POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Systems and Methods for Big and Unstructured Data Project

Author(s): **Sebastian Ballesteros - 10945208**
Abril Cano - 10992251
Cesar Carbajal - 11001933

Group Number: **29**

Academic Year: 2024-2025

Contents

1 | Introduction

The modern landscape of big data demands efficient storage, indexing, and querying mechanisms for different data structures. This is fundamental not only to be able to manage correctly the immense quantity of data that is being produced at each moment, but also to be able to observe the data in a way to gain insightful information from it and make informed business decisions. With this in mind, in this project, we aim to manage and analyze two datasets:

1. The Amazon Reviews Dataset: a large-scale, textual dataset containing customer reviews of multiple products.
2. The Stack Overflow Developer Survey Dataset: a structured dataset that captures developers' opinions, demographics, and career details.

To handle these datasets effectively, we selected Elasticsearch for the Amazon Reviews dataset and MongoDB for the Stack Overflow dataset due to their distinct features and strengths:

- Elasticsearch is ideal for textual, search-intensive datasets like reviews due to its full-text search capabilities, scalability, and analytical power.
- MongoDB excels with hierarchical and semi-structured data like survey responses due to its schema-less nature and JSON-like document model.

The objective of this project is to perform meaningful queries on these datasets, demonstrating their real-world applications and exploring the functionality of these non-relational database technologies.

2 | Data Wrangling

All the files referred to in this section can be found in the corresponding Github repository.

2.1. Elasticsearch Dataset Preparation

Preprocessing Steps:

From the source of the data 2 files were obtained, one regarding the items and the other the reviews. The details of these will be explained in more detail in a next section. Upon first inspection of the data, the unnecessary or empty columns were removed. Specifically for the items dataset the *details* key that contained multiple values were flattened in order to manage and standardize the information with ease. In the same way, the *store* key, that contained information regarding both the artist and the format, was split and identified using Regex, to be able to create two new keys containing this information and perform a better analysis.

- Merged and cleaned the two files into one dataset containing information from both datasets using *merge_dataset.py*
- Converted JSON files into Elasticsearch-compatible JSON objects i.e. by adding an index line between each JSON object with *process_file.py*.
- Created an index and provided an explicit mapping (which will be described in the next section) using a bash script *create_index.sh*.
- Ingested the JSON into our index in the elasticsearch instance with another bash script called *ingest_data.sh*. .

2.2. MongoDB Dataset Preparation

The preparation of this data set was much easier as the data had better quality and only few adjustments need to be done. All the preprocessing steps are implemented by a single python script called *preprocessing.py* which:

- Imported CSV using pandas.
- Dropped irrelevant columns and those with missing values higher than 60%.
- Converted CSV data into JSON format.
- Imported JSON file into our mongodb Atlas cluster.

3 | Datasets

3.1. Dataset 1: Amazon Reviews

The reviews dataset was extracted from Amazon Reviews 2023[1]. This is a large-scale dataset, collected in 2023 by McAuley Lab, and it includes two main files:

- User Reviews (ratings, text, helpfulness votes, etc.);
- Item Metadata (descriptions, price, raw image, etc.);

Each object in the User Reviews represents a product review with the following attributes:

Key	**Explanation**	**Type**
title	Title of the user review	String
rating	Rating of the product (from 1.0 to 5.0)	Float
text	Text body of the user review. (from 1.0 to 5.0)	String
images*	Images that users post after they have received the product	Array
asin	ID of the product.	String
parent_asin	Parent ID of the product	String
user_id	ID of the reviewer	String
timestamp	Time of the review (unix time)	Integer
verified_purchase	User purchase verification	String
helpful_vote	Helpful votes of the review	Integer

Table 3.1: Description of the reviews file and its structure.
*This key was removed due to its lack of content or relevance in the project.

In addition, the JSON file containing the items i.e. (products) has the following structure:

[1] https://amazon-reviews-2023.github.io/

Key	Explanation	Type
main_category	Main category of the product	String
title	Name of the product	String
average_rating	Average rating of the product	Float
rating_number	Number of ratings in the product	Integer
features*	Bullet-point format features of the product	Array
description	Description of the product	Array
price	Price in US dollars	Float
images	Images of the product	Array
videos	Videos of the product including title and url	Array
store	Store name of the product	Integer
categories*	Hierarchical categories of the product	Array
details	Product details, including materials, etc	Object
parent_asin	Parent ID of the product	String
bought_together*	Recommended bundles from the websites	Array

Table 3.2: Description of the items file and its structure
*This key was removed due to its lack of content or relevance in the project.

As explained in chapter 2, we merged these two files and created a final JSON file containing all the information. For each review, we retrieved the item it reviewed in the item metadata file, and nested it inside the review object. We defined an index mapping:

```
{
  "mappings": {
    "properties": {
      "asin":    { "type": "keyword" },
      "parent_asin":  { "type": "keyword"  },
      "user_id":  { "type": "keyword"  },
      "title":  { "type": "text"  },
      "text":  { "type": "text"  },
      "rating":  { "type": "integer"  },
      "timestamp":  { "type": "date", "format": "epoch_millis"  },
      "helpful_vote":  { "type": "integer"  },
      "verified_purchase":  { "type": "boolean"  },
      "item":  {
            "type": "nested",
            "properties": {
              "main_category":    { "type": "keyword"  },
              "title": { "type": "text"  },
              "average_rating":     { "type": "half_float"   },
              "rating_number":   { "type": "integer"   },
              "description":   { "type": "text"   },
              "price":   { "type": "float"   },
              "store":   { "type": "text"   },
              "artist":   { "type": "keyword"   },
              "format":   { "type": "keyword"   },
              "parent_asin":   { "type": "keyword"   },
              "detail_Package_Dimensions": { "type": "text" },
              "detail_Manufacturer": { "type": "keyword" },
              "detail_Date_First_Available": {"type": "date", "format": "yyyy-MM-dd"},
              "detail_Label": { "type": "keyword" },
              "detail_Number_of_discs": { "type": "integer" },
              "detail_Is_Discontinued_By_Manufacturer": { "type": "boolean" }
            }
          }
        }
      }
    }
```

Below, we group and explain our choices by field type:

1. Text Fields (text): **title, text, description, store, detail_Package_Dimensions**. These fields represent textual data that requires tokenization and full-text search capabilities. By using text, we ensure effective querying and relevance scoring for content analysis.

2. Exact Match Fields (keyword): **asin, parent_asin, user_id, main_category**. These fields are identifiers or categorical data that require exact matches without tokenization. Using keyword ensures precision and enables filtering and aggregations.

   - The derived types: **artist, format, and detail_Manufacturer** were also set as keywords mainly to exploit as well the properties of this data type when constructing the dashboard in Kibana.

3. Numerical Fields (integer, float, half_float): **helpful_vote, rating, price, average_rating, rating_number. detail_Number_of_discs**. These fields are used for numerical comparisons, range queries, and aggregations. We chose float or half_float to balance memory efficiency with precision, especially for fractional ratings and prices.
4. Date Fields (date): **timestamp, detail_Date_First_Available**. Date fields allow efficient range queries and time-based filtering, critical for analyzing trends and time-series data.

3.2. Dataset 2: Developer Survey

Data Source: The Stack Overflow Developer Survey 2024 dataset was extracted from Kaggle[2].

The results of the survey include more than 80 questions regarding different topics related to coding activities. For the scope of this project only the main questions and those addressed in the queries will be described, but a complete explanation of all the survey is detailed in *"survey_results_schema.csv"* and *"2024 Developer Survey.pdf"*

[2]`https://www.kaggle.com/datasets/berkayalan/stack-overflow-annual-developer-survey-2024`

Key	**Description**
MainBranch	Describes the relation of the person with coding (Hobby, Developer, learning, ...)
Age	Range of age of the person taking the survey
Employment	Best describes your current employment status
RemoteWork	If either the work modality is Hybrid, Remote, In-person
EdLevel	Highest level of formal education that's been completed
LearnCode	Where did they learned to code
YearsCode	Total years of coding experience
OrgSize	Approximately size of the company currently working for
DevType	Area of coding the person is currently working on
Country	Where do the the surveyed is living
Currency	Currency used day-to-day
CompTotal	Current total annual compensation
JobSat	Satisfaction in the current professional role
DatabaseHaveWorkedWith	Database environments where the surveyed worked most in the past year
DatabaseWantToWorkWith	Database environments that the surveyed wants to works with

Table 3.3: Description of the main fields of The Stack Overflow Developer Survey 2024.

4 | Queries

For each query, provide a title, a description of what the query is supposed to do, the (text of the) query, and a screenshot/table representing the (possibly partial) outcome.

4.1. Queries for the Amazon Reviews dataset

Note that for the some results of these queries we just kept the first objects returned by the query to limit the output.

Query 1 Highly Rated Reviews of Latin American Music

Description: This query retrieves reviews related to Mexican, Colombian, or Peruvian music. It ensures the reviews mention these keywords in their title and gives priority to those with a rating of 4 or higher.
Query:

```
"_source": ["user_id","title", "text", "rating", "item.title",
"item.average_rating", "item.artist", "item.price"],
"query": {
  "bool": {
    "must": {
      "match": {
        "title": "Mexican, Colombia, Peruvian"
      }
    },
    "should": {
      "range": {
        "rating": { "gte" : 4 }
      }
    }
  }
}
```

Result: For formatting purposes, only the objects in the response will be presented.

```
{
  [
    {
        "_index" : "reviews_index",
        "_id" : "GSErtpMBycm5KepJcxnu",
        "_score" : 11.958555,
        "_source" : {
          "rating" : 4.0,
          "title" : "Mexican electronica",
          "text" : "This is a super collection of songs from the duo of Bostich &amp;
          Fussible that was released around the same time as another of their efforts
          was released under the name of Nortec Collective. Similar in style to the
          Nortec Collective CD, this disc features hyper-beats, wailing  jazzy horns
          and crashing and droping percussive syncopation, infused with Banda samples
          that sound nothing like the original horns. A popular song from the disc
          entitled &quot;Odyssea 2000&quot; is performed twice, once as an extended
          version by a Dj that has many &quot;false starts&quot; that keep the song
          sounding like it is just beginning. It is a radio friendly , hence the radio
          edit version, sort of song that gets occasional airplay on one of the local
          Tijuana radio stations down here in southernmost Cali. Much of the music is
          the type of stuff perfect for bumper music on radio shows or for those
          snazzy car commercials on tv. The music is very clever  that features dance
          type grooves or if your best  dancing days are behind you perfect for an
          additional adrenaline rush to compliment your aerobic activity.
          &quot;Unicornio&quot; is another done twice song that probably could have
          suficed with the extended version only but I guess they felt the Dj version
          was neccessary.The problem is that it makes for much of the same songs as
          opposssed to two new fresh songs being on the disc. &quot;Unicornio&quot;
          is a hybrid that bears the qualities of chill and trip hop with a trippy
          Banda flavor. &quot;Mona B&quot; is an odd song that sounds like the inside
          of  a martian craft from a b-movie with someone tweeking the control panel
          resulting in a bunch of odd  fluctuating hi-pitched sounds. The percussion
          picks the song up and the odd mixture of sythn sounds results in an exotic
          dance groove. &quot;Tj&quot; has the feel of a human  cabaret, people walking
          thestreets, horns blarring and buses spewing black toxins in the air, while
          people scurry about in the most visited city in the world, vendors hawking
          their wares to tourists as the heartbeat of the city pumps  a um-pah-pah-
          banda beat. If you have never heard electronica from Mexico this is a good
          example but there are many others too, including Nopalbeat, Wakal, Tocadisco,
          Nortec Collective and various variations on a genre like Sidestepper, Plastilli
          Mosh or Kinky. If you want to try something different in the electronica vein
          than this might be a good place to start your exploration  from down Mexcio way
          "user_id" : "AFIA5YDIOGEGNRECRIKC2U72H3SQ",
          "item" : {
            "title" : "Nortec",
            "average_rating" : 4.8,
            "price" : 7.97,
            "artist" : "Bostich and Fussible"
          }
        }
      },...
    ]
  }
```

Query 2 Verified Boring Purchases

DESCRIPTION: This query finds reviews where the purchase is verified and the title contains the word "boring," suggesting dissatisfaction with the item. This helps identify low-quality items among verified purchases.

