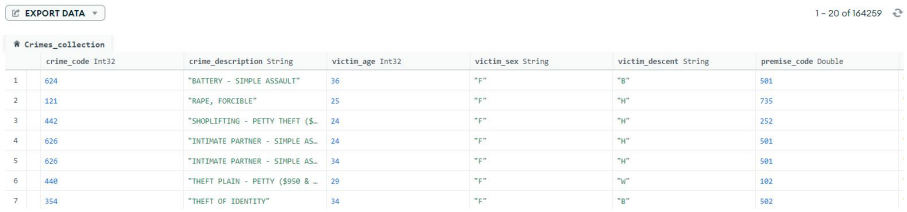
3.1.2. Number of adult female victims

This query wants to count the number of crimes committed against victims whose gender is "female" and whose age is comprised between 21 and 40.

```
LAcrimes.LAcrimes_collection.countDocuments({
    "$and": [
        {"victim_sex": "F"},
        {"victim_age": {"$gte": 21, "$lte": 40}}
    ]
})
```

The query makes use of the *countDocuments()* function that returns the number of documents matching the pattern provided, that in this case is composed by the conditions specified above put together with an AND

logic operator.
The results of a *find()* query with equal pattern to match are shown in figure 3.2


EXPORT DATA 1 - 20 of 164259

Crimes_collection

	crime_code Int32	crime_description String	victim_age Int32	victim_sex String	victim_descent String	premise_code Double
1	624	"BATTERY - SIMPLE ASSAULT"	36	"F"	"B"	501
2	121	"RAPE, FORCIBLE"	25	"F"	"H"	735
3	442	"SHOPLIFTING - PETTY THEFT ($…	24	"F"	"H"	252
4	626	"INTIMATE PARTNER - SIMPLE AS…	24	"F"	"H"	501
5	626	"INTIMATE PARTNER - SIMPLE AS…	34	"F"	"H"	501
6	440	"THEFT PLAIN - PETTY ($950 & …	29	"F"	"W"	102
7	354	"THEFT OF IDENTITY"	34	"F"	"B"	502

Figure 3.2: query 3.1.2 results.

3.1.3. Location and victim age of crimes happened in some coordinates range

This query will retrieve the premise description and age of all crime reports that refer to crimes happened in neighborhood “woodland hills”, that is comprised between the latitude 34.12 to 34.20 and the longitude from -118.65 to -118.55, and will sort the results by victim age in descending order.

```
LAcrimes.LAcrimes_collection.find(
    {"$and": [
        {"latitude": {"$gte": 34.12}},
        {"latitude": {"$lte": 34.2}},
        {"longitude": {"$gte": -118.65}},
        {"longitude": {"$lte": -118.55}}
      ]
    },
    {"premise_description": 1, "victim_age": 1}
).sort({"victim_age": -1})
```

The query works by matching the document with the provided conditions on the coordinates, and then applies a projection to remove all the fields that are not of interest, and lastly orders the results using the victim_age attribute. The results are shown in figure 3.3


Crimes_collection

	_id ObjectId	victim_age Int32	premise_description String
1	ObjectId('658dea60363dd5e8b45…	98	"NURSING/CONVALESCENT/RETIREM…
2	ObjectId('658dea0c363dd5e8b45…	98	"MULTI-UNIT DWELLING (APARTME…
3	ObjectId('658de9f1363dd5e8b45…	98	"MULTI-UNIT DWELLING (APARTME…
4	ObjectId('658de9f0363dd5e8b45…	98	"VEHICLE, PASSENGER/TRUCK"
5	ObjectId('658de9eb363dd5e8b45…	97	"SINGLE FAMILY DWELLING"
6	ObjectId('658dea27363dd5e8b45…	97	"DRIVEWAY"
7	ObjectId('658dea28363dd5e8b45…	97	"NURSING/CONVALESCENT/RETIREM…
8	ObjectId('658de99c363dd5e8b44…	96	"SINGLE FAMILY DWELLING"
9	ObjectId('658de9fa363dd5e8b45…	96	"SINGLE FAMILY DWELLING"

Figure 3.3: query 3.1.3 results.

