Pokročilé databázové technológie

**Zadanie č. 5 – Elasticsearch**

ZADANIE:

5. zadanie zamerané na Elastic:

Odovzdanie do 12.12. 23:59. Zadania 1-8 je dokopy za 7,5 boda tak isto ako aj 9 – 11.

Dokopy teda za 15 bodov. Odovzdávate dokument s popisom a queries separátne každú ako

json čo posielate a json čo dostanete ako odpoveď.

1. Rozbehajte si 3 inštancie Elasticsearch-u

2. Vytvorte index pre Tweety, ktorý bude mať “optimálny“ počet shardov a replík pre 3 nody

(aby tam bola distribúcia dotazov vo vyhľadávaní, aj distribúcia uložených dát)

3. Vytvorte mapping pre normalizované dáta z Postgresu - Tweet musí obsahovať údaje

rovnaké ako máte už uložené v PostgreSQL. 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.

Mapovanie musí byť striktné.

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

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

1.2. fitre: english\_possessive\_stemmer, lowercase, english\_stop, english\_stemmer,

1.3. char\_filter: html\_strip

1.4. tokenizer: štandardný

- ukážku nájdete na stránke elastic.co pre anglický analyzér

2. Analyzér custom\_ngram:

2.2. Filtre: lowercase, asciifolding, filter\_ngrams (definujte si ho sami na rozmedzie 1-

10)

2.3. char\_filter: html\_strip

2.4. tokenizer: štandardný

3. Analyzér custom\_shingles:

3.2. Filtre: lowercase, asciifolding, filter\_shingles (definujte si ho sami a dajte

token\_separator: “”)

3.3. char\_filter: html\_strip

3.4. tokenizer: štandardný

Do mapovania pridajte:

1. každý anglický text (rátajme že každý tweet a description u autora je primárne

v angličtine) nech je analyzovaný novým analyzérom "englando"

2. Priraďte analýzery

a. author.name nech má aj mapovania pre custom\_ngram, a custom\_shingles,

b. author. screen\_name nech má aj custom\_ngram,

c. author.description nech má aj custom\_shingles. Toto platí aj pre mentions, ak tam tie záznamy máte.

3. Hashtagy indexujte ako lowercase

5. Vytvorte bulk import pre vaše normalizované Tweety.

6. Importujete dáta do Elasticsearchu prvych 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?

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 tom

ako zabíjate a spúšťate nódy.

9. Zrušte repliky a importujete všetky tweety

10. Vyhľadajte vo vašich tweetoch spojenie "gates s0ros vaccine micr0chip". V query použite

function\_score, kde jednotlivé medzikroky sú nasledovné:

Query:

1. Must - vyhľadajte vo viacerých poliach (konkrétne: author.name (pomocou shingle), content (cez analyzovaný anglický text), author.description (pomocou shingles), author.screen\_name (pomocou ngram)) spojenie "gates s0ros vaccine micr0chip", zapojte podporu pre preklepy, operátor je OR.

2.1 tieto polia vo vyhľadávaní boost-nite nasledovne - author.name \* 6, content \* 8,

author.description \* 6, author.screen\_name \* 10.

3. Filter - vyfiltrujte len tie, ktoré majú author.statuses\_count > 1000 a tie, ktoré majú

hashtag „qanon“

4. Should – boost-nite 10 krat tie, ktoré obsahujú v mentions.name (tento objekt je typu nested) cez ngram string "real".

5. Nastavte podmienené váhy cez functions nasledovne:

5.1. retweet\_count, ktorý je väčší rovný ako 100 a menší rovný ako 500 na 6,

5.2. author.followers\_count väčší ako 100 na 3

Zobrazte agregácie pre výsledky na konci. Vytvorte bucket hashtags podľa hashtagov

a spočítajte hodnoty výskytov (na webe by to mohli byť facety).

11. Konšpiračné teórie podľa Elasticu. Pracujte zo všetkými tweetami, ktoré máte. Následne

pre všetky týždne zistite pomocou vnorených agregácii, koľko retweet\_count sumárne majú

tweety ktoré majú hashtagy z prvého zadania. Teda na základe hashtagov znova rozdeľte

tweety do konšpiračných teórii ale pomocou agregácii.

Riešenie:

Časť zaoberajúcu sa importovaním dát ako aj samotných query v úlohách 1-3 podľa nášho modelu sme vykonali v jupyter notebooku pomocou jazyku Python a s ním súvisiacou knižnicou pre prácu s mongo databázou pymongo.

**Úloha 1:**

Ako prvé sme si najprv stiahli windows subsystem for linux, cez ktorý sme spúšťali docker a teda aj samotný elastic, ktorý sme najprv stiahli a setupli pomocou inštalačného guidu na oficiálnej stránke elasticu a tak sme si rozbehali 3 inštancie/nody elasticu. Dolu na obrázku môžeme cez postmana vidieť naše inštancie.

Obrázok, na ktorom je text, snímka obrazovky, monitor, vnútri

Automaticky generovaný popis

**Úloha 2:**

Obrázok, na ktorom je text

Automaticky generovaný popis

**Úloha 3:**

Mapping som vytvoril tak, ako bolo opisané v zadaní úlohy, teda striktné mapovanej, ktoré je priložené v priečinku s riešením, a atribúty by viac menej zodpovedať tým čo sú v postgrese. Pri riešení mám vlastne dva mappingy prvý, ktorý obsahuje parenta(embednuty tweet) a druhý, ktorý parenta nemá, dovôd prečo to takto mám, že pri exportovaní dát z postgresu vzťah medzi parent tweetom a child tweetom dosť času žral a tak som sa od úlohy 9 rozhodol používať mapping bez parenta, kedže bolo samotne naznačené aj na prednáške, že štruktúra toho mappingu záleží najmä od use casu, tak mi to neprišiel ako veľký problém odstrániť parenta keďže v ďalších úlohách sa s ním nepracuje. Rovnako ako prvý tak aj druhý mapping je priložený v súbore.

**Úloha 4:**

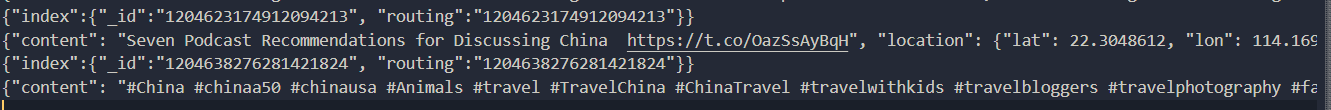
Dané analyzéry sú súčasťou mappingu.

Obrázok, na ktorom je text

Automaticky generovaný popis

**Úloha 5:**

Predtým ako som si bulk importol tweety, tak som si ich najprv cez python sqlalchemy ORM vytiahol a uložil v bulk json formáte, ktorý môžeme vidieť nižšie.



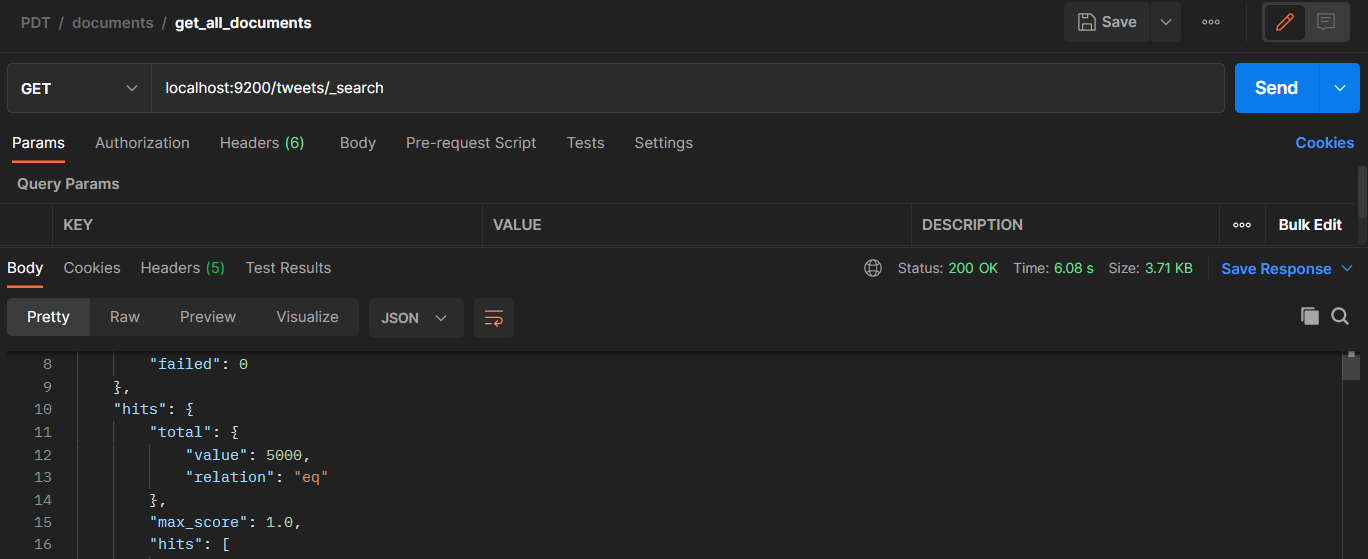
Potom som si spravil bulk script v bashy, ktorý mi dané dáta po 25 000 tweetov pridával do elasticu cez CURL.

**Úloha 6:**

Prvý 5000 tweetov som importoval, tak že pôvodný script, čo vyberal dáta z postgresu som upravil aby namiesto 25 000 mi dal len 5 000 tweetov a tie som jednucho cez curl v cmd bulk importol do elasticu.

*Curl prikaz:*

curl -H "Content-Type: application/x-ndjson" -XPOST localhost:9200/tweets/\_bulk --data-binary "@psql\_dump\_with\_parent2.txt"



**Úloha 7 :**

3 nodes (1 hlavny 2 vedlajsie) :

Obrázok, na ktorom je text

Automaticky generovaný popis

* Vyhladavanie funguje
* Prezeranie funguje
* Pridavanie funguje
* Mazanie funguje
* Update (script – zmena retweet\_count) funguje

Obrázok, na ktorom je text, oranžové, tmavé

Automaticky generovaný popis

2 nodes (1 hlavny 1 vedlajsi)

Obrázok, na ktorom je text

Automaticky generovaný popis

* Vyhladavanie funguje
* Prezeranie funguje
* Pridavanie funguje
* Mazanie funguje
* Update (script – zmena retweet\_count) funguje

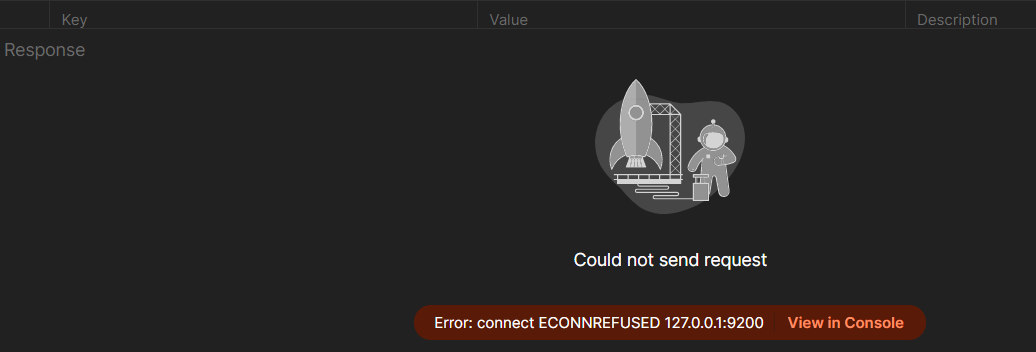
Obrázok, na ktorom je text, oranžové

Automaticky generovaný popis

Obrázok, na ktorom je text, oranžové, nastaviť

Automaticky generovaný popis

2 nodes (2 vedlajsie )

****

Postman uz nefunguje tak skusime cez docker ps.

Obrázok, na ktorom je text

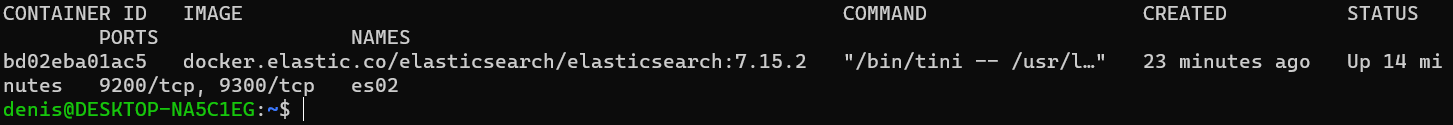
Automaticky generovaný popis

* Vyhladavanie nefunguje
* Prezeranie nefunguje
* Pridavanie nefunguje
* Mazanie nefunguje
* Update (script – zmena retweet\_count) nefunguje

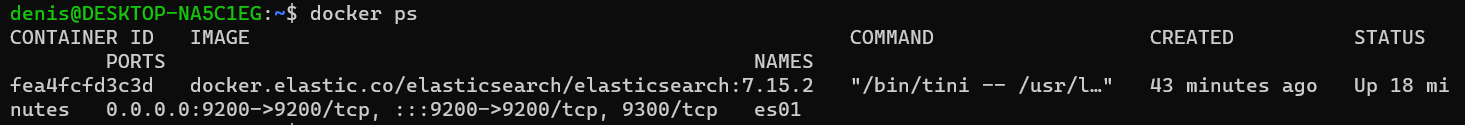
Teda nefunguje nic, ako v prípade bežného dňa fitkára.

1 node( vedlajsi)

* Nefunguje nic



1 node( hlavny)



* Neda sa pingnut ale funguje vyhladavanie
* Prezeranie funguje
* Mazanie nefunguje
* Pridavanie nefunguje
* Update (script – zmena retweet\_count) nefunguje

Obrázok, na ktorom je text

Automaticky generovaný popis

**Úloha 8:**

Pouzival som dva skripty:

Obrázok, na ktorom je text

Automaticky generovaný popis Obrázok, na ktorom je text

Automaticky generovaný popis

\_seq\_no rástol a \_primary\_term sa nemenil

**Úloha 9:**

**Úloha 10:**

**1.**

**2.1.**

**3.**

**4.**

**5.**

**5.1.**

**5.2.**

**Úloha 11:**