

Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Vyhľadávanie a indexovanie

Martin Schön, Bc.

AIS ID: 103121

E-mail: xschon@stuba.sk

GitHub repozitár: <https://github.com/FIIT-DBS/zadanie-pdt-xSchon>

Predmet: Pokročilé databázové technológie

Zimný semester 2022/2023

Vyhľadávanie a indexovanie

1. Vyhľadanie mfa_russia	2
2. Workeri	2
3. Btree index autori	4
4. Followers count	4
5. Index nad followers countom	5
6. 3 Btree indexy insert	9
7. Dĺžka vytvárania indexu	10
8. Porovnanie indexov	10
9. Conversations content meno "Gates"	13
10. "There are no excuses..." tweet	15
11. Index na vyhľadávanie podľa konca reťazca	18
12. Jednoduché indexy nad countami	19
13. Zložený index nad countami	20
14. Index pre Putin a New World Order cez GiST a GIN	21
15. Linky darujme.sk	23
16. Query pomocou FTS	24
Záver	26

1. Vyhľadanie mfa_russia

```

1  ✓ SELECT * FROM authors WHERE username = 'mfa_russia';

Output (to_tsvector('english','Володимир') × pdt_tweets.public.authors)
1 row
id | name | username | description
---+---+---+---
255471924 | MFA Russia | mfa_russia | Ministry of Foreign Affairs of Russia (Official account) | Countr...

3  ✓ EXPLAIN ANALYZE
4  SELECT * FROM authors WHERE username = 'mfa_russia';

Output Result 76
12 rows
QUERY PLAN
1  Gather (cost=1000.00..135068.52 rows=1 width=125) (actual time=206.884..223.672 rows=1 loops=1)
2    Workers Planned: 4
3    Workers Launched: 4
4    -> Parallel Seq Scan on authors (cost=0.00..134068.42 rows=1 width=125) (actual time=152.082..186.790 rows=0 loops=5)
5          Filter: ((username)::text = 'mfa_russia'::text)
6          Rows Removed by Filter: 1179035
7  Planning Time: 0.068 ms
8  JIT:
9    Functions: 10
10   Options: Inlining false, Optimization false, Expressions true, Deforming true
11   Timing: Generation 1.299 ms, Inlining 0.000 ms, Optimization 0.910 ms, Emission 15.901 ms, Total 18.109 ms
12  Execution Time: 223.858 ms

```

Plánovač vybral paralelný sekvenčný sken. Bez indexov nemá pomôcku a musí tak prehľadať všetky riadky tabuľky - čo sa vykonáva sekvenčným skenom. Ideálne je, ak je možné prehľadávať viacero častí DB na disku súčasne - preto paralelný. DB je rozdelená na bloky, ktoré následne jednotliví paralelizovaní workeri prehľadávajú sekvenčne.

2. Workeri

Výsledok selectu je rovnaký, ako bol pri úlohe 1 - počet workerov nič nemení. Na selecte pracovali 4 workeri, je to spôsobené tým, že v súbore postgresql.conf mám pre túto DB nastavený parameter `max_parallel_workers_per_gather = 4`. Toto číslo som si nastavoval pri zrýchľovaní databázy v zadaní 1, defaultne je nastavené na hodnotu 2. Využili sa všetci pracovníci na vhodný paralelný sken (spomínaný v bode 1).

Maximálne nastaviteľné číslo paralelných pracovníkov na query je 1024 - na základe obmedzení PostgreSQL. Minimálne sa dá vybrať 0 workerov, kedy query nebude paralelizovaná - bude to sekvenčný sken.

```

12
13  ✗ ALTER TABLE authors SET (parallel_workers = 1025);

[22023] ERROR: value 1025 out of bounds for option "parallel_workers"
Detail: Valid values are between "0" and "1024".

```

Parameter `parallel_workers` (počet workerov spustiteľných pre procesy danej tabuľky) tabuľky a `max_parallel_workers_per_gather` spoločne s `max_parallel_workers` (počet workerov, ktoré DB podporí) databázy spolu úzko súvisia. `parallel_workers` je

podradené databázovému parametru. Query teda v praxi využije maximálne taký počet workerov, ktorý predstavuje menšie číslo z dvoch spomínaných parametrov. Ak chcem zmeniť `max_parallel_workers_per_gather`, dá sa to pomocou:

`ALTER SYSTEM SET max_parallel_workers_per_gather = 64;` avšak následne je potrebné reštartovať DB. Toto všetko mi teda určí **maximálny** počet workerov, čo ale neznamená, že ich query planner vždy všetkých využije. Proces určovania počtov je taktiež ovplyvnený cez `max_worker_processes` (default 8), určuje počet background procesov, ktoré zvládne systém podporiť. Ideálne je nastaviť ho na hodnotu `max_parallel_workers`. Rovnako tak je dobré nastaviť aj `max_parallel_maintenance_workers`, ktoré určuje maximálny počet workerov, ktoré môžu byť spustené jedným utility príkazom.