3.1.4. Crime code and area-based crime statistics

For each crime code, the query lists all the areas of Los Angeles ordered by the number of crimes with that code that happened in that area.

```
LAcrimes.LAcrimes_collection.aggregate([
    {"$group": {
        "_id": {"code": "$crime_code", "area": "$area_name"},
        "numCrimes": {"$sum": 1}
      }
    },
    {"$sort": {"numCrimes": -1}}
])
```

The purpose of this aggregation is to find and display the count of crimes for each unique combination of "crime_code" and "area_name", sorted by the number of crimes in descending order. The result will show the most common crime combinations first. The results are shown in figure 3.4

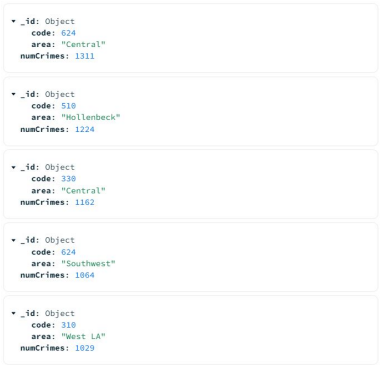


Figure 3.4: query 3.1.4 results.

3.1.5. Crimes with the same weapon

This query computes the number of crimes committed with the same weapon type in each area. Then it selects only the areas in which the number of crimes with that specific weapon is greater than 3.

```
LAcrimes.LAcrimes_collection.aggregate([
    {"$group": {
        "_id": {"type": "$weapon_code", "area": "$area_name"},
        "countCrimes": {"$sum": 1}
      }
    },
    {"$match": {"countCrimes": {"$gt": 3}}}
])
```

The aggregation pipeline is designed to find and count crimes grouped by the combination of "weapon_code" and "area_name," and then to filter those groups where the count of crimes is greater than 3.
The results are shown in figure 3.5

Figure 3.5: query 3.1.5 results.

3.1.6. Average victim age per crime with AO status

This query, for each kind of crime, computes the average age of the victims with crime reports whose status is 'AO'.

```
LAcrimes.LAcrimes_collection.aggregate([
    {"$match": {"status": {"$eq": "AO"}}},
    {"$group": {
        "_id": {
            "crime_code": "$crime_code",
            "crime_description": "$crime_description"
        },
        "average_victim_age": {"$avg": "$victim_age"}
      }
    }
])
```

First, all the documents whose status is not 'AO' are discarded, and then a grouping is made utilizing the kind of crime as discriminant and for each group the average victim age is computed by averaging the victim_age fields of the documents of each group. The results are shown in figure 3.6

3.1.7. Top 6 districts for number of open investigations

This query finds the 6 districts in the city that have the highest number of currently open investigations.

```
LA_crime.LA_crime_collection.aggregate([
    {"$match": {"status": "IC"}},
    {"$group": {
        "_id": {"district": "$reporting_district"},
        "open_investigations": {"$sum": 1}
      }
    },
    {"$sort": {"open_investigations": -1}},
    {"$limit": 6}
])
```

Figure 3.6: query 3.1.6 results.

First, all the reports that are about investigations with status different from "investigation Continues" are discarded. Then, the remaining reports are grouped by the district and a sum of all the grouped crime reports for each district is computed. At this point, the results are sorted by descending number of open investigations and only the top 6 districts are kept in the output.
The results are shown in figure 3.7

Figure 3.7: query 3.1.7 results.