QUERY:

```
    "_source": ["user_id","title", "text", "rating", "item.title",
    "item.average_rating", "item.artist","item.price"],
      "query": {
        "bool": {
          "must": {
            "match": { "verified_purchase": true }
          },
          "should": {
            "match": { "title": "Boring" }
          }
        }
      }
```

RESULT:

```
{
```

```
[
  {
      "_index" : "reviews_index",
      "_id" : "uCErtpMBycm5KepJ6iA_",
      "_score" : 12.812708,
      "_source" : {
        "rating" : 1.0,
        "title" : "Boring, boring, boring",
        "text" : "If you’re a fan of jam bands, you may enjoy this.
        I, however, am not.  I bought it on the recommendation of a
        friend who used to see the band all the time.
        Adequate musicianship, boring songs.",
        "user_id" : "AFIV2CAC4YSTG6GCCGJQWN6ZKGJQ",
        "item" : {
          "title" : "It’s a Brand New Day",
          "average_rating" : 1.0,
          "price" : null,
          "artist" : "The Homel-Alaniz Band"
        }
      }
    },
    {
      "_index" : "reviews_index",
      "_id" : "e7IqtpMBwFltyBCO4mwV",
      "_score" : 11.157477,
      "_source" : {
        "rating" : 3.0,
        "title" : "Boring",
        "text" : "I am disappointed with this album. I was expecting
        a variety but got easy listening.  These artists have
        remarkable talents but they did not display too many of
        those talents in this album.  Unfortunately, this is the first
        Vocal Majority album I have purchased so I cannot recommend
        a better one.  Try to find one with classical music not popular.",
        "user_id" : "AFBI5C4W3COTCIVUHXKU5G7A4CBA",
        "item" : {
          "title" : "I’ll Be Seeing You (1990)",
          "average_rating" : 3.7,
          "price" : 24.99,
            "artist" : "Vocal Majority"
          }
        }
      },...
    ]
}
```

Query 3 Negative Review of Elton John Items Priced Above $29.99

DESCRIPTION: This query identifies negative reviews for items by Elton John where the item's price is at least $29.99. It filters for text containing phrases like "expensive," "cheaper," or "refund," indicating dissatisfaction.

QUERY:

```
  "_source": ["user_id","title", "text", "rating", "item.title",
  "item.average_rating", "item.artist", "item.price"],
  "query": {
    "nested": {
      "path": "item",
      "query": {
        "bool": {
          "should": {
            "match": { "text": "Expensive, Cheaper, Refund" }
          },
          "must": [
            { "range": {
                "item.price": { "gte" : 29.99 }}
            },
            { "match": {"item.store": "Elton John"} }
          ]
        }
      }
    }
  }
```

RESULT:

```
{
  [
{
   "_index" : "reviews_index",
   "_id" : "dbMvtpMBwFltyBCONZUZ",
   "_score" : 15.1933,
   "_source" : {
     "rating" : 1.0,
     "title" : "Missing all of the inserts that were pictured in
     the description!",
     "text" : "Album came today, but it was NOT as described!
     Photos in description were misleading; my package was missing
     the scraps booklet, the lyrics booklet, poster & fan club
     sheet! I will be returning it! Not happy!",
     "user_id" : "AHSL7TARDLCZR5JMRIO6UFBRBP4A",
     "item" : {
       "title" : "Elton John: Captain Fantastic And The Brown
       Dirt Cowboy (12\" 33 rpm) Vinyl Record MCA-2142",
       "average_rating" : 3.6,
       "price" : 100.49,
       "artist" : "Elton John"
     }
   }
 },
 {
   "_index" : "reviews_index",
   "_id" : "XLMutpMBwFltyBCOL1ci",
   "_score" : 15.1933,
   "_source" : {
     "rating" : 5.0,
     "title" : "Great Christmas gift",
     "text" : "Was in excellent shape with original poster and
     was shipped promptly.",
     "user_id" : "AF3WDGA42OADCP6DLEKR72PCDE5Q",
     "item" : {
       "title" : "Elton John: Captain Fantastic And The Brown
       Dirt Cowboy (12\" 33 rpm) Vinyl Record MCA-2142",
       "average_rating" : 3.6,
       "price" : 100.49,
       "artist" : "Elton John"
     }
```

```
        }
      },...
    ]
  }
```

Query 4 Top 5 Artists with the Most Reviews

Description: This query identifies the top five artists with the highest number of reviews, excluding irrelevant entries like "Various Artists" or non-specific labels. It aggregates review counts for each artist.
Query:

```
      "size": 0,
      "query": {
        "nested": {
          "path": "item",
          "query": {
            "bool": {
              "must_not": [
                {"term": {"item.artist": "Format: Audio CD" }},
                {"term": {"item.artist": "Various Artists" }},
                {"term": {"item.artist": "Various" }},
                {"term": {"item.artist": "VARIOUS ARTISTS" }}
              ]
            }
          }
        }
      },
      "aggs": {
        "most_frequent_artist": {
          "nested": {
            "path": "item"
          },
          "aggs": {
            "artist_aggregation": {
              "terms": {
                "field": "item.artist",
                "size": 5
              }
          }
        }
      }
    }
```

Result:

```
{
   "took":0,
   "timed_out":false,
   "_shards":{
      "total":1,
      "successful":1,
      "skipped":0,
      "failed":0
   },
   "hits":{
      "total":{
         "value":10000,
         "relation":"gte"
      },
      "max_score":null,
      "hits":[

      ]
   },
   "aggregations":{
      "most_frequent_artist":{
         "doc_count":107937,
         "artist_aggregation":{
            "doc_count_error_upper_bound":0,
            "sum_other_doc_count":97520,
            "buckets":[
               {
                  "key":"BTS",
                  "doc_count":1355
               },
               {
                    "key":"The Beatles",
                    "doc_count":1050
                 },
                 {
                    "key":"Elvis Presley",
                    "doc_count":782
                 },
                 {
                    "key":"Pink Floyd",
                    "doc_count":490
                 },
                 {
                    "key":"Prince",
                    "doc_count":316
                 }
              ]
           }
        }
     }
  }
```

Query 5 Most Active Reviewer

Description: This query finds the user who has submitted the most reviews overall. It aggregates and ranks users based on their total review count.

Query:

```
"size": 0,
"aggs": {
  "most_reviews_user": {
    "terms": {
      "field": "user_id",
      "size": 1
    }
  }
}
```

Result:

```
{
  "took":0,
  "timed_out":false,
  "_shards":{
     "total":1,
     "successful":1,
     "skipped":0,
     "failed":0
  },
  "hits":{
     "total":{
        "value":10000,
        "relation":"gte"
     },
     "max_score":null,
     "hits":[

     ]
  },
  "aggregations":{
     "most_reviews_user":{
        "doc_count_error_upper_bound":0,
        "sum_other_doc_count":130093,
        "buckets":[
           {
              "key":"AGAFM74L2RIJ5O36NNYH4Z5ISQNQ",
              "doc_count":341
           }
        ]
     }
  }
}
```

Query 6 Positive Reviews by the Top User

Description: This query calculates the number of positive reviews submitted by the top reviewer. It looks for reviews with the user ID of the most active user and searches for positive sentiments such as "good," "great," or "excellent" in the review text.
Query:

```
  "_source": ["user_id","title", "text", "rating", "item.title",
  "item.average_rating", "item.artist", "item.price"],
"size": 2,
"query": {
  "bool": {
    "must": {
      "term": {
        "user_id": "AGAFM74L2RIJ5O36NNYH4Z5ISQNQ"
      }
    },
    "should": {
      "match": {
        "text": "Good, Great, Excellent, Amazing, Awesome"
      }
    }
  }
}
```

Result:

```
{
   "took":2,
   "timed_out":false,
   "_shards":{
      "total":1,
      "successful":1,
      "skipped":0,
      "failed":0
   },
   "hits":{
      "total":{
         "value":341,
         "relation":"eq"
      },
      "max_score":8.5523205,
      "hits":[
         {
            "_index":"reviews_index",
            "_id":"07IstpMBwFltyBCOTMkx",
            "_score":8.5523205,
"_source":{
   "rating":4.0,
   "title":"excellent recital of Italian Renaissance music,
   impaired by the absence of texts",
   "text":"Excellent recital of Italian Renaissance music,
   recorded in 1986 by IMP Classics, centered around Florenc
   e and music composed for the Carnival period (May 1 - June 14)
   in the days of Lorenzo de' Medici (1469 - 1492) and
   the Dutch composer established in Florence, Heinrich Isaac.
   Good and varied mixture of instrumental (lute & guitar,
   drum, recorder & shawn, sackbuts, cornett, viol) and
   vocal (always multiple voices, madrigal like), and, within
   the madrigals, of a cappella and accompanied, and good
   and varied mixture of the lively-playful and meditative-solemn.
   Other than the instruments, th booklet credits three (male)
   singers as comprising London Pro Musica, counter-tenor and two
   tenors, but I hear four including a basso, so I am supposing
   that conductor Bernard Thomas also sings. London Pro Muica was
   an ensemble specializing in Renaissance music, especially
   that of Italy, widely toured in Great Britain and continental
   Europe. There's still a London Pro Musica choir active today,
   but it is based in London... Ontario, Canada (!!!), and I doubt
   that it has anything to do with this earlier Renaissance ensemble.
   There's also a London Pro Musica publisher of ancient music,
   and this one based in London England, so maybe that is a
   distant offshoot of the performing ensemble, but I don't
   know.<br /><br />All that is fine, except for the absence of
   texts and translations of the madrigals sung, which hampers
   enjoyment. As everyone with a taste for Italian madrigals
   will know, madrigals aren't just \"pure\" music, intertwining
   melodic lines meant to be vaguely listened to. No, they are
   setting of words, poetry, and if you can't follow closely the
   words, you just remain at the surface, you are condemned to
   listen only vaguely. Layer of disappointment, then, in a recital
   which could have been highly enjoyable.<br /><br />IMP was an
   interesting English label, first set up by Pickwick in the
   early days of the CD, then passed on to Carlton Classics in the
   mid 1990s, and they disappeared around the end of that decade.
   Many of their installments sell for 1 penny and many more for
```

```
      less than a pound on the UK sister company. How can one resist.
      I’ve bought hundreds.",
      "user_id":"AGAFM74L2RIJ5O36NNYH4Z5ISQNQ",
      "item":{
         "title":"A Florentine Carnival",
         "average_rating":4.0,
         "price":75.0,
         "artist":"Heinrich Isaac"
      }
   }
},
{
   "_index":"reviews_index",
   "_id":"ZrIstpMBwFltyBCOTMox",
   "_score":8.17034,
   "_source":{
      "rating":4.0,
      "title":"This is so good, there is simply not enough of it,
      and Philips/Universal are really milking the cow here",
      "text":"First, let me vent my frustrations: this is so good
      and entertaining, why so little of it? The original LP -
      recorded and released in 1965 according to the apparently
      well-informed Wikipedia entry, although the CD bears a copyright
      date of 1967 - was short even for an LP: 28 minutes 35.
      This was the fourth album of the original, Paris-based Swingle
      ensemble, that had burst into the limelight in 1963 with
      their chart-making \"Jazz S&eacute;bastien Bach\"
      (\"Bach’s Greatest Hits\" in the US, [[ASIN:B00004UFOL J
      azz Sebastian Bach, Vol. 1]]). Two members had changed
      since the original album: one mezzo and one bass. Not that
      you can hear any difference. The Paris-based Swingles used a
      jazz drum and double-bass accompaniment section - here, Guy
      Pedersen, also Michel Legrand’s double-bassist, and Daniel
      Humair at the drums, who became (and still is) one of France’s
      top jazz drummers. When the group reformed in 1973 in London,
      they became a purely \"a capella\" group; not that they
      dispensed with the jazz accompaniment: they now sung it.<br
      /><br />And then there’s this \"original jacket collection\"
      type of thing by which Philips or Universal or whoever holds
```

```
the copyrights on these recordings (both are indicated on the
CD’s back cover)  reissue the original LPs \"straight\",
short timings and all. Now come on, what’s the point!?? Getting
the glands of nostalgia of the older generation of listeners
to salivate? Short thinking. The more you go, the less of
the older generation there will be to salivate anyway, and
personally I don’t buy nostalgia, I buy music. And there aren’t
even any substantial liner notes from the original LP that might
justify it, as sometimes with Sony/CBS’ original jacket collection
reissues. So why give us the scanty 30-minutes minus, when two
of the original LPs would easily fit on one CD? And this is indeed
what Polygram did in the early 1990s when they started reissuing
the Swingles’ original recordings: the two Bach albums
from 1963 and 1968 were reissued as [[ASIN:B0000046WP
Jazz Sebastian Bach]], and the 1964 album \"Going Baroque\"
was paired with the 1965 LP \"Anyone for Mozart?\" on [
[ASIN:B0000046Y7 Anyone For Mozart, Bach, Handel, Vivaldi?]]
(except, who knows why, for one track, that could have easily
fit). So really, this 1965 album, \"Getting Romantic\"
(released in France under the title \"Les Romantiques\")
should have been paired with, say, the Swingle’s next from 1966,
\"Rococo A Go Go/Swingling Telemann\" (see my review of
[[ASIN:B00005V9SM Swingling Telemann]]). Anyway, as it is,
find it priced accordingly.<br /><br />And there’s this habit
of the Swingles to play only snippets: the finale of Beethoven’s
5th Violin-Piano Sonata (the \"Spring\") but not the rest -
the first movement in particular has a gorgeously lyrical
theme that would have been ideally suited to the Swingles -,
Mussorgky’s \"Market of Limoges\" from \"Pictures at an Exhibition\",
but why not more ? All the more frustrating as, in the original,
\"Market of Limoges\" leads directly, by way of a great
crescendo-accelerando, to the awesome and funeral Catacombae.
In fact, why not the complete Pictures ? Two Etudes and a Waltz
from Chopin, fine, great, but are you telling me that in 50 years
of existence (personnel constantly changing, of course) the
Swingles never found the time to adapt and perform Chopin’s
complete Preludes, Etudes and Waltzes ?!!! The andante from
Schubert’s 13th String Quartet, but where’s the rest, and where
is \"Death and the Maiden\", Schubert’s awesome 14th String
```

```
Quartet?<br /><br />In fact, the Swingles' arrangements and
realizations are so enticing, entertaining, illuminating,
that I crave for complete traversals of everything: Schubert's
complete quartets and two quintets, Scarlatti's 555 sonatas,
Mozart's and Beethoven's symphonies, concertos  and piano
sonatas (in fact the only complete works the Swingles have
recorded in their 50 years of existence is, if I am not
forgetting anything, although in abridged form, are Mozart's
Kleine Nachtmusik in the \"Anyone for...\" album mentioned
above and 40th Symphony, in [[ASIN:B000002SND A Cappella Amadeu
s - A Mozart Celebration]], a 1981 recording), Bach's Goldberg
Variations and Well-Tempered Clavier, Vivaldi's complete
Four Seasons, and.... Oh, well, everything.<br /><br />I've
written a number of reviews of the Swingle's discs recently,
and I've contended that the fascination and entertainment of
their arrangements and realizations derive from the fact that
they elevate to the status of high art what we all do, almost
inadvertently, without even thinking about it, in a very
rudimentary form, walking on the street, washing the dishes,
vacuuming and ironing - and, of course, under the notorious
shower: humming our favorite tunes. But the Swingles are to
our shower humming what a Picasso is to your kid's drawings.
And I've also opined that by so doing, they return instrumental
music to its very origin and essence: the human voice - as,
presumably, all these great composers, Bach, Mozart, Vivaldi,
Handel, Beethoven, Chopin and all the rest first hummed these
tunes for themselves before committing them to the music sheet.
Consequently, it is asked of and expected from a pianist playing
Chopin (or Beethoven, or Schumann, or Schubert) to make his
instrument SING, even if (or precisely because) its acoustical
properties are radically different from those of the human voice:
no sustained out-breath, but the vibration of a struck chord
that will start decaying as soon as it is produced.<br /><br
/>The Swingles' arrangements and realizations are great.
You get here their customary mixture of the dazzlingly virtuosic
(the two Beethoven, Chopin's Etude op. 25/2 with great tossing
of the melody between the two sopranos, Mendelssohn's Song
without Words \"Spinner's Song/Spinnerlied\", here titled
\"La Fileuse\", Mussorgsky), and the sensuously caressing soprano(s)
```

```
                    over the silky tapestry of the vocal accompaniment
                    (Chopin's Etude op. 10/6 and Waltz, Schubert's Rosamunde)
                    .<br /><br />But 28 minutes! It took me longer to write the
                    review: not a good ratio. It's an enthusiastic five stars for
                    the music, a frustrated two for the disappointing timing. I
                    love the music so much, I'll flunk my maths and average it to
                    four. For more on/of the Swingle Singers, see my comprehensive
                    discography on Listmania! (links in the comments section).",
                    "user_id":"AGAFM74L2RIJ5O36NNYH4Z5ISQNQ",
                    "item":{
                       "title":"Les Romantiques",
                       "average_rating":4.7,
                       "price":39.47,
                       "artist":"Swingle Singers"
                    }
                 }
              }
           ]
        }
     }
```

