

Pokročilé databázové technológie

Zadanie 5 – Elasticsearch

Marek Adamovič

Obsah

1. Rozbehánie inštancií	3
2. Index pre Tweety	5
3. Mapping pre normalizované dáta	6
4. Vlastné analyzéry	8
5. Bulk import.....	12
6. Import	14
7. Uzly.....	16
8. Script	17
9. Veľký import.....	20
10. Vyhľadávanie.....	23

1. Rozbehanie inštancií

Otázka:

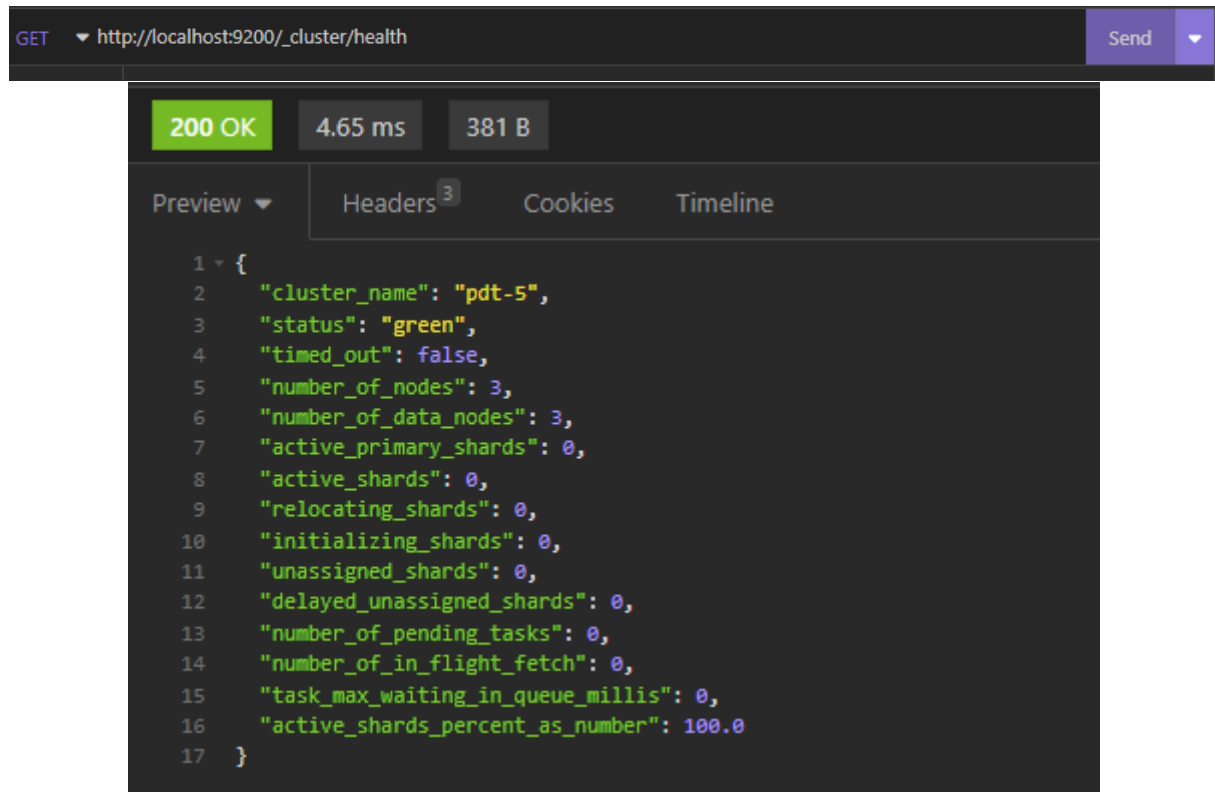
Rozbehajte si 3 inštancie Elasticsearch-u.

Odpoveď:

Inštancie sme vytvárali na windowse. Stiahli sme si Elasticsearch a rozbalili ho do troch zložiek (každá zložka tvorí jednu inštanciu). Následne sme nakonfigurovali elasticsearch.yml súbor (obrázok č.1) v každej z troch zložiek a postupne spustili jednotlivé inštancie pomocou súboru elasticsearch.bat. Konfigurácie sa líšia minimálne, len v názve uzlu a portoch. Na obrázku č.2 vidíme náš cluster, ktorý obsahuje 3 uzly.

```
elasticsearch-node1 > config > elasticsearch.yml > ...
1  ##### my configuration #####
2  cluster.name: pdt-5
3  node.name: 'node-1'
4  http.port: 9200
5  transport.port: 9300
6  discovery.seed_hosts:
7    - "127.0.0.1:9300"
8    - "127.0.0.1:9301"
9    - "127.0.0.1:9302"
10 cluster.initial_master_nodes: ["node-1"]
11
12 # Disable security features
13 xpack.security.enabled: false
14 xpack.security.enrollment.enabled: false
15
16 ingest.geoip.downloader.enabled: false
17 ##### end of my configuration #####
```

Obrázok 1 Konfigurácia prvého uzlu v clusteri



Obrázok 2 Kontrola clusteru

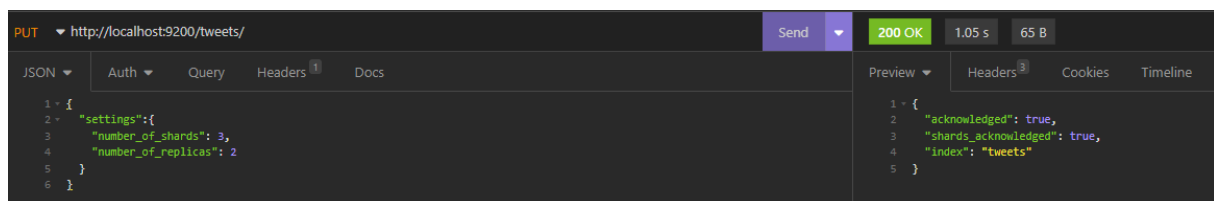
2. Index pre Tweety

Otázka:

Vytvorte index pre Tweety, ktorý bude mať “optimálny” počet shardov a replík pre 3 nódy (aby tam bola distribúcia dotazov vo vyhľadávaní, aj distribúcia uložených dát).

Odpoveď:

Rozhodli sme sa zvoliť počet shardov 3, aby sme vedeli naplno využiť paralelizmus pri 3 uzloch. Počet replík sme zvolili 2, keďže nemá zmysel dávať väčšie číslo ako počet uzlov mínus jedna (keďže jedny dáta máme vždy defaultne).



Obrázok 3 Vytvorenie indexu

3. Mapping pre normalizované dáta

Otázka:

