Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 2, 842 16 Bratislava 4

Vyhľadávanie informácii

Vyhľadávanie nad otázkami a odpoveďami

ElasticSearch a StackOverflow dataset

Zadanie 1

Autori: Ladislav Gallay
Cvičiaci: Ing. Michal Kompan, PhD.
Cvičenie: štvrtok, 13:00
Semester: ZS 2016

1. Motivácia

Systémy otázok a odpovedí (angl. Community Question Answering, CQA) sa vďaka svojej otvorenosti širokej verejnosti stali rozsiahlym zdrojom množstva informácii. V mnohých povolaniach je každodennou súčasťou práce a neoddeliteľnou pomôckou najmä programátorom. Jeden z najznámejších CQA systémov StackOverflow obsahuje v súčasnosti cez 12 miliónov otázok. V tomto obrovskom množstve dát je potrebné sa vhodne orientovať. Cieľom tejto práce je pomocou databázy ElasticSearch otvoriť nový pohľad na časť dát z tohto systému. Okrem štandartného textového vyhľadávania StackOverflow neobsahuje špeciálne verejne dostupné nástroje, ktoré by umožňovali robiť detailnejšie pohľadý na dáta a prepojenia medzi nimi.

V rámci tejto práce som priblížil a implementoval vybrané dopyty nad dátami StackOverflow.
Za použitia agregácií a grafov táto práca prináša nové zaujímavé vlastnosti a informácie o vzťahoch medzi otázkami, odpoveďami a používateľmi.

2. Získavanie dát

V rámci zadania som pracoval s datami z portálu StackOverflow[1], ktoré som si svojpomocne stiahol z verejnej webovej stránky, spracoval, a vložil do ElasticSearch databázy. Automatický skript, ktorý získaval data, tzv. crawler, som implementoval v jazyku Ruby. Použil som pri tom niekoľko voľne dostupných gemov ako napríklad Nokogiri pre prácu s HTML formátom.

Program umožuje zadať tag, ktorého otázky sa majú získať a taktiež začiatočnú stránku (záložku) a počet stránok na StackOverflow. Každá stránka zo zoznamu otázok pre konkrétny tag obsahuje predvolene 50 otázok. Program po spustení nájprv identifikuje všetky odkazy na otázky na stránke a následne jednu po druhej otvára a obsah spracuváva do databázy.

Pokiaľ je požadované stiahnuť väčší počet otázok, než sa na aktuálne stránke nachádza, program inkrementálne prechádza cez ďalšie stránky a rovnakým spôsobom získava obsah. Príklad adresy je: http://stackoverflow.com/questions/tagged/php?sort=votes&pageSize=50. Program je pritom možné nakonfigurovať pomocou prepínačov tak, aby sťahoval otázky pre ľubovoľnú značku (tag).

Vzhľadom na limitovaný počet dopytov na server som obmedzil rýchlosť prechádzania cez stránky. A to tak, že medzi jednotlivými kliknutiami na otázky som ponechal voľný čas 0.7 sekundy a medzi jednotlivými stránkami 5 sekúnd. V prípade, že server odpovedal so s návratovým kódom *429 - Too many requests*, program v tomto prípade čakal 10 minút a potom opätovne pokračoval v práci.

Pre urýchlenie sťahovania som bol schopný aplikáciu nasadiť na rôzne servery, ktoré vystupovali pod rôznymi IP adresami. Lokálnu anonymizáciu, prípadne zmenu proxy, som neriešil aj vzhľadom na nespoľahlivosť a nízku garanciu stability pripojenia, ako aj vyššiu technickú náročnosť pri implementácii. Keďže skripty boli nasadená na vzdialených serveroch, bolo potrebné nastaviť lokálnu ElasticSearch databázu tak, aby bola prístupná z vonku na verejnej IP adrese.

Samotný kód crawlera sa nachádza v priloženom súbore *crawler.rb*. Počas získavania dát nenastali žiadne neočakávané komplikácie.

[1] http://stackoverflow.com

3. Opis dát

Dáta získané v bode 2 som uložil do ElasticSearch databázy. Pritom zo samotnej stránky boli na základe identifikovanách HTML značiek extrahované elementy opísané v tabuľke 1. Stiahnuté data boli primárne v jazyku angličtina.

Dáta som špeciálne nespracoval. Základným spracovaním bola identifikácia dátových typov. V prípade dátumu s časom (napr. časová stopa vytvorenia otázky) som do databázy kvôli jednoduchosti ukladal iba dátum bez časovej informácie.

Ku každej otázke som získal aj príslušné komentáre, odpovede a komentáre k odpovediam. Tieto údaje sú uložené vo vnorenej štruktúre, ktorú dokumenty v databáze ElasticSearch podporujú. Kvoli zachovaniu podobnej štruktúry som aj texty otázok ukladal do atribútu s názvom *question*.

