1. Rozbehajte si 3 inštancie Elasticsearch-u

Najprv som zväčšil ram pamäť pre WSL tým pádom pre docker tým pádom pre Elasticsearch. Urobil som to nasledovne.

V priečinku C:\Users\pplev som vytvoril súbor s názvom .wslconfig a obsahom:

[wsl2]

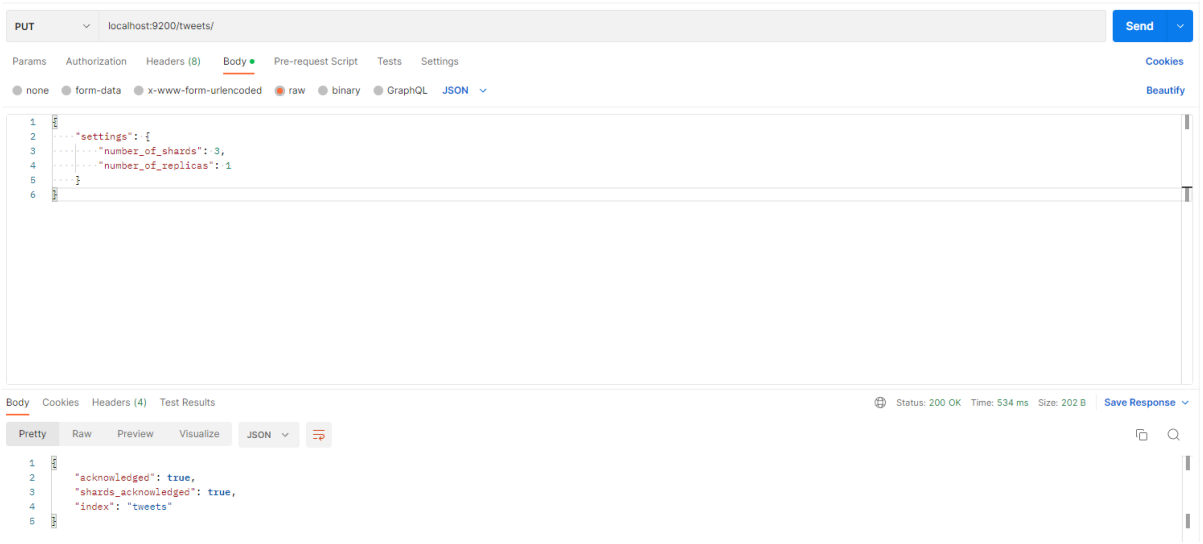
memory=8GB

kernelCommandLine = "sysctl.vm.max_map_count=262144"

Teraz som potreboval vytvoriť 3 inštacie elasticsearchu. Za pomoci tejto stránky: https://www.elastic.co/guide/en/elasticsearch/reference/current/docker.html#docker-compose-file som vytvoril .env a docker-compose. Tieto súbory môžeme nájsť v priloženom súbore Elasticsearch. Všetky requesty ktoré použijem sa budú nachádzať aj exportnuté v priečinku postman pomenované podľa úloh, nájde sa tam aj response z úlohy 10.

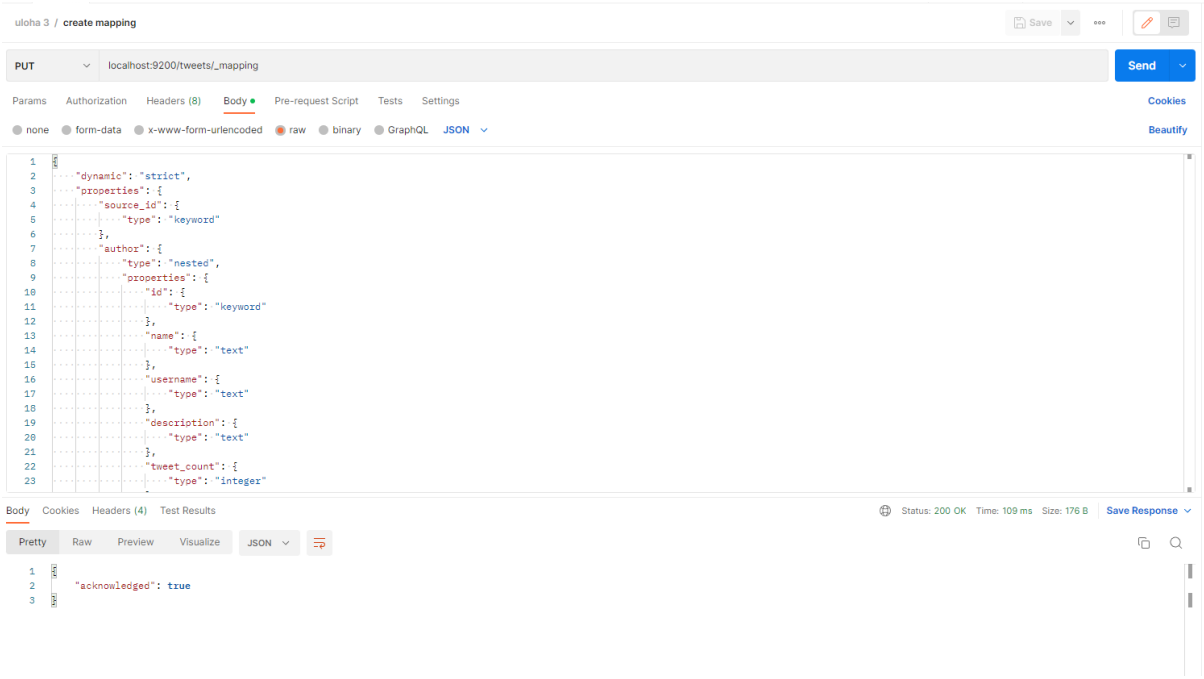
2. Vytvorte index pre Tweety, ktorý bude mať "optimálny" počet shardov a replík pre 3 nódy (aby tam bola distribúcia dotazov vo vyhľadávaní, aj distribúcia uložených dát)

Keďže mám 3 inštancie tak som vytvoril 3 shardy. Viac ako jeden shard by už spomaľovalo index. Každý shard ma jednu repliku na ostatných inštanciách aby sa zabezpečil backup.