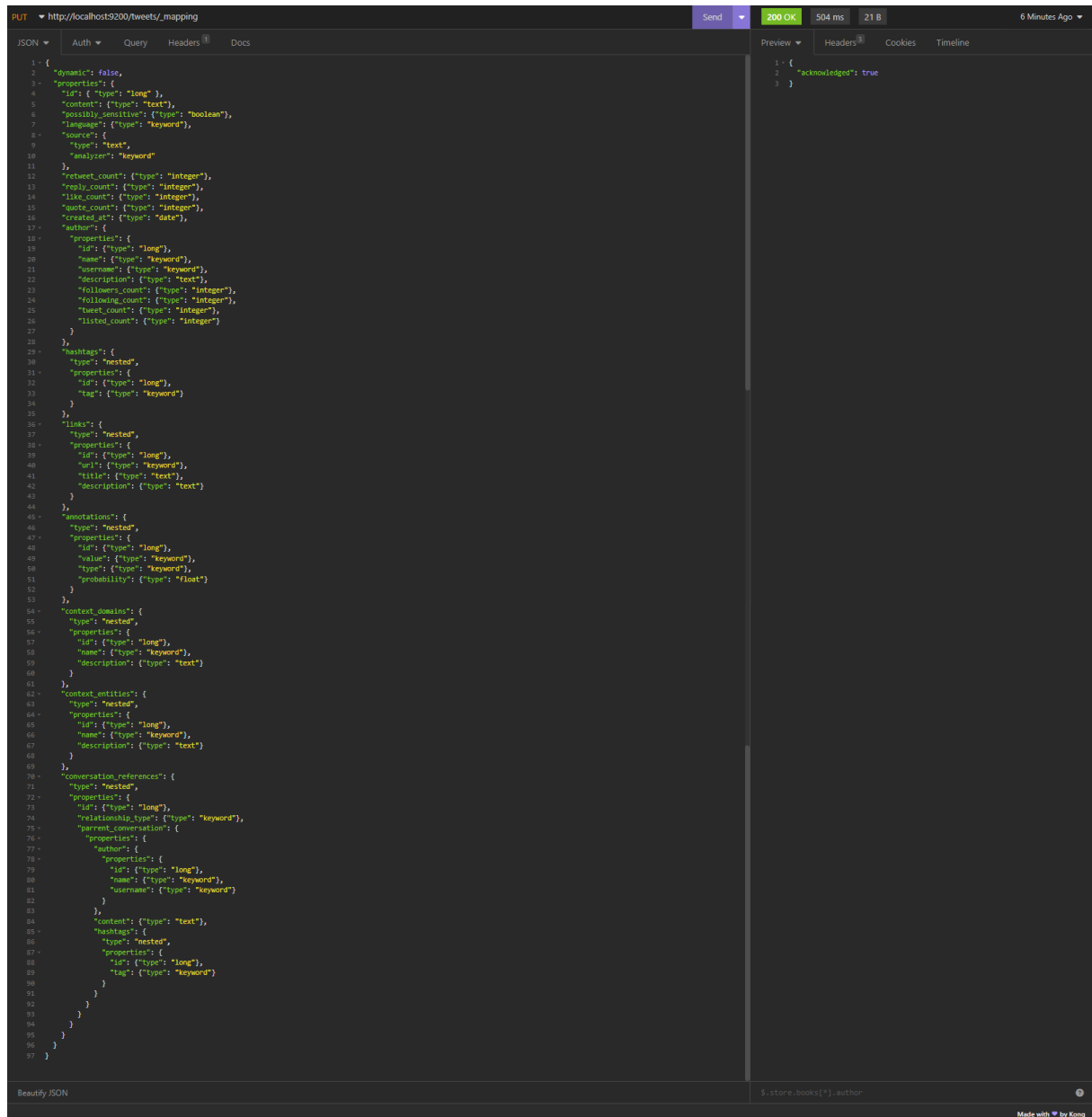
Vytvorte mapping pre normalizované dáta z Postgresu (denormalizujte ich) – Každý Tweet teda musí obsahovať údaje rovnaké ako máte už uložené v PostgreSQL (všetky tabuľky). Dbajte na to, aby ste vytvorili polia v správnom dátovom type (polia ktoré má zmysel analyzovať analyzujte správne, tie ktoré nemá, aby neboli zbytočne analyzované (keyword analyzer)) tak aby index nebol zbytočne veľký, pozor na nested – treba ho použiť správne. Mapovanie musí byť striktné. Čo sa týka väzieb cez referencies – pre ne zaindexujte type vzťahu, id, autor (id, name, username), content a hashtags.

Odpoveď:

Pre striktné mapovanie sme použili dynamic -> false, čo nám zakazuje pridávať nové polia do indexu. Typ nested sme použili len pre také polia, ktoré majú typ vzťahu ku tweetom many-to-one (to znamená, že na jednu konverzáciu ich môže byť viacero). Pri vzťahu one-to-one alebo one-to-many nebolo potrebné využívať typ nested. Niekoľkokrát sme použili typ keyword (v úlohe 4 sme ho potom niekde vymenili za text kvôli vlastným analyzátom), keďže tento typ má už daný analyzér keyword a jeho jediné obmedzenie je maximálna dĺžka stringu 32766 znakov. Preto sme ho využívali pre polia, ktoré túto dĺžku nemôžu presiahnuť a zároveň ich nechceme analyzovať (keďže typ text sa automaticky analyzuje). Ak sme mali polia, ktoré môžu presahovať spomenutú dĺžku a napriek tomu ich nechceme analyzovať, použili sme typ text a explicitne sme nastavili analyzér na hodnotu keyword. Na nasledujúcich obrázkoch uvádzame naše mapovanie.

Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií



Obrázok 4 Ukážka mapovania

4. Vlastné analyzéry

Otázka:

Pre index tweets vytvorte 3 vlastné analyzéry (v settings) nasledovne:

a. Analyzér "englando". Tento analyzér bude obsahovať nasledovné:

- i. filtre: english_possessive_stemmer, lowercase, english_stop, english_stemmer,
- ii. char_filter: html_strip
- iii. tokenizer: štandardný - ukážku nájdete na stránke elastic.co pre anglický analyzér

b. Analyzér custom_ngram:

- i. filtre: lowercase, asciifolding, filter_ngrams (definujte si ho sami na rozmedzie 1- 10)
- ii. char_filter: html_strip
- iii. tokenizer: štandardný

c. Analyzér custom_shingles:

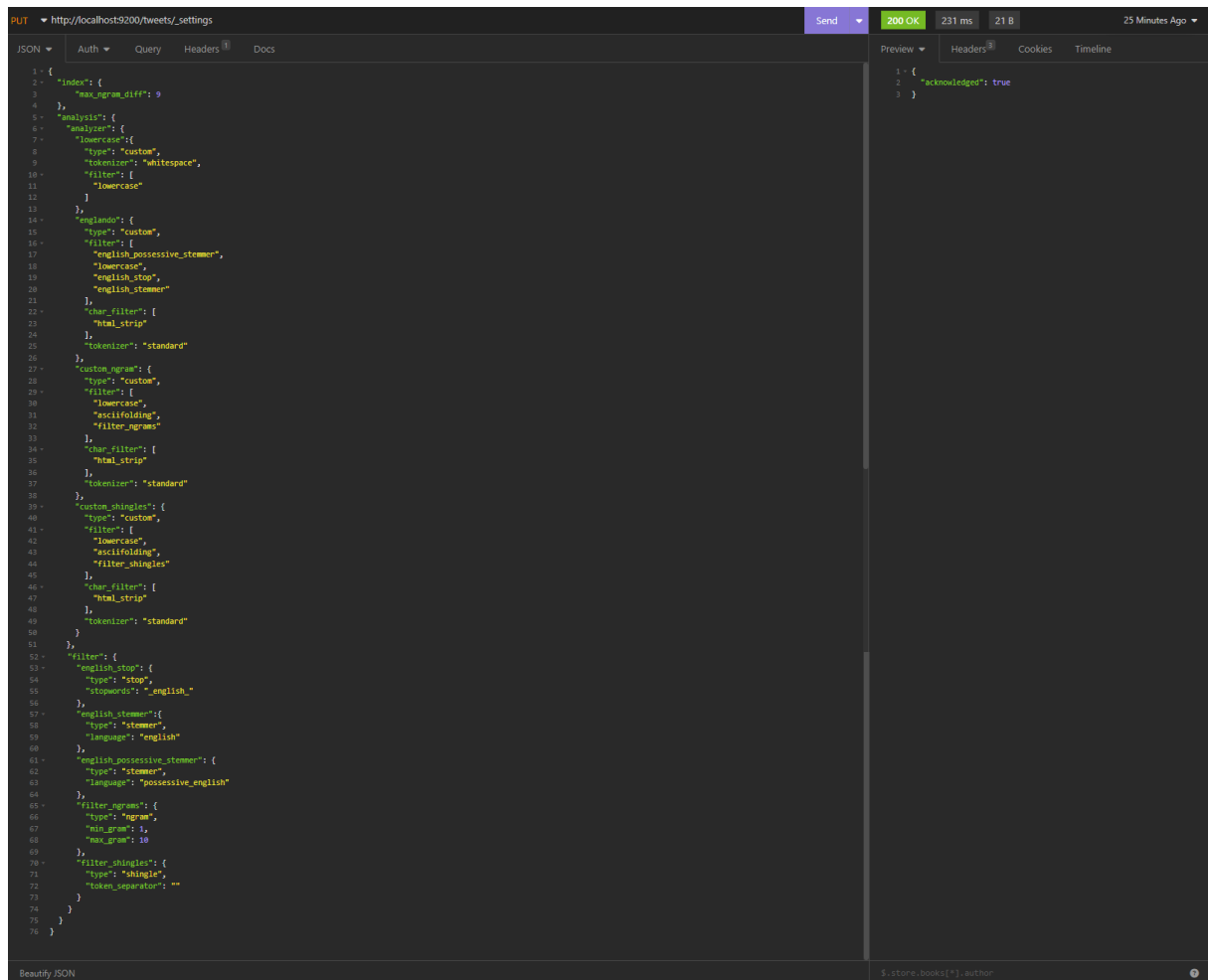
- i. filtre: lowercase, asciifolding, filter_shingles (definujte si ho sami a dajte token_separator: "")
- ii. char_filter: html_strip
- iii. tokenizer: štandardný

d. Do mapovania pridajte:

- i. každý anglický text (rátajme že každý tweet a description u autora je primárne v angličtine) nech je analyzovaný novým analyzérom "englando"
- ii. Priradte analyzery
 - 1. a. author.name nech má aj mapovania pre custom_ngram, a custom_shingles
 - 2. b. author. screen_name nech má aj custom_ngram,
 - 3. c. author.description nech má aj custom_shingles. Toto platí aj pre mentions, ak tam tie záznamy máte.
- iii. Hashtagy indexujte ako lowercase

Odpoveď:

Najskôr sme vytvorili analyzéry v settings (obrázok č.5). Aby sme ich vytvorili, museli sme pozastaviť index pomocou POST dopytu http://localhost:9200/tweets/_close. Po vytvorení sme ho znovu spustili pomocou http://localhost:9200/tweets/_open. Následne sme upravili pôvodný mapping, aby využíval dané analyzéry tam, kde treba (obrázok č.6). Tým pádom mapping musíme vytvárať až po vytvorení našich analyzéro.



```
PUT http://localhost:9200/tweets/_settings

{
  "index": {
    "max_ngram_diff": 9
  },
  "analysis": {
    "tokenizer": {
      "lowercase": {
        "type": "custom",
        "tokenizer": "whitespace",
        "filter": [
          "lowercase"
        ]
      },
      "english": {
        "type": "custom",
        "filter": [
          "english_possessive_stemmer",
          "lowercase",
          "english_stop",
          "english_stemmer"
        ],
        "char_filter": [
          "html_strip"
        ],
        "tokenizer": "standard"
      },
      "custom_ngram": {
        "type": "custom",
        "filter": [
          "lowercase",
          "asciifolding",
          "filter_ngrams"
        ],
        "char_filter": [
          "html_strip"
        ],
        "tokenizer": "standard"
      },
      "custom_shingles": {
        "type": "custom",
        "filter": [
          "lowercase",
          "asciifolding",
          "filter_shingles"
        ],
        "char_filter": [
          "html_strip"
        ],
        "tokenizer": "standard"
      },
      "filter": {
        "english_stop": {
          "type": "stop",
          "stopwords": "_english_"
        },
        "english_stemmer": {
          "type": "stemmer",
          "language": "english"
        },
        "english_possessive_stemmer": {
          "type": "stemmer",
          "language": "possessive_english"
        },
        "filter_ngrams": {
          "type": "ngram",
          "min_ngram": 1,
          "max_ngram": 10
        },
        "filter_shingles": {
          "type": "shingle",
          "token_separator": ""
        }
      }
    }
  }
}
```

```
{
  "acknowledged": true
}
```

Obrázok 5 Vytvorenie custom analyzéro

Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií

PUT

http://localhost:9200/tweets/_mapping

Send

200 OK

337 ms

21 B

Just Now

JSON

Auth

Query

Headers

Docs

```
1 {
2   "dynamic": false,
3   "properties": {
4     "id": { "type": "long" },
5     "content": {
6       "type": "text",
7       "analyzer": "englando"
8     },
9     "possibly_sensitive": { "type": "boolean" },
10    "language": { "type": "keyword" },
11    "source": {
12      "type": "text",
13      "analyzer": "keyword"
14    },
15    "retweet_count": { "type": "integer" },
16    "reply_count": { "type": "integer" },
17    "like_count": { "type": "integer" },
18    "quote_count": { "type": "integer" },
19    "created_at": { "type": "date" },
20    "author": {
21      "properties": {
22        "id": { "type": "long" },
23        "name": {
24          "type": "text",
25          "analyzer": "englando",
26          "fields": {
27            "name_custom_ngram": {
28              "type": "text",
29              "analyzer": "custom_ngram"
30            },
31            "name_custom_shingles": {
32              "type": "text",
33              "analyzer": "custom_shingles"
34            }
35          }
36        },
37        "username": {
38          "type": "text",
39          "analyzer": "englando",
40          "fields": {
41            "username_custom_ngram": {
42              "type": "text",
43              "analyzer": "custom_ngram"
44            }
45          }
46        },
47        "description": {
48          "type": "text",
49          "analyzer": "englando",
50          "fields": {
51            "description_custom_shingles": {
52              "type": "text",
53              "analyzer": "custom_shingles"
54            }
55          }
56        },
57        "followers_count": { "type": "integer" },
58        "following_count": { "type": "integer" },
59        "tweet_count": { "type": "integer" },
60        "listed_count": { "type": "integer" }
61      }
62    },
63    "hashtags": [
64      {
65        "type": "nested",
66        "properties": {
67          "id": { "type": "long" },
68          "tag": {
69            "type": "text",
70            "analyzer": "lowercase"
71          }
72        }
73      },
74      {
75        "type": "nested",
76        "properties": {
77          "id": { "type": "long" },
78          "url": { "type": "keyword" },
79          "title": {
80            "type": "text",
81            "analyzer": "englando"
82          },
83          "description": {
84            "type": "text",
85            "analyzer": "englando"
86          }
87        }
88      },
89      {
90        "type": "nested",
91        "properties": {
92          "id": { "type": "long" },
93          "value": { "type": "keyword" },
94          "type": { "type": "keyword" },
95          "probability": { "type": "float" }
96        }
97      }
98    ],
99    "context_domain": {
100     "type": "nested",
101     "properties": {
102       "id": { "type": "long" },
103       "name": { "type": "keyword" },
104       "description": {
105         "type": "text",
```

Preview

Headers

Cookies

Timeline

```
1 {
2   "acknowledged": true
3 }
```

```
184     "analyzer": "englando"
185   }
186 }
187 },
188 "context_entities": {
189   "type": "nested",
190   "properties": {
191     "id": {"type": "long"},
192     "name": {"type": "keyword"},
193     "description": {
194       "type": "text",
195       "analyzer": "englando"
196     }
197   }
198 },
199 "conversation_references": {
200   "type": "nested",
201   "properties": {
202     "id": {"type": "long"},
203     "type": {"type": "keyword"},
204     "parent_conversation": {
205       "properties": {
206         "custom": {
207           "properties": {
208             "id": {"type": "long"},
209             "name": {
210               "type": "text",
211               "analyzer": "englando",
212               "fields": {
213                 "name_custom_ngram": {
214                   "type": "text",
215                   "analyzer": "custom_ngram"
216                 },
217                 "name_custom_shingles": {
218                   "type": "text",
219                   "analyzer": "custom_shingles"
220                 }
221               }
222             },
223             "username": {
224               "type": "text",
225               "analyzer": "englando",
226               "fields": {
227                 "username_custom_ngram": {
228                   "type": "text",
229                   "analyzer": "custom_ngram"
230                 }
231               }
232             }
233           }
234         }
235       }
236     },
237     "content": {
238       "type": "text",
239       "analyzer": "englando"
240     },
241     "hashtags": {
242       "type": "nested",
243       "properties": {
244         "id": {"type": "long"},
245         "tag": {
246           "type": "text",
247           "analyzer": "lowercase"
248         }
249       }
250     }
251   }
252 }
253 },
254 },
255 },
256 },
257 },
258 },
259 },
260 },
261 },
262 },
263 },
264 },
265 },
266 },
267 },
268 },
269 },
270 },
271 },
272 },
273 },
274 }
```

```
2 "acknowledged": true
3 }
```

Obrázok 6 Upravený mapping

5. Bulk import

Otázka:

Vytvorte bulk import pre vaše normalizované Tweety.