3.1.8. Crimes against women each year in each area

The query computes the number of female victims per year for each area.

```
LAcrimes.LAcrimes_collection.aggregate([
    {‘‘$match’’: {‘‘victim_sex’’: "F"}},
    {‘‘$project’’: {‘‘year’’: {‘‘$year’’: "$date_occurred"},
                   ‘‘area’’: "$area_name’’}
    },
    {‘‘$group’’: {
        ‘‘_id’’: { ‘‘area’’: "$area",’’year’’: "$year" },
        ‘‘crimesOnFemale’’: {‘‘$sum’’: 1}
      }
    }
])
```

This aggregation pipeline first filters documents with a female victim, then projects a new document by only keeping the "year" and "area" fields. While execting this projection, the year is extracted from the "date_occurred" field. Finally, the documents are grouped by "area" and "year", and the number of crimes on females in each group is counted.
The results are shown in figure 3.8

Figure 3.8: query 3.1.8 results.

3.1.9. Number of crime codes in most dangerous areas

This query will count the number of distinct crime codes that have been reported in each of the areas in which the number of different reported crime codes is higher than 90. This way, we will find the areas that present the bigger diversity in terms of crimes committed.

```
LAcrimes.LAcrimes_collection.aggregate([
    {‘‘$group’’: {
        "_id": {"area": "$area", "crime code":"$crime_code"}
      }
    },
    {‘‘$group’’: {
        "_id": {"area_code": "$_id.area"},
        "different_crime_codes_per_area": {"$sum":1}
      }
    },
    {‘‘$match’’: { "different_crime_codes_per_area": {"$gt": 90}}}
])
```

This query works by first aggregating the crime reports by area and crime code, forming groups composed by all the found couples (area, crime_code), and then makes a second aggregation on these first groups by grouping together all of those which happened the same area, while counting how many are put together. At the end, the results that have a count of crime codes that is less than a threshold are filtered out.
The results are shown in figure 3.9

Figure 3.9: query 3.1.9 results.

3.1.10. Crimes gender-based

This query, for each kind of weapon, counts the number of crimes against females and males victims.

```
LAcrimes.LAcrimes_collection.aggregate([
    {"$group":{
        "_id":{"weapon_code":"$weapon_code", "weapon":"$weapon_description"},
        "femaleVictims":{
            "$sum":{"$cond":{
                "if":{"$eq":["$victim_sex","F"]},
                "then":1,
                "else":0
                }
            }
        },
        "maleVictims":{
            "$sum":{"$cond":{
                "if":{"$eq":["$victim_sex","M"]},
                "then":1,
                "else":0
                }
            }
        }
      }
    }
])
```

This query is grouping documents based on the "weapon_code" and "weapon_description" fields. It then calculates the count of crimes for each unique combination of weapon code and description, categorized by the number of female and male victims. The "$cond" expressions are conditional operators that in this case check whether the sex of the victims is "Male" (M) or "Female" (F), and increment the count accordingly. The results are shown in figure 3.10

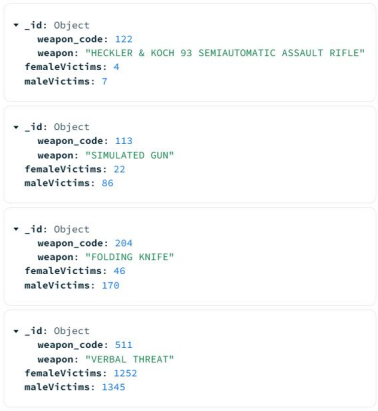


Figure 3.10: query 3.1.10 results.