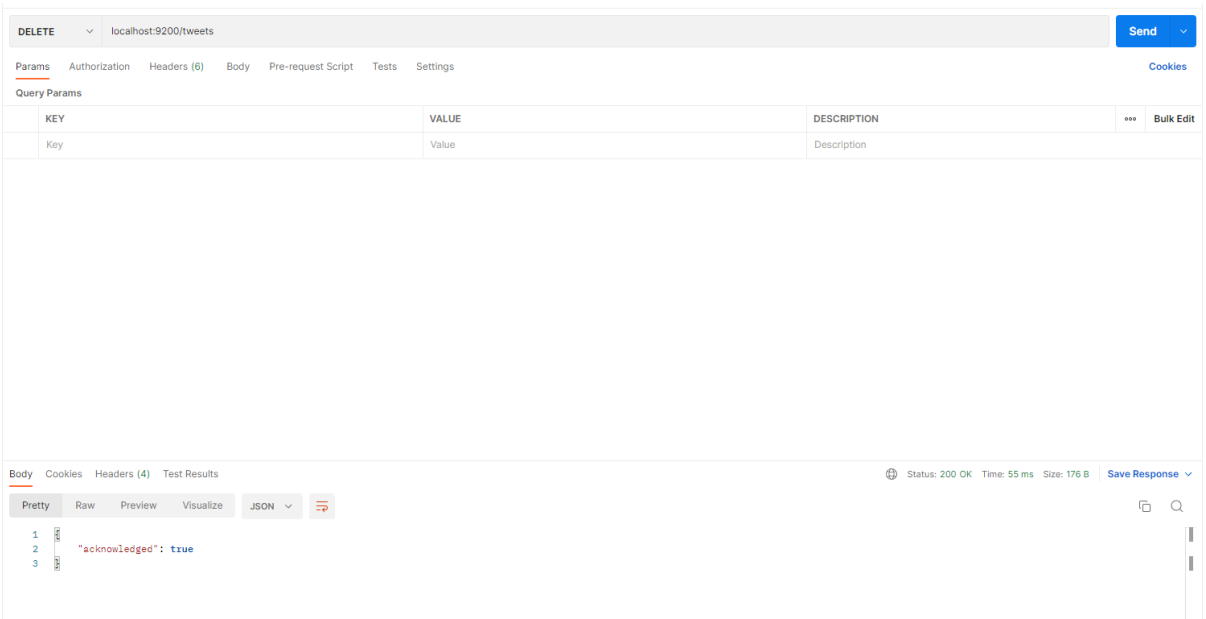
3. Vytvorte mapping pre normalizované dáta z Postgresu (denormalizujte ich) – Každý Tweet teda musí obsahovať údaje rovnaké ako máte už uložené v PostgreSQL (všetky tabuľky). Dbajte na to, aby ste vytvorili polia v správnom dátovom type (polia ktoré má zmysel analyzovať analyzujte správne, tie ktoré 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 vsťahu, id, autor (id, name, username), content a hashtags.

Pomocou requestu ktorý môžeme vidieť nižšie som vytvoril mapping. Ako response som dostal acknowledged true to znamená, že všetko prebehlo úspešne.

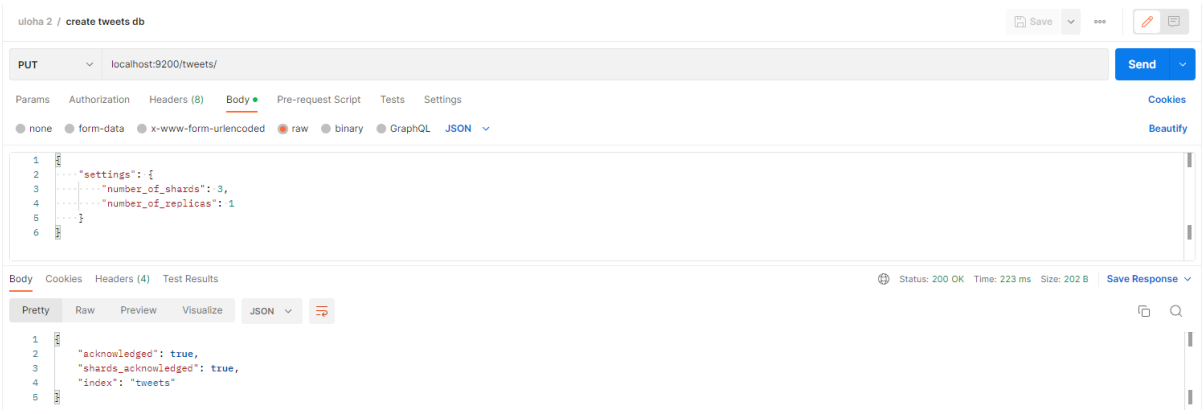


4. Pre index tweets vytvorte 3 vlastné analyzéry (v settings) nasledovne: a. Analyzér "englando". Tento analyzér bude obsahovať nasledovné: i. fitre: english_possessive_stemmer, lowercase, english_stop, english_stemmer, ii. char_filter: html_strip iii. tokenizer: štandardný - ukážku nájdete na stránke 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. Priraďte analýzery 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

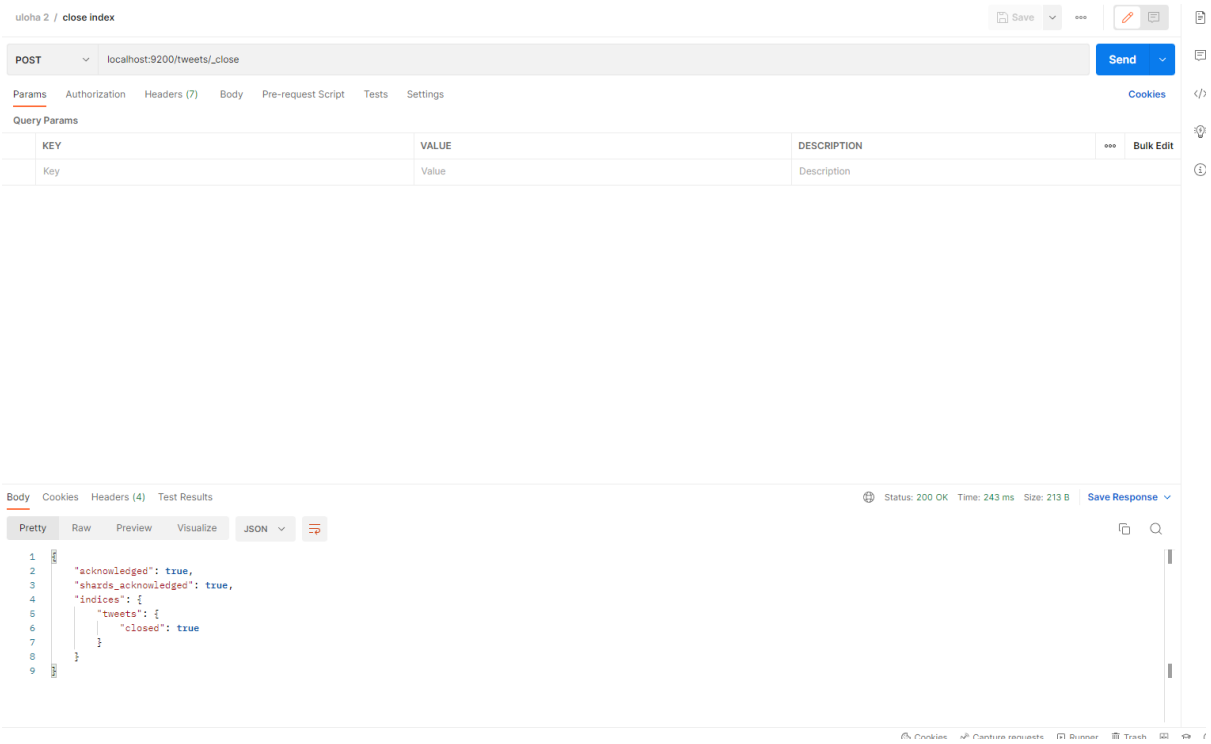
Najprv bolo potrebné staré mapovanie vymazať a to pomocou nasledovného requestu.



Keď už som mal mapovanie vymazané bolo potrebné znovu vytvoriť index a to pomocou nasledujúceho príkazu.