Odpoveď:

Pre bulk import sme si najskôr vytvorili indexy (obrázok č.7) pre všetky cudzie kľúče. Následne sme napísali dopyt (obrázok č.8), ktorý denormalizuje dáta zo všetkých tabuliek a spojí ich do jedného JSON stringu na jeden riadok.

```
76 -- create indexes for foreign keys --
77 CREATE INDEX conversations_index ON conversations USING btree(author_id);
78 CREATE INDEX hashtag_index_1 ON conversation_hashtags USING btree(conversation_id);
79 CREATE INDEX hashtag_index_2 ON conversation_hashtags USING btree(hashtag_id);
80 CREATE INDEX link_index ON links USING btree(conversation_id);
81 CREATE INDEX annotations_index ON annotations USING btree(conversation_id);
82 CREATE INDEX context_annotations_index_1 ON context_annotations USING btree(conversation_id);
83 CREATE INDEX context_annotations_index_2 ON context_annotations USING btree(context_domain_id);
84 CREATE INDEX context_annotations_index_3 ON context_annotations USING btree(context_entity_id);
85 CREATE INDEX conversation_references_index_1 ON conversation_references USING btree(conversation_id);
86 CREATE INDEX conversation_references_index_2 ON conversation_references USING btree(parent_id);
```

Obrázok 7 vytvorenie indexov

Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií

Query Query History

```
1 COPY(
2   SELECT json_build_object(
3     'id', c.id,
4     'content', c.content,
5     'possibly_sensitive', c.possibly_sensitive,
6     'language', c.language,
7     'source', c.source,
8     'retweet_count', c.retweet_count,
9     'reply_count', c.reply_count,
10    'like_count', c.like_count,
11    'quote_count', c.quote_count,
12    'created_at', c.created_at,
13    'author.id', author.id,
14    'author.name', author.name,
15    'author.username', author.username,
16    'author.description', author.description,
17    'author.followers_count', author.followers_count,
18    'author.following_count', author.following_count,
19    'author.tweet_count', author.tweet_count,
20    'author.listed_count', author.listed_count,
21    'hashtags', (
22      SELECT json_agg(json_build_object('id', h.id, 'tag', h.tag))
23      FROM hashtags h
24      JOIN conversation_hashtags ch ON ch.hashtag_id = h.id
25      WHERE ch.conversation_id = c.id
26    ),
27    'links', (
28      SELECT json_agg(json_build_object('id', l.id, 'url', l.url, 'title', l.title, 'description', l.description))
29      FROM links l
30      WHERE l.conversation_id = c.id
31    ),
32    'annotations', (
33      SELECT json_agg(json_build_object('id', a.id, 'value', a.value, 'type', a.type, 'probability', a.probability))
34      FROM annotations a
35      WHERE a.conversation_id = c.id
36    ),
37    'context_domains', (
38      SELECT json_agg(json_build_object('id', cd.id, 'name', cd.name, 'description', cd.description))
39      FROM context_domains cd
40      JOIN context_annotations ca ON ca.context_domain_id = cd.id
41      WHERE ca.conversation_id = c.id
42    ),
43    'context_entities', (
44      SELECT json_agg(json_build_object('id', ce.id, 'name', ce.name, 'description', ce.description))
45      FROM context_entities ce
46      JOIN context_annotations ca ON ca.context_domain_id = ce.id
47      WHERE ca.conversation_id = c.id
48    ),
49    'conversation_references', (
50      SELECT json_agg(json_build_object(
51        'id', cr.id,
52        'type', cr.type,
53        'parent_conversation.author.id', author2.id,
54        'parent_conversation.author.name', author2.name,
55        'parent_conversation.author.username', author2.username,
56        'parent_conversation.content', c2.content,
57        'parent_conversation.hashtags', (
58          SELECT json_agg(json_build_object('id', h.id, 'tag', h.tag))
59          FROM conversation_hashtags ch
60          JOIN hashtags h ON ch.hashtag_id = h.id
61          WHERE ch.conversation_id = c2.id
62        ))
63      )
64      FROM conversation_references cr
65      JOIN conversations c2 ON c2.id = cr.parent_id
66      JOIN authors author2 ON author2.id = c2.author_id
67      WHERE cr.conversation_id = c.id
68    )
69  )
70  FROM conversations c
71  JOIN authors author ON author.id = c.author_id
72  LIMIT 5000
73 ) TO 'D:\skola2022_2023\PDT\zadanie5\file.json' WITH (FORMAT CSV, QUOTE ' ');
```

Data output Messages Notifications

COPY 5000

Query returned successfully in 3 min 38 secs.

Total rows: 1 of 1 Query complete 00:03:38.201 Ln 59, Col 30

Obrázok 8 Dopyt denormalizujúci dáta z tabuliek

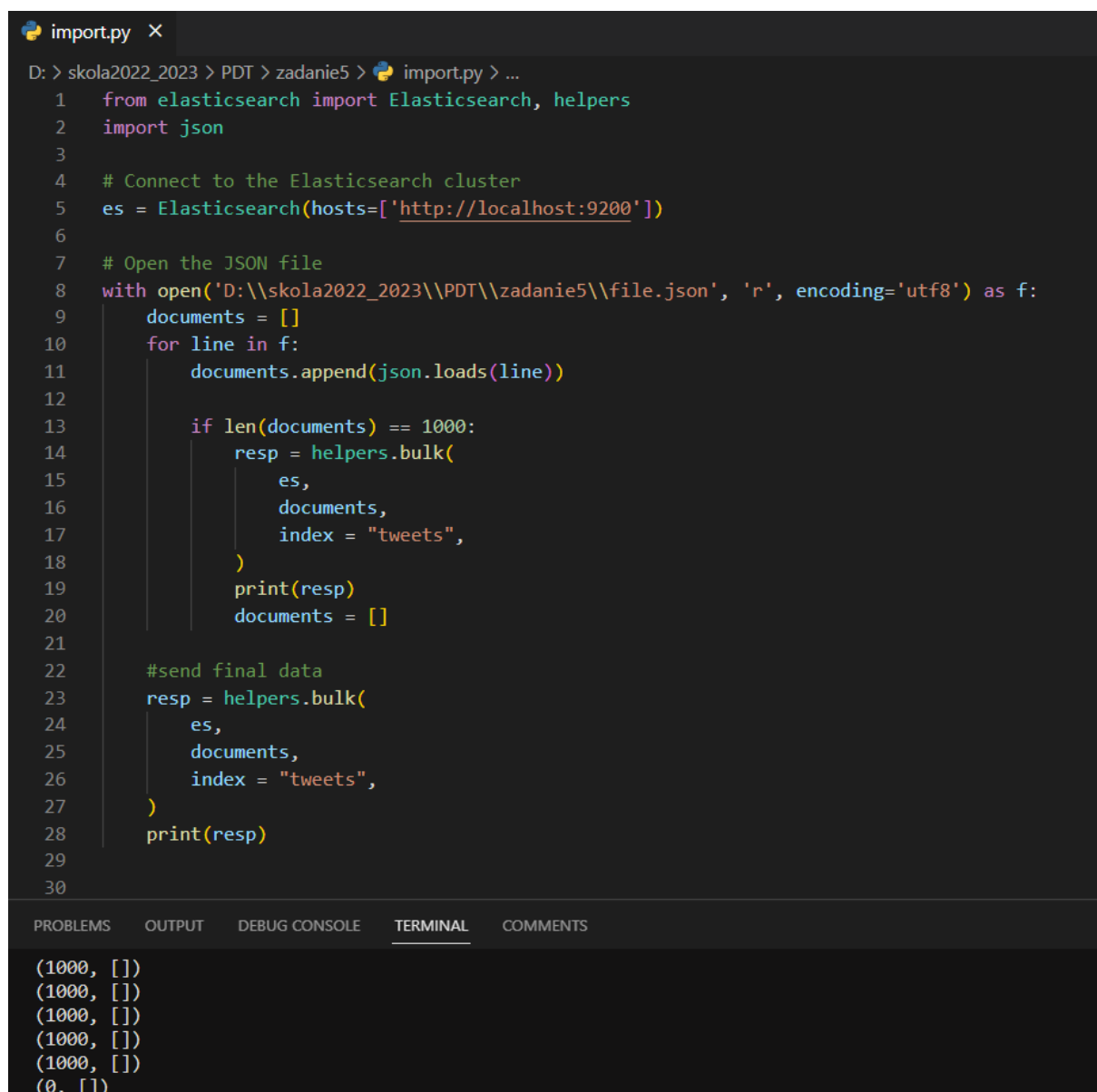
6. Import

Otázka:

Importujete dáta do Elasticsearchu prvych 5000 tweetov.