Porovnanie rýchlostí podľa počtu workerov

12 rows	12 rows
<pre> QUERY PLAN 1 Gather (cost=1000.00..116718.06 rows=1 width=125) (actual time=375.984..4243.039 rows=1 loops=1) 2 Workers Planned: 1024 3 Workers Launched: 1023 4 -> Parallel Seq Scan on authors (cost=0.00..115717.96 rows=1 width=125) (actual time=10.219..10.499 rows=0 loops=1024) 5 Filter: ((username)::text = 'mfa_russia'::text) 6 Rows Removed by Filter: 5757 7 Planning Time: 0.061 ms 8 JIT: 9 Functions: 2048 10 Options: Inlining false, Optimization false, Expressions true, Deforming true 11 Timing: Generation 1076.040 ms, Inlining 0.000 ms, Optimization 33.943 ms, Emission 996.222 ms, Total 2106.204 ms 12 Execution Time: 4243.213 ms </pre>	<pre> QUERY PLAN 1 Gather (cost=1000.00..125857.31 rows=1 width=125) (actual time=168.257..184.050 rows=1 loops=1) 2 Workers Planned: 8 3 Workers Launched: 8 4 -> Parallel Seq Scan on authors (cost=0.00..124857.21 rows=1 width=125) (actual time=121.377..135.808 rows=0 loops=9) 5 Filter: ((username)::text = 'mfa_russia'::text) 6 Rows Removed by Filter: 655019 7 Planning Time: 0.090 ms 8 JIT: 9 Functions: 18 10 Options: Inlining false, Optimization false, Expressions true, Deforming true 11 Timing: Generation 2.870 ms, Inlining 0.000 ms, Optimization 1.896 ms, Emission 61.555 ms, Total 66.321 ms 12 Execution Time: 184.283 ms </pre>

Väčší počet workerov nemusí nevyhnutne znamenať lepší (rýchlejší) beh dopytu. Pri 8 workeroch som dosiahol lepší čas, ako pri 4, ale aj 16. Viac workerov viac zaťažuje CPU, teda ich efektívne využitie závisí od typu a počet jadier CPU. Okrem toho - úzke hrdlo práce s databázami spravidla býva nedostatočne rýchly prístup na disk (úložisko dát), čo viac workerov nevyrieši.

3. Btree index autori

```

15 CREATE INDEX authors_usernames_btree_index ON authors
16     USING btree
17     (username);
18 EXPLAIN ANALYZE
19     SELECT * FROM authors WHERE username = 'mfa_russia';
20

```

Output Result 105

4 rows

QUERY PLAN

```

1  Index Scan using authors_usernames_btree_index on authors  (cost=0.43..2.65 rows=1 width=125) (actual time=0.022..0.023 rows=1 loops=1)
2    Index Cond: ((username)::text = 'mfa_russia'::text)
3  Planning Time: 0.060 ms
4  Execution Time: 0.037 ms

```

Po vytvorení indexu je vyhľadávanie násobne rýchlejšie, vďaka využitiu index scanu. Ten nepoužíva viacero workerov, index scan neprechádza cez všetky dáta uložené v DB, ale namiesto toho používa prechádzanie cez indexy uložené v B-tree štruktúre. Vďaka tomu, že bol vyhľadávaný iba jeden záznam, bolo toto prechádzanie veľmi rýchle - strom bolo potrebné prejsť iba raz. Vyhľadávanie v Btree strome má časovú náročnosť $O(\log n)$ - teda máme veľmi veľkú rýchlosť výmenou za pamäť potrebnú na uloženie indexu.

Výsledok zostáva nemenný, keďže sa mení prístup k databáze, ale nie obsah požiadavky, ani tabuľky v databáze:

```

25 SELECT * FROM authors WHERE username = 'mfa_russia';

```

Output pdt_tweets.public.authors

1 row

id	name	username	description	followers_count	following_count	tweet_count	listed_count
255471924	MFA Russia	mfa_russia	Ministry of Foreign Affairs of Ru...	556912	1443	69213	5300

4. Followers count

```

24 EXPLAIN ANALYZE
25     SELECT * FROM authors WHERE (followers_count >= 100) AND (followers_count <= 200);
26

```

Output Result 121

12 rows

QUERY PLAN

```

1  Gather  (cost=1000.00..201791.08 rows=740885 width=125) (actual time=4.088..260.776 rows=760088 loops=1)
2    Workers Planned: 8
3    Workers Launched: 8
4    -> Parallel Seq Scan on authors  (cost=0.00..126702.58 rows=92611 width=125) (actual time=7.512..173.337 rows=84454 loops=9)
5          Filter: ((followers_count >= 100) AND (followers_count <= 200))
6          Rows Removed by Filter: 570565
7  Planning Time: 0.082 ms
8  JIT:
9    Functions: 18
10   Options: Inlining false, Optimization false, Expressions true, Deforming true
11   Timing: Generation 4.569 ms, Inlining 0.000 ms, Optimization 2.564 ms, Emission 64.755 ms, Total 71.889 ms
12  Execution Time: 298.284 ms

```

```

27 ✓ EXPLAIN ANALYZE
28 SELECT * FROM authors WHERE (followers_count >= 100) AND (followers_count <= 120);
29

```

Output Result 122 x

12 rows

QUERY PLAN

```

1 Gather (cost=1000.00..145660.38 rows=179578 width=125) (actual time=3.857..241.020 rows=199937 loops=1)
2   Workers Planned: 8
3   Workers Launched: 8
4   -> Parallel Seq Scan on authors (cost=0.00..126702.58 rows=22447 width=125) (actual time=5.603..163.693 rows=22215 loops=9)
5     Filter: ((followers_count >= 100) AND (followers_count <= 120))
6     Rows Removed by Filter: 632804
7 Planning Time: 0.050 ms
8 JIT:
9   Functions: 18
10  Options: Inlining false, Optimization false, Expressions true, Deforming true
11  Timing: Generation 4.245 ms, Inlining 0.000 ms, Optimization 2.396 ms, Emission 47.737 ms, Total 54.377 ms
12 Execution Time: 251.045 ms

```

Pri zvýšenom počte workerov nie je rozdiel - pri 8 prebehne pre oboch paralelný sekvenčný sken, akurát s rôznou rýchlosťou.