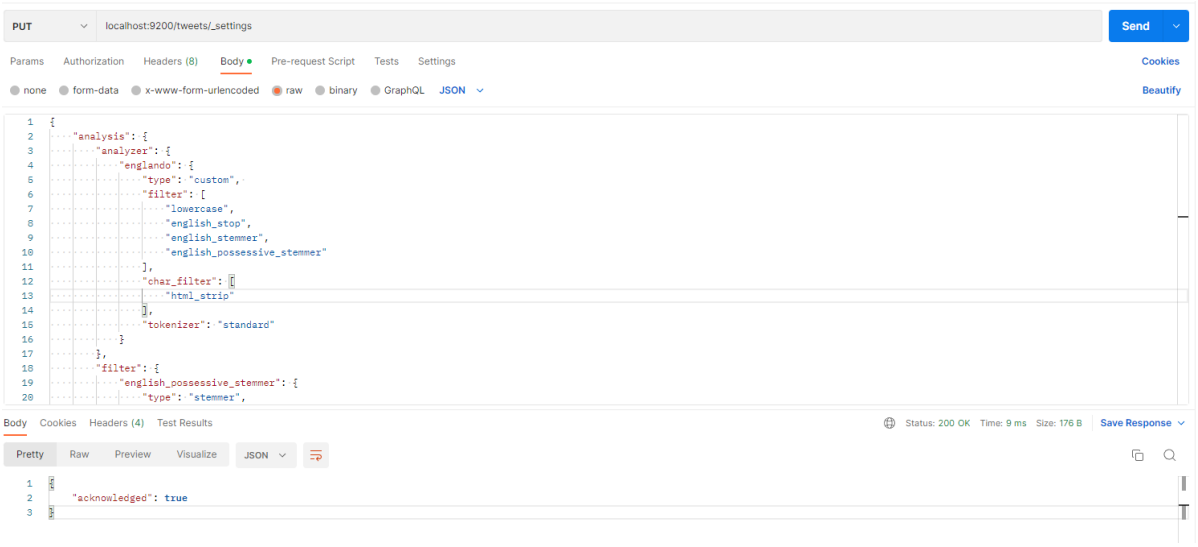


Nato aby som mohol analyzátor a tým pádom mapping vytvárať bolo potrebné zatvoriť index pomocou nasledujúceho requestu.



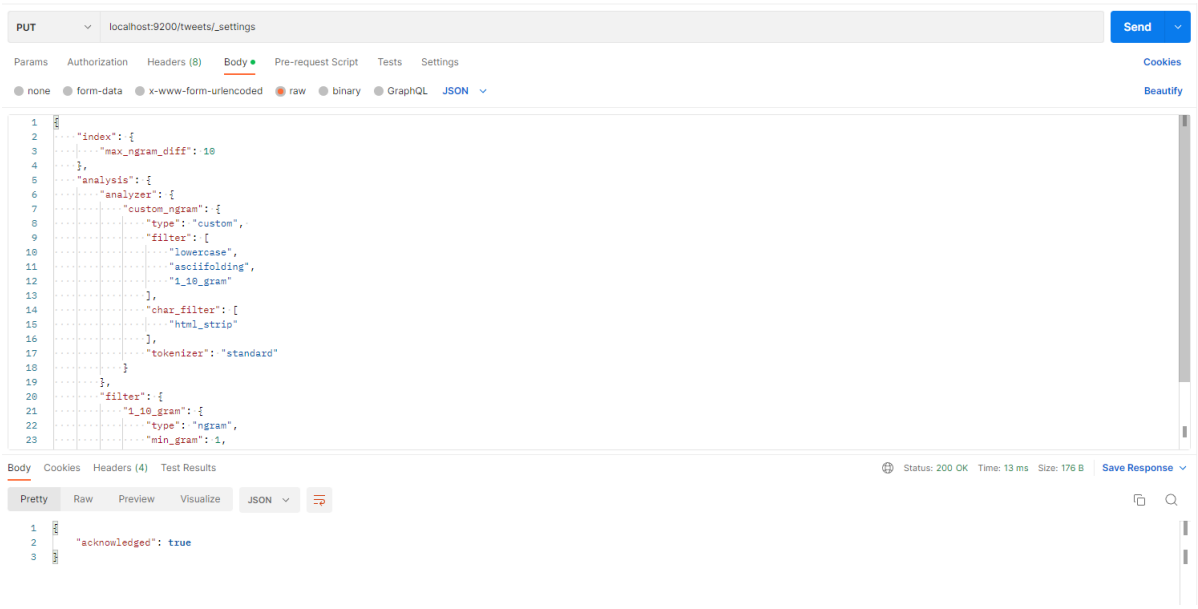
Úloha 4.1 Analyzér „englando“

Tento analyzér bude obsahovať nasledovné: i. fitre: 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