Odpoveď:

Ked' sme mali tieto dáta v json súbore, importovali sme ich do elasticu pomocou python scriptu (obrázok č.9) využívajúcim bulk api po batchoch o veľkosti 1000. Následne sme skontrolovali (obrázok č.10), či sedí počet dát v elasticsearch pomocou search dopytu.



```
import.py X
D: > skola2022_2023 > PDT > zadanie5 > import.py > ...
1  from elasticsearch import Elasticsearch, helpers
2  import json
3
4  # Connect to the Elasticsearch cluster
5  es = Elasticsearch(hosts=['http://localhost:9200'])
6
7  # Open the JSON file
8  with open('D:\\skola2022_2023\\PDT\\zadanie5\\file.json', 'r', encoding='utf8') as f:
9      documents = []
10     for line in f:
11         documents.append(json.loads(line))
12
13     if len(documents) == 1000:
14         resp = helpers.bulk(
15             es,
16             documents,
17             index = "tweets",
18         )
19         print(resp)
20         documents = []
21
22     #send final data
23     resp = helpers.bulk(
24         es,
25         documents,
26         index = "tweets",
27     )
28     print(resp)
29
30
```

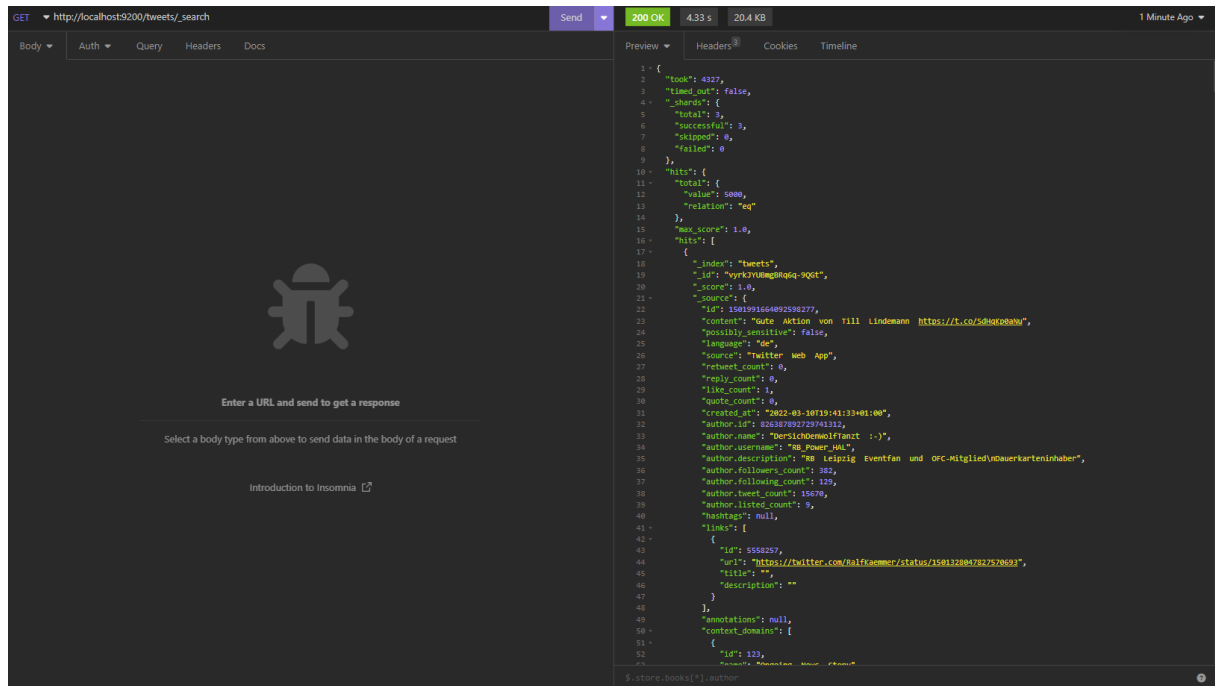
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
(1000, [])
(1000, [])
(1000, [])
(1000, [])
(1000, [])
(0, [])
```

Obrázok 9 Python script na importovanie záznamov do Elasticsearch

Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií



Obrázok 10 Kontrola počtu vložených dát

7. Uzly

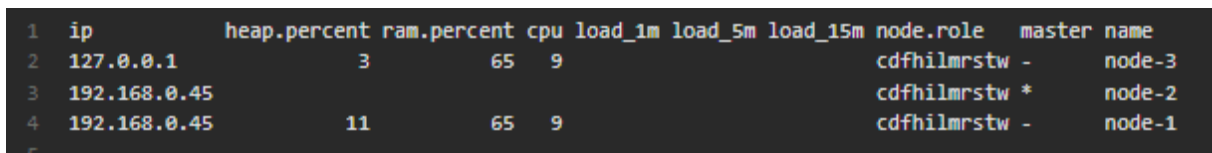
Otázka:

Experimentujte s nódami, a zistíte koľko nódov musí bežať (a ktoré) aby vám Elasticsearch vedel pridávať dokumenty, mazať dokumenty, prezerať dokumenty a vyhľadávať nad nimi? Dá sa nastaviť Elastic tak, aby mu stačil jeden nód? Čo je dôvodom toho že existuje nejaké kvórum?

Odpoveď:

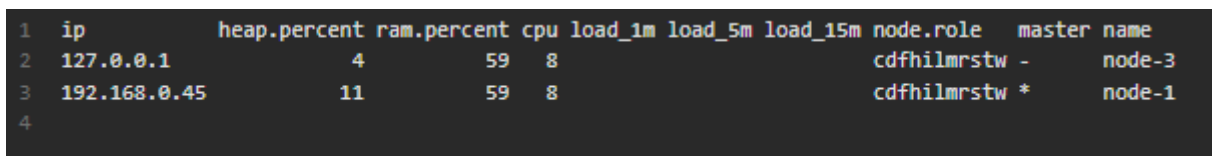
3 uzly – všetko ide

2 uzly – skúsili sme vymazať aj master uzol, aj nie-master uzol .. v oboch prípadoch všetko išlo. Keď sme vypli uzol, ktorý bol master (obrázok č.11), tak iný uzol bol vybraný za mastera (obrázok č.12)



	ip	heap.percent	ram.percent	cpu	load_1m	load_5m	load_15m	node.role	master	name
2	127.0.0.1	3	65	9				cdfhilmrstw	-	node-3
3	192.168.0.45							cdfhilmrstw	*	node-2
4	192.168.0.45	11	65	9				cdfhilmrstw	-	node-1

Obrázok 11 Vypnutie master uzla



	ip	heap.percent	ram.percent	cpu	load_1m	load_5m	load_15m	node.role	master	name
2	127.0.0.1	4	59	8				cdfhilmrstw	-	node-3
3	192.168.0.45	11	59	8				cdfhilmrstw	*	node-1
4										

Obrázok 12 Iný uzol zvolený za master

1 uzol – ak spustíme samostatne jeden uzol, tak nefunguje nič, nevieme spraviť ani dopyt na cluster health. Ak spustíme 2 uzly a potom jeden vypneme, cluster taktiež prestáva byť funkčný. Keď sa pozrieme na logy uzlu, vidíme, že na zvolenie mastera treba aspoň 2 uzly.

0 uzlov – nefunguje nič 🐼

Aby zostal cluster k dispozícii, nesmieme stratiť polovicu alebo viacej uzlov, ktoré môžu byť mastermi. V našej konfigurácii to znamená to, na čo sme prišli -> maximálne môžeme stratiť len jeden uzol, 2 musia byť vždy dostupné. Elastic sa dá nastaviť aj bez clusteru, aby fungoval len na jednom uzle. Kvórum v clusteri je potrebné, aby sme zachovali funkčnosť clusteru aj v prípade, že niektoré uzly spadnú. Vďaka tomu, že kvórum musí odsúhlasiť dôležité akcie, vieme zabezpečiť konzistenciu dát a zabrániť prípadnej strate dát. Pri distribuovanej architektúre potrebujeme takýto alebo obdobný mechanizmus pre jej správne fungovanie.

8. Script

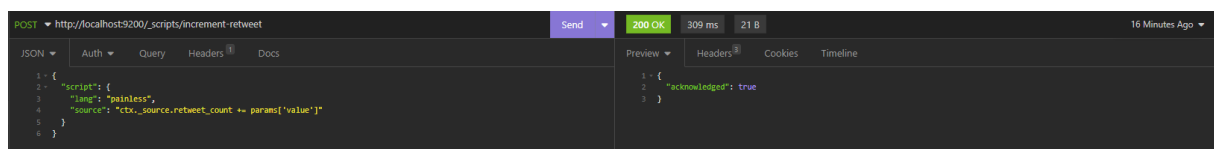
Otázka:

Upravujte počet retweetov pre vami vybraný tweet pomocou vašeho jednoduchého scriptu (v rámci Elasticsearchu) a sledujte ako sa mení `_seq_no` a `_primary_term` pri tom ako zabíjate a spúšťate nódy.

Odpoveď:

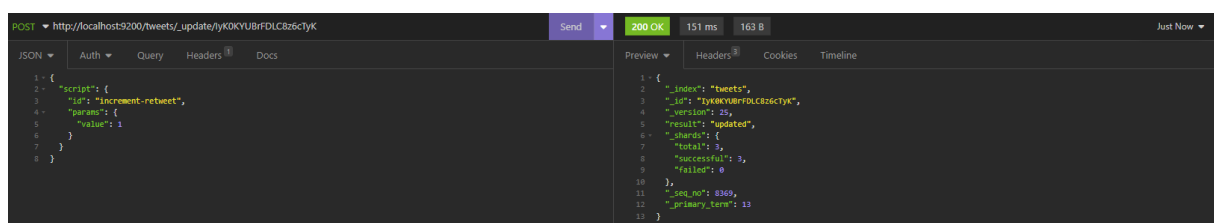
Najskôr sme si vytvorili script (obrázok č. 13), ktorý budeme používať na inkrementovanie dokumentu. Keďže na funkčnosť clusteru potrebujeme aspoň 2 uzly z 3, vykonáme nasledujúce testy, počas ktorých budeme sledovať zmeny `_seq_no` a `_primary_term`:

1. test -> všetky uzly aktívne
2. test -> vypneme prvý uzol
3. test -> zapneme prvý uzol
4. test -> vypneme druhý uzol
5. test -> zapneme druhý uzol
6. test -> vypneme tretí uzol
7. test -> zapneme tretí uzol



Obrázok 13 Vytvorenie scriptu na inkrementáciu

1. test (obrázok č. 14) - `_seq_no` sa nám zväčšuje o jedna, zatiaľ čo `_primary_term` ostáva rovnaké

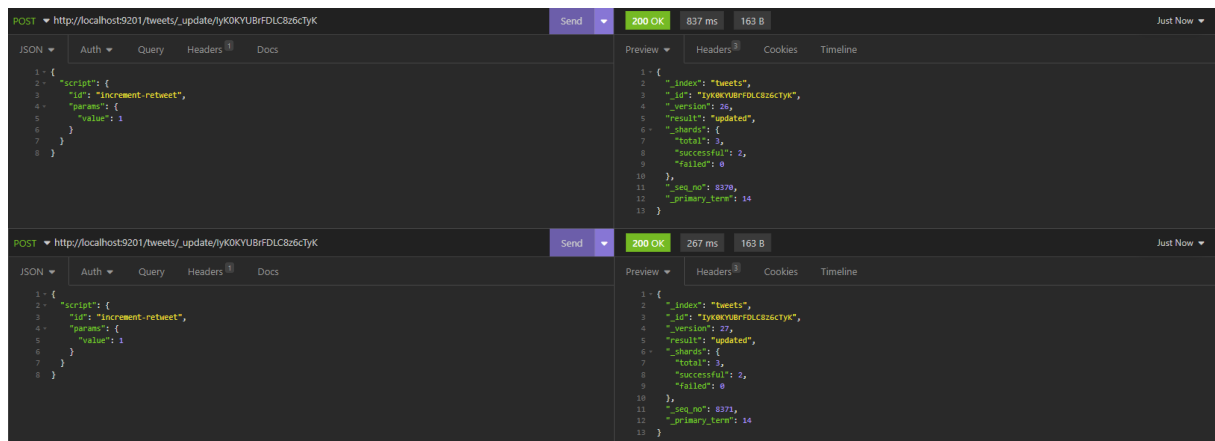


Obrázok 14 Test 1

2. test (obrázok č.15) - `_seq_no` sa nám zvýšilo o jedna a `_primary_term` sa nám taktiež zvýšilo o jedna, avšak len jedenkrát (od vypnutia)

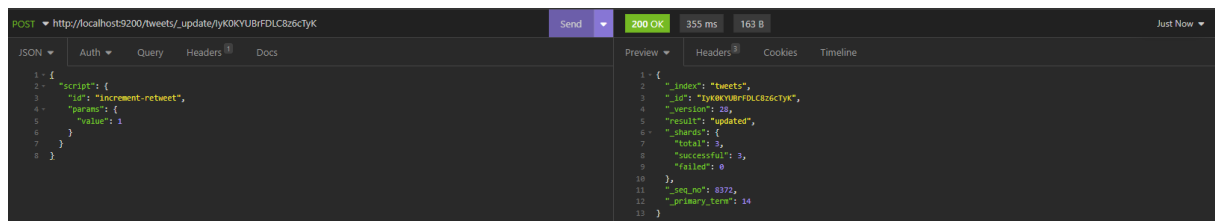
Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií



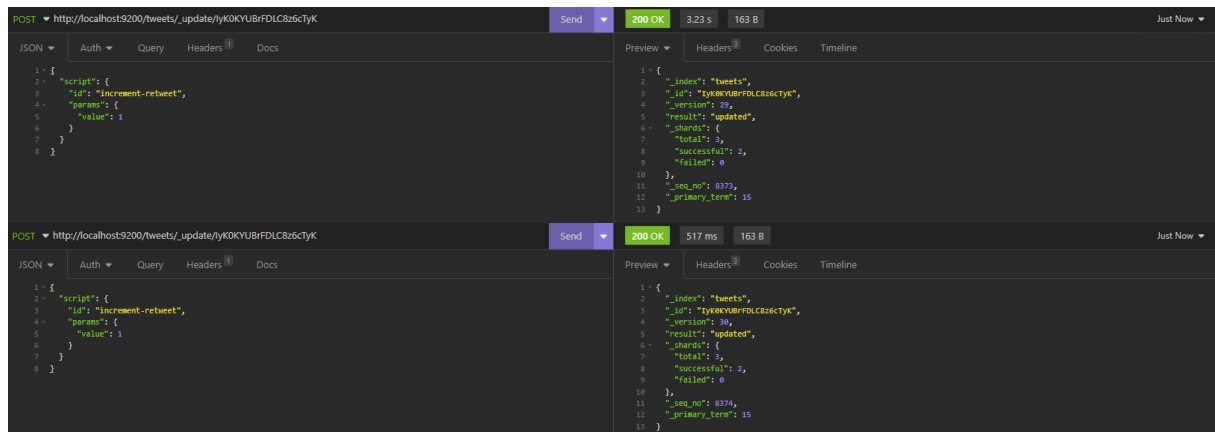
Obrázok 15 Test 2

3. test (obrázok č.16) – `_seq_no` sa nám zväčšilo o jedna, zatiaľ čo `_primary_term` ostáva rovnaké



Obrázok 16 Test 3

4. test (obrázok č.17) – `_seq_no` sa nám zvýšilo o jedna a `_primary_term` sa nám taktiež zvýšilo o jedna, avšak len jedenkrát (od vypnutia)

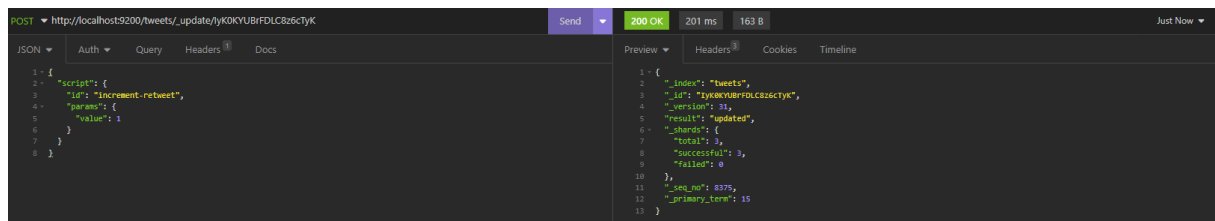


Obrázok 17 Test 4

5. test (obrázok č.18) - `_seq_no` sa nám zväčšilo o jedna, zatiaľ čo `_primary_term` ostáva rovnaké

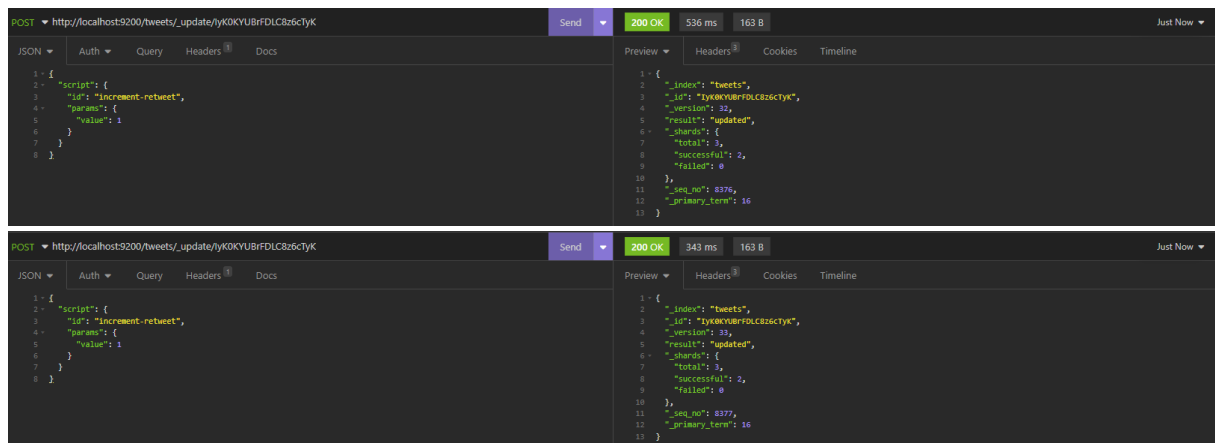
Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií



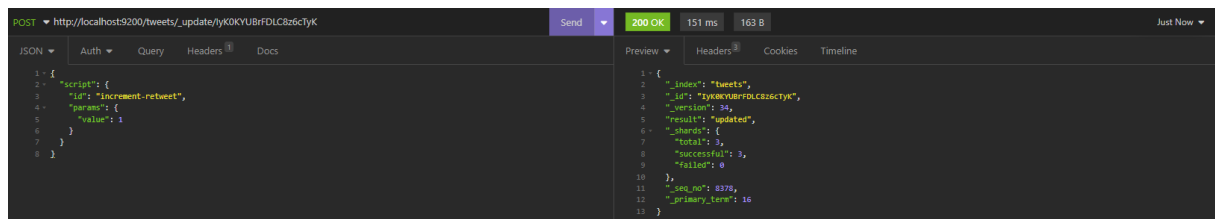
Obrázok 18 Test 5

6. test (obrázok č.19) – `_seq_no` sa nám zvýšilo o jedna a `_primary_term` sa nám taktiež zvýšilo o jedna, avšak len jedenkrát (od vypnutia)



Obrázok 19 Test 6

7. test (obrázok č.20) - `_seq_no` sa nám zväčšilo o jedna, zatiaľ čo `_primary_term` ostáva rovnaké



Obrázok 20 Test 7

Tieto dve hodnoty nám slúžia ako počítadlá, vďaka ktorým zabezpečíme kontrolu konkurencie (napríklad, aby nám starší update neprepísal novší update). `Primary_term` sa inkrementuje vždy keď sa z iného shardu stane primárny, teda pri páde pôvodného primárneho shardu. `Seq_no` je jednoduché počítadlo, ktoré sa inkrementuje vždy pri vykonanej operácii.

9. Veľký import

Otázka:

Zrušte repliky a importujete všetky tweety.

Odpoveď:

Bohužiaľ, nepodarilo sa nám vytvoriť query, ktorá by zvládla denormalizovať všetky dáta konverzácií v prijateľnom čase. Vyskúšali sme 3 hlavné typy queries, a to query pomocou subselectov, query pomocou left joinov a na koniec query využívajúcu kombináciu lateral joinov a klasických joinov (obrázok č.21). Prvotné vytvorenie indexov na všetky cudzie kľúče bolo samozrejmosťou. Nakoniec sa nám podarilo vyexportovať viac ako 2 milióny denormalizovaných záznamov do json súboru. Neskôr sme však zistili, že veľmi **úzke hrdlo je pri tomto importe pravdepodobne autorove HDD** .. skúšali sme zbehnúť túto query (pre 5000 záznamov) na inom stroji, patriacemu autorovmu spolužiakovi, kde je schéma uložená na SSD disku .. z pôvodných 4 minút, čo trebalo starému stroju na denormalizáciu 5000 záznamov, sme sa dostali na 10 sekúnd, čo je markantné zrýchlenie okolo 96%. Pri importe všetkých dát to znamená rozdiel medzi 17 hodinami a 430 hodinami.

Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií

```
39 ) AS l_l ON true
40 LEFT JOIN LATERAL(
41     SELECT json_agg(json_build_object('id', a.id, 'value', a.value, 'type', a.type, 'probability', a.probability))
42     FROM annotations a
43     WHERE a.conversation_id = c.id
44 ) AS a_l ON true
45 LEFT JOIN LATERAL(
46     SELECT json_agg(json_build_object('id', cd.id, 'name', cd.name, 'description', cd.description))
47     FROM context_domains cd
48     JOIN context_annotations ca ON ca.context_domain_id = cd.id AND ca.conversation_id = c.id
49 ) AS cd_l ON true
50 LEFT JOIN LATERAL(
51     SELECT json_agg(json_build_object('id', ce.id, 'name', ce.name, 'description', ce.description))
52     FROM context_entities ce
53     JOIN context_annotations ca ON ca.context_domain_id = ce.id AND ca.conversation_id = c.id
54 ) AS ce_l ON true
55 LEFT JOIN LATERAL(
56     SELECT json_agg(json_build_object(
57         'id', cr.id,
58         'type', cr.type,
59         'parent_conversation.author.id', author2.id,
60         'parent_conversation.author.name', author2.name,
61         'parent_conversation.author.username', author2.username,
62         'parent_conversation.content', c2.content,
63         'parent_conversation.hashtags', (
64             SELECT json_agg(json_build_object('id', h.id, 'tag', h.tag))
65             FROM conversation_hashtags ch
66             JOIN hashtags h ON ch.hashtag_id = h.id
67             WHERE ch.conversation_id = c2.id
68         ))
69     )
70     FROM conversation_references cr
71     JOIN conversations c2 ON c2.id = cr.parent_id
72     JOIN authors author2 ON author2.id = c2.author_id
73     WHERE cr.conversation_id = c.id
74 ) AS cr_l ON true
75 LIMIT 500
76 ) TO 'D:\skola2022_2023\PDT\zadanie5\file.json' WITH (FORMAT CSV, QUOTE ' ');