3.2. News headlines - queries

3.2.1. David Moye talks about crimes

This query finds all the articles for a specific category and a specific author. In this case, we want to search for all the articles about crimes written only by David Moye alone.

```
GET news/_search
{
  "query": {
    "bool": {
      "must": [
        { "term": { "category": "CRIME" }},
        { "term": { "authors.keyword": "David Moye" }}
      ]
    }
  }
}
```

A "must" clause composed of two term queries is used to search for documents where the "category" field is "CRIME," and the "authors" field contains exactly the term "David Moye."
The results are shown in figure 3.11


```
"hits": {
  "total": {
    "value": 269,
    "relation": "eq"
  },
  "max_score": 16.13507,
  "hits": [
    {
      "_index": "news",
      "_id": "EIYC0YwBYy7or0GWAfuU",
      "_score": 16.13507,
      "_source": {
        "date": "2022-06-22",
        "short_description": "The woman placing the order asked restaurant employees to “call the police” and have
          them come with the delivery, but “please don’t make it obvious.”",
        "@timestamp": "2022-06-22T00:00:00.000+02:00",
        "link": "https://www.huffpost.com/entry/chipper-truck-cafe-grubhub-hostage_n_62b38196e4b04a617365121e",
        "category": "CRIME",
        "headline": "Woman Allegedly Being Held Hostage Gets Rescued Thanks To Grubhub Note",
        "authors": "David Moye"
      }
    },
    {
      "_index": "news",
      "_id": "h4cC0YwBYy7or0GWAwOZ",
      "_score": 16.13507,
      "_source": {
        "date": "2021-05-26",
        "short_description": """Witnesses said Virginia Brown yelled "no vaccine" while almost hitting seven people
          with her SUV.""",
        "@timestamp": "2021-05-26T00:00:00.000+02:00",
        "link": "https://www.huffpost.com/entry/virginia-brown-vaccination-site-protest_n_60aecbeae4b03135479fee17",
        "category": "CRIME",
        "headline": "Woman Protests COVID-19 Vaccine By Speeding Car Through Vaccination Site",
        "authors": "David Moye"
```


Figure 3.11: query 3.2.1 results.

3.2.2. Post-2019 articles containing the term 'COVID'

This query finds all articles which contain the word 'COVID' written from 01/01/2019 to the present day.

```
GET news/_search
{
  "query": {
    "bool": {
      "must": [
        {"match": {"headline": "COVID"}}
      ],
      "filter": [
        {
          "range": {"date": {"gte": "2019-01-01"}}}
      ]
    }
  }
}
```

This query is designed to retrieve documents in which the "headline" field contains the word 'COVID' and to remove from the results those without a date of publication that is after January 1, 2019. The results are shown in figure 3.12


```
"hits": {
  "total": {
    "value": 301,
    "relation": "eq"
  },
  "max_score": 7.38609,
  "hits": [
    {
      "_index": "news",
      "_id": "rIYC0YwBYy7or0GWAf2W",
      "_score": 7.38609,
      "_source": {
        "date": "2022-02-20",
        "short_description": """The 95-year-old British monarch has cold-like symptoms and "Will continue to receive
          medical attention," Buckingham Palace said.""",
        "@timestamp": "2022-02-20T00:00:00.000+01:00",
        "link": "https://www.huffpost.com/entry/queen-elizabeth-covid-19_n_62122eb7e4b06212585e8e84",
        "category": "POLITICS",
        "headline": "Queen Elizabeth Tests Positive For COVID-19",
        "authors": ""
      }
    },
    {
      "_index": "news",
      "_id": "WYcC0YwBYy7or0GWBAum",
      "_score": 7.38609,
      "_source": {
        "date": "2020-05-25",
        "short_description": "Despite silent streets and nearly nonexistent traffic, vehicle larcenies shot up 63% in
          New York and nearly 17% in Los Angeles from Jan. 1 through mid-May.",
        "@timestamp": "2020-05-25T00:00:00.000+02:00",
        "link": "https://www.huffpost.com/entry/car-thefts-spike-coronavir_n_5ecc08c6c5b6396df96b7b33",
        "category": "CRIME",
        "headline": "Car Thefts Spike During COVID-19 Pandemic",
        "authors": "Stefanie Dazio, AP"
```


Figure 3.12: query 3.2.2 results.