Query 7 Average Price of Taylor Swift's Items

Description: This query calculates the average price of items associated with one of the most popular artists nowadays: Taylor Swift. It filters the dataset for items with "Taylor Swift" listed as the artist.
Query:

```
  "size": 0,
  "query": {
    "bool": {
      "must": {
        "term": {
          "user_id": "AGAFM74L2RIJ5O36NNYH4Z5ISQNQ"
        }
      },
      "should": {
        "match": {
          "text": "Good, Great, Excellent, Amazing, Awesome"
        }
      }
    }
  }
```

Result:

```
    {
      "took" : 0,
      "timed_out" : false,
      "_shards" : {
        "total" : 1,
        "successful" : 1,
        "skipped" : 0,
        "failed" : 0
      },
      "hits" : {
        "total" : {
          "value" : 305,
          "relation" : "eq"
        },
        "max_score" : null,
        "hits" : [ ]
      },
          "doc_count" : 305,
          "average_price" : {
            "value" : 34.952183822105674
          }
        }
      }
    }
```

Query 8 Average Rating of High-Priced Items

Description: This query calculates the average rating of items with a price greater than or equal to $49.99. It focuses on premium items to assess customer satisfaction at higher price points.
Query:

```
  "size": 0,
  "query": {
    "nested": {
      "path": "item",
      "query": {
        "bool": {
          "must": {
            "range": {
              "item.price": { "gte" : 49.99 }
            }
          }
        }
      }
    }
  },
  "aggs": {
    "average_rating": {
      "nested": {
        "path": "item"
      },
      "aggs": {
        "average_rating": {
          "avg": {
            "field": "item.average_rating"
          }
        }
      }
    }
  }
```

Result:

```
    {
      "took" : 4,
      "timed_out" : false,
      "_shards" : {
        "total" : 1,
        "successful" : 1,
        "skipped" : 0,
        "failed" : 0
      },
    "hits" : {
      "total" : {
        "value" : 10000,
        "relation" : "gte"
      },
      "max_score" : null,
      "hits" : [ ]
    },
    "aggregations" : {
      "average_rating" : {
        "doc_count" : 13039,
        "average_rating" : {
          "value" : 4.561759807716275
        }
      }
    }
  }
```

Query 9 Price Statistics for Items

Description: This query provides statistical data (minimum, maximum, average, and standard deviation) for item prices. It includes only items with valid price entries.

Query:

```
"size": 0,
"query": {
  "nested": {
    "path": "item",
    "query": {
      "bool": {
        "must": {
          "exists": {
            "field": "item.price"
          }
        }
      }
    }
  }
},
  "aggs": {
    "correlation": {
      "nested": {
        "path": "item"
      },
      "aggs": {
        "correlation": {
          "stats": {
            "field": "item.price"
          }
        }
      }
    }
  }
```

Result:

```
    {
      "took" : 0,
      "timed_out" : false,
      "_shards" : {
        "total" : 1,
        "successful" : 1,
        "skipped" : 0,
        "failed" : 0
      },
      "hits" : {
        "total" : {
          "value" : 10000,
          "relation" : "gte"
        },
        "max_score" : null,
        "hits" : [ ]
      },
      "aggregations" : {
        "correlation" : {
          "doc_count" : 81492,
          "correlation" : {
            "count" : 81492,
            "min" : 0.009999999776482582,
            "max" : 2200.0,
            "avg" : 35.66876870320232,
            "sum" : 2906719.2991613634
          }
        }
      }
    }
```

Query 10 Distribution and Average Ratings

Description: This query generates a histogram of item prices within the range of $0 to $200, with intervals of $10. It calculates the average rating for items within each price range to analyze the relationship between price and customer satisfaction.
Query:

```
  "size": 0,
  "query" : {
    "nested": {
      "path": "item",
      "query": {
        "bool": {
          "must": [
            { "exists": { "field": "item.price" } },
            { "exists": { "field": "item.average_rating" } },
            { "range": { "item.price": { "gt": 0 , "lt": 200 } } }
          ]
        }
      }
    }
  },
  "aggs": {
    "item": {
      "nested": {
        "path": "item"
      },
      "aggs": {
        "price_ranges": {
          "histogram": {
            "field": "item.price",
            "interval": 10
          },
          "aggs": {
            "average_rating": {
              "avg": {
                "field": "item.average_rating"
              }
            }
          }
        }
      }
    }
  }
```

Result:

```
    {
      "took" : 0,
      "timed_out" : false,
      "_shards" : {
        "total" : 1,
        "successful" : 1,
        "skipped" : 0,
        "failed" : 0
      },
      "hits" : {
        "total" : {
          "value" : 10000,
          "relation" : "gte"
        },
        "max_score" : null,
        "hits" : [ ]
      },
      "aggregations" : {
        "item" : {
          "doc_count" : 80089,
          "price_ranges" : {
            "buckets" : [
              {
                "key" : 0.0,
    "doc_count" : 14594,
    "average_rating" : {
      "value" : 4.545313129004214
    }
  },
  {
    "key" : 10.0,
    "doc_count" : 24386,
    "average_rating" : {
      "value" : 4.60378060958388
    }
  },
  {
    "key" : 20.0,
    "doc_count" : 17386,
    "average_rating" : {
      "value" : 4.614182376894484
    }
  },
  {
    "key" : 30.0,
    "doc_count" : 7645,
    "average_rating" : {
      "value" : 4.6023244614535646
    }
  },
  {
    "key" : 40.0,
    "doc_count" : 4980,
    "average_rating" : {
      "value" : 4.588615202999498
    }
  },
  {
    "key" : 50.0,
    "doc_count" : 2402,
    "average_rating" : {
      "value" : 4.5275868742194
    }
```

```
          },
          {
            "key" : 60.0,
            "doc_count" : 1558,
            "average_rating" : {
              "value" : 4.534646030568036
            }
          },
          {
            "key" : 70.0,
            "doc_count" : 1530,
            "average_rating" : {
              "value" : 4.553624131944445
            }
          },
          {
            "key" : 80.0,
            "doc_count" : 1027,
            "average_rating" : {
              "value" : 4.508147212755599
            }
          },
          {
            "key" : 90.0,
            "doc_count" : 1216,
            "average_rating" : {
              "value" : 4.582052130448191
            }
          },
          {
            "key" : 100.0,
            "doc_count" : 606,
            "average_rating" : {
              "value" : 4.570592899133663
            }
          },
          {
            "key" : 110.0,
            "doc_count" : 510,
      "average_rating" : {
        "value" : 4.574789368872549
      }
    },
    {
      "key" : 120.0,
      "doc_count" : 470,
      "average_rating" : {
        "value" : 4.533884640957447
      }
    },
    {
      "key" : 130.0,
      "doc_count" : 394,
      "average_rating" : {
        "value" : 4.603143837246193
      }
    },
    {
      "key" : 140.0,
      "doc_count" : 304,
      "average_rating" : {
        "value" : 4.525152909128289
      }
    },
    {
      "key" : 150.0,
      "doc_count" : 176,
      "average_rating" : {
        "value" : 4.583296342329546
      }
    },
    {
      "key" : 160.0,
      "doc_count" : 94,
      "average_rating" : {
        "value" : 4.67860704787234
      }
    },
          {
            "key" : 170.0,
            "doc_count" : 145,
            "average_rating" : {
              "value" : 4.674191810344827
            }
          },
          {
            "key" : 180.0,
            "doc_count" : 169,
            "average_rating" : {
              "value" : 4.518930288461538
            }
          },
          {
            "key" : 190.0,
            "doc_count" : 497,
            "average_rating" : {
              "value" : 4.6435212839537225
            }
          }
        ]
      }
    }
  }
}
```

4.2. Queries for the Developer Survey dataset

Just like the previous set of queries, for the some results of we just kept the first object returned by the query. In addition, we got rid of the null values to shorten the output and make it more readable.

Query 1 Italian hobbyist coders aged 18-24

Description: Identify young Italian hobbyist coders.
Query:

```
[
    { $match: {
        Country: "Italy",
        MainBranch: "I code primarily as a hobby",
        Age: "18-24 years old"
      }
    },
    { $replaceWith:
      { $arrayToObject: {
          $filter: {
            input: { $objectToArray: "$$ROOT" },
            as: "field",
            cond: {
              $and: [
                { $ne: ["$$field.v", NaN] },
                { $ne: [{ $type: "$$field.v" }, "array"] }
              ]
            }
          }
        }
      }
    }
]
```

Result:

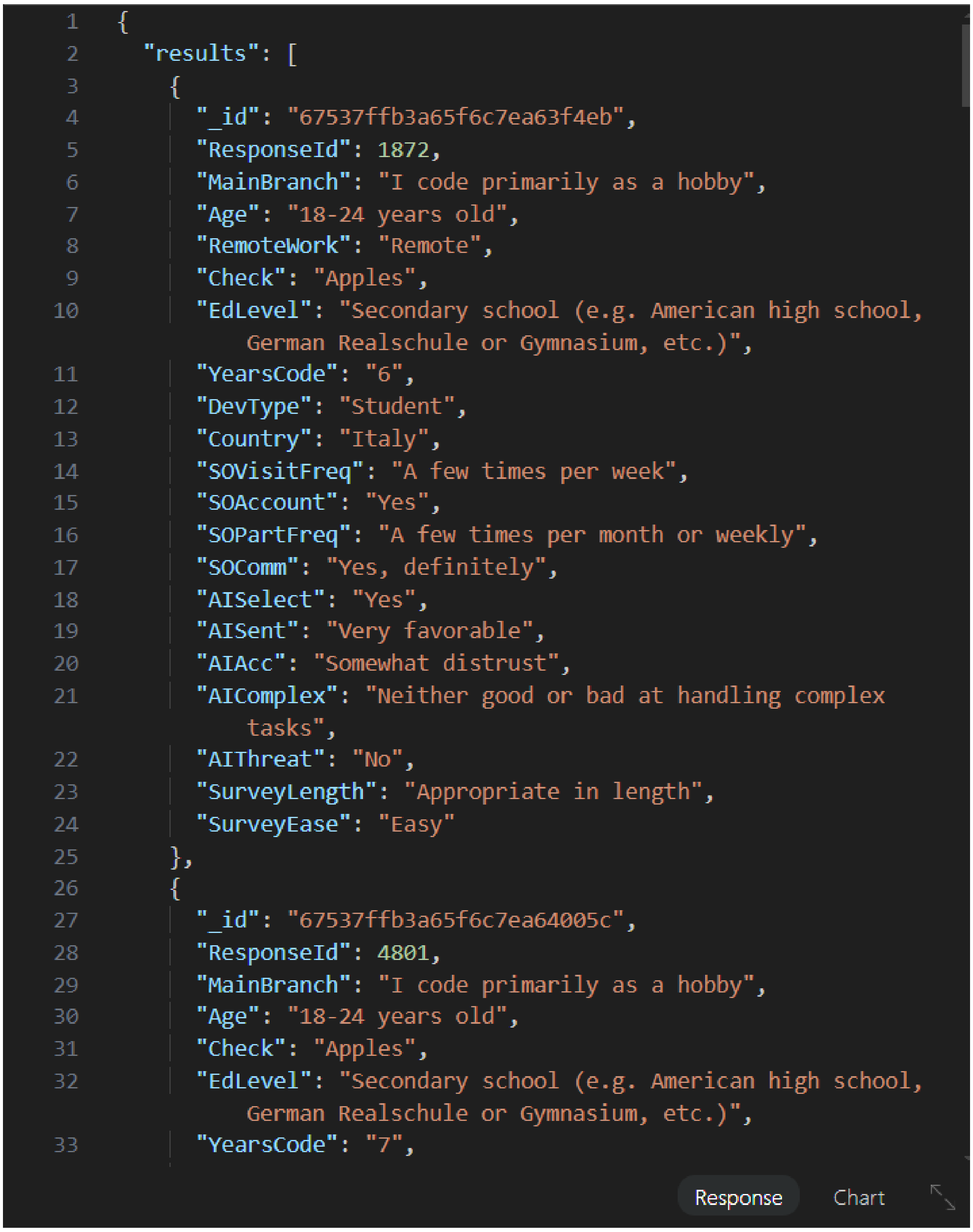

```
{
  "results": [
    {
      "_id": "67537ffb3a65f6c7ea63f4eb",
      "ResponseId": 1872,
      "MainBranch": "I code primarily as a hobby",
      "Age": "18-24 years old",
      "RemoteWork": "Remote",
      "Check": "Apples",
      "EdLevel": "Secondary school (e.g. American high school,
          German Realschule or Gymnasium, etc.)",
      "YearsCode": "6",
      "DevType": "Student",
      "Country": "Italy",
      "SOVisitFreq": "A few times per week",
      "SOAccount": "Yes",
      "SOPartFreq": "A few times per month or weekly",
      "SOComm": "Yes, definitely",
      "AISelect": "Yes",
      "AISent": "Very favorable",
      "AIAcc": "Somewhat distrust",
      "AIComplex": "Neither good or bad at handling complex
          tasks",
      "AIThreat": "No",
      "SurveyLength": "Appropriate in length",
      "SurveyEase": "Easy"
    },
    {
      "_id": "67537ffb3a65f6c7ea64005c",
      "ResponseId": 4801,
      "MainBranch": "I code primarily as a hobby",
      "Age": "18-24 years old",
      "Check": "Apples",
      "EdLevel": "Secondary school (e.g. American high school,
          German Realschule or Gymnasium, etc.)",
      "YearsCode": "7",
```

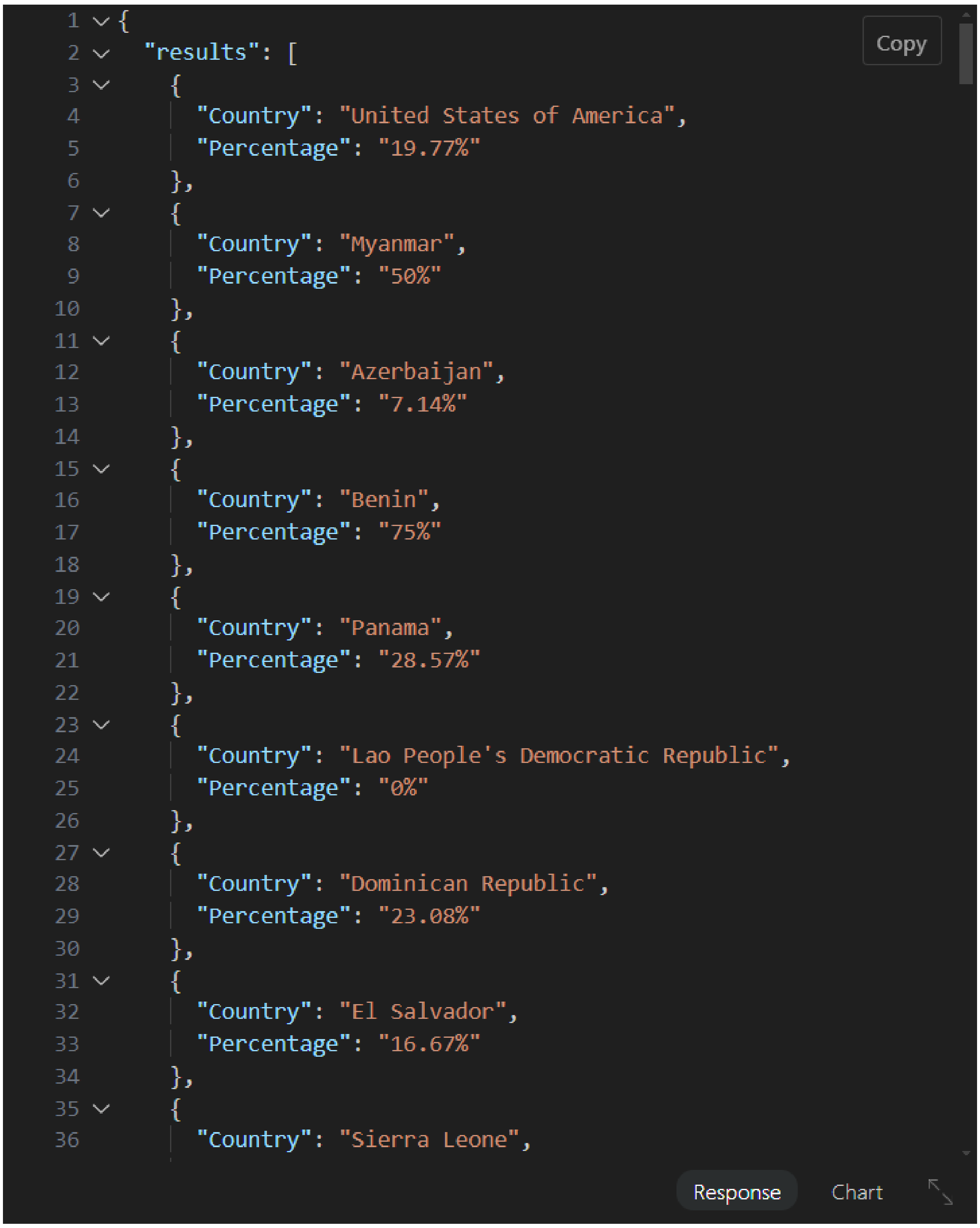
Figure 4.1: Query 1

Query 2 Countries percentage of job dissatisfaction

Description: Insights into job dissatisfaction in every country.
Query:

```
[
    { $match: {
        JobSat: { $ne: NaN },
        Country: { $ne: null }
      }
    },
    { $group: {
        _id: "$Country",
        total: { $sum: 1 }, // Total documents per country
        condition: { // If condition matches, increment by 1
          $sum: {
            $cond: [{ $lte: ["$JobSat", 5.0] }, 1, 0]
          }
        }
      }
    },
    { $project: {
        _id: 0,
        Country: "$_id",
        Percentage: {
          $concat: [
            { $toString: {
                $round: [
                  { $multiply: [ { $divide: ["$condition", "$total"] }, 100]},
                  2
                ]
              }
            },"%"
          ]
        }
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "Country": "United States of America",
      "Percentage": "19.77%"
    },
    {
      "Country": "Myanmar",
      "Percentage": "50%"
    },
    {
      "Country": "Azerbaijan",
      "Percentage": "7.14%"
    },
    {
      "Country": "Benin",
      "Percentage": "75%"
    },
    {
      "Country": "Panama",
      "Percentage": "28.57%"
    },
    {
      "Country": "Lao People's Democratic Republic",
      "Percentage": "0%"
    },
    {
      "Country": "Dominican Republic",
      "Percentage": "23.08%"
    },
    {
      "Country": "El Salvador",
      "Percentage": "16.67%"
    },
    {
      "Country": "Sierra Leone",
```

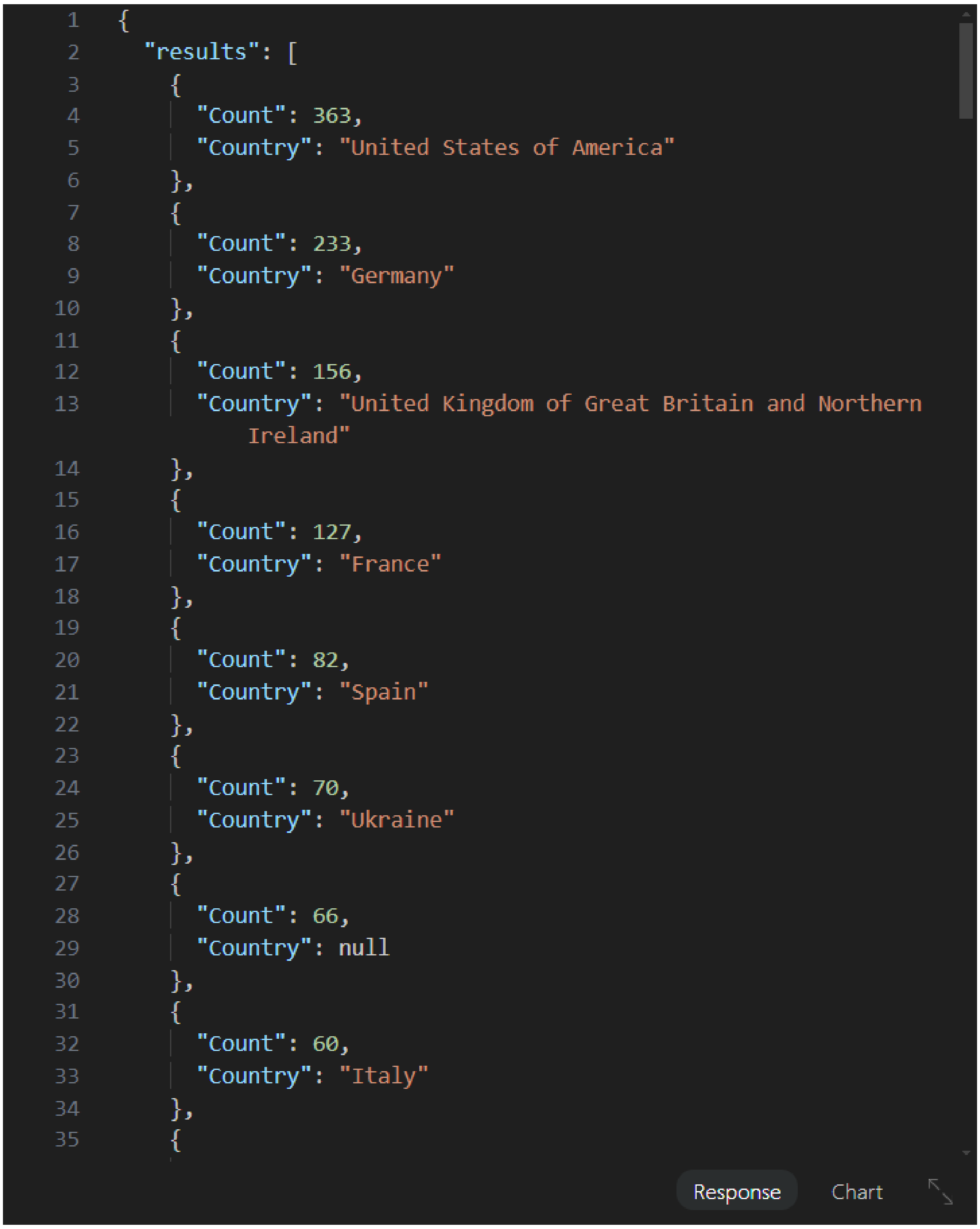
Figure 4.2: Query 2

Query 3 Professional developers with degrees grouped by country.