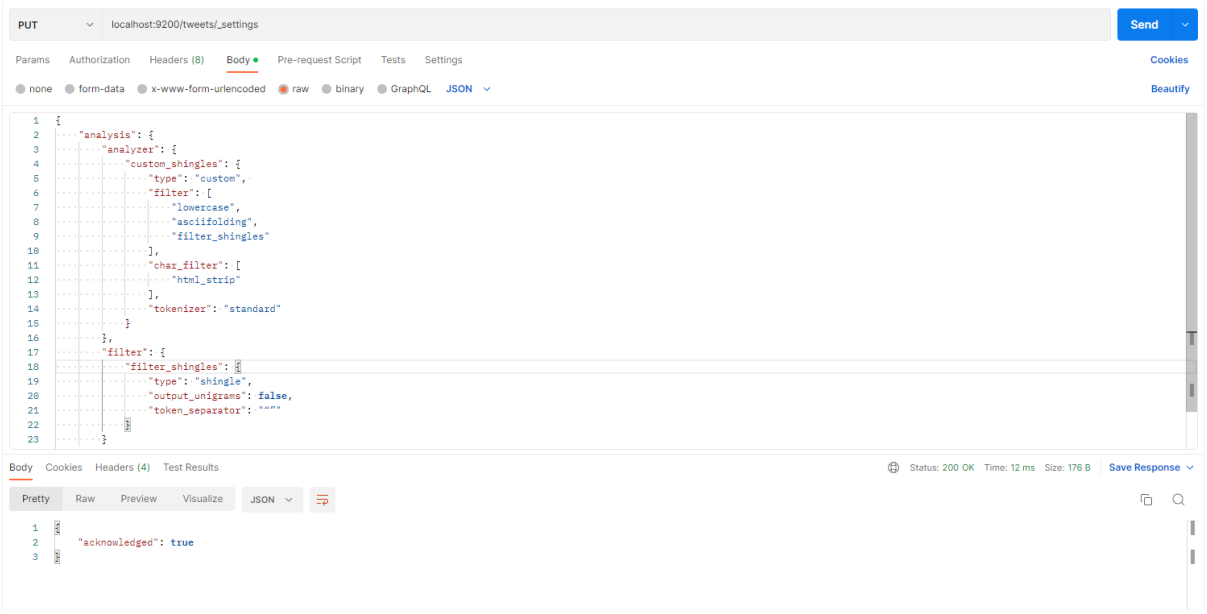
4.2 Analyzér custom_ngram

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



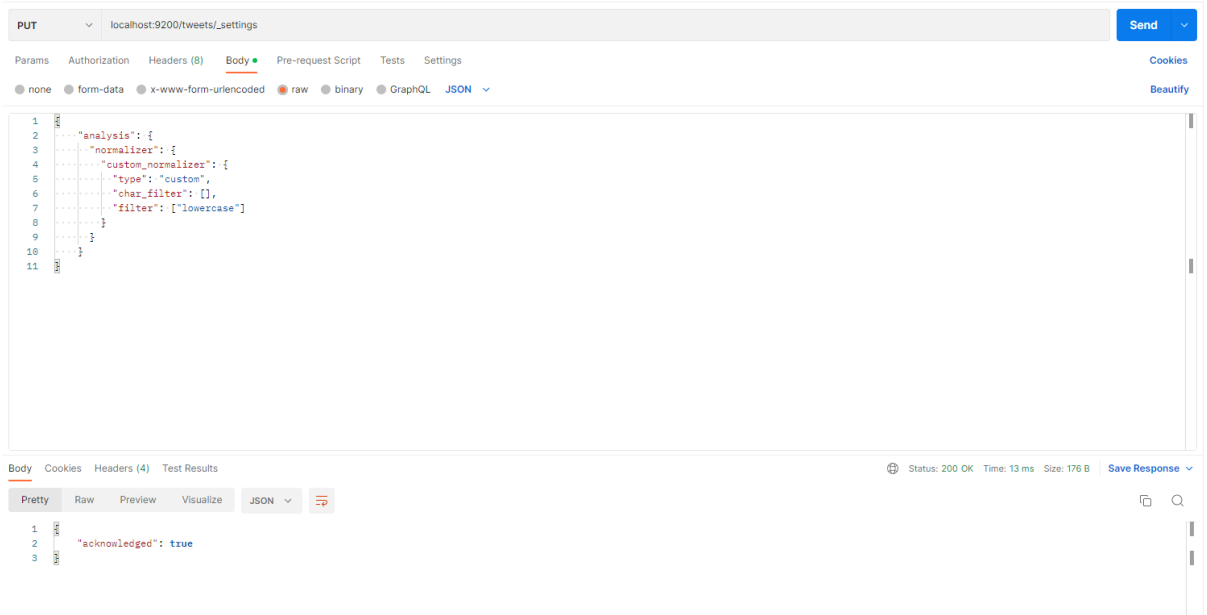
4.3 Analyzér custom_shingles:

Tento analyzér bude obsahovať nasledovné: i. filtre: lowercase, asciifolding, filter_shingles (definujte si ho sami a dajte token_separator: “”) ii. char_filter: html_strip iii. tokenizer: štandardný



4.4 lowercase normalizer

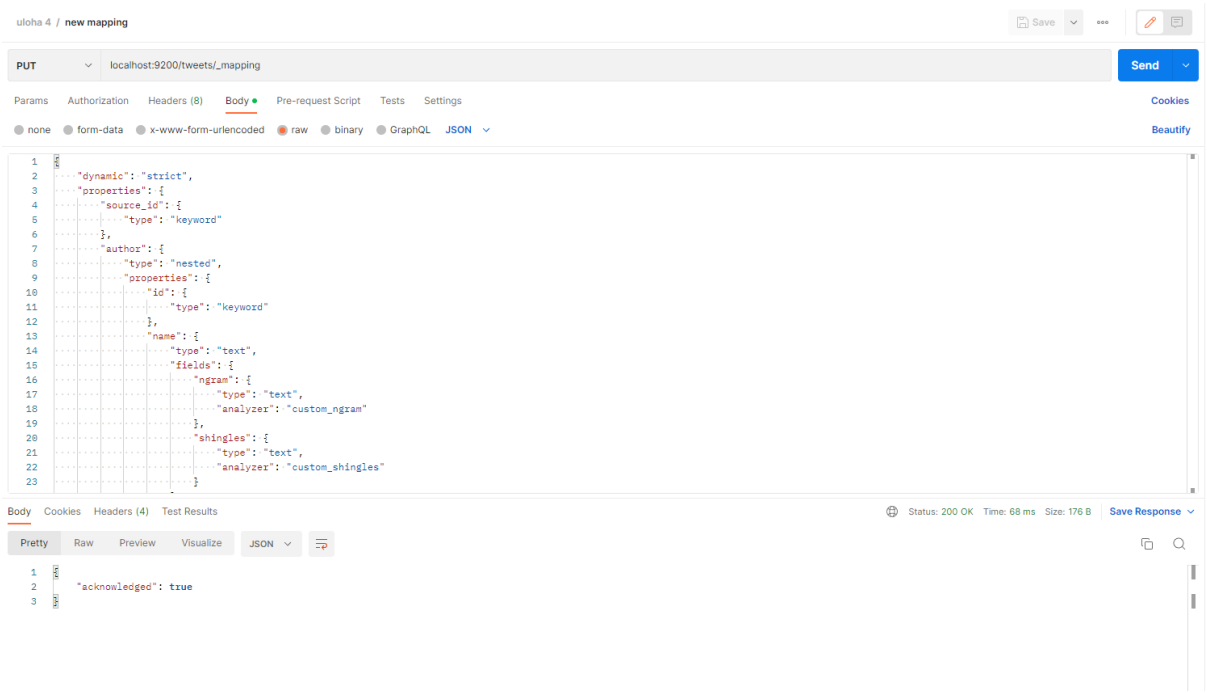
Tento normalizér bude obsahovať nasledovné: i. filter: lowercase



4.5 Do mapovania pridajte:

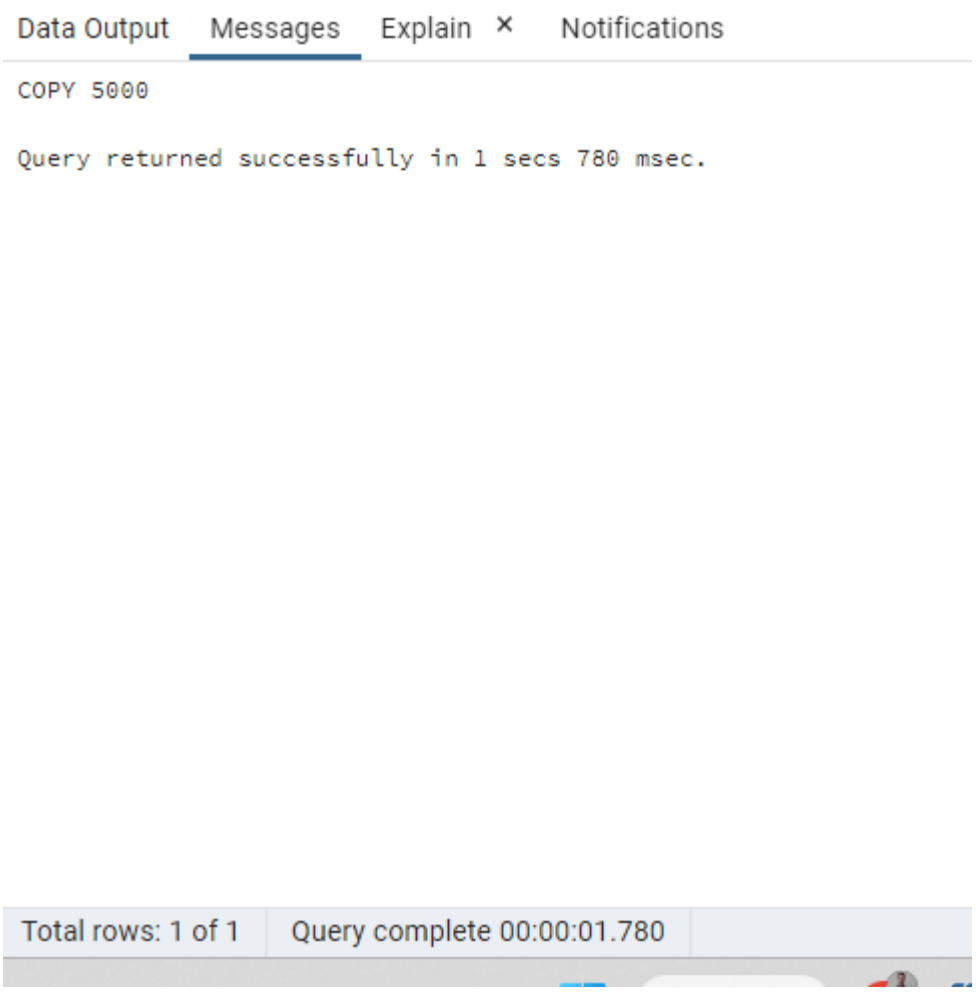
i. každý anglický text (rátajme že každý tweet a description u autora je primárne v angličtine) nech je analyzovaný novým analyzérom "englando" ii. Priraďte analýzery 1. a. author.name nech má aj mapovania pre custom_ngram, a custom_shingles 2. b. author. screen_name nech má aj custom_ngram, 3. c. author.description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