3.2.3. Number of articles about U.S. published by huffpost or new york times

This query returns the count of the headlines of the articles that are in the category "U.S. NEWS" and that have a reference to the *huffpost* or to the *new york times* in their link.

```
GET news/_count
{
  "query": {
    "bool": {
      "must": [
        {
          "match": {
            "link": {
              "query": "www.huffpost.com www.nytimes.com",
              "operator": "or"
            }
          }
        },
        {"term": {"category": "U.S. NEWS"}}
      ]
    }
  }
}
```

To obtain the results, the links of the huffpost and of the nytimes are searched in the analyzed “link” textual field, and the documents containing at least one of the two are kept. Moreover, it’s requested that all the documents have as category the value "U.S. NEWS".
The results are shown in figure 3.13

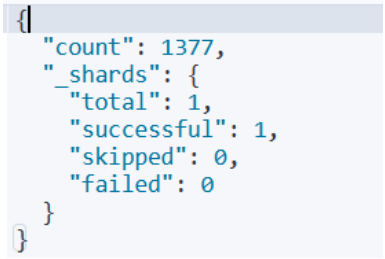


Figure 3.13: query 3.2.3 results.

3.2.4. Christmas without Santa Claus

With this query we want to select all the articles that talk about a particular topic, while avoiding certain characteristics. This is a specific example in which we want articles that talk about Christmas, giving priority to those that contain the words "present" and "gift", but we don't want them to talk about Santa Claus.

```
GET news/_search
{
  "query": {
    "bool": {
      "must": [
        {"match": {"short_description": "Christmas"}}
      ],
      "should": [
        {"match": {"short_description": "present"}},
        {"match": {"short_description": "gift"}}
      ],
      "must_not": [
        {"match": {"short_description": "Santa Claus"}}
      ]
    }
  }
}
```

The query is searching for documents where the "short_description" field contains the term "Christmas." Additionally, it favours documents containing either "present" or "gift" in the short description but excludes documents containing "Santa Claus." The "should" clause allows to check additional conditions, which are not mandatory but increase the overall score of the results, and the "must_not" clause excludes documents that match a specific criteria. The results are shown in figure 3.14


```
"hits": {
  "total": {
    "value": 535,
    "relation": "eq"
  },
  "max_score": 17.934711,
  "hits": [
    {
      "_index": "news",
      "_id": "c4kC0YwBYy7or0GWNSxY",
      "_score": 17.934711,
      "_source": {
        "date": "2013-12-25",
        "short_description": "May we humbly present our Christmas gift to you, our 10 favorite unorthodox depictions
          of JC himself. The following collection",
        "@timestamp": "2013-12-25T00:00:00.000+01:00",
        "link": "https://www.huffingtonpost.com/entry/jesus-art_us_5bb268e9e4b0171db6a048d5",
        "category": "CULTURE & ARTS",
        "headline": "The 10 Most Unorthodox Artistic Depictions Of Jesus",
        "authors": ""
      }
    },
    {
      "_index": "news",
      "_id": "C4cC0YwBYy7or0GWCTrZ",
      "_score": 17.07571,
      "_source": {
        "date": "2018-01-10",
        "short_description": "What a colorful Christmas gift.",
        "@timestamp": "2018-01-10T00:00:00.000+01:00",
        "link": "https://www.huffingtonpost.com/entry/man-girlfriend-drawing-animation
          -styles_us_5a565fe5e4b08a1f624ad6ce",
        "category": "ARTS & CULTURE",
        "headline": "Man Surprises Girlfriend By Drawing Them In Different Animation Styles",
        "authors": "Elyse Wanshel"
```


Figure 3.14: query 3.2.4 results.

