# ZADANIE ELASTIC SEARCH

Adam Štuller

1. **Rozbehajte si 3 inštancie Elasticsearch-u**

Link na projekt na Githube je <https://github.com/AdamStuller/tweets-elastic-search/tree/develop>. Všetky curl requesty sú v projekte v adresári requests.

Rozbehal som troj-uzlový cluster pomocou docker-composu (samotny *docker-compose.yml* sa nachádza v projekte, elastic sa spúšta pomocou príkazu *npm run start:elasticsearch*)

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)**Vytvoril som index *tweets*s dvoma shardami a dvoma replikami. Rozhodol som sa tak preto, že aj na prednáške ste použivali takéto nastavenie. Vo všeobecnosti neplatí žadne zlaté pravidlo koľko shardov a replík by malo byt v pomere k uzlom. 2 shardy sú ale dobré riešenie, z dôvodu, že dotazy sú distribuované medzi aspon dva shardy. 2 repliky sú zasa dobre z dôvodu, že exstuje kópia shardu na každom node a teda sa nemusim hned báť, že stratím dáta.

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.**

Vytvoril som mapping už pri vytváraní indexu (curl request je v projekte v requests/create\_index.sh ). Dá sa tam dopodrobna pozrieť celý mapping. Dynamic som nastavil na strict.

V zásade som sa rozhodol analyzovať content tweetu. Je typu text a analyzátor ho teda rozbije na keywords. Ostatné textové properties som nechal ako keywords (určite nechceme rozbíjať meno Donal Trump na dva keywordy ale budeme vyhľadávať meno).   
Objekt tweet\_mentions, ktorý reprezentuje accounty, ktoré spomenuli konkrétny tweet sú typu nested, aby sa na pozadí vytvorili skryté dokumenty v korektnom tvare a dalo sa skrz tweet\_mentions správne vyhľadávať.   
Tags som tiež uložil ako typ text (keďže pole tagov bude na pozadi pretransformované na kywordy aj cheme).

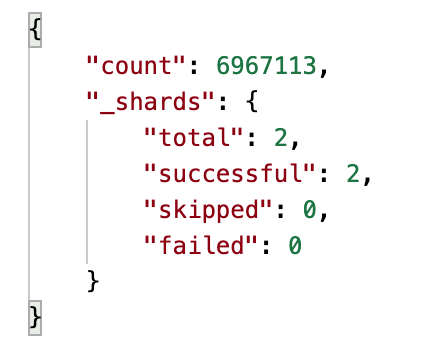
4. **Vytvorte bulk import pre vaše normalizované Tweety.**Na bulk import som použil oficiálneho klienta elasticsearchu pre typescript. Celý script som napísal v typescripte, za pomoci stream api. Script vykoná zašpecifikovaný dotaz nad databázov a pretransformuje ho na stream objektov. Tie objekty namapujeme na náš potrebný tvar a stringifyujeme, aby sme mali potrebný formát pre bulk import. Potom takýto stream použijeme ako datasource pre bulk import oficiálneho klienta.

5. **Importujete dáta do Elasticsearchu**

Dáta som importoval do elastiku, zopár krát mi to na začiatku nevyšlo kvôli typos, čo som urobil. Script som ešte zrýchlil tým, že som vytvoril tabuľku zo spomínaného requestu a teda pri importe ju iba prečítam.  
Experimentoval som aj s nastaveniami bulk importu... nakoniec som ostal pri defaultných nastaveniach a síce 5mb dávka.

Importoval som presne rovnaký počet tweetov ako mám v databáze a síce 6967113.

**Obrázok, na ktorom je text

Automaticky generovaný popis**

6. **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?**  
Zabil som najprv iba jeden node aby som vedel ako sa cluster bude správať. Health endpoint mi vrátil yellow stav. Prídavať dokument som mohol, rovnako ako aj vymazávať, getovať aj updatovať aj vyhladávať.

Obrázok, na ktorom je text

Automaticky generovaný popis

Stále žili dva shardy, každý s jednou replikou. Primárne shardy boli na es01.

Skúsim som zabiť aj druhý uzol (es3), na ktorom sa nachádzali repliky oboch prmárnych uzlov. Posledný živý uzol logoval kopu zaujímavých java errorov. V tomto prípade health a sharps endpointy odpovedali chybou s kódom 503.   
  
Obrázok, na ktorom je text

Automaticky generovaný popis

Dokument som stale dokázal získať pomocou GET na id dokumentu. Search stale fungoval. Update, create ale nie. Keď som znova naštartoval aspoň jeden node, všetko fungovalo.

7. **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.**

Pridal som dokument s id 1 a *retweet*\_*count* 0. \_seq\_no začínalo na 3484451. \_primary\_term na 3. Updatol som dokument niekoľko krát tak, že som pomocu scriptu inkrementoval *retweet\_count* o 1. Pri každom update sa o jedno zvýšil aj \_seq\_no až na 3484457. \_primary\_term sa nemenil.   
  
Keďže oba primarne shardy boli na es01, vypol som tento uzol. Primarne uzly boli potom na node es02.   
  
Keďže ale pracujem cez docker a vystavený je iba port 9200 z es01 musel som ho opäť naštartovať, aby som sa vedel dopytovať. Opäť som ho teda zapol a updatol dokument. \_seq\_no sa zvýšilo opäť o jedno. \_primary\_term bol po týchto dvoch operáciach 5.   
  
Rozhodol som sa ešte zabiť es02 a updatnuť iba pri dvoch uzloch. Primárne shardy sa posunuli opäť na es01. Po update sa \_seq\_no inkrementoval ako vždy a \_primary\_term bol 6, teda zväčšený o 1.

Vyskúšal som práve tieto prípady aby som videl ako sa obe tie properties správajú v rôznych situáciach.