Description: Aggregate professional developers by education and country.
Query:

```
[
    { $match: {
        MainBranch: "I am a developer by profession",
        EdLevel: {$regex: "Professional degree"}
      }
    },
    { $group:
      {
        _id: "$Country",
        Count: { $sum: 1 }
      }
    },
    { $project: {
        _id: 0,
        Country: '$_id',
        Count: 1
      }
    },
    { $sort: {
        Count: -1
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "Count": 363,
      "Country": "United States of America"
    },
    {
      "Count": 233,
      "Country": "Germany"
    },
    {
      "Count": 156,
      "Country": "United Kingdom of Great Britain and Northern
          Ireland"
    },
    {
      "Count": 127,
      "Country": "France"
    },
    {
      "Count": 82,
      "Country": "Spain"
    },
    {
      "Count": 70,
      "Country": "Ukraine"
    },
    {
      "Count": 66,
      "Country": null
    },
    {
      "Count": 60,
      "Country": "Italy"
    },
    {
```

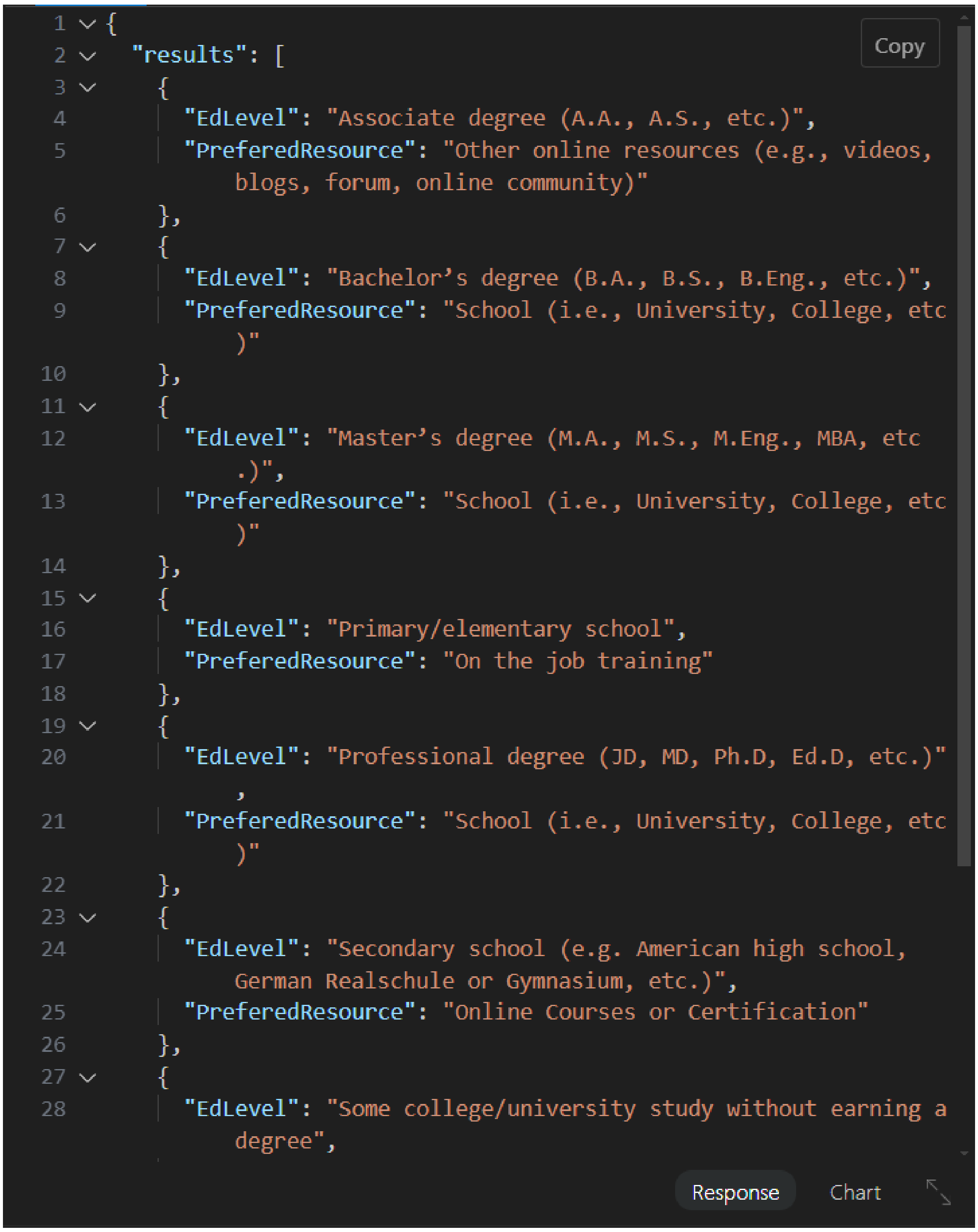
Figure 4.3: Query 3

Query 4 Preferred learning resource of by Education Level

Description: Insights of the most preferred learning resource depending on the education level.
Query:

```
[
   { $unwind: "$LearnCode" },
    { $group: {
        _id: { EdLevel: "$EdLevel", LearnCode: "$LearnCode" },
        count: { $sum: 1 }
      }
    },
    { $group: {
        _id: "$_id.EdLevel",
        mostPreferredResource: { $first: "$_id.LearnCode" },
        maxCount: { $first: "$count" }
      }
    },
    { $sort: {
        _id: 1
      }
    },
    { $project: {
        _id: 0,
        EdLevel: "$_id",
        PreferedResource: "$mostPreferredResource"
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "EdLevel": "Associate degree (A.A., A.S., etc.)",
      "PreferedResource": "Other online resources (e.g., videos,
          blogs, forum, online community)"
    },
    {
      "EdLevel": "Bachelor’s degree (B.A., B.S., B.Eng., etc.)",
      "PreferedResource": "School (i.e., University, College, etc
          )"
    },
    {
      "EdLevel": "Master’s degree (M.A., M.S., M.Eng., MBA, etc
          .)",
      "PreferedResource": "School (i.e., University, College, etc
          )"
    },
    {
      "EdLevel": "Primary/elementary school",
      "PreferedResource": "On the job training"
    },
    {
      "EdLevel": "Professional degree (JD, MD, Ph.D, Ed.D, etc.)"
          ,
      "PreferedResource": "School (i.e., University, College, etc
          )"
    },
    {
      "EdLevel": "Secondary school (e.g. American high school,
          German Realschule or Gymnasium, etc.)",
      "PreferedResource": "Online Courses or Certification"
    },
    {
      "EdLevel": "Some college/university study without earning a
          degree",
```

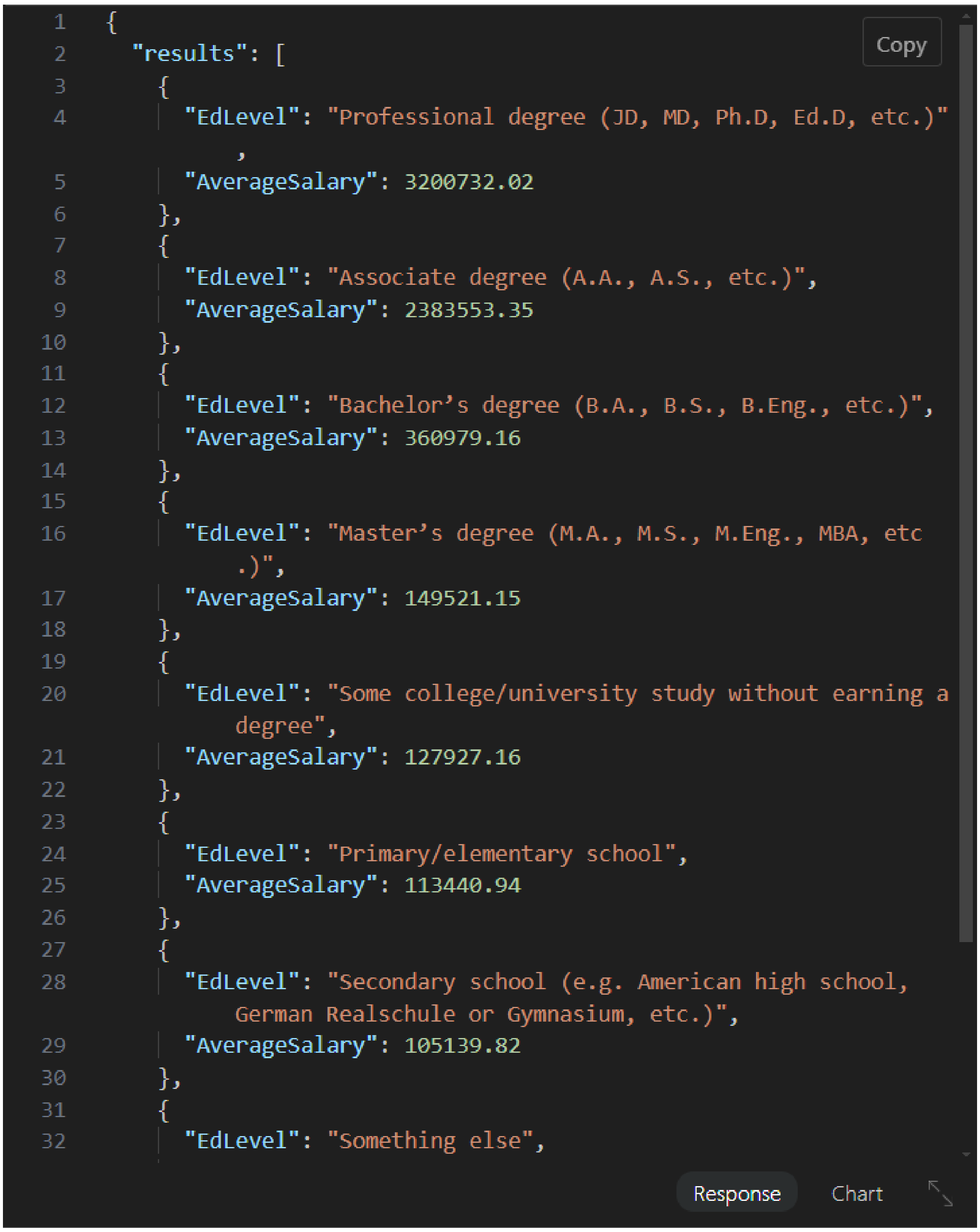
Figure 4.4: Query 4

Query 5 Annual salary by Education Level

Description: Average annual salary of developers grouped by their education level
Query:

```
[
    { $match: {
        CompTotal: { $ne: NaN },
        EdLevel: { $ne: null},
        Currency: { $eq: "USD\tUnited States dollar" }
      }
    },
    { $group: {
        _id: {EdLevel:"$EdLevel"},
        averageSalary: { $avg: "$CompTotal" }
      }
    },
    { $project: {
        _id: 0,
        EdLevel: "$_id.EdLevel",
        AverageSalary: { $round: ["$averageSalary", 2] }
      }
    },
    { $sort: { AverageSalary: -1 } }
]
```

Result:



```
{
  "results": [
    {
      "EdLevel": "Professional degree (JD, MD, Ph.D, Ed.D, etc.)"
        ,
      "AverageSalary": 3200732.02
    },
    {
      "EdLevel": "Associate degree (A.A., A.S., etc.)",
      "AverageSalary": 2383553.35
    },
    {
      "EdLevel": "Bachelor’s degree (B.A., B.S., B.Eng., etc.)",
      "AverageSalary": 360979.16
    },
    {
      "EdLevel": "Master’s degree (M.A., M.S., M.Eng., MBA, etc
        .)",
      "AverageSalary": 149521.15
    },
    {
      "EdLevel": "Some college/university study without earning a
        degree",
      "AverageSalary": 127927.16
    },
    {
      "EdLevel": "Primary/elementary school",
      "AverageSalary": 113440.94
    },
    {
      "EdLevel": "Secondary school (e.g. American high school,
        German Realschule or Gymnasium, etc.)",
      "AverageSalary": 105139.82
    },
    {
      "EdLevel": "Something else",
```