3.2.5. Covid articles about vaccines or boosters in "omicron" period

This query finds all articles which contain the word 'COVID' prioritizing those that talk about "Omicron" and immediately after those that talk about vaccines or boosters.

```
GET news/_search
{
  "query": {
    "bool": {
      "must": [
        {"match": {"headline": "COVID"}}
      ],
      "should": [
        {"match": {"short_description": "vaccines vaccine boosters booster"}},
        {
          "match": {
            "short_description": {
              "query": "Omicron",
              "boost": 2
            }
          }
        }
      ]
    }
  }
}
```

The query is designed to search articles whose headlines contain the term "COVID" ("must" condition). Additionally, it uses the "should" clause to rank the results, with two optional conditions:

- Articles with "short description" field containing any of the terms "vaccines," "vaccine," "boosters," or "booster";
- Articles with "short description" field containing the term "Omicron," with a boost factor of 2.

The words are specified both in singular form and in plural form because the analyzer in use makes a distinction between them, and we are not interested in this distinction but just in the presence of the words, whatever the number.
The results are shown in figure 3.15

```
"hits": [
  {
    "_index": "news_headlines",
    "_id": "-9q3T4wBvJTkmD6J7kxL",
    "_score": 25.313404,
    "_source": {
      "date": "2021-12-13",
      "short_description": "The doctor said Sunday the U.S. had all the tools needed to “protect ourselves” amid
        the dual threat from the delta and omicron variants.",
      "@timestamp": "2021-12-13T00:00:00.000+01:00",
      "link": "https://www.huffpost.com/entry/fauci-omicron-optimally-protected_n_61b6bcbfe4b068effecc47f1",
      "category": "POLITICS",
      "headline": "Fauci Says Those Who Want To Be 'Optimally Protected' Should Get COVID-19 Booster",
      "authors": "Nick Visser"
    }
  },
  {
    "_index": "news_headlines",
    "_id": "sdq3T4wBvJTkmD6J7kxL",
    "_score": 24.965158,
    "_source": {
      "date": "2021-12-27",
      "short_description": "The president changed his tone after previously going on the defense in response to
        questions about the lack of testing in the face of the omicron variant.",
      "@timestamp": "2021-12-27T00:00:00.000+01:00",
      "link": "https://www.huffpost.com/entry/biden-acknowledges-effort-expand-covid-testing-not
        -enough_n_61ca3d3ae4b04b42ab6d8ab9",
      "category": "POLITICS",
      "headline": "Biden Acknowledges Federal Effort To Expand COVID Testing ‘Clearly Not Enough’",
      "authors": "Sanjana Karanth"
    }
  },
```


Figure 3.15: query 3.2.5 results.

3.2.6. Recent Jon Gambrell's articles

Let's assume that news readers are interested in articles written by a particular author in a specific year, with a preference for more recent content. The following query serves as an example: it retrieves all articles written by Jon Gambrell after the year 2021, assigning a higher score to those published in the year 2022."

```
GET news/_search
{
  "query": {
    "bool": {
      "must": [
        {"match":{"authors": "Jon Gambrell"}},
        {"range": {"date": {"gte": "2021-01-01||/y"}}}
      ],
      "should": [
        {
          "range": {
            "date": {
              "gte": "2022-01-01||/y",
              "lte": "2022-12-31||/y",
              "boost": 3.0
            }
          }
        }
      ]
    }
  }
}
```