Tipujem ale, že pointou tohto zadania bola ukázať fakt, že pri rozptyle 100 až 200 bude použitý sekvenčný sken. Ak nastavím workerov na 4 (prípadne rozsah na 100 až 250), tak sa ukáže sekvenčný sken. To sa deje preto, že query chce vrátiť veľký počet záznamov - odhaduje približne 740000 riadkov. Pri takomto množstve by získavanie stránok daných záznamov bolo najefektívnejšie, ak pristupujem sekvenčným skenom - má zmysel prehľadávať celú databázu a tak získať všetky vyhovujúce záznamy.

```

24 ALTER TABLE authors SET (parallel_workers = 4);
25 ✓ EXPLAIN ANALYZE
26 SELECT * FROM authors WHERE (followers_count >= 100) AND (followers_count <= 200);
27

```

Output Result 140 x

9 rows

QUERY PLAN

```

1 Seq Scan on authors (cost=0.00..204098.66 rows=740885 width=125) (actual time=2.459..616.114 rows=760088 loops=1)
2   Filter: ((followers_count >= 100) AND (followers_count <= 200))
3   Rows Removed by Filter: 5135088
4 Planning Time: 0.048 ms
5 JIT:
6   Functions: 2
7   Options: Inlining false, Optimization false, Expressions true, Deforming true
8   Timing: Generation 0.172 ms, Inlining 0.000 ms, Optimization 0.143 ms, Emission 2.303 ms, Total 2.617 ms
9 Execution Time: 638.067 ms

```

5. Index nad followers countom

```

3 DROP INDEX authors_followers_count_index;
4 CREATE INDEX authors_followers_count_index ON authors
5   USING btree
6   (followers_count);
7
8 EXPLAIN ANALYZE
9   SELECT * FROM authors WHERE followers_count BETWEEN 100 AND 200;

```

Output Result 159

8 rows

QUERY PLAN

```

1 Index Scan using authors_followers_count_index on authors (cost=0.43..142749.57 rows=740676 width=125) (actual time=0.021..464.313 rows=760888 loops=1)
2   Index Cond: ((followers_count >= 100) AND (followers_count <= 200))
3 Planning Time: 0.060 ms
4 JIT:
5   Functions: 2
6   Options: Inlining false, Optimization false, Expressions true, Deforming true
7   Timing: Generation 0.338 ms, Inlining 0.000 ms, Optimization 0.000 ms, Emission 0.000 ms, Total 0.338 ms
8 Execution Time: 482.505 ms

```

```

61 EXPLAIN ANALYZE
62   SELECT * FROM authors WHERE followers_count BETWEEN 100 AND 120;

```

Output Result 160

8 rows

QUERY PLAN

```

1 Index Scan using authors_followers_count_index on authors (cost=0.43..114944.38 rows=179527 width=125) (actual time=0.020..133.780 rows=199937 loops=1)
2   Index Cond: ((followers_count >= 100) AND (followers_count <= 120))
3 Planning Time: 0.058 ms
4 JIT:
5   Functions: 2
6   Options: Inlining false, Optimization false, Expressions true, Deforming true
7   Timing: Generation 0.229 ms, Inlining 0.000 ms, Optimization 0.000 ms, Emission 0.000 ms, Total 0.229 ms
8 Execution Time: 138.847 ms

```

Bitmap Index Scan je vhodný na použitie pre výber takého počtu prvkov, ktorý je priveľký pre index, ale zároveň primalý pre sekvenčný sken. Každá stránka je zobrazená iba raz. Bitmap Heap sken funguje podobne ako sekvenčný sken, ale navštívi len tie stránky, ktoré obsahujú potrebné dáta - čo sa hodí pre zrýchlenie procesu. Klasický index prechádza riadok po riadku a navštevuje všetky stránky, ktoré potrebuje pre daný riadok, koľkokrát potrebuje za beh programu. Bitmap index scan získa zoznam stránok a z každej vytiahne vhodné riadky- teda vzniká recheck cond.

Recheck cond sa dostáva do funkcie, keď by bitmapa bola priveľká - bitmapa si začne pamätať, aké stránky obsahujú potrebné hodnoty, namiesto toho aby si pamätala hodnoty samotné. Keď sa to stane, recheck condition kontroluje všetky hodnoty na zapamätanej stránke a vytiahne z nej vhodné hodnoty.

Inými slovami - sken vyberie stránky na ktorých si myslí, že budú správne hodnoty a recheck condition potom skontroluje, či táto domnienka platí a či sú hodnoty naozaj správne.

Na dosiahnutie Bitmap Heap scan som potreboval znížiť rozsah na <100, 110>, pretože pri väčšom rozsahu nefungoval.

```
51 ✓ EXPLAIN ANALYZE
52 SELECT * FROM authors WHERE followers_count BETWEEN 100 AND 110;
53
```

Output Result 175

7 rows

QUERY PLAN

```
1 Bitmap Heap Scan on authors (cost=961.82..66341.70 rows=85530 width=125) (actual time=30.253..92.603 rows=108131 loops=1)
2   Recheck Cond: ((followers_count >= 100) AND (followers_count <= 110))
3   Heap Blocks: exact=69915
4   -> Bitmap Index Scan on authors_followers_count_index (cost=0.00..940.43 rows=85530 width=0) (actual time=15.937..15.937 rows=108131 loops=1)
5       Index Cond: ((followers_count >= 100) AND (followers_count <= 110))
6 Planning Time: 0.078 ms
7 Execution Time: 95.149 ms
```


Výsledky queries pre bod 4 a 5:

<100, 200> bez indexu

```

38 EXPLAIN ANALYZE
39 SELECT * FROM authors WHERE (followers_count >= 100) AND (followers_count <= 200);

```

id	name	username	description	followers_count	following_count
1	1391483261748338695 SJ Wealth	SJWealth	Financial Planner with specialty in wealth creation and risk management...	196	196
2	1204043515505729537 369498089 Zehra Aydın	qptt14	İstanbul Sultan Selim Mesleki ve Teknik Anadolu Lisesi=Moda Tasarım Tekn...	144	144
3	1107461437918633984 Chava	_chavafloresr	:p	185	185
4	1142639089962721280 Emilio Lara	cheche28061	ig:cheche28061	194	194
5	1350076379278967297 İlhan Güneşer	IlhanGuneser	Welcome!!!	168	168
6	3344881810 Mohamed Fahs	fahs77		172	172