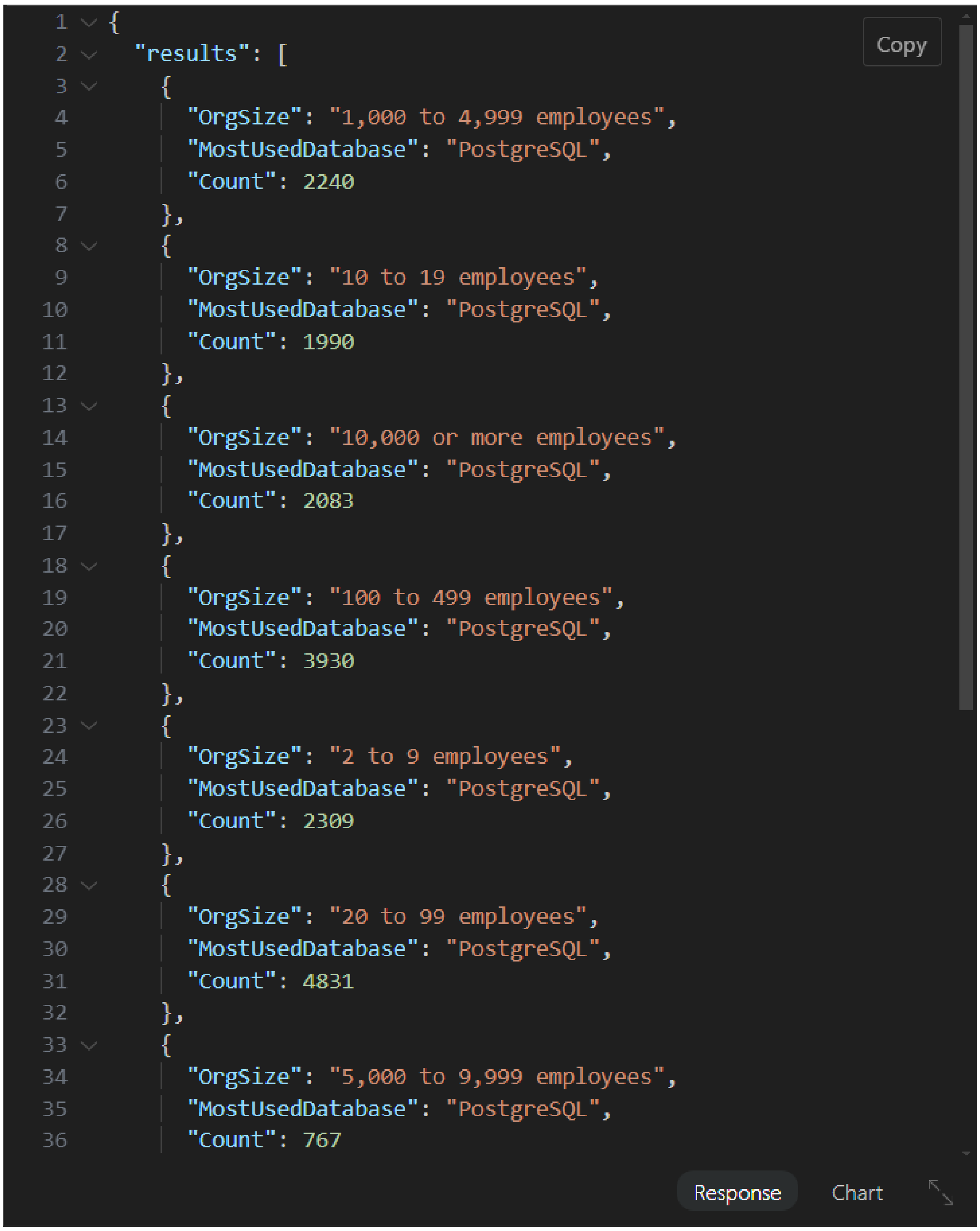
Figure 4.5: Query 5

Query 6 Database preferred by organization size

Description: Insights of the most used database on the organization size
Query:

```
[
    { $match: {
        OrgSize: { $ne: NaN },
        DatabaseHaveWorkedWith: { $ne: null, $not: { $size: 0 } }
      }
    },
    { $unwind: "$DatabaseHaveWorkedWith" },
    { $group: {
        _id: { OrgSize: "$OrgSize", Database: "$DatabaseHaveWorkedWith" },
        count: { $sum: 1 }
      }
    },
    { $sort: {
        "_id.OrgSize": 1,
        count: -1
      }
    },
    { $group: {
        _id: "$_id.OrgSize",
        mostUsedDatabase: { $first: "$_id.Database" },
        count: { $first: "$count" }
      }
    },
    { $project: {
        _id: 0,
        OrgSize: "$_id",
        MostUsedDatabase: "$mostUsedDatabase",
        Count: "$count"
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "OrgSize": "1,000 to 4,999 employees",
      "MostUsedDatabase": "PostgreSQL",
      "Count": 2240
    },
    {
      "OrgSize": "10 to 19 employees",
      "MostUsedDatabase": "PostgreSQL",
      "Count": 1990
    },
    {
      "OrgSize": "10,000 or more employees",
      "MostUsedDatabase": "PostgreSQL",
      "Count": 2083
    },
    {
      "OrgSize": "100 to 499 employees",
      "MostUsedDatabase": "PostgreSQL",
      "Count": 3930
    },
    {
      "OrgSize": "2 to 9 employees",
      "MostUsedDatabase": "PostgreSQL",
      "Count": 2309
    },
    {
      "OrgSize": "20 to 99 employees",
      "MostUsedDatabase": "PostgreSQL",
      "Count": 4831
    },
    {
      "OrgSize": "5,000 to 9,999 employees",
      "MostUsedDatabase": "PostgreSQL",
      "Count": 767
```

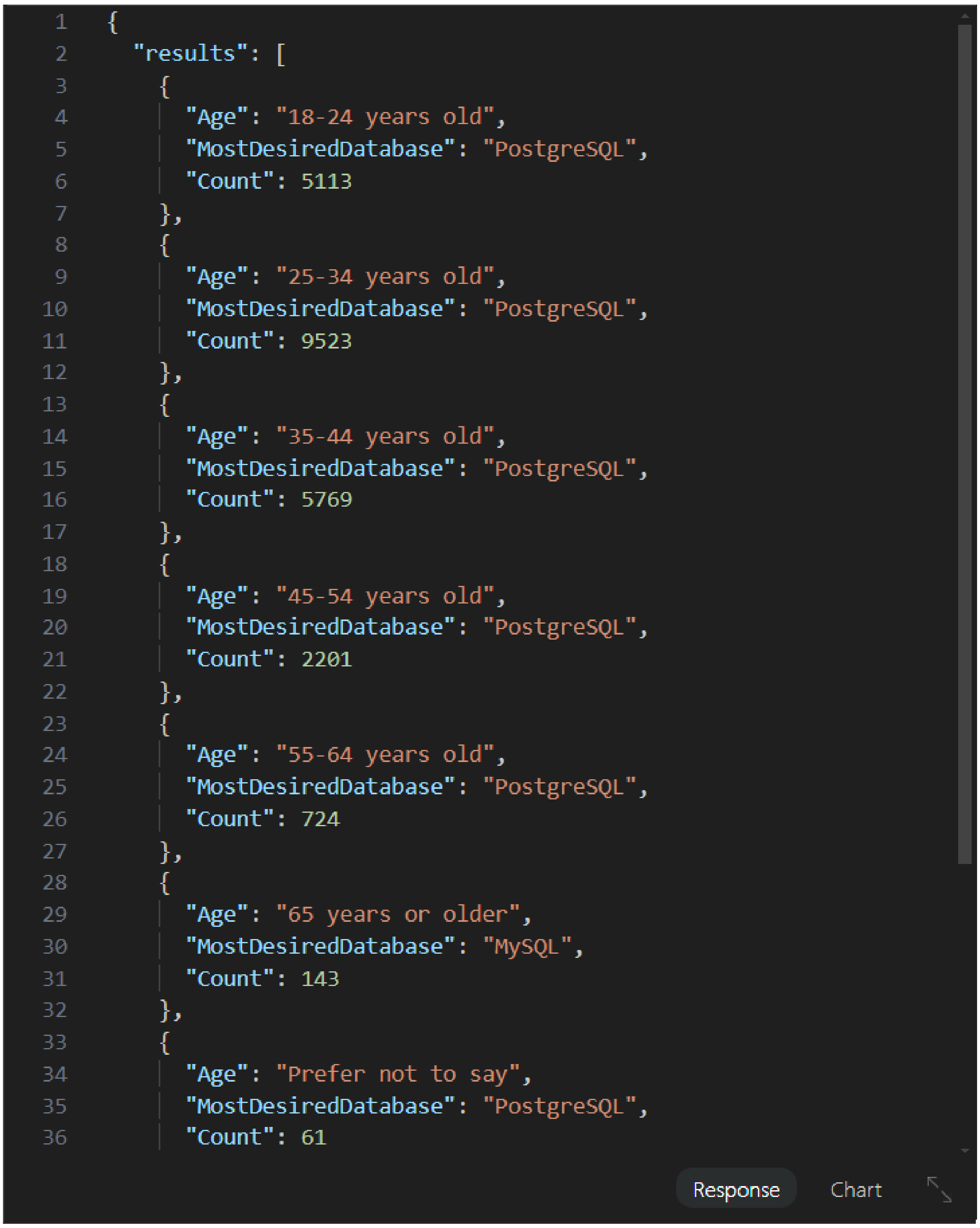
Figure 4.6: Query 6

Query 7 Most interesting database by age group

Description: Most desired database to work with grouped by age group
Query:

```
[
    { $match: {
        DatabaseWantToWorkWith: { $ne: null },
        Age: { $ne: null },
      }
    },
    { $unwind: "$DatabaseWantToWorkWith" },
    { $group: {
        _id:
        {
          Database: "$DatabaseWantToWorkWith",
          Age: "$Age"
        },
        count: { $sum: 1 }
      }
    },
    { $project: {
        _id: 0,
        Database: "$_id.Database",
        Age: "$_id.Age",
        Count: "$count"
      }
    },
    { $sort: {
        Age: 1,
        Count: -1
      }
    },
    { $group: {
        _id: "$Age", // Group by age only
        Database: { $first: "$Database" },
        Count: { $first: "$Count" }
      }
    },
    { $sort: {
        _id: 1,
        Count: -1
      }
    },
    { $project: {
        _id: 0,
        Age: "$_id",
        MostDesiredDatabase: "$Database",
        Count: "$Count"
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "Age": "18-24 years old",
      "MostDesiredDatabase": "PostgreSQL",
      "Count": 5113
    },
    {
      "Age": "25-34 years old",
      "MostDesiredDatabase": "PostgreSQL",
      "Count": 9523
    },
    {
      "Age": "35-44 years old",
      "MostDesiredDatabase": "PostgreSQL",
      "Count": 5769
    },
    {
      "Age": "45-54 years old",
      "MostDesiredDatabase": "PostgreSQL",
      "Count": 2201
    },
    {
      "Age": "55-64 years old",
      "MostDesiredDatabase": "PostgreSQL",
      "Count": 724
    },
    {
      "Age": "65 years or older",
      "MostDesiredDatabase": "MySQL",
      "Count": 143
    },
    {
      "Age": "Prefer not to say",
      "MostDesiredDatabase": "PostgreSQL",
      "Count": 61
```

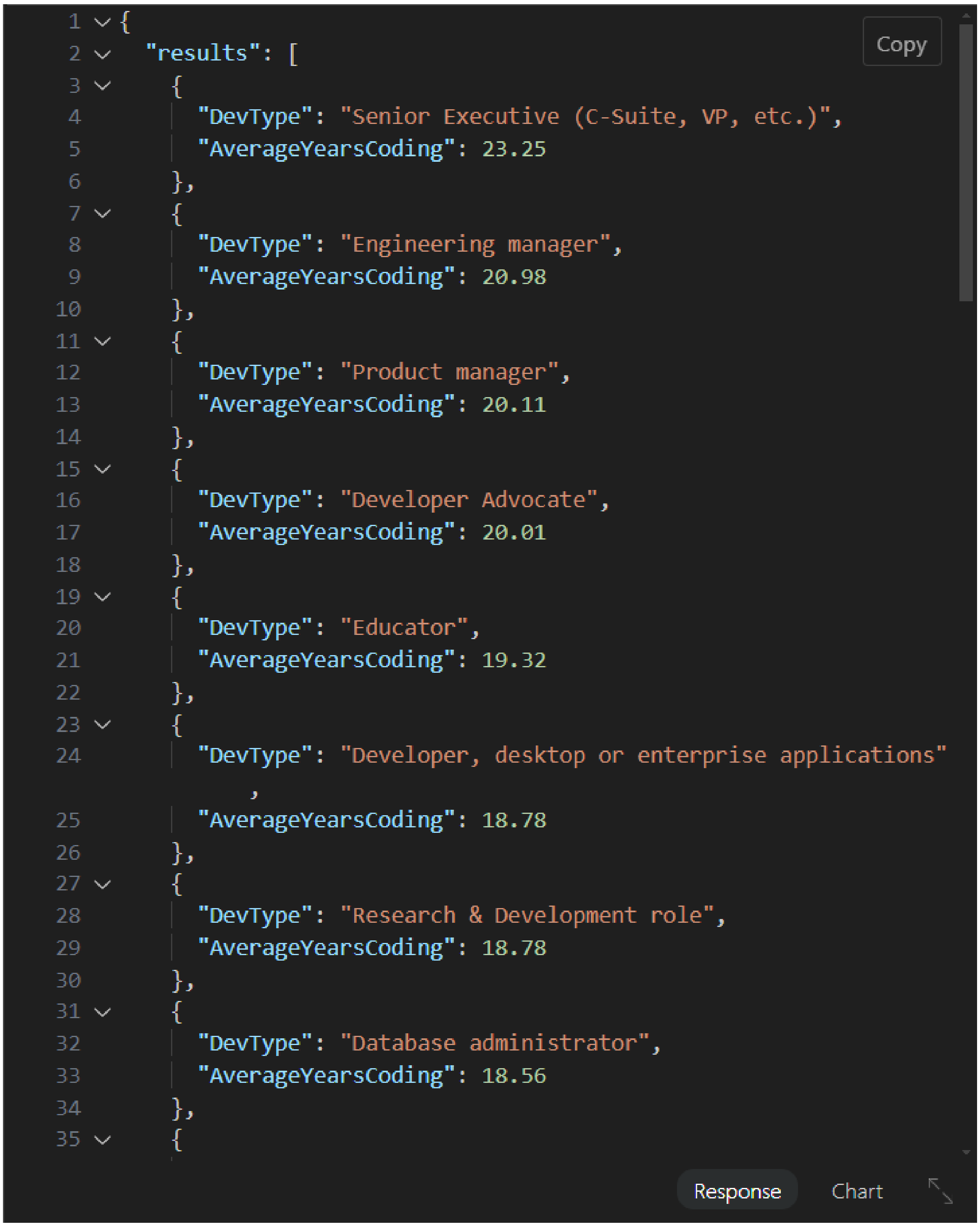
Figure 4.7: Query 7

Query 8 Years coding by developer type

Description: Average years coding grouped by developer type
Query:

```
[
    { $match:
      {
        YearsCode: {$ne: NaN},
        DevType: {$ne: NaN}
      }
    },
    { $project:
      {
        DevType: "$DevType",
        YearsCodeNumeric: {
          $cond: {
            if: { $eq: ["$YearsCode", "More than 50 years"] },
            then: 50,
            else: {
              $cond: {
                if: { $eq: ["$YearsCode", "Less than 1 year"] },
                then: 0,  // Convert "Less than a year" to 0
                else: { $toInt: "$YearsCode" }  // Convert to integer
              }
            }
          }
        }
      }
    },
    {
      $group: {
        _id: {DevType: "$DevType"},
        YearsCode: { $avg: "$YearsCodeNumeric" }
      }
    },
    { $project: {
        _id: 0,
        DevType: "$_id.DevType",
        AverageYearsCoding: { $round: ["$YearsCode", 2] } ,
      }
    },
    { $sort: {
        "AverageYearsCoding": -1
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "DevType": "Senior Executive (C-Suite, VP, etc.)",
      "AverageYearsCoding": 23.25
    },
    {
      "DevType": "Engineering manager",
      "AverageYearsCoding": 20.98
    },
    {
      "DevType": "Product manager",
      "AverageYearsCoding": 20.11
    },
    {
      "DevType": "Developer Advocate",
      "AverageYearsCoding": 20.01
    },
    {
      "DevType": "Educator",
      "AverageYearsCoding": 19.32
    },
    {
      "DevType": "Developer, desktop or enterprise applications"
        ,
      "AverageYearsCoding": 18.78
    },
    {
      "DevType": "Research & Development role",
      "AverageYearsCoding": 18.78
    },
    {
      "DevType": "Database administrator",
      "AverageYearsCoding": 18.56
    },
    {
```