The query is searching for documents where the "authors" field contains "Jon Gambrell" and the "date" field is greater than or equal to January 1, 2021. Additionally, it provides a boost to documents with a "date" field within the range of January 1, 2022, to December 31, 2022. The "should" clause allows for additional conditions, and the boosting emphasizes the importance of documents matching the second range condition. (The || syntax, is used to round the date to the beginning of the specified interval)
The results are shown in figure 3.16

```
"hits": {
  "total": {
    "value": 10,
    "relation": "eq"
  },
  "max_score": 22.412544,
  "hits": [
    {
      "_index": "news",
      "_id": "Q4YC0YwBYy7or0GWAfmT",
      "_score": 22.412544,
      "_source": {
        "date": "2022-09-20",
        "short_description": "The concerted and quickening Kremlin-backed efforts to swallow up four regions could
          set the stage for Moscow to escalate the war.",
        "@timestamp": "2022-09-20T00:00:00.000+02:00",
        "link": "https://www.huffpost.com/entry/russian-controlled-ukrainian-regions
          -referendum_n_6329d53ae4b07198f012f023",
        "category": "WORLD NEWS",
        "headline": "4 Russian-Controlled Ukrainian Regions Schedule Votes This Week To Join Russia",
        "authors": "Jon Gambrell, AP"
      }
    },
    {
      "_index": "news",
      "_id": "DYYC0YwBYy7or0GWAfqU",
      "_score": 22.412544,
      "_source": {
        "date": "2022-08-15",
        "short_description": "While Iran hasn't focused on the writer in recent years, a decades-old fatwa demanding
          his killing still stands.",
        "@timestamp": "2022-08-15T00:00:00.000+02:00",
        "link": "https://www.huffpost.com/entry/iran-denies-involvement-but-justifies-salman-rushdie
          -attack_n_62fa275ee4b095e78881df24",
        "category": "WORLD NEWS"
```


Figure 3.16: query 3.2.6 results.

3.2.7. Most frequent category

This query selects the category with the highest number of correlated documents to return the most covered topic.

```
GET news/_search
{
  "size": 0,
  "aggs": {
    "agg_for_category": {
      "terms": {
        "field": "category",
        "size": 1,
        "order": {
          "_count": "desc"
        }
      }
    }
  }
}
```

The query consists in an aggregation on the "category" field, aimed at returning only the top category (the one with the highest count of documents). The initial size parameter is set to 0, indicating that no actual documents should be returned, only the aggregation results.
The aggregation is specified using the "aggs" key and documents are grouped based on the values in the "category" field. The resulting groups are ordered based on the count of documents for each category in descending order and a second *size* parameter is set to 1 to limit the number of resulting aggregations. As a consequence of this process, only the most frequent category will be displayed.
The results are shown in figure 3.17

```
"aggregations": {
  "agg_for_category": {
    "doc_count_error_upper_bound": 0,
    "sum_other_doc_count": 173925,
    "buckets": [
      {
        "key": "POLITICS",
        "doc_count": 35602
      }
    ]
  }
}
```


Figure 3.17: query 3.2.7 results.

3.2.8. Articles about Biden or Trump

This query counts the number of articles for each author and ranks them by preferring those that contain in the description the words “Biden” or “Trump”.

```
GET news/_search
{
  "size": 0,
  "query": {
    "bool": {
      "should": [
        { "match": { "short_description": "Trump" } },
        { "match": { "short_description": "Biden" } }
      ]
    }
  },
  "aggs": {
    "authors": {
      "terms": {
        "field": "authors.keyword"
      }
    }
  }
}
```

The query specifies that the short description of the articles should contain either the term "Trump" or "Biden" using the "should" clause. Additionally, it includes an aggregation to count the number of articles per author. The results are shown in figure 3.18

```
"aggregations": {
  "authors": {
    "doc_count_error_upper_bound": 0,
    "sum_other_doc_count": 3236,
    "buckets": [
      {
        "key": "",
        "doc_count": 487
      },
      {
        "key": "Lee Moran",
        "doc_count": 192
      },
      {
        "key": "Mary Papenfuss",
        "doc_count": 184
      },
      {
        "key": "Marina Fang",
        "doc_count": 96
      },
      {
        "key": "Ed Mazza",
        "doc_count": 81
      },
      {
        "key": "S.V. Date",
        "doc_count": 76
      },
      {
        "key": "Igor Bobic",
        "doc_count": 68
      },
```


Figure 3.18: query 3.2.8 results.