Už máme všetky analyzátory vytvorené takže nám nič nebraní v tom vytvoriť finálny mapping.



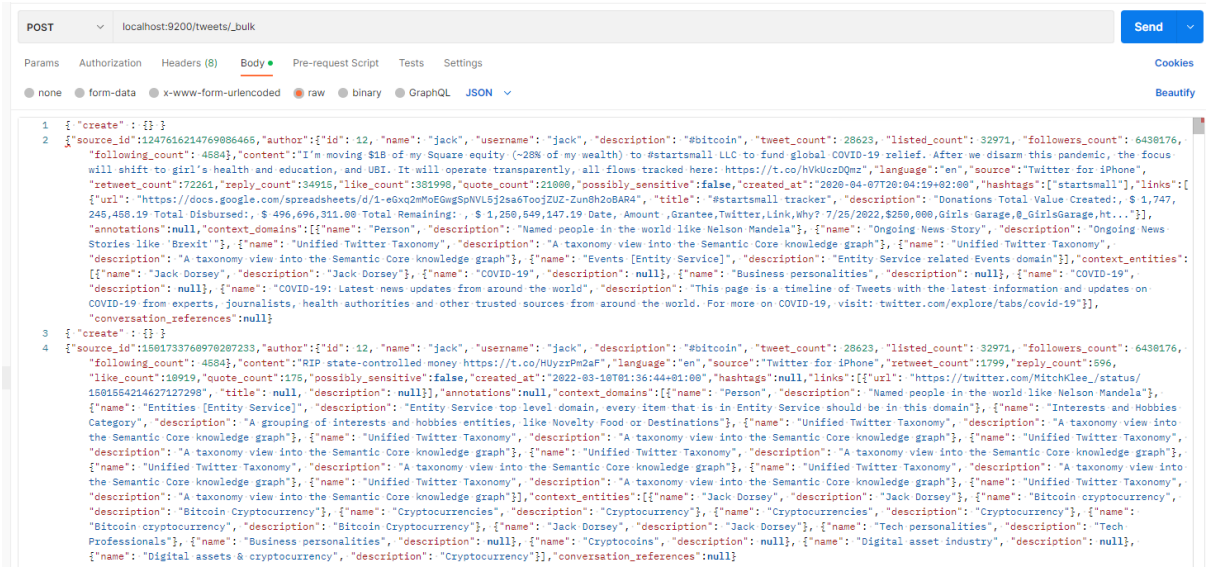
5. Vytvorte bulk import pre vaše normalizované Tweety.

Najprv bolo potrebné dáta z postgresu normalizovať toho sme dosiahli pomocou query v postgrese ktorú môžeme nájsť v priečinku queries s názvom export_5000.txt. Táto query nám vráti JSON objekty. Aby query zbehla rýchlo boli vytvorené indexy ktoré môžeme nájsť v priečinku queries a súbor indexes.txt. Na screene nižšie môžeme vidieť, že query zbehla za približne 2 sekundy. Následne potrebujem tieto JSON objekty pripraviť pre bulk API elasthicu nato som si vytvoril program v pythone. Tento program môžeme nájsť v priečinku programs s názvom import_5000.py. Jedine čo robí tento program je, že pred každý JSON object vloží string create aby bulk API vedelo, že má tento object vložiť.

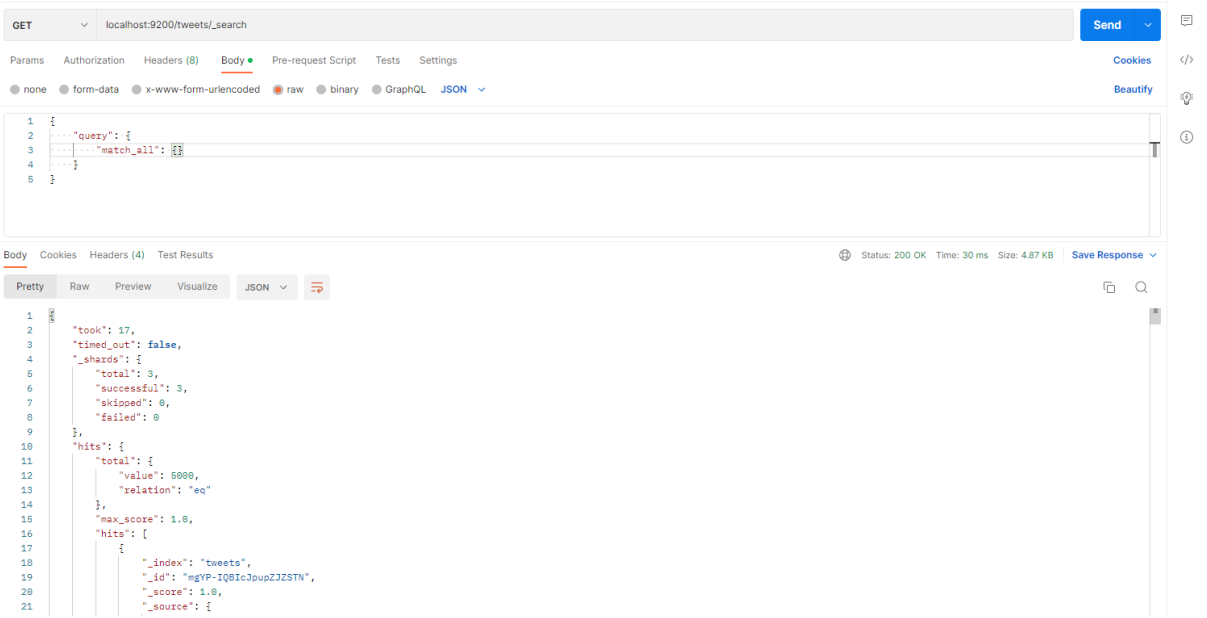


6. Importujete dáta do Elasticsearchu prvych 5000 tweetov

Import prebehol tak, že som urobil post na bulk a do tela vložil všetkých 5000 tweetov vo formáte ktoré mi vrátil môj python program.

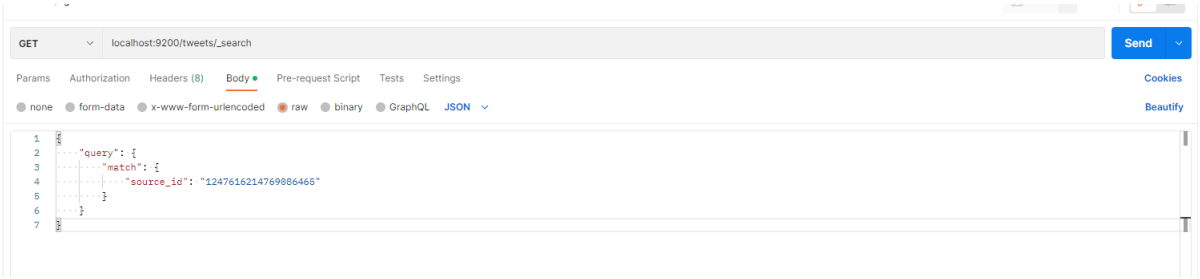


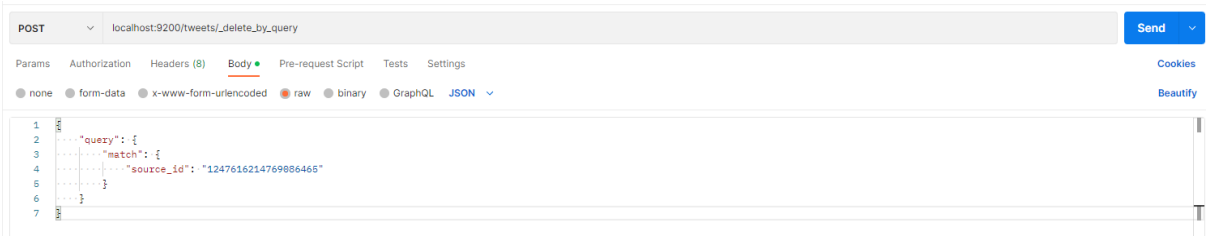
Na nasledujúcom screene môžeme vidieť, že sa nám vložilo všetkých 5000 tweetov.

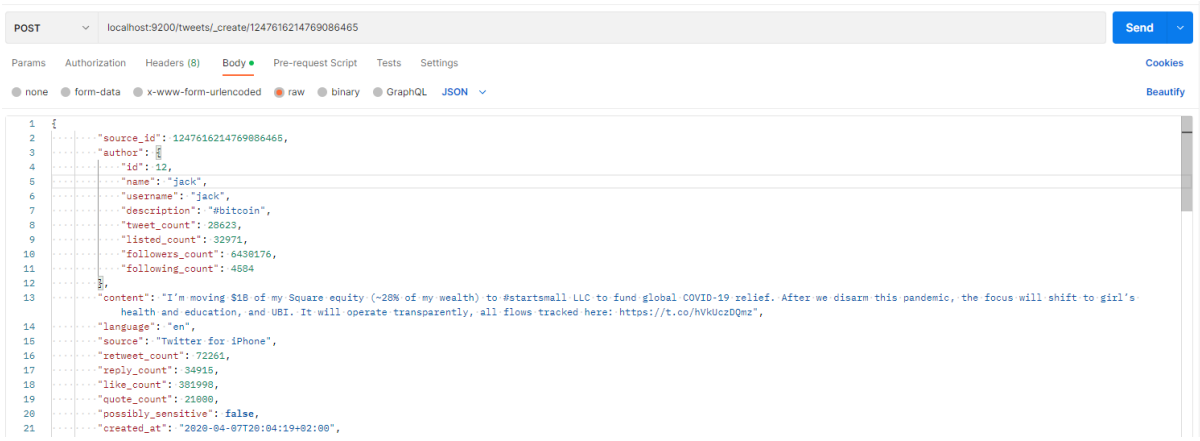


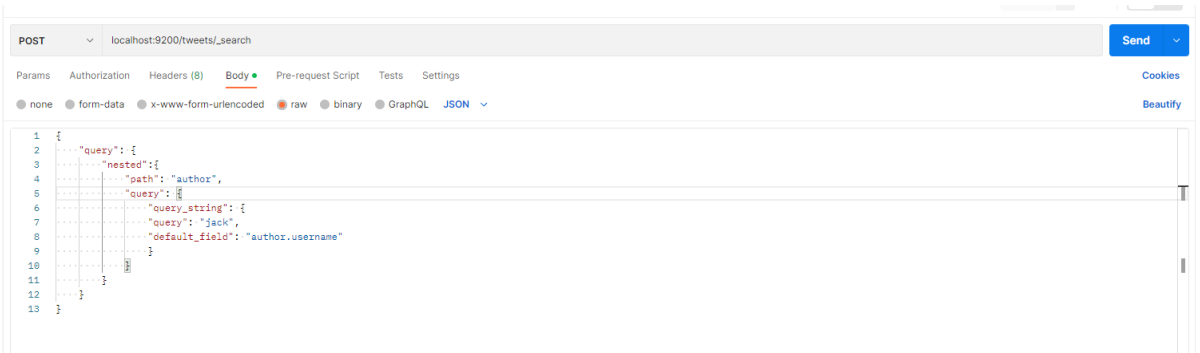
7. Experimentujte s nódami, a zistite koľko nódov musí bežať (a ktoré) aby vám Elasticsearch vedel pridávať dokumenty, mazať dokumenty, prezerať dokumenty a vyhľadávať nad nimi? Dá sa nastaviť Elastic tak, aby mu stačil jeden nód? Čo je dôvodom toho že existuje nejaké kvórum?

Na obrázkoch nižšie môžeme vidieť, že na testovanie som vytvoril get, delete, create a search









7.1 Všetko zapnuté

	took	s_total	s_succesfull	s_skipped	s_failed
get	4	3	3	0	0
delete	32	-	-	-	-
create	37	2	2	-	0
search	11	3	3	0	0

7.2 Všetko vypnuté

	took	s_total	s_succesfull	s_skipped	s_failed
get	-	-	-	-	-
delete	-	-	-	-	-
create	-	-	-	-	-
search	-	-	-	-	-

Všetky requesty Could not send request

7.3 Zapnutý node es01-1 a es02-1

	took	s_total	s_succesfull	s_skipped	s_failed
get	2	3	3	0	0
delete	21	-	-	-	-
create	22	2	2	-	0
search	24	3	3	0	0

7.4 Zapnutý node es01-1 a es03-1

	took	s_total	s_succesfull	s_skipped	s_failed
get	3	3	3	0	0
delete	27	-	-	-	-
create	36	2	2	-	0
search	13	3	3	0	0

7.5 Zapnutý node es02-1 a es03-1

	took	s_total	s_succesfull	s_skipped	s_failed
get	-	-	-	-	-
delete	-	-	-	-	-
create	-	-	-	-	-
search	-	-	-	-	-

Každý request Could not send request

7.6 Zapnutý node es01-1

	took	s_total	s_succesfull	s_skipped	s_failed
get	12	3	3	0	0
delete	-	-	-	-	-
create	-	-	-	-	-
search	13	3	3	0	0

Delete, create

"type": "cluster_block_exception",

"reason": "blocked by: [SERVICE_UNAVAILABLE/2/no master];"

7.7 Zapnutý node es02-1

	took	s_total	s_succesfull	s_skipped	s_failed
get	-	-	-	-	-
delete	-	-	-	-	-
create	-	-	-	-	-
search	-	-	-	-	-

Všetky requesty Could not send request

7.8 Zapnutý node es03-1

	took	s_total	s_succesfull	s_skipped	s_failed
get	-	-	-	-	-
delete	-	-	-	-	-
create	-	-	-	-	-
search	-	-	-	-	-

Všetky requesty Could not send request

7.9 Záver

Môj node es01-1 je master node to znamená keď ho vypnem nejde mi nič. Ak by som chcel aby som mohol tento node vypnúť musel by som všetky nody nastaviť na master nody. Pri skúšaní som si všimol, že keď mi beží iba node es01-1 tak síce viem si getnut document a aj ho vyhľadávať no neviem document pridať ani zmazať. Vidíme, že najrýchlejšie sa robia requesty keď máme všetky 3 nody zapnuté je to aj logické.

8. 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 om ako zabíjate a spúšťate nódy.

8.1 Všetko zapnuté

	_seq_no	_primary_term
update	12	4

8.2 Zapnutý node es01-1 a es02-1

	_seq_no	_primary_term
update	13	4

8.3 Zapnutý node es01-1 a es03-1

	_seq_no	_primary_term
update	14	4

"type": "unavailable_shards_exception",

"reason": "[tweets][0] [1] shardIt, [0] active : Timeout waiting for [1m], request: indices:data/write/update"

8.4 Zapnutý node es01-1

	_seq_no	_primary_term
update	-	-

"type": "cluster_block_exception",

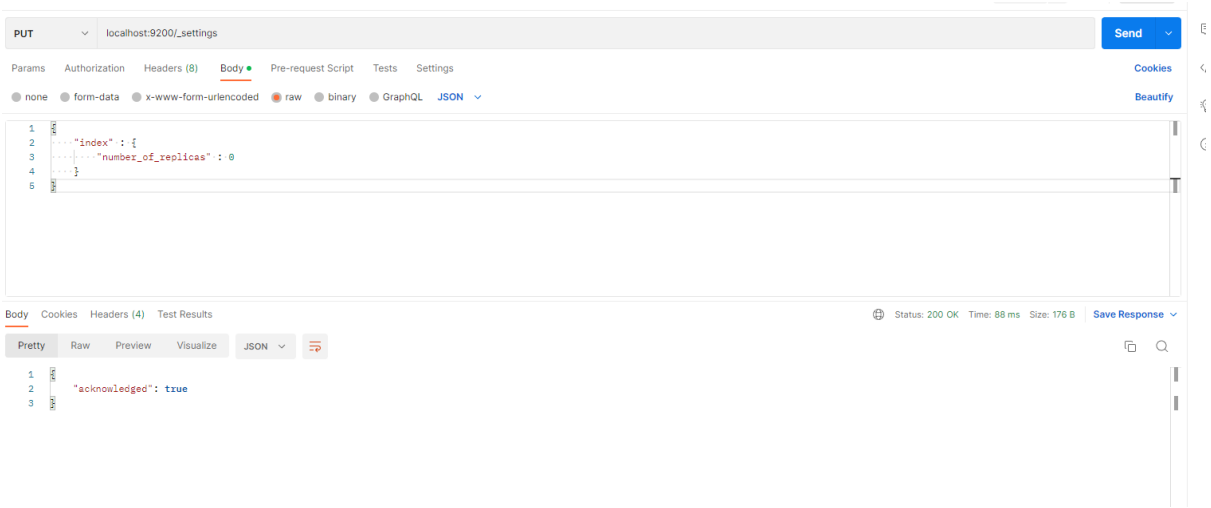
"reason": "blocked by: [SERVICE_UNAVAILABLE/2/no master];"

8.5 Záver

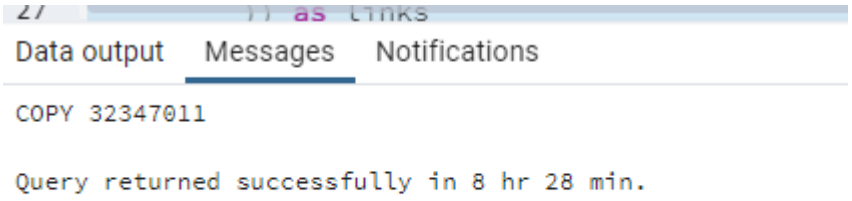
Sequence number sa mení vždy keď vykonáme update. Keby vola napríklad get nemenil by sa. Primary term je pri pustení všetkých nodov uložený na sharde 4. keď vypnem node tri nezmení sa primary term to znamená, že dokument nebol uložený na primary sharde v node 3. Z toho vyplýva, že dokument je uložený na node 1. Ak vypnem node 2 a 3 neviem vyhľadávať.

9. Zrušte repliky a importujete všetky tweety

Predtým vložených 5000 tweetov a celu databázu som zmazal. Následne som vytvoril index zavrel ho vytvoril analyzátory pridal mapping a pomocou nasledovného requestu som zrušil repliky.

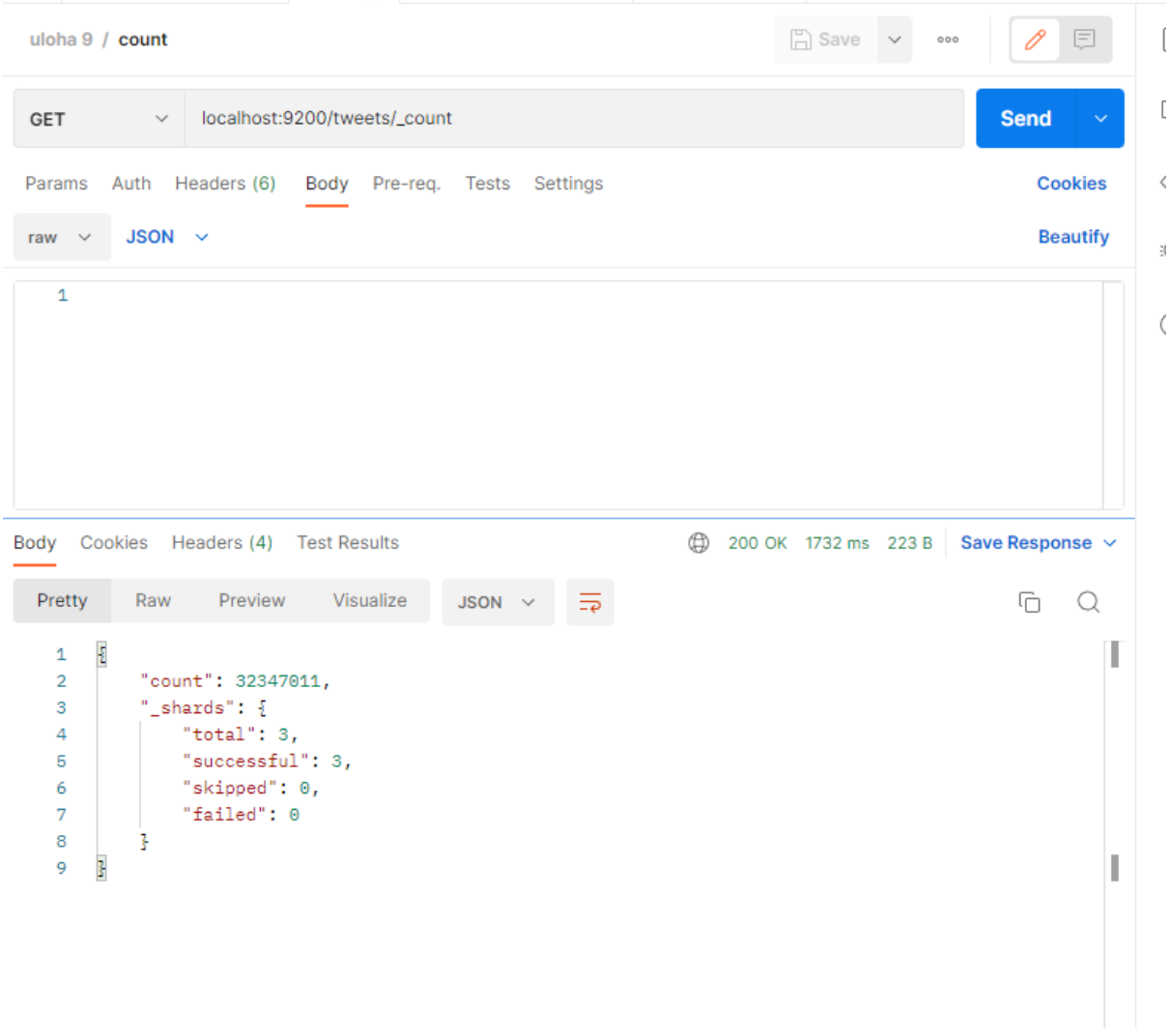


Tento import script je rovnaký ako ten pre 5000 akurát som odstránil order by a limit. Import script môžeme nájsť v priečinku queries s názvom export_all.txt. Aby query zbehla rýchlo boli vytvorené indexy ktoré môžeme nájsť v priečinku queries a súbor indexes.txt. Na nasledujúcom screene môžeme vidieť, že export trval 8 hodín a 28 minút a vrátil nám 32 347 011 JSON objektov.



Keďže tu už ide o veľmi veľa dát približne 32 miliónov už nie úplne viem urobiť to čo pre 5000 to znamená vytvoriť create a ručne vložiť do bulk importu cez postmana. Musel som si vyrobiť nový import program ktorý môžeme nájsť v priečinku programs s názvom import_all.py Na obrázkoch nižšie môžeme vidieť, že za 8 hodín sa nám importli do elasticu všetky dokumenty. Postgresql export mal 32 347 011 a v elasticu mám 32 347 011 to znamená, že sa importli všetky dokumenty.

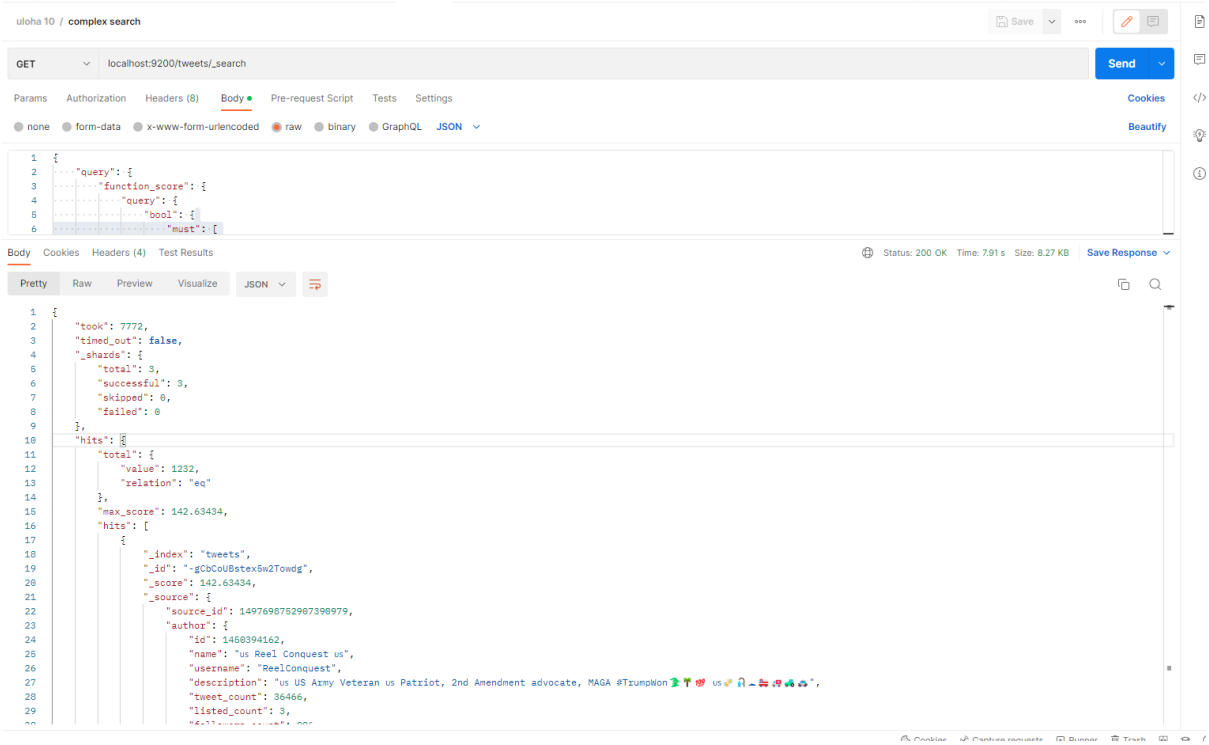
```
Execution time: 82.55353212356567 seconds
Execution time: 108.5751473903656 seconds
Execution time: 107.5488429069519 seconds
Execution time: 89.09803318977356 seconds
Execution time: 57.66910243034363 seconds
Execution time: 102.48526883125305 seconds
Execution time: 95.70412349700928 seconds
Execution time: 22.69034481048584 seconds
Execution time: 29248.828378677368 seconds

D:\dump>
```



10. 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 referencies.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" boostinte 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 ze sa môže stat jedna výmena slov boostnite o 5 d. Agregácie: i. Vytvorte bucket pro-russia ktorý obsahuje hastagy používane 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

Na obrázku nižšie môžeme vidieť request ktorý nám vrátil tweet zo skóre 142 taktiež môžeme vidieť, že daným vyhľadávacím kritériám vyhovuje 1232 tweetov. Na konci responsu sú agregácie. Cely response môžeme nájsť v priečinku postman súbor uloha-10-response.json



Keďže aj request body aj response sú moc dlhé na to aby som ich tu po kúskoch screnshotoval nachádzajú sa oba v priečinku postman. Teraz si poďme túto query prejsť. Pomocou funkcie function score som vedel pre dokumenty nastaviť aké majú mať score takže ich v podstate sortnúť od najväčšieho skóre po to najmenšie. Bolo potrebné zabezpečiť vyhľadávanie vo viacerých fieldoch naraz nato som použil funkciou multi match. Do jej argumentov som pridal polia v ktorých som chcel vyhľadávať to znamená content a author description shingles. Boostovanie som zabezpečil pomocou ^ notácie. Ďalej nasledovalo filtrovanie tam som musel použiť nested query keďže som vyhľadával v nested objekte. Vyfiltroval som, že chcem len tweety ktorých autor ma followers count väčší ako sto a following count takisto. V neposlednom rade chcel som taký tweet ktorý má nejaké linky to znamená linka není null. Ďalej bolo potrebné boostnut scóre ak sa v domain name nachádza person. Ak sa v entity name nachádza string soros. Nakoniec som ešte boostol ak som našiel string put1n chr1stian fake jew jednu výmenu slov som zabezpečil kľúčovým slovom slop 1. Nakoniec som vytvoril agregácie to znamená našiel som si všetky tweety kde sa nachádza hashtag kremeľskej propagandy. Rozdelil som to podľa týždňov a nastavil formátovanie dátumu. Nakoniec som ešte zagregoval podľa hashtagov.