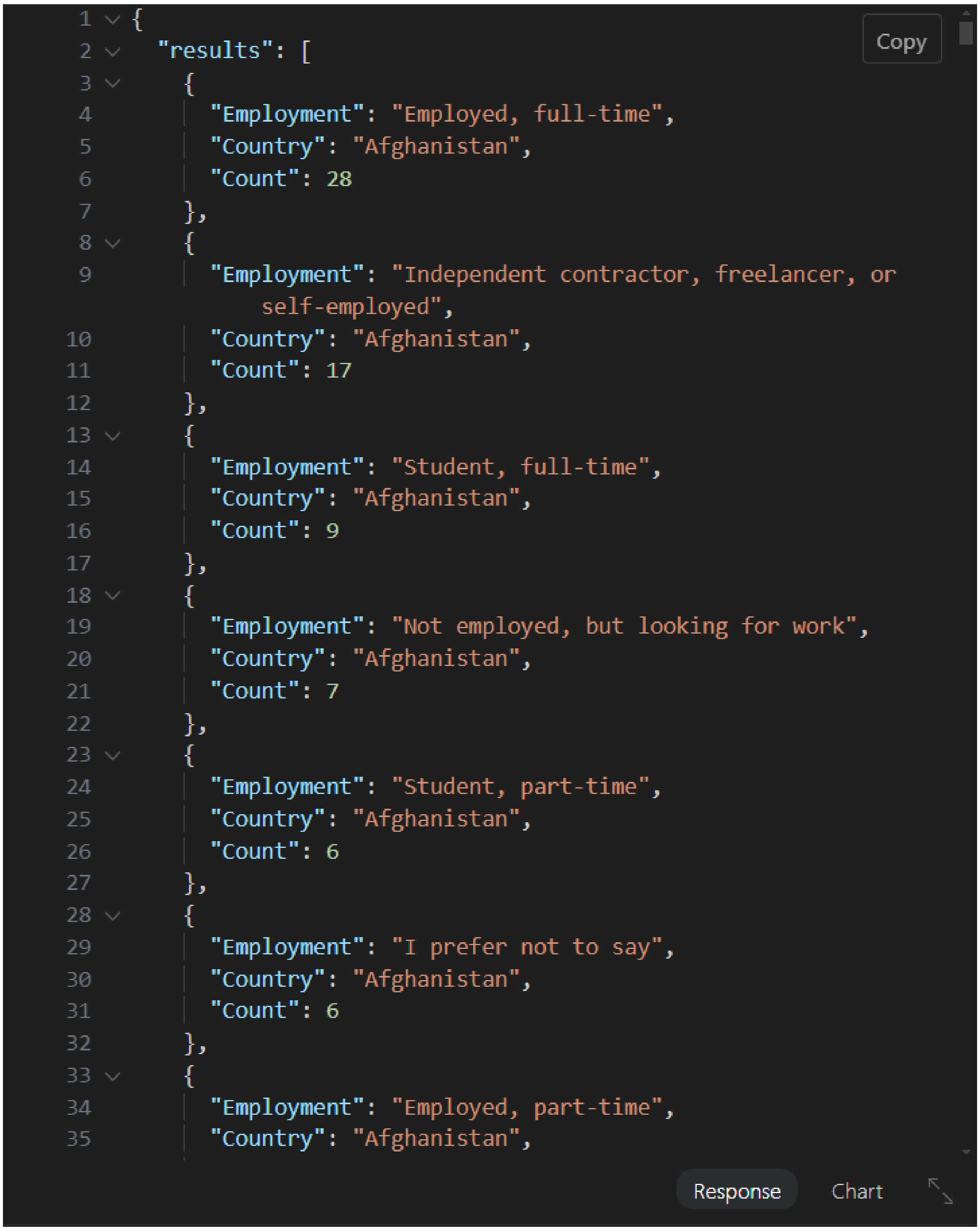
Figure 4.8: Query 8

Query 9 Employment status by country

Description: Insights of the employment status grouped by country
Query:

```
[
    {
      $unwind: "$Employment"
    },
    {
      $match: {
        Employment: {$ne: null},
        Country: {$ne: NaN}
      }
    },
    {
      $group: {
        _id: {
          Employment: '$Employment',
          Country: '$Country'
        },
        count: {$sum: 1}
      }
    },
    {
      $project: {
        _id: 0,
        Employment: '$_id.Employment',
        Country: '$_id.Country',
        Count: '$count'
      }
    },
    {
      $sort: {
        Country: 1
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "Employment": "Employed, full-time",
      "Country": "Afghanistan",
      "Count": 28
    },
    {
      "Employment": "Independent contractor, freelancer, or
        self-employed",
      "Country": "Afghanistan",
      "Count": 17
    },
    {
      "Employment": "Student, full-time",
      "Country": "Afghanistan",
      "Count": 9
    },
    {
      "Employment": "Not employed, but looking for work",
      "Country": "Afghanistan",
      "Count": 7
    },
    {
      "Employment": "Student, part-time",
      "Country": "Afghanistan",
      "Count": 6
    },
    {
      "Employment": "I prefer not to say",
      "Country": "Afghanistan",
      "Count": 6
    },
    {
      "Employment": "Employed, part-time",
      "Country": "Afghanistan",
```

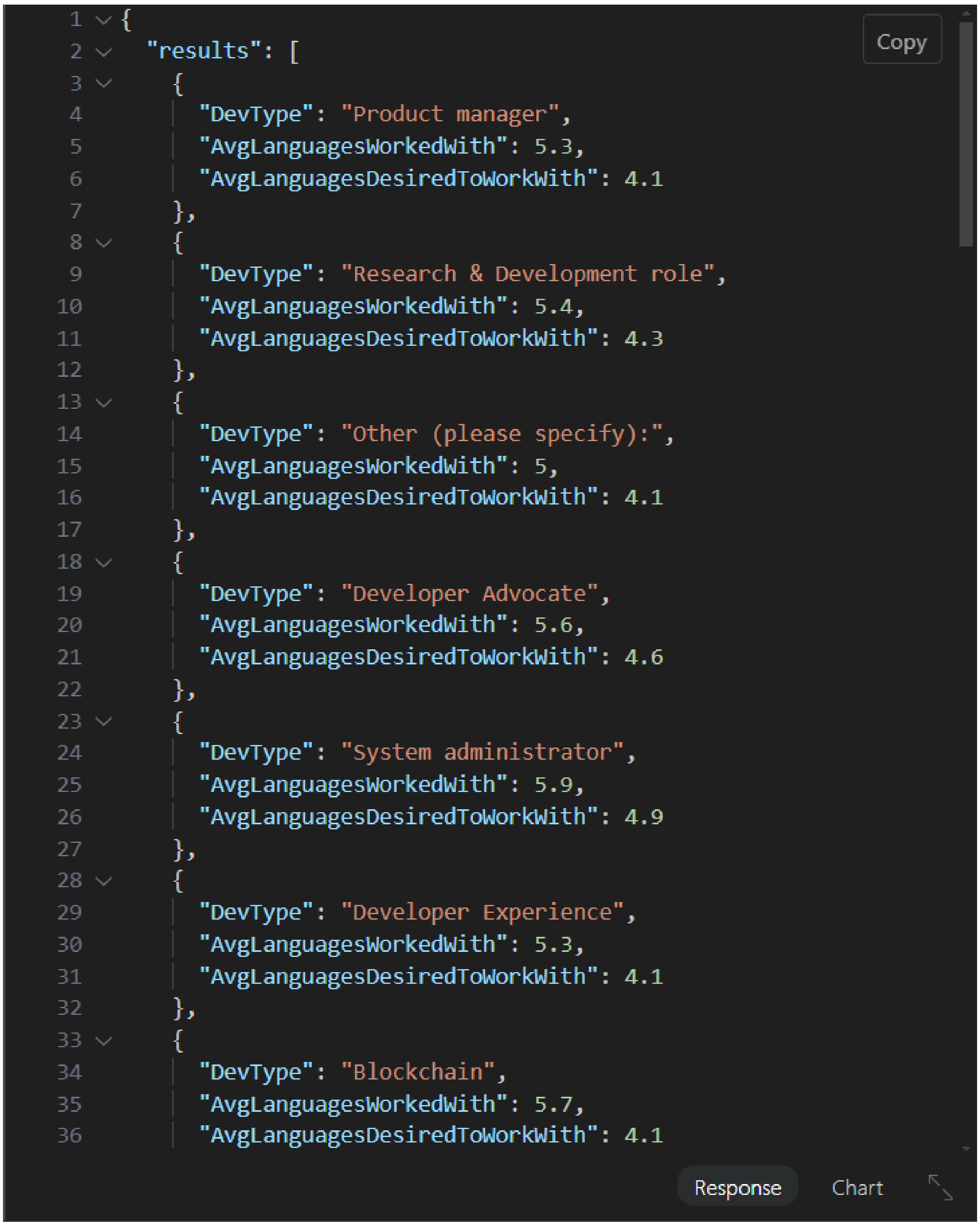
Figure 4.9: Query 9

Query 10 Number of languages worked with and desired to work with by developer type

Description: Average number of languages worked with and desired to work with grouped by developer type
Query:

```
[
   { $match: {
        LanguageHaveWorkedWith: {$ne: null},
        DevType: {$ne: NaN},
      }
    },
    {
      $project: {
        DevType: "$DevType",
        LanguagesWorkedWithCount: {
          $size: "$LanguageHaveWorkedWith"
        },
        LanguagesDesiredToWorkWithCount: {
          $size: "$LanguageWantToWorkWith"
        },
      }
    },
    {
      $group: {
        _id: '$DevType',
        AvgLanguagesWorkedWith: {
          $avg: "$LanguagesWorkedWithCount"
        },
        AvgLanguagesDesiredToWorkWith: {
          $avg: "$LanguagesDesiredToWorkWithCount"
        },
      }
    },
    {
      $project: {
        _id: 0,
        DevType: ’$_id’,
        AvgLanguagesWorkedWith: {
          $round: ["$AvgLanguagesWorkedWith", 1]
        },
        AvgLanguagesDesiredToWorkWith: {
          $round: ["$AvgLanguagesDesiredToWorkWith", 1]
        },
      }
    }
]
```

Result:



```
{
  "results": [
    {
      "DevType": "Product manager",
      "AvgLanguagesWorkedWith": 5.3,
      "AvgLanguagesDesiredToWorkWith": 4.1
    },
    {
      "DevType": "Research & Development role",
      "AvgLanguagesWorkedWith": 5.4,
      "AvgLanguagesDesiredToWorkWith": 4.3
    },
    {
      "DevType": "Other (please specify):",
      "AvgLanguagesWorkedWith": 5,
      "AvgLanguagesDesiredToWorkWith": 4.1
    },
    {
      "DevType": "Developer Advocate",
      "AvgLanguagesWorkedWith": 5.6,
      "AvgLanguagesDesiredToWorkWith": 4.6
    },
    {
      "DevType": "System administrator",
      "AvgLanguagesWorkedWith": 5.9,
      "AvgLanguagesDesiredToWorkWith": 4.9
    },
    {
      "DevType": "Developer Experience",
      "AvgLanguagesWorkedWith": 5.3,
      "AvgLanguagesDesiredToWorkWith": 4.1
    },
    {
      "DevType": "Blockchain",
      "AvgLanguagesWorkedWith": 5.7,
      "AvgLanguagesDesiredToWorkWith": 4.1
```