3.2.9. Top 10 authors of 2022: monthly article count analysis

This query counts the number of articles written per month by the top 10 authors that have written more articles in 2022.

```
GET news/_search
{
  "size": 0,
  "query": {
    "bool": {
      "filter": [
        {
          "range": {
            "date": {
              "gte": "2022-01-01",
              "lte": "2022-12-31"
            }
          }
        }
      ]
    }
  },
  "aggs": {
    "article_per_author": {
      "terms": {
        "field": "authors.keyword",
        "size": 10
```

```
      },
      "aggs": {
        "monthly_articles": {
          "date_histogram": {
            "field": "date",
            "calendar_interval": "month"
          }
        }
      }
    }
  }
}
```

The query initially filters documents written between 01/01/2022 and 31/12/2022, then there is an aggregation in which each author is associated with the number of documents written by him, retrieving top 10. Of these, there is a "date_histogram" aggregation too, which is used for analyzing time-based data. The document will be grouped on the "date" field with a calendar interval set to "month," counting the number of articles for each month. The results are shown in figure 3.19


```
"aggregations": {
  "article_per_author": {
    "doc_count_error_upper_bound": 0,
    "sum_other_doc_count": 890,
    "buckets": [
      {
        "key": "",
        "doc_count": 140,
        "monthly_articles": {
          "buckets": [
            {
              "key_as_string": "2022-01-01T00:00:00.000Z",
              "key": 1640995200000,
              "doc_count": 7
            },
            {
              "key_as_string": "2022-02-01T00:00:00.000Z",
              "key": 1643673600000,
              "doc_count": 22
            },
            {
              "key_as_string": "2022-03-01T00:00:00.000Z",
              "key": 1646092800000,
              "doc_count": 23
            },
            {
              "key_as_string": "2022-04-01T00:00:00.000Z",
              "key": 1648771200000,
              "doc_count": 15
            },
            {
              "key_as_string": "2022-05-01T00:00:00.000Z",
              "key": 1651363200000,
              "doc_count": 17
            },
```


Figure 3.19: query 3.2.9 results.

3.2.10. Daily article count per author combination: in the current Month but 3 years ago

This query returns the list of all articles that have been written this month but 3 years ago, grouping them by the combination of authors that wrote them and then by date.

```
GET news/_search
{
  "size": 10,
  "query": {
    "range": {
      "date": {
        "gte": "now-3y/M",
        "lte": "now-3y/M+1M"
      }
    }
  },
  "aggs": {
    "articles_per_author": {
      "terms": {
        "field": "authors.keyword"
      },
      "aggs": {
        "articles_per_day": {
          "terms": {
            "field": "date"
          }
        }
      }
    }
  }
}
```

To obtain the desired results, first a range query is run to get all the articles published in the desired time of the month starting 3 years ago from the current moment. Then, an aggregation query is performed, using the sub-field authors.keyword of the field author to group by the combinations of authors writing the article. The choice to use this sub-field is motivated by the fact that using the textual field "authors" would produce insignificant results, because it would for example create a group of all articles having the word "and" in the authors field, or single names that may belong to more authors. At the end, a second aggregation is performed on the date to group again the articles in each already existing group by date of publication.
The results are shown in figure 3.20

```
},
{
  "key": "Mary Papenfuss",
  "doc_count": 21,
  "articles_per_day": {
    "doc_count_error_upper_bound": 0,
    "sum_other_doc_count": 6,
    "buckets": [
      {
        "key": 1609632000000,
        "key_as_string": "2021-01-03T00:00:00.000Z",
        "doc_count": 3
      },
      {
        "key": 1610236800000,
        "key_as_string": "2021-01-10T00:00:00.000Z",
        "doc_count": 2
      },
      {
        "key": 1613260800000,
        "key_as_string": "2021-02-14T00:00:00.000Z",
        "doc_count": 2
      },
      {
        "key": 1613865600000,
        "key_as_string": "2021-02-21T00:00:00.000Z",
        "doc_count": 2
      },
      {
        "key": 1609545600000,
        "key_as_string": "2021-01-02T00:00:00.000Z",
        "doc_count": 1
      },
```


Figure 3.20: query 3.2.10 results.