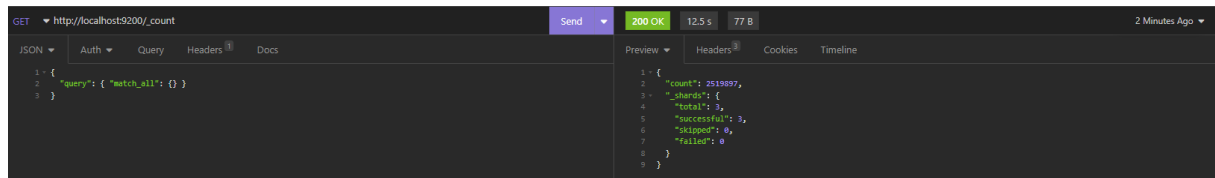
1 COPY(
2     SELECT json_build_object(
3         'id', c.id,
4         'content', c.content,
5         'possibly_sensitive', c.possibly_sensitive,
6         'language', c.language,
7         'source', c.source,
8         'retweet_count', c.retweet_count,
9         'reply_count', c.reply_count,
10        'like_count', c.like_count,
11        'quote_count', c.quote_count,
12        'created_at', c.created_at,
13        'author.id', author.id,
14        'author.name', author.name,
15        'author.username', author.username,
16        'author.description', author.description,
17        'author.followers_count', author.followers_count,
18        'author.following_count', author.following_count,
19        'author.tweet_count', author.tweet_count,
20        'author.listed_count', author.listed_count,
21        'hashtags', h_l,
22        'links', l_l,
23        'annotations', a_l,
24        'context_domains', cd_l,
25        'context_entities', ce_l,
26        'conversation_references', cr_l
27    )
28    FROM conversations c
29    JOIN authors author ON author.id = c.author_id
30    LEFT JOIN LATERAL(
31        SELECT json_agg(json_build_object('id', h.id, 'tag', h.tag))
32        FROM hashtags h
33        JOIN conversation_hashtags ch ON ch.hashtag_id = h.id AND ch.conversation_id = c.id
34    ) AS h_l ON true
35    LEFT JOIN LATERAL(
36        SELECT json_agg(json_build_object('id', l.id, 'url', l.url, 'title', l.title, 'description', l.description))
37        FROM links l
38        WHERE l.conversation_id = c.id
```

Obrázok 21 Query pre denormalizáciu dát

V ďalšom kroku sme využili python script z úlohy č.6, aby sme dáta z json súboru dostali na elasticsearch cluster. Museli sme upraviť zopár parametrov, aby viacej zodpovedali importu veľkých dát. Konkrétne sa jednalo o veľkosť posiadaného batchu a request timeout (keďže defaultne je nastavený na 10 sekúnd, čo nebolo dostačujúce). Po veľkom importe máme v indexe 2 519 897 dokumentov (obrázok č.22).

Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií



Obrázok 22 Počet dokumentov v indexe

10. Vyhľadávanie

Otázka:

Vyhľadajte vo vašich tweetoch, kde použite `function_score` pre jednotlivé medzikroky nasledovne:

a. Must:

i. Vyhľadajte vo viacerých poliach naraz (konkrétne: `author.description.shingles` (pomocou `shingle`) – boost 10, `content` (cez analyzovaný anglický text) spojenie – boost 6 "put1n chr1stian fake jew", zapojte podporu pre preklepy, operátor je OR.

ii. V poly `references.content` slovo "nazi"

iii. Hashtag "ukraine"

b. Filter:

i. vyfiltrujte len tie, ktoré majú `author.following_count > 100`, tie ktoré majú `author.followers_count > 100` a tie, ktoré majú nejakú linku

c. Should:

i. Ak sa v `context_annotations.domain.name` nachádza "Person" boostnite o 5

ii. Ak sa v `context_annotations.entity.name` nachádza "Soros" boostnite o 10

iii. Ak je vyhľadaný string "put1n chr1stian fake jew" aj fráza s tým že sa môže stať jedna výmena slov boostnite o 5

d. Agregácie:

i. Vytvorte bucket `pro-russia` ktorý obsahuje hashtagy používané Kremľom na propagandu: `istandwithputin`, `racism`, `1trillion`, `istandwithrussia`, `isupportrussia`, `blacklivesmatter`, `racism`, `racistukraine`, `africansinukraine`, `palestine`, `israel`, `freepalestine`, `istandwithpalestine`, `racisteu`, `putin`

1. Pre neho spravte týždňový histogram, kde pre každý týždeň zobrazte štatistiky

Odpoveď:

Naša query (obrázok č.23) je rozdelená na 2 časti -> bool query a agregácie. V bool query máme podľa zadania vyhľadávanie v Must, Filter a Should. V agregácií vytvárame buckety podľa hashtagov. Pri tvorbe bucketov sme si museli pridať do mappingu na hashtagy "*fielddata*": *true*, keďže ich máme uložené ako text (kvôli analyzátorom). Týždňový histogram sa nám nepodarilo spraviť. Skóre relevancie sme menili pomocou \wedge (číslo) – notácie (pri jednotlivých slovách) a boostu (celých dokumentov). Pre operácie vykonávané nad poľami (respektíve nad nested fieldami) sme používali nested operátor.

Slovenská Technická Univerzita v Bratislave

Fakulta Informatiky a Informačných Technológií

```
GET http://localhost:9200/tweets/_search Send
JSON Auth Query Headers 1 Docs
1 {
2   "query": {
3     "bool": {
4       "must": [
5         {
6           "multi_match": {
7             "query": "putin christian fake jew",
8             "fields": ["author.description.description_custom_shingles^10", "content^6"],
9             "fuzziness": "AUTO",
10            "operator": "OR"
11          },
12          {
13            "nested": {
14              "path": "conversation_references",
15              "query": {
16                "match": {
17                  "conversation_references.parent_conversation.content": "nazi"
18                }
19              }
20            },
21            {
22              "nested": {
23                "path": "hashtags",
24                "query": {
25                  "match": {
26                    "hashtags.tag": "ukraine"
27                  }
28                }
29              }
30            }
31          ],
32          "filter": [
33            {
34              "range": {
35                "author.following_count": {
36                  "gt": 100
37                }
38              },
39              {
40                "range": {
41                  "author.followers_count": {
42                    "gt": 100
43                  }
44                }
45              }
46            },
47            {
48              "nested": {
49                "path": "links",
50                "query": {
51                  "bool": {
52                    "must": [
53                      {
54                        "exists": {
55                          "field": "links"
56                        }
57                      }
58                    ]
59                  }
60                }
61              }
62            }
63          ],
64          "should": [
65            {
66              "nested": {
67                "path": "context_domains",
68                "query": {
69                  "match": {
70                    "context_domains.name": "Person"
71                  }
72                },
73                "boost": 5
74              }
75            },
76            {
77              "nested": {
78                "path": "context_entities",
79                "query": {
80                  "match": {
81                    "context_entities.name": "Soros"
82                  }
83                },
84                "boost": 10
85              }
86            }
87          ],
88          "match_phrase": {
89            "content": {
90              "query": "putin christian fake jew",
91              "slop": 1,
92              "boost": 5
93            }
94          }
95        ]
96      },
97      "aggs": {
98        "pro-russia": {
99          "nested": {
100            "path": "hashtags"
101          },
102          "aggs": {
103            "hashtags": {
104              "terms": {
105                "field": "hashtags.tag",
106                "size": 5,
107                "include": "istandwithputin|racism|trillion|istandwithrussia|isupportrussia|blacklivesmatter|racism|racistukraine|africansinukraine|palestine|israel|freepalestine|istandwithpalestine|racisteu|putin"
108              }
109            }
110          }
111        }
112      }
113    }
114  }
115 }
```

Obrázok 23 Vyhľadávacia query