Figure 4.10: Query 10

5 | Extra

5.1. Kibana Dashboard

To fully show the capabilities of Elasticsearch and simplify data exploration, we created a dashboard using Kibana. Its user-friendly interface enabled visualization and analysis of our dataset, making it easier to identify trends, patterns, and outliers. The dashboard can be found at the following link: `https://f41b6f517c314135be645865ca6e7f27.us-central1.gcp.cloud.es.io/app/dashboards#/view/5dd07599-7a67-445d-b1d6-8dc887e1f3 _g=(refreshInterval:(pause:!f,value:10000),time:(from:now-15y,to:now))&_a=().`

To login you have you click on *Log in with Elasticsearch* and input the following credentials:

```
username: guest
password: 123456789
```

The dashboard should appear as seen in figure 5.1 :

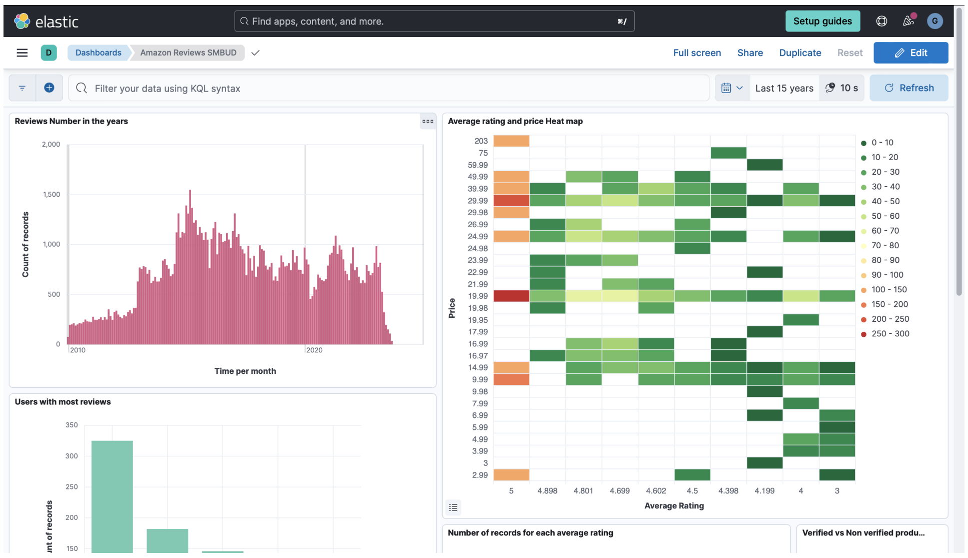


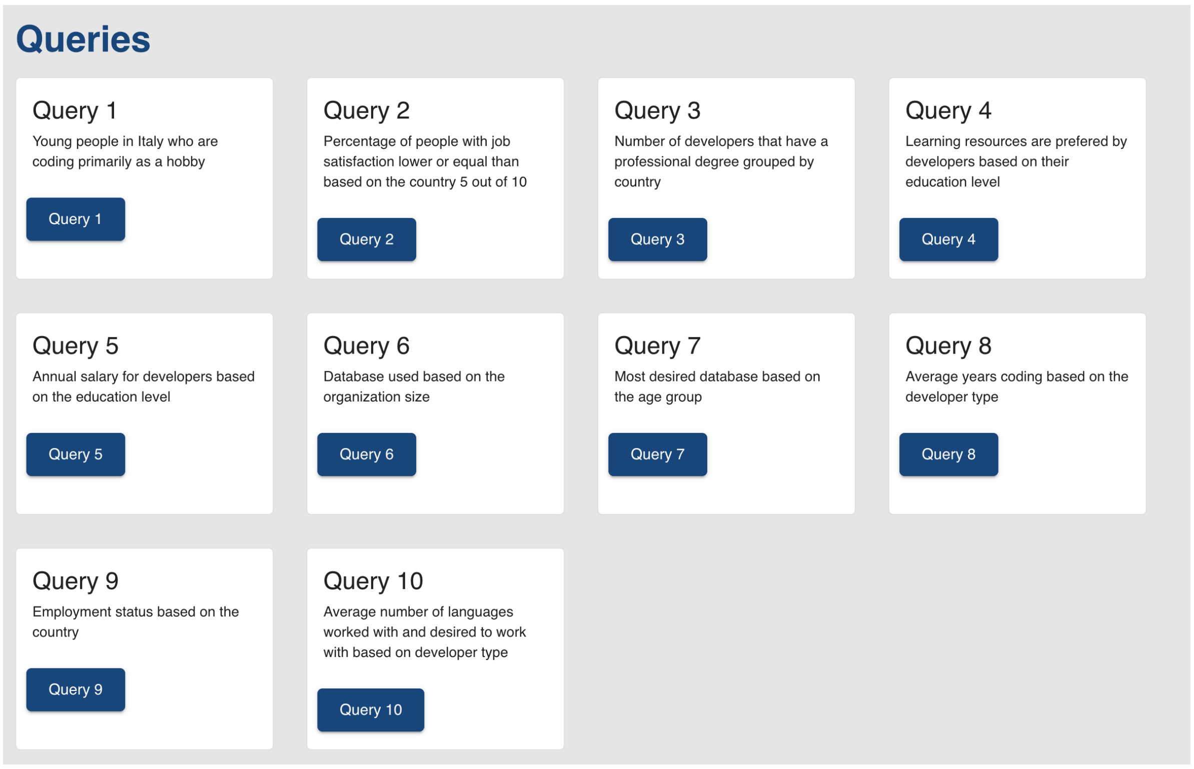
Figure 5.1: The landing page Kibana dashboard we have created for elasticsearch.

5.2. Web Application for MongoDB

We developed a React/Node.js application to visualize MongoDB queries. Its main features are: user-specified queries and deployment on Render. The front-end is deployed in the following link: `https://smbud-project-gwum.onrender.com/`, while the server is deployed here: `https://smbud-project-19y6.onrender.com/`. Please note that the server must be accessed before querying the front-end since Render puts the servers to sleep after a period of inactivity. When the server is queried, it should display the following message after a few minutes:

```
"Successfully connected to mongoDB"
```

Once the previous message appears, the front-end can be queried. The landing page user interface can be seen in figure 5.2.


Queries

Query 1
Young people in Italy who are coding primarily as a hobby

Query 2
Percentage of people with job satisfaction lower or equal than based on the country 5 out of 10

Query 3
Number of developers that have a professional degree grouped by country

Query 4
Learning resources are prefered by developers based on their education level

Query 5
Annual salary for developers based on the education level

Query 6
Database used based on the organization size

Query 7
Most desired database based on the age group

Query 8
Average years coding based on the developer type

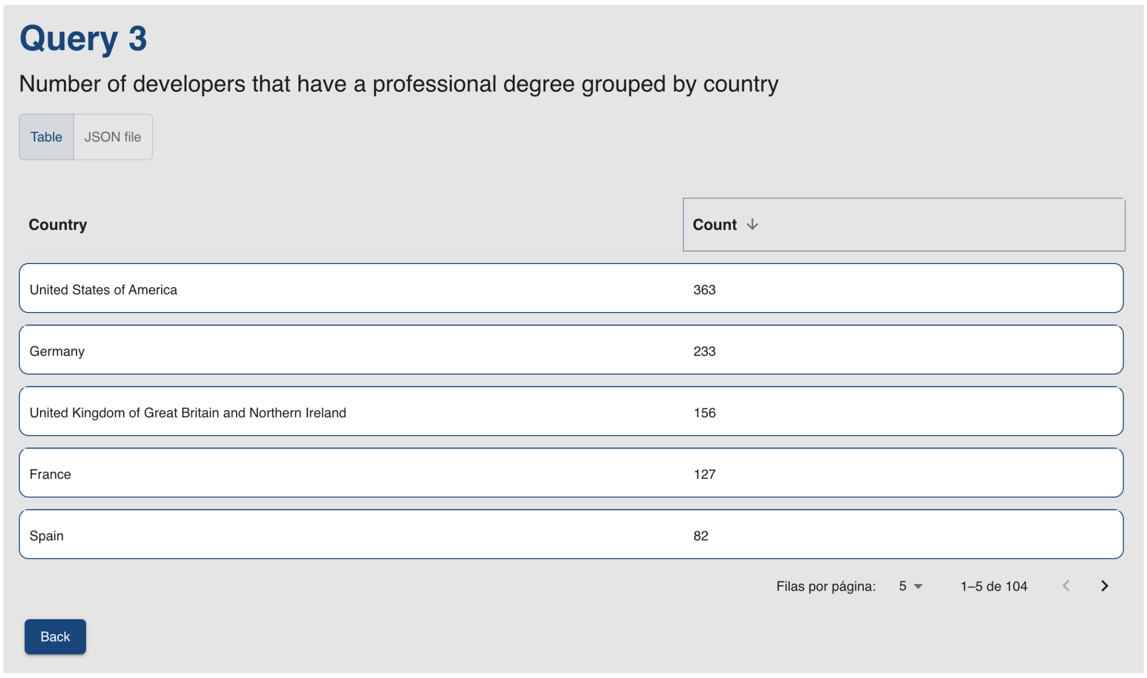
Query 9
Employment status based on the country

Query 10
Average number of languages worked with and desired to work with based on developer type

Figure 5.2: The landing page of the React application to visualize the Mongodb queries.

The queries can be accessed one by one. We implemented custom sorting for the attributes returned by the query, as seen in figure 5.3. In addition, the queries can be visualized either in a tabular form or in a JSON format, as shown in figure 5.4.

The code for our entire project and implementation can be found on github: `https://github.com/sebastianballesteros/SMBUD-Project`. Permission may be required.



Query 3

Number of developers that have a professional degree grouped by country

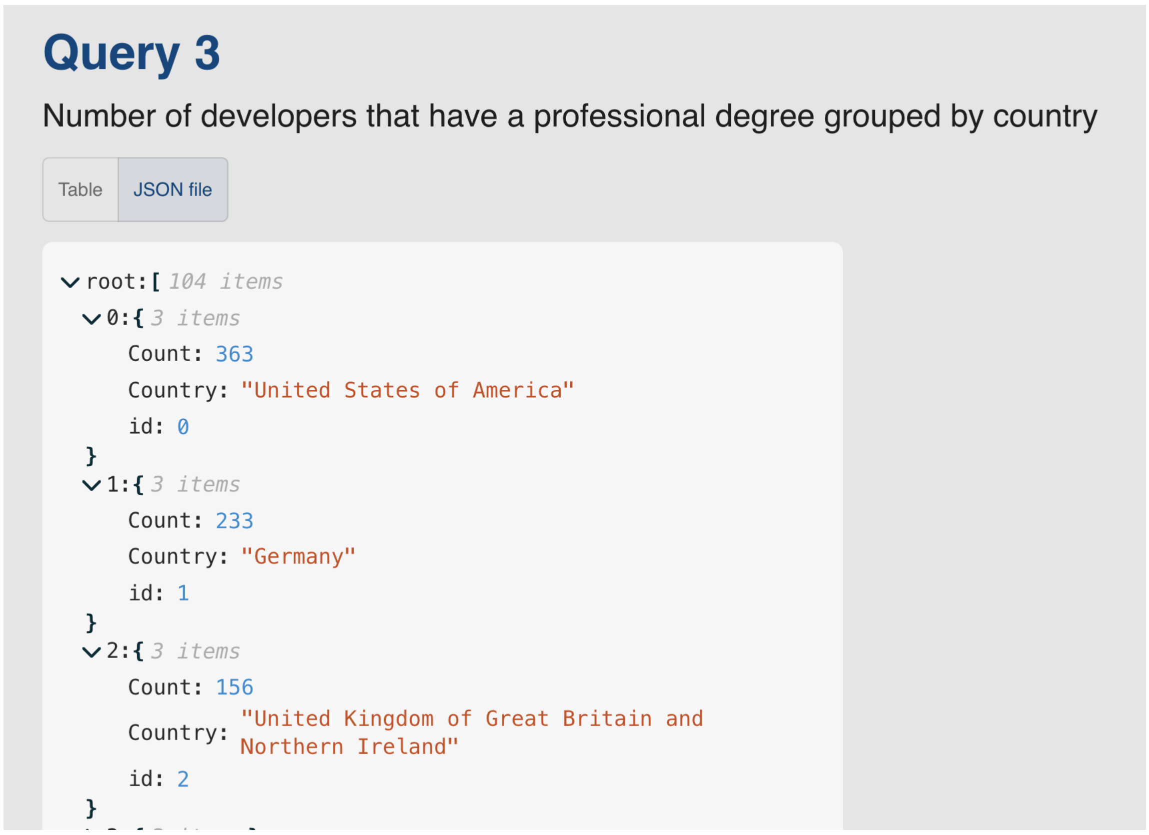
Table | JSON file

Country	Count ↓
United States of America	363
Germany	233
United Kingdom of Great Britain and Northern Ireland	156
France	127
Spain	82

Filas por página: 5 1–5 de 104

Back

Figure 5.3: Query number three sorted by the aggregate count.



Query 3

Number of developers that have a professional degree grouped by country

Table | JSON file

```
root:[ 104 items
  0:{ 3 items
     Count: 363
     Country: "United States of America"
     id: 0
  }
  1:{ 3 items
     Count: 233
     Country: "Germany"
     id: 1
  }
  2:{ 3 items
     Count: 156
     Country: "United Kingdom of Great Britain and
              Northern Ireland"
     id: 2
  }
```

Figure 5.4: Query number three viewed in a JSON format