Tabuľka č. 1: *Opis vybraných získaných atribútov pre jednu otázku.*

Názov	**Dátový typ**	**Popis**
id	celé číslo	Interný CQA identifikátor
title	text	Nadpis otázky
votes	celé číslo	Počet hlasov pre danú otázku
favorites	celé číslo	Počet obľúbeností otázky
question	text	Samotný text otázky
tags	pole textov	Zoznam značiek (angl. tag) priradených k otázke
created	dátum	Dátum vytvorenia otázky
owner.username	text	Používateľské meno autora otázky
owner.reputation	celé číslo	CQA reputácia autora otázky
owner.gold	celé číslo	Počet zlatých ocenení autora otázky
owner.silver	celé číslo	Počet strieborných ocenení autora otázky
owner.brozne	celé číslo	Počet bronzových ocenení autora otázky
comments	pole	Zoznam komentárov k otázke
comments.score	celé číslo	Skóre v prospech otázky
comments.comment	text	Samotný text komentáru
comments.owner	text	Používateľské meno autora komentáru

comments.created	dátum	Dátum pridania komentáru
answers	pole	Zoznam odpovedí k danej otázke
answers.question	text	Samotný text odpovede
answers.created	dátum	Dátum vytvorenia odpovede
answers.owner	štruktúra	Štruktúra používateľa - rovnaká ako pri otázke
answers.comments	pole	Zoznam komentárov k otázke - podobná štruktúra ako pri komentároch k otázke

Celkovo som zozbieral 210 700 otázok v konečnej veľkosti 1.1 GB dát. Dáta pozostavjú zo otázok získaných pre tagy *php* a *python*. Otázky pre php boli vyberané náhodne. Pre Python boli kvôli vtedajšiemu malému objemu dát zvolené podľa najlepšieho hodnotenia a najväčšieho počtu odpovedí.

V rámci dopytu 4 som si vytvoril nový index, ktorý využíval pokročilejší analyzátor, než ten štandartný v ElasticSearch. Detaily nového idnexu sú opísané v kapitole 4, v sekcii *Dopyt 4*.

4. Dopyty a vyhodnotenie

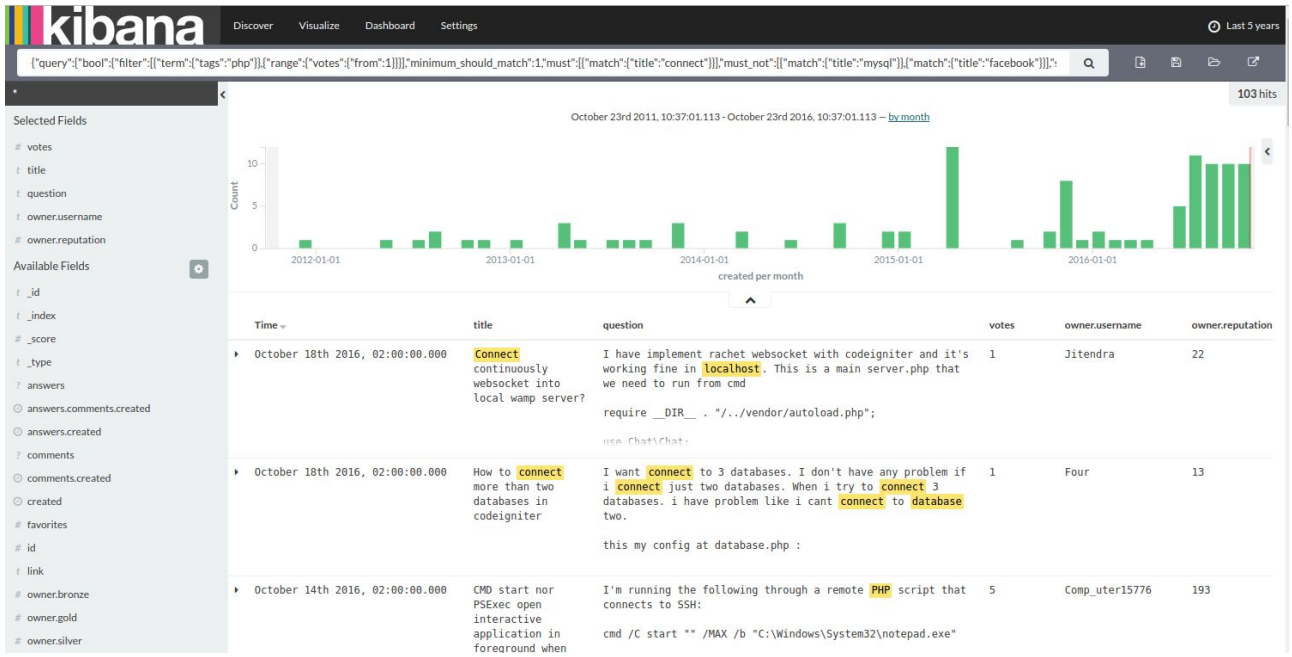
Nad získanými dátami som vykonal niekoľko dopytov do ElasticSearch databázy. Dopyty som pri testovaní spúšťal pomocou Sense doplnku pre prehliadač Chrome. Ako výstupné grafické prostredie som zvolil Kibanu, ktorá bola aj odporúčaná v zadaní úlohy. Kibana umožňovala nadefinovať a vizualizovať aj grafy, ktoré sú neskôr v tomto dokumente zmienené. Všetky použité dopyty, prípadne adresy na výsledné vizualizácie sú súčasťou textových príloh toho dokumentu.

Dopyt 1

V prvom dopyte som vyhľadával otázky, ktoré majú priradenú značku *php* a majú aspoň jeden hlas (vote). Cieľom bolo nájsť otázky, ktoré sa týkajú problému s pripojením. *Title* musel obsahovať slovo *connect*, no zároveň som vyradil problémy s *facebook* a *mysql* pripojením. Pomocou *should* som vyzdvihol otázky, ktoré mali v texte *database localhost* alebo sa týkali značky *cakephp*, pričom museli spĺňať aspoň jedno z týchto pravidiel.

Výsledky som následne zobrazil v prostredí Kibana, kde som filtroval stĺpce: *created, title, question, votes, owner.username, owner.reputation.* Obrázok 1 znázorňuje daný výstup.

Na prvom mieste bola otázka s názvom *"Connect continuously websocket into local wamp server?"*, ktorá mala 1 hlas. V texte obsahovala slovo *localhost* a taktiež mala priradenú značku *cakephp*, čím spĺňala všetky podmienky. Ďalšie otázky potom už spĺňali iba časť podmienok. Táto otázka mala preto najvyššie skóre a nachádzala sa navrchu. Celkovo bolo v tomto datasete nájdených 103 otázok, ktoré vyhovovali zadaným kritériam.



Obrázok 1: *Výsledok prvého dopytu v prostredí Kibana.*

Dopyt 2

V tomto dopyte som použil vlastnú skórovaciu funkciu, ktorá uprednostňovala odpovede s vyšším počtom otázok. Ukážka 1 zobrazuje použitú skórovaciu funkciu. Filtroval som iba tie otázky, ktoré mali tri a menej komentárov. Cieľom bolo vyfiltrovať otázky, ktoré sú celkom jasne zadané a nepotrebovali veľa komentárov a bolo možné na ne odpovedať. Zároveň aspoň jedna z odpovedí musela obsahovať slovo *elasticsearch* a hodnotenie otázky muselo byť nad 10. Tento dopyt by teda mal filtrovať kvalitné otázky, ktorých riešením bol *elasticsearch*. Zoradenie otázok podľa najväčšieho počtu odpovedí by malo uprednostiť otázky, na ktoré existuje viacero riešení, prípadne sú jednoduché a veľké množstvo ľudí z komunity dokázalo otázku zdopovedať.

```
"functions": [{
  "script_score": {
    "script": "return _source.answers.size() * 10 + _source.votes;"
  }
}]
```

Ukážka 1*: Vlastná skórovacia funkcia*

Obrázok 2 znázorňuje výsledky zobrazené v Kibane. Zo stĺpcov sú zobrazené *title, question, votes, owner.reputation* a *answers*. Výsledok obashoval 17 otázok. Prvá otázka "*How to use Bulk API to store the keywords in ES by using Python*" dosiahla skóre 76.68. Už prvé z odpovedí obsahovali slová *elasticsearch*.

Obrázok 2: *Výsledok druhého dopytu v prostredí Kibana.*

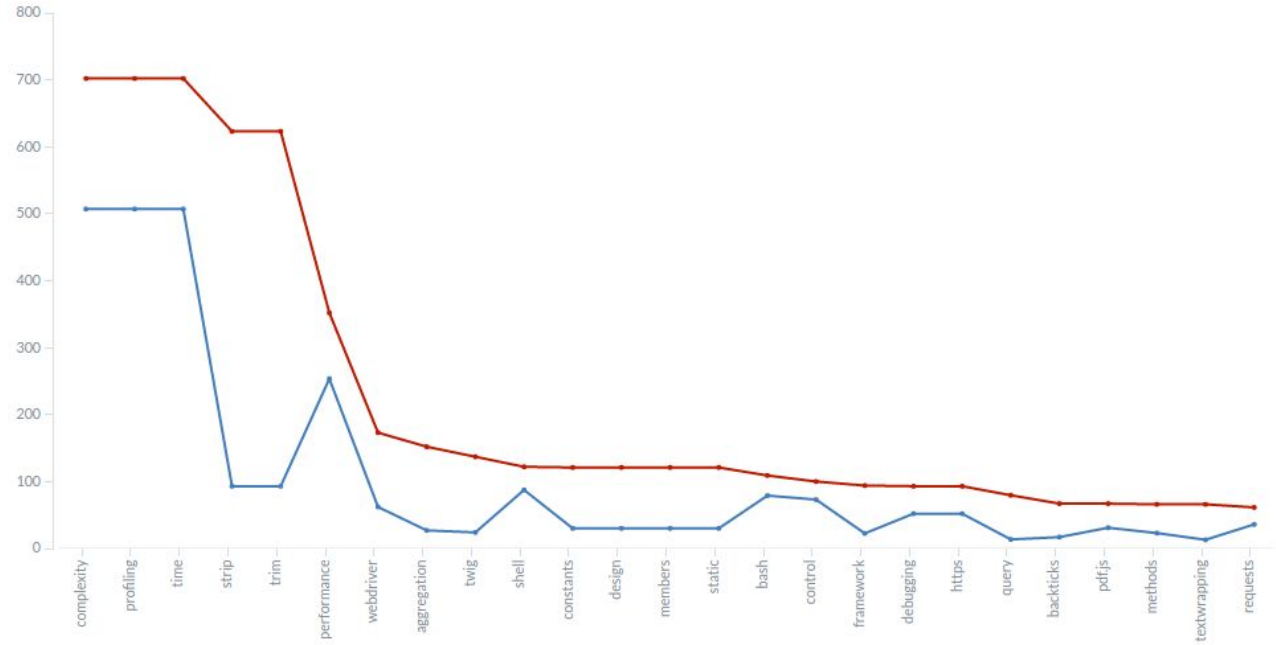
Dopyt 3

Ako ďalší dopyt som si zvolil vizualizáciu v Kibane. Vizualizácie umožňujú rôzne pokročilejšie možnosti filtrovania dát a následne agregácie.

Dopyt filtroval otázky, ktoré sa opýtal používateľ s menom *Chris*. Vo vizualizácii som následne na X-ovej osi zobrazil 25 značiek (tag). Tieto značky boli zoradené zostupne podľa hodnôt na Y-ovej osi. Y-ová os zobrazuje priemerný počet hlasov (votes červená čiara) a priemerný počet obľúbeností (favorites, modrá čiara) na otázkach pre každú značku.

Z grafu, ktorý je výstupom z Kibany, možno pozorovať, že v danom datasete je najpopulárnejšou značkou *complexity* s priemerným počtom hlasov 700 a priemerným počtom obľúbeností okolo 500. Z grafu tiež možno pozorovať, že existuje priamoúmerná závislosť medzi počtom hlasov a počom obľúbeností pre jednotlivé značky.

Medzi ďalšie populárne značky patria *profiling, time, strip, trim, performance* a *webdriver*, či *aggregation*.



Obrázok 3: *Počet hlasov (červená čiara) a počet obľúbeností (modrá čiara) pre prvých 25 použitých pri otázkach, ktoré sa opýtal Chris.*

Dopyt 4

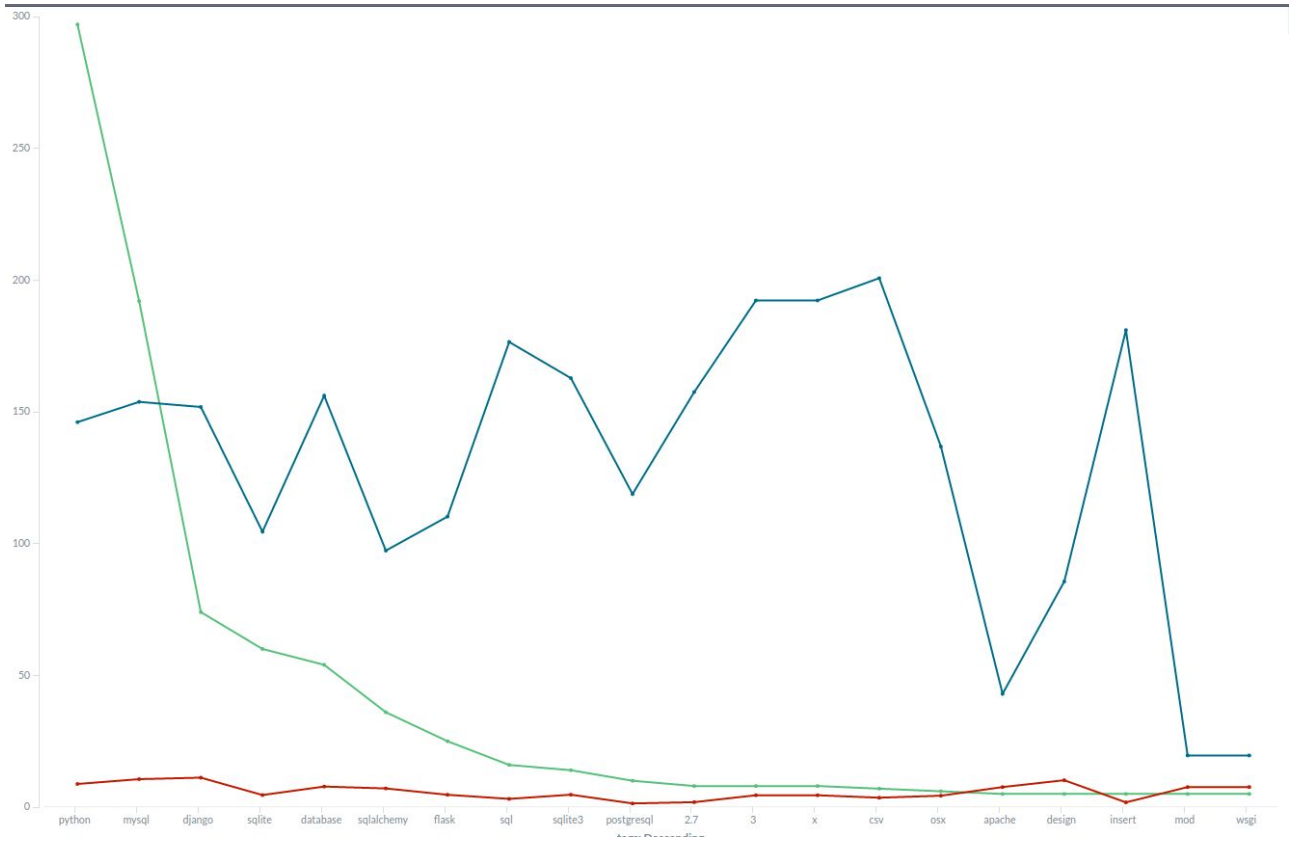
Pre ďalšiu prácu som zistil, že aktuálny polo-automaticky vygenerovaný index v databáze ElasticSearch nemusí byť dostatočný. Preto som si vytvoril vlastný index s názvom *new_questions* a pomocou *_reindex* som do neho vložil časť dát z pôvodného indexu. Vybratl som iba tie otázky, ktoré obsahujú tag *python*.

V rámci indexu som si vytvoril vlastný analyzátor *vi_analyzer*, ktorý odstraňoval HTML značky, mapoval ampersand znak na slovo *and*, odstraňoval vybrané stop-slová (*the, a, an, uhm, ah, heh, hey*), tokenizoval slová s maximálnou dĺžkou tokenu do 10 a všetky písmená považoval za malé. Tento index už obsahoval iba 72 tisíc otázok o celkovej veľkosti 552 MB. Použitý príkaz pre vytvorenie a reindexáciu je v priloženom súbore *query_4_reindex.txt*.

Pre porovnanie indexov som si skúsil v pôvodnom a novom indexe vyhľadať otázky pod tagom *python*, ktoré v otázke obsahujú *an*. V pôvodnom indexe bolo nájdených 3 769 výsledkov, pričom v novom indexe ich bolo 0. To preto, lebo toto slovo je považované ako *stop*-slovo. Týmto som si overil čiastočnú správnosť nového indexu.

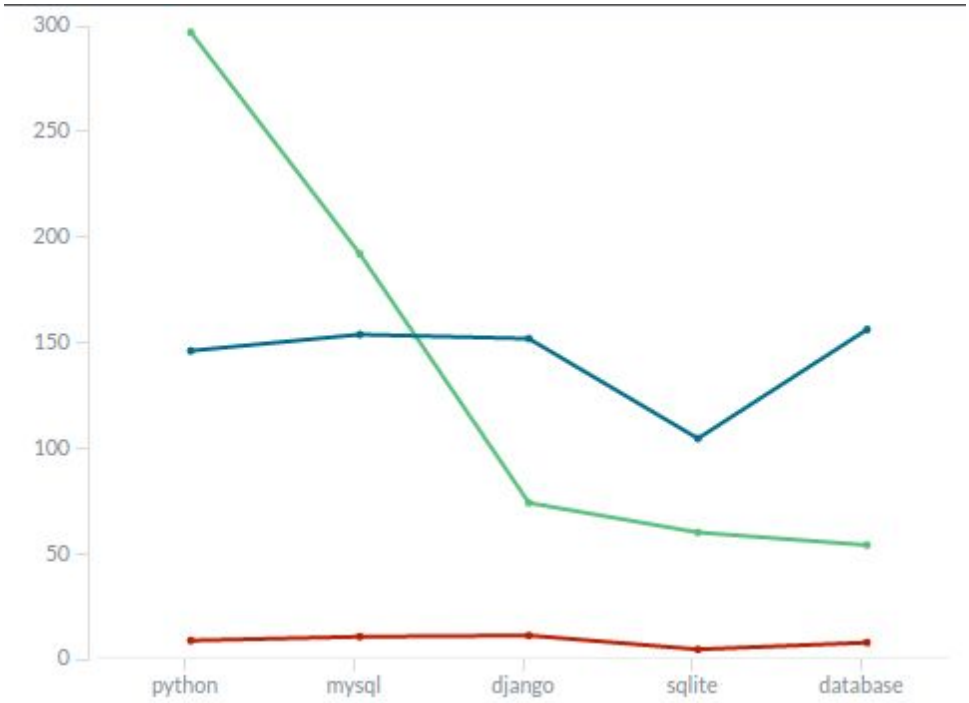
V tomto dopyte som hľadal otázky, ktoré majú aspoň 1 hlas a v názve sa musia týkať problematiky *error database*. Pomocou *should* som potom preferoval otázky, ktoré mali v texte *sqlite* alebo medzi tagmi *mysql*. Následne som si pomocou *aggregations* v ElasticSearch databáze dal vypísať tiež štatistické informácie hlasov a obľúbenosti pre jednotlivé značky. Tieto štatistické informácie obsahovali data ako priemerný počet, minimum, maximum, suma. Minimum pre počety hlasov bol všade 1, čo zodpovedalo zadanému dopytu. Cieľom dopytu je teda zobraziť otázky, ktoré sa týkajú problémov s MySQL alebo SQLite databázami v Pythone.

Najvyšší výskyt mal pochopiteľne tag *python* - na všetkých 297 otázkach, ktoré dopyt vrátil. To preto, lebo tento index bol skonštruovaný iba z *python* otázok. Za ním bol populárny tag *mysql*, taktiež preto, lebo som ho najviac preferoval vo výsledkoch. Tag *sqlite* bol až na štvrtok mieste. To preto, lebo som nepreferoval priamo tento tag, ale otázky, ktoré v texte obsahovali slovo *sqlite*. Pochopiteľne ale väčšina týchto otázok mala potom aj priradený samotný tag *sqlite*. Na obrázku 4 možno vidieiť vizualizáciu z prostredia Kibana pre prvých 20 tagov a vzťah medzi počtom výskytov, priemerným počtom hlasov a priemernou reputáciou používateľov. Z grafu možno pozorovať, že všetky otázky sú si podobné z hľadiska hodnotenia a neexistuje pritom závislosť medzi použitými tagmi a reputáciou používateľa. Tieto tagy používajú rovnako skúsení aj neskúsení používatelia systému StackOverflow.



Obrázok 4: *Prvých 20 značiek a ich počet výskytov (zelená čiara), priemerný počet hlasov (červená čiara) a priemerná reputácia používateľa (modrá čiara).*

Hoci na obrázku 4 nie su vizualizované aj obľúbenosti (z dôvodu prehľadnosti), v rámci dopytu som vyhodnotil aj výsledky z tohto parametra. Pozoroval som opäť priamu úmernosť medzi počtom hlasov a počtom obľúbeností pre jednotlivé značky. Pre lepšiu viditeľnost je na obrázku 5 priblížených iba prvých 5 značiek z obrázka 4.



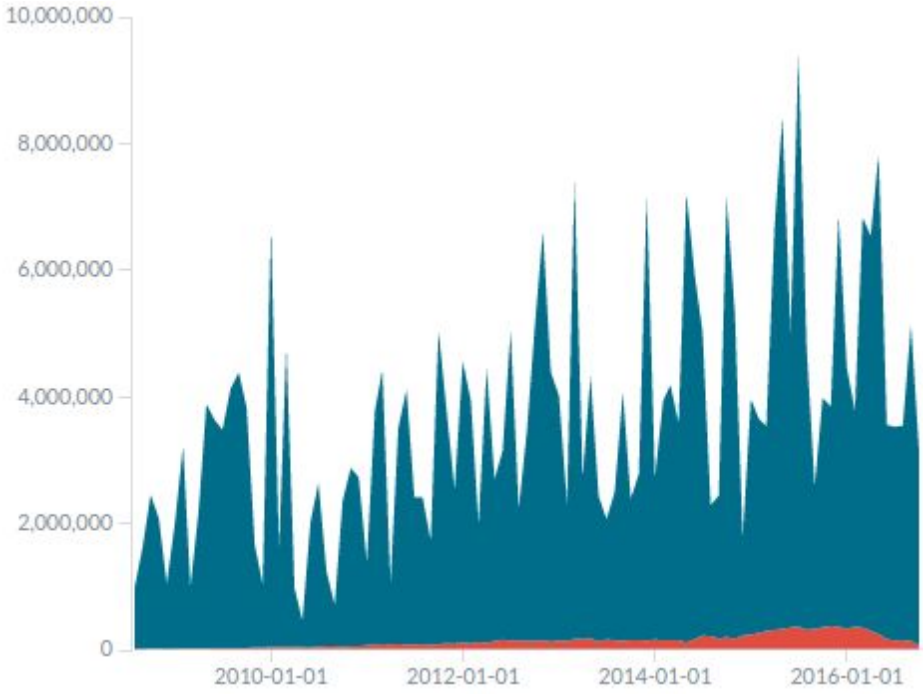
Obrázok 5: *Detailný pohľad na prvých 5 značiek a ich počet výskytov (zelená čiara), priemerný počet hlasov (červená čiara) a priemerná reputácia používateľa (modrá čiara).*

Dopyt 5

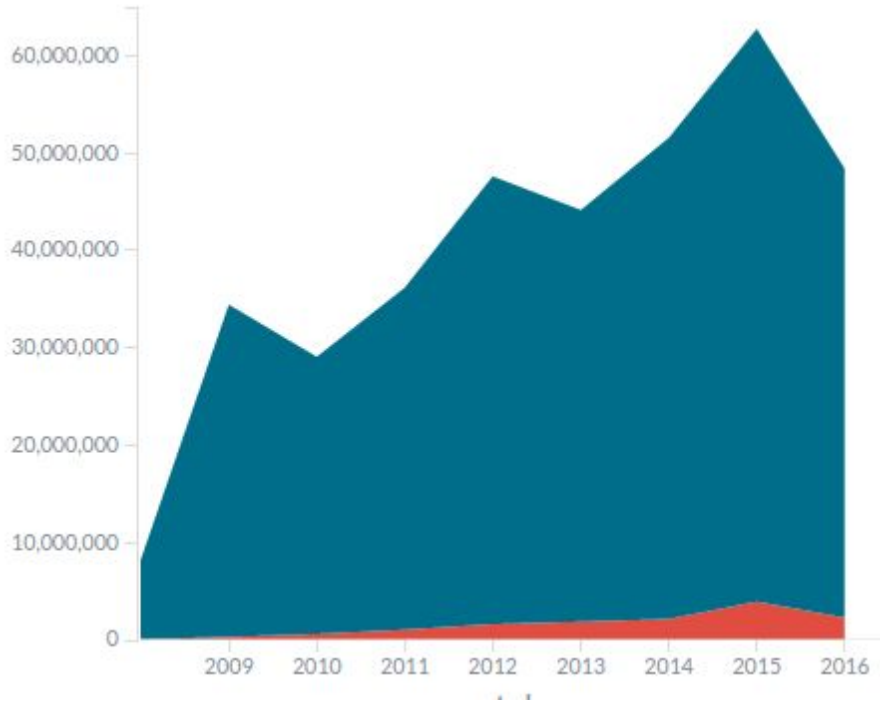
Posledný dopyt som tiež vykonal nad novým indexom *new_questions*. V rámci tohto dopytu som sledoval, ako sa vyvíjala celková reputácia medzi autormi otázok a odpovedajúcimi. Otázky boli rozdelené po mesiacoch a pre každý mesiac bol výslednou hodnotou súčet reputácii používateľov.

Na obrázku 6 možno pozorovať, že reputácia pýtajucich sa (červená) je omnoho nižšia ako reputácia tých, ktorí odpovedajú. To svedčí aj o celej myšlienke CQA systémov, kde sa otázky pýtajú najmä neskúsení používatelia a opovedajú hlavne tí skúsení. Na základe hodnotení odpovedí im potom rapídnejšie stúpa reputácia. Keďže reputácia pýtajúcich sa je neustále nízka a počas obdobia siedmych rokov, ktoré sú zachytené, prekvapivo nestúpla, svedčí to o tom, že skúsení používatelia sa otázky nepýtajú.

Keďže mesačné štatistiky ukazujú mnoho výchyliek, kedy pravdepodobne sú používatelia menej aktívny, napíklad v obodbí sviatkov alebo prázdnin, zobrazil som na obrázku 6 rovnaké dáta a štatistiku, s časovým oknom 1 rok. Na tomto grafe možno pozorovať plynulejší nárast celkovej reputácie v systéme. Na príčine sú dva faktory: zvyšujúci sa počet používateľov a zvyšujúce sa skúsenosti existujúcich používateľov.



Obrázok 5: *Súčet reputácii pýtajucich sa (červená farba) a odpovedajúcich (modrá farba) pre každý mesiac za obdobie siedmich rokov.*



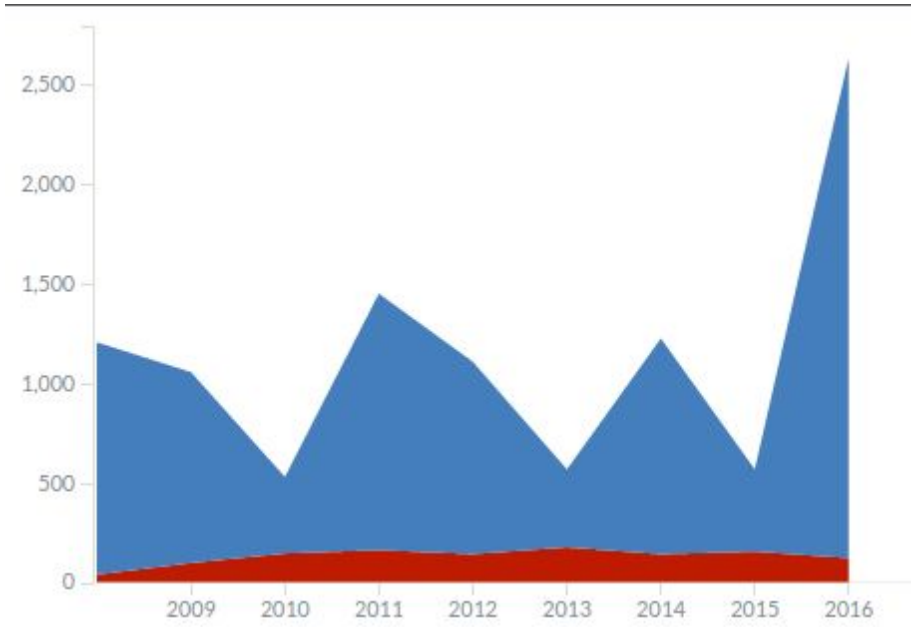
Obrázok 6: *Súčet reputácii pýtajucich sa (červená farba) a odpovedajúcich (modrá farba) pre každý rok za obdobie siedmich rokov.*

Tieto výsledky ilustrujú dopyt na celým indexom *new_questions*. Pre pokročilejšiu prácu s výsledkami som ešte využil funkciu *query_string*. Pomocou nej som iným spôsobom vyskladal dopyt, v ktorom som filtroval otázky, ktoré v texte obsahovali pojmy *parse?* a *web* alebo **TML* a zároveň majú jednu odpoveď. Tento dopyt je znázornený v ukážke 2.

```
"query": "parse? AND (web OR *tml)"
```

Ukážka 2: *Pokročilejší textový dopyt*

Keďže v Kibane sa mi nepodarilo spusiť dopyt s filtrom, ktorý obsahoval vlastný skript, výsledky vizualizované na obrázku 7 sú bez podmienky s počtom odpovedí. Z výsledku možno pozorovať prudký nárast v reputácii používateľov v roku 2016, ktorí sa vyznajú v téme spracovania textov webu. To môže súvisieť s nárastom záujmu u webové technológie, ktoré uchovávajú a ponúkajú k dispozícii čoraz viac dát.



Obrázok 7: *Premerná reputácia pýtajucich sa (červená farba) a odpovedajúcich (modrá farba) pre každý rok v téme týkajucej sa spracovania textu z webu.*

5. Záver

V tejto práci som ukázal pokročilejšie možnosti vyhľadávania a filtrovania pomocou databázy ElasticSearch. Výhodou tohto riešenia je flexibilita, keďže samotný systém StackOverflow neponúka až takú pokročilú možnosť filtrácie a orientovania sa v dátach. Použité dopyty boli veľmi doménovo špecifické, nakoľko aj dátový set bol z veľmi úzkej domény - primárne otázky okolo témy *php* a *python* z programátorského prostredia. Vizualizácia aj používanie prostredia Kibana bolo intuitívne a jednoduché. Toto riešenie je vhodné najmä pri výskume alebo potrebe rýchleho prehľadu nad dátami. Taktiež spôsob spracovania a vyhodnocovania dát je primárne určený pre skúseného používateľa a nie je vhodné pre bežnú verejnú prevádzku napr. na internete.

Aktuálne riešenie je funkčné a testované primárne pre anglický jazyk. Ďalšími možnosťami, ako obohatiť celkové vyhľadávanie a zlepšiť tak aj výsledky, by bolo lepšie analyzovať niektoré parametre, vykonať lepšie predspracovanie dát, vo forme vlastných indexov, ktoré by si poradili aj s cudzími znakmi.