<100, 200> s indexom

```

38 EXPLAIN ANALYZE
39 SELECT * FROM authors WHERE followers_count BETWEEN 100 AND 200;

```

id	name	username	description	followers_count	following_count
1	952489626857439232 Shivashish Tripathi	shivashish	---Conquer from Within---	100	251
2	1352534170522816513 numan	numan61272207	unutma;her gelen sevmez!!!hiç bir seven de gitmez...gidemez...	100	159
3	1397125922898198528 Sercan Ceylan	SercanCyln	Yeni Türk Edb Arg. G6r. «Eşikte ve Eksik (Öykü, 2021)	100	151
4	1128440406931378176 talhabuyuk	talhabuyuk2		100	150
5	154932275 Mehmet Müftüoğlu	MuftuogluMehmet	Senior software developer	100	501
6	1248263514751295489 Rjmalhotra	rjmalhotra807	Out Of Difficulties Grow Miracles...	100	184
7	3978609281 VIKASH RAJ KUSHWAHA	Only4Vikash	#ProudIndian & #ProudHindu	100	284

<100, 120> bez indexu

```

39 SELECT * FROM authors WHERE (followers_count >= 100) AND (followers_count <= 120);

```

id	name	username	description	followers_count	following_count
1	1428186765785582728 Тимофеев	BF34KmsBiffiQLE		112	112
2	1167833716 Anichu	Anichu_u	19. Auroner de coraza0. Sigo a rubius desde el 2013 y aqui estamoh. Vici...	102	102
3	816767174 Able	aeAble05	The limits of tyrants are prescribed by the endurance of those whom they...	120	120
4	55795174 mrafieq	mrafieq		105	105
5	1493830171 hasan zorlu	hzorlu83		102	102
6	1106117393716142080 @@@	MuskaanJain27	Free-spirited girl with an unpopular opinion. «May offend you!!!!!! So g...	113	113
7	1324025570880049153 Nielen	nielen_n	Creating my own sunshine.	111	111
8	144141326145531335 sicanh	MLMB695	Dwight Shrute stan Account. Ive been to space. top tier taste in music. ...	103	103

<100, 200> s indexom

```

43 EXPLAIN ANALYZE
44 SELECT * FROM authors WHERE followers_count BETWEEN 100 AND 120;

```

id	name	username	description	followers_count	following_count
1	952489626857439232 Shivashish Tripathi	shivashish	---Conquer from Within---	100	251
2	1352534170522816513 numan	numan61272207	unutma;her gelen sevmez!!!hiç bir seven de gitmez...gidemez...	100	159
3	1397125922898198528 Sercan Ceylan	SercanCyln	Yeni Türk Edb Arg. G6r. «Eşikte ve Eksik (Öykü, 2021)	100	151
4	1128440406931378176 talhabuyuk	talhabuyuk2		100	150
5	154932275 Mehmet Müftüoğlu	MuftuogluMehmet	Senior software developer	100	501
6	1248263514751295489 Rjmalhotra	rjmalhotra807	Out Of Difficulties Grow Miracles...	100	184
7	3978609281 VIKASH RAJ KUSHWAHA	Only4Vikash	#ProudIndian & #ProudHindu	100	284

Pri výsledkoch queries je možné vidieť, že výsledky sú v rôznom poradí - výsledky samotné sa nemenia, ale ich poradie môže byť rôzne, nakoľko každý systém pristupuje k pamäti iným spôsobom a teda získa dáta v inom poradí.

Na dôkaz tohto tvrdenia sú v odovzdaní priložené aj dva csv súbory obsahujúce výstupy rozmedzia <100, 120> pre indexovanú aj neindexovanú query:

z02_04_index_100_120_schon.csv a z02_04_noindex_100_120_schon.csv.

6.3 Btree indexy insert

Vkladanie s existujúcimi indexami:

```

69 ✓ EXPLAIN ANALYZE
70 INSERT INTO authors (id, name, username, description,
71                      followers_count, following_count, tweet_count, listed_count)
72 VALUES (8, 'Martin Schon', 'xschon',
73          'Ziak ktory dostane Acko z PDT <3', 7, 3, 358, 2);
i authors_names_btree_index

```

Output Result 29

4 rows

QUERY PLAN

1	Insert on authors	(cost=0.00..0.01 rows=0 width=0)	(actual time=1.103..1.104 rows=0 loops=1)
2	-> Result	(cost=0.00..0.01 rows=1 width=1088)	(actual time=0.001..0.002 rows=1 loops=1)
3	Planning Time: 0.033 ms		
4	Execution Time: 1.126 ms		

Vkladanie po dropnutí indexov:

```

83 ✓ EXPLAIN ANALYZE
84 INSERT INTO authors (id, name, username, description,
85                      followers_count, following_count, tweet_count, listed_count)
86 VALUES (10, 'Martin Schon', 'xschon',
87          'Ziak ktory dostane Acko z PDT <3', 7, 3, 358, 2);

```

Output Result 36

4 rows

QUERY PLAN

1	Insert on authors	(cost=0.00..0.01 rows=0 width=0)	(actual time=0.022..0.022 rows=0 loops=1)
2	-> Result	(cost=0.00..0.01 rows=1 width=1088)	(actual time=0.001..0.001 rows=1 loops=1)
3	Planning Time: 0.021 ms		
4	Execution Time: 0.031 ms		

Vkladanie do tabuľky v ktorej neexistujú indexy je násobne rýchlejšie. Akonáhle totižto vkladám dáta do indexovanej tabuľky, je potrebné zaindexovať aj vložené hodnoty - čo zaberie čas. Pokiaľ vkladám záznam do tabuľky bez indexov, stačí nájsť na disku miesto iba pre takýto záznam - teda kľudne prvé vyhovujúce. Pre každý index je potom tiež potrebné nájsť miesto. A vo finále - pre btree indexy je potrebné správne uložiť index. To znamená nájsť správne miesto v správnom liste - ak nie je, je potrebné rozdeliť listy a vytvoriť nové. Následne je potrebné tiež overiť vybalancovanie stromu, keďže btree strom musí byť vybalancovaný podľa jeho definície.

7. Dĺžka vytvárania indexu

```

pdt_tweets.public> CREATE INDEX conversations_retweet_count_index ON conversations
                    USING btree
                      (retweet_count)
[2022-10-17 10:55:47] completed in 43 s 920 ms
pdt_tweets.public> CREATE INDEX conversations_content_index ON conversations
                    USING btree
                      (content)
[2022-10-17 10:59:24] completed in 3 m 9 s 6 ms

```

Ako vidno, index nad kontentom konverzácií sa vytvára 4-násobne pomalšie. Dĺžka vytvárania indexu je ovplyvnená viacerými faktormi - softvérom, konfiguráciou DB, hardvérom, počtom riadkov tabuľky, vyťaženosťou zariadenia,... V tomto prípade je ale väčšina týchto “externých” parametrov rovnaká.

Rozdiel medzi contentom a retweet_countom je ten, že retweet_count obsahuje jedno číslo pre každý riadok tabuľky. Pri contente ale máme dlhé vety, ktoré obsahujú desiatky slov, ktoré je celé potrebné uložiť do listov stromu a je potrebné k nim spraviť cestu. Je potrebné nájsť väčšie bloky voľnej pamäte, nakoľko by sa dlhá veta nezmestila na menšie miesto. Preto je takéto vytváranie násobne pomalšie a veľkosť indexu kontentu je väčšia, ako pre retweet_county - preto, že dlhý text zaberie viac miesta ako číslo. Taktiež sú čísla na porovnanie jednoduchšie, ako porovnávanie stringov, kde treba porovnať viacero znakov na správne zaradenie do stromu.

8. Porovnanie indexov

Conversation content:

```

111 ✓ SELECT tree_level, index_size, root_block_no, internal_pages, leaf_pages FROM pgstatindex( relname: 'conversations_content_index');

```

tree_level	index_size	root_block_no	internal_pages	leaf_pages
1	5	2517655552	245302	12253

```

112 ✓ SELECT type, avg_item_size, page_size FROM bt_page_stats( relname: 'conversations_content_index', blkno: 3);

```

type	avg_item_size	page_size
1 i	204	8192

```

113 ✓ SELECT type, avg_item_size, page_size FROM bt_page_stats( relname: 'conversations_content_index', blkno: 10003);

```

type	avg_item_size	page_size
1 i	233	8192

114 ✓ `SELECT type, avg_item_size, page_size FROM bt_page_stats(relname: 'conversations_content_index', blkno: 1);`

Output Result 157 ×

	type	avg_item_size	page_size
1	l	214	8192

115 ✓ `SELECT type, avg_item_size, page_size FROM bt_page_stats(relname: 'conversations_content_index', blkno: 1515);`

Output Result 158 ×

	type	avg_item_size	page_size
1	l	312	8192

Conversation retweet count:

111 ✓ `SELECT tree_level, index_size, root_block_no, internal_pages, leaf_pages FROM pgstatindex(relname: 'conversations_retweet_count_index');`

Output Result 159 ×

	tree_level	index_size	root_block_no	internal_pages	leaf_pages
1	2	225804288	209	136	27427

112 ✓ `SELECT type, avg_item_size, page_size FROM bt_page_stats(relname: 'conversations_retweet_count_index', blkno: 3);`

Output Result 167 × Result 166 ×

	type	avg_item_size	page_size
1	i	23	8192

113 ✓ `SELECT type, avg_item_size, page_size FROM bt_page_stats(relname: 'conversations_retweet_count_index', blkno: 10003);`

Output Result 167 × Result 168 ×

	type	avg_item_size	page_size
1	l	729	8192

114 ✓ `SELECT type, avg_item_size, page_size FROM bt_page_stats(relname: 'conversations_retweet_count_index', blkno: 1);`

Output Result 169 × Result 168 ×

	type	avg_item_size	page_size
1	l	729	8192

115 ✓ `SELECT type, avg_item_size, page_size FROM bt_page_stats(relname: 'conversations_retweet_count_index', blkno: 1515);`

Output Result 169 × Result 170 ×

	type	avg_item_size	page_size
1	l	729	8192

Authors followers count:

```
111 ✓ SELECT tree_level, index_size, root_block_no, internal_pages, leaf_pages FROM pgstatindex( relname: 'authors_followers_count_index');
```

	tree_level	index_size	root_block_no	internal_pages	leaf_pages
1	2	43294720	269	26	5258

```
112 ✓ SELECT type, avg_item_size, page_size FROM bt_page_stats( relname: 'authors_followers_count_index', blkno: 3);
```

	type	avg_item_size	page_size
1	i	23	8192

```
113 ✓ SELECT type, avg_item_size, page_size FROM bt_page_stats( relname: 'authors_followers_count_index', blkno: 1);
```

	type	avg_item_size	page_size
1	l	729	8192

Authors name:

```
111 ✓ SELECT tree_level, index_size, root_block_no, internal_pages, leaf_pages FROM pgstatindex( relname: 'authors_names_btree_index');
```

	tree_level	index_size	root_block_no	internal_pages	leaf_pages
1	3	195411968	21550	162	23691

```
112 ✓ SELECT type, avg_item_size, page_size FROM bt_page_stats( relname: 'authors_names_btree_index', blkno: 3);
```

	type	avg_item_size	page_size
1	i	29	8192

```
113 ✓ SELECT type, avg_item_size, page_size FROM bt_page_stats( relname: 'authors_names_btree_index', blkno: 1);
```

	type	avg_item_size	page_size
1	l	95	8192

```
113 ✓ SELECT type, avg_item_size, page_size FROM bt_page_stats( relname: 'authors_names_btree_index', blkno: 51)
```

bt_page_stats		
type	avg_item_size	page_size
l	33	8192

Vysvetlenie:

Level stromu sa líši v závislosti od veľkosti indexu - konkrétne závisí od počtu nódov, ktoré sa v strome nachádzajú. Tieto stromy sú veľmi široké, ale nie až tak veľmi hlboké - vidíme, že sa dostaneme na level 5 pri conversations retweets. Pri menších indexoch vznikajú menej hlboké stromy.

V stromoch sa nachádzajú dva rôzne typy nódov - inner nodes (i), ktoré slúžia na prechádzanie stromom - obsahujú rozhodovacie keys, nie teda samotné values. Hodnoty z databázy sa nachádzajú v listoch (l), teda na poslednej vrstve. Počet vnútorných nodes (v screenshotoch ako root_block_no) je tiež závislý od veľkosti vkladanej hodnôt.

Priemerná veľkosť predmetu zapísaného v indexe sa líši podľa toho, či sa jedná o numerickú hodnotu, alebo o texty. V prípade inner nódov pri stringoch dostanem rôzne hodnoty, podľa veľkosti textu a naplnenia nódu. Pri číslach má každý kľúč preddefinovanú veľkosť - počet bitov potrebných na uloženie čísla v DB. Pri leaf nódoch platí to isté - textové polia sa budú pohybovať, zatiaľ čo čísla potrebujú jednotnú veľkosť.

9. Conversations content meno “Gates”

Bez indexu:

```
110 DROP INDEX conversations_content_index;
111 ✓ EXPLAIN ANALYZE
112 SELECT content FROM conversations WHERE content LIKE '%Gates%';
```

QUERY PLAN	
1	Gather (cost=1000.00..1115441.71 rows=3053 width=159) (actual time=39.561..7737.187 rows=4199 loops=1)
2	Workers Planned: 4
3	Workers Launched: 4
4	-> Parallel Seq Scan on conversations (cost=0.00..1114136.41 rows=763 width=159) (actual time=114.985..7692.999 rows=840 loops=5)
5	Filter: (content ~ '%Gates% '::text)
6	Rows Removed by Filter: 6468562
7	Planning Time: 0.234 ms
8	JIT:
9	Functions: 20
10	Options: Inlining true, Optimization true, Expressions true, Deforming true
11	Timing: Generation 1.968 ms, Inlining 408.362 ms, Optimization 74.463 ms, Emission 54.897 ms, Total 539.690 ms
12	Execution Time: 7738.120 ms

S indexom:

```

119 ✓ EXPLAIN ANALYZE
120 SELECT content FROM conversations WHERE content LIKE '%Gates%';

```

Output Result 192 x pdt_tweets.public.conversations x

13 rows

QUERY PLAN

```

1 Gather (cost=1000.81..641816.16 rows=3053 width=159) (actual time=17.562..5179.449 rows=4199 loops=1)
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Parallel Index Only Scan using conversations_content_index on conversations (cost=0.81..640510.86 rows=763 width=159) (actual time=17.562..5179.449 rows=4199 loops=1)
5     Filter: (content ~ '%Gates% '::text)
6     Rows Removed by Filter: 6468562
7     Heap Fetches: 1014361
8 Planning Time: 0.321 ms
9 JIT:
10   Functions: 5
11   Options: Inlining true, Optimization true, Expressions true, Deforming true
12   Timing: Generation 0.865 ms, Inlining 267.189 ms, Optimization 16.567 ms, Emission 19.840 ms, Total 304.460 ms
13 Execution Time: 5179.994 ms

```

Pri práci so zaindexovanými dátami prebehlo použitie “parallel index only” skenu. Ten je v aktuálnej verzii postgresu podporovaný iba pre btree indexy. Funguje tak, že každý worker vezme jeden blok indexu, prejde ním a vráti všetky platné hodnoty v danom bloku. Tento prístup sa ale použije iba v prípade, že hľadáme content - ak by som vyhľadával všetky stĺpce tabuľky (nie len zaindexovaný content), tak použijem paralelný sekvenčný sken, hoci počet riadkov ostane rovnaký. Index sa nedá použiť, ak slovo nie je na začiatku stringu.

```

104 CREATE INDEX IF NOT EXISTS conversations_content_btree_index ON conversations
105 USING btree (content);
106
107 ✓ EXPLAIN ANALYZE
108 SELECT * FROM conversations WHERE content LIKE '%Gates%';
109
110 -----

```

Output Result 46 x

12 rows

QUERY PLAN

```

1 Gather (cost=1000.00..1064899.51 rows=3053 width=221) (actual time=208.558..14005.718 rows=4199 loops=1)
2   Workers Planned: 8
3   Workers Launched: 8
4   -> Parallel Seq Scan on conversations (cost=0.00..1063594.21 rows=382 width=221) (actual time=208.558..14005.718 rows=4199 loops=1)

```

Výsledky:

```

1 $URG and look who it is in this article, that's right its URG=@Un_Energy @TerraPower @cameconews @BillGates @Something cooking in the kitchen ...
2 09/07/2021 - Une femme tient dans ses mains un journal de 2011 qui dit noir sur blanc que Bill Gates va commencer "la dépopulation par la vaccin...
3 100%#Politics #BillGates #Russia #Ukraine https://t.co/FqVfbi9FI
4 @10 DowningStreet @BorisJohnson Urgent! @mundorecord #PutinHitler@SF_Moro @POTUS @SecDef @PressSec @SecBlinken @BillGates @yourano @g1 @Metropo...
5 (1/2) Un dem Film "Enemy at Gates" geht es um einen Zweikampf zweier Scharfschützen in Stalingrad. Während der Deutsche eine Filmerfindung ist, ...
6 1/Bill Gates PAHO are very concerned about under vaccinated Ukraine refugees entering the Americas and suggests you kindly vax them. As the #Ukr...
7 (1) Ca nous rappelle le laboratoire P4 de #wuhan d'où s'est "échappé" le #covid19 et dans lequel les #USA étaient mouillés . On dirait que cette...
8 1/7@Hey, Folks. Here are some information about #ESOFamAgainstWar (-&gt; thread)@Some streamers will go live tomorrow evening to make a Peace ...
9 @1Marzia2 I RUSSI AVANZANO ancora. Mariupol sta per cadere e i media IMPAZZISCONO di RABBIA! Ora vogliono che saltino i negoziati e la guerra mo...
10 @1 Under the guise of #Ukraine #WHO and #BillGates are pulling a fast one friends! This cant be allowed. Some non-elected officials should NO...
11 1/ We always overestimate the change that will occur in the short term and underestimate the change that will occur in the long term.@People ov...
12 "2017/ #KlausSchwab:-> "Gurur duyduğumuz şey,#WEF Genç küresel liderlerimizle ülkelerin kabinelerine,nüfuz etmemizdir."@#YoungGlobalLeaders@92'de...
13 2019 Event 201: #COVID pandemic simulation put on by the Gates Foundation. A few months later, the simulated crises... @2019 Plan A: Pentagon s...
14 2187 residents of Mariupol were killed by russian air strikes!@jensstoltenberg@OlafScholz@a_schallenberg@BillGates https://t.co/sCzXxZNRaX
15 2/2 @seanhannity @Bill Gates, Charles Schwab & possibly Fauci are also globalists involved in the Great Reset. Build Back Better is the Gre...
16 24 month global pandemic... @Seamlessly transitioned into full scale war... @Theatre to keep you scared (If you die in the process of the lie, bon...
17 2ème Partie@Bill Gates de du lâcher de moustiques stériles en Amérique et en Europe pour combattre des maladies mortelles.@#UkraineRussie@#C...
18 3ème Partie@Bill Gates Le lâcher de moustiques stériles en Amérique et en Europe pour combattre des maladies mortelles.@#UkraineRussie@#Covi...
19 468: DEEPFAKE UFOBAMA (Live)@#Facelikesun #CCNT #Christian #Jesus #Bible #LiveStream #Space #Technology #Flippy #Crypto #Bitcoin #BitcoinCash...
20 4/to enforce a no-fly zone, military air superiority is a prerequisite to protect #NATO pilots. Fmr. SECDEF Robert Gates made this point to Cong...
21 7 Sins, 7 Gates, 7 Ways to die: Follow the Way of 7 Sins through the 7 Gates of Hell @SEASON 1: @Madison But

```

Ako som už vyššie uvádzal, výsledky sa pre nezaindexované tabuľky nemenia, maximálne tak poradie. Všetky contenty je možné nájsť v súbore `z02_09_schon.csv`.

10. “There are no excuses...” tweet

```

112 ✓ EXPLAIN ANALYZE
113 SELECT content, possibly_sensitive FROM conversations WHERE content LIKE 'There are no excuses%'
114 AND possibly_sensitive;
115
Output
-- S10 -----
-- S10 ----- 2 X
12 rows
113
QUERY PLAN
1 Gather (cost=1000.00..1115139.61 rows=32 width=160) (actual time=7113.908..7129.262 rows=1 loops=1)
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Parallel Seq Scan on conversations (cost=0.00..1114136.41 rows=8 width=160) (actual time=6432.188..7094.769 rows=0 loops=5)
5     Filter: (possibly_sensitive AND (content ~ 'There are no excuses%':text))
6     Rows Removed by Filter: 6469402
7 Planning Time: 0.157 ms
8 JIT:
9   Functions: 20
10  Options: Inlining true, Optimization true, Expressions true, Deforming true
11 Timing: Generation 1.945 ms, Inlining 312.973 ms, Optimization 64.400 ms, Emission 44.226 ms, Total 423.543 ms
12 Execution Time: 7129.636 ms

```

Index sa nepoužil, pretože dopyt obsahoval podmienku cez nezaindexovaný stĺpec.

Ak by som používal samotné vyhľadávanie cez There are no excuses... na začiatku vety, tak takýto dopyt dokáže Btree index spracovať:

```

113 ✓ EXPLAIN ANALYZE
114 ✓ SELECT content FROM conversations WHERE content LIKE 'There are no excuses%';

```

Output S10 Result 58

13 rows

QUERY PLAN

```

1 Gather (cost=1000.81..641816.16 rows=3053 width=159) (actual time=2327.895..2414.086 row
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Parallel Index Only Scan using conversations_content_btree_index on conversations

```

Výsledok:

```

111 ---- S10 -----
112 EXPLAIN ANALYZE
113 ✓ SELECT content, possibly_sensitive FROM conversations WHERE content LIKE 'There are no excuses%'
114 AND possibly_sensitive;
115 ----- S10 -----

```

Output pdt_tweets.public.conversations

1 row

content possibly_sensitive

```

1 There are no excuses, zero, non, zilch, nada.«The US and NATO have been arming scum, 14000 dead by Zelensky and Azov's hand. No amount of ap... true

```

Tento výsledok bude zhodný pre všetky následovné riešenia bez ohľadu na indexáciu.

Zlepšiť dopyt môžeme tak, že

- pridáme index aj nad possibly sensitive

```

128 ✓ EXPLAIN ANALYZE
129 ✓ SELECT content, possibly_sensitive FROM conversations WHERE content LIKE '%There are no excuses%'
130 AND possibly_sensitive;

```

Output pdt_tweets.public.conversations Result 213

9 rows

QUERY PLAN

```

1 Gather (cost=1000.44..29739.28 rows=32 width=160) (actual time=11636.611..11694.081 rows=1 loops=1)
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Parallel Index Scan using conversations_sensitive_index on conversations (cost=0.44..28736.08 rows=8 width=160) (actual time=11440.334..11501.667 rows=0 loops=5)
5     Index Cond: (possibly_sensitive = true)
6     Filter: (content ~ '%There are no excuses% '::text)
7     Rows Removed by Filter: 71664
8 Planning Time: 0.416 ms
9 Execution Time: 11694.106 ms

```

Ak mám dva indexy nad dvoma stĺpcami, čas sa zlepší približne o 15%.

b) Vytvorenie jedného, zloženého indexu nad množinou (content, possibly_sensitive)

```

135 ✓ EXPLAIN ANALYZE
136 SELECT content, possibly_sensitive FROM conversations WHERE content LIKE '%There are no excuses%'
137 AND possibly_sensitive;

```

Output pdt_tweets.public.conversations x Result 218

9 rows

QUERY PLAN

```

1 Gather (cost=1000.44..29739.28 rows=32 width=160) (actual time=6995.803..7043.608 rows=1 loops=1)
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Parallel Index Scan using conversations_sensitive_index on conversations (cost=0.44..28736.08 rows=8 width=160) (actual time=6946.019..6947.617 rows=0 loops=5)
5       Index Cond: (possibly_sensitive = true)
6       Filter: (content ~ '%There are no excuses% '::text)
7       Rows Removed by Filter: 71664
8 Planning Time: 0.313 ms
9 Execution Time: 7043.625 ms

```

Týmto sa dostanem na hodnotu približne 7 sekúnd, čo je opäť zlepšenie oproti rozdeleným indexom.

c) Najrýchlejšie riešenie - vytvorenie zloženému indexu nad dvoma stĺpcami v poradí (possibly_sensitive, content)

```

135 ✓ EXPLAIN ANALYZE
136 SELECT content, possibly_sensitive FROM conversations WHERE content LIKE '%There are no excuses%'
137 AND possibly_sensitive;

```

Output pdt_tweets.public.conversations x Result 215

10 rows

QUERY PLAN

```

1 Gather (cost=1000.81..13667.02 rows=32 width=160) (actual time=134.717..176.686 rows=1 loops=1)
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Parallel Index Only Scan using conversations_sen_con_index on conversations (cost=0.81..12663.82 rows=8 width=160) (actual time=94.346..108.664 rows=0 loops=5)
5       Index Cond: (possibly_sensitive = true)
6       Filter: (content ~ '%There are no excuses% '::text)
7       Rows Removed by Filter: 71664
8       Heap Fetches: 13992
9 Planning Time: 0.334 ms
10 Execution Time: 176.714 ms

```

Takéto riešenie, podobne ako b) používa Paralelný index scan. Toto riešenie je však neporovnateľne najrýchlejšie, vďaka štýlu query a typu, akým vyberáme hodnoty. Tento index najskôr zahodí všetky výsledky, ktoré nie sú possibly_sensitive - tie tvoria 98.9% výsledkov. Vďaka tomu je vyhľadávanie nad contentom extrémne rýchle. Samozrejme, tento konkrétny výsledok dostanem len pre takto zadanú úlohu - ak by som napríklad mal vyhľadať tie, ktoré NIE SÚ possibly_sensitive, tak stále ostane preskúmať skoro všetky riadky a teda rýchlosť nebude takto závažne rýchlejšia.

11. Index na vyhľadávanie podľa konca reťazca

```

146 CREATE INDEX reverse_content_index ON conversations
147 USING btree
148 (reverse(content));
149
150 EXPLAIN ANALYZE
151 SELECT * FROM conversations WHERE reverse(content) LIKE reverse('%https://t.co/pkFwLXZlEm');

```

Output pdt_tweets.public.conversations x Result 265 x

12 rows

QUERY PLAN

```

1 Gather (cost=1000.00..1151526.79 rows=161735 width=221) (actual time=12764.635..12800.690 rows=1 loops=1)
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Parallel Seq Scan on conversations (cost=0.00..1134353.29 rows=40434 width=221) (actual time=11844.330..12737.073 rows=0 loops=5)
5     Filter: (reverse(content) ~ 'mELZXLwFkp/oc.t//:sptth%':text)
6     Rows Removed by Filter: 6469402
7 Planning Time: 0.063 ms
8 JIT:
9   Functions: 10
10  Options: Inlining true, Optimization true, Expressions true, Deforming true
11 Timing: Generation 1.792 ms, Inlining 342.935 ms, Optimization 60.366 ms, Emission 27.797 ms, Total 432.889 ms
12 Execution Time: 12800.914 ms

```

V tejto úlohe je podstatné správne vytvoriť index. Btree indexy nedokážu vyhľadávať values podľa konečných hodnôt, je potrebné správať sa k tomuto zadaniu tak, ako by znenie bolo “...tweet, ktorý začína reťazcom...”. Preto teda otočím celý content. Pri vytváraní indexu použijem reverse nad kontentom, teda sa mi zaindexuje naopak. text_pattern_ops je potom dobrým parametrom, pretože podporuje kontrolovanie po charakteroch poľa textu (preto text_pattern_ops).

Následne pri SELECTe vyberám otočený content a porovnávam ho s otočeným textom, keďže chcem nájsť link v tomto tvare na začiatku textu contentu.

Výsledok je len jeden:

```

146 SELECT * FROM conversations WHERE reverse(content) LIKE reverse('%https://t.co/pkFwLXZlEm');
147
148

```

reverse_content_index

Output S10 x S10 x 2 x pdt_tweets.public.conversations x

1 row

Tx: Auto DDL

	id	author_id	possibly_sensitive	content	language	
1	1502105081151311873	1501402131030364163	false	Vladimir Putin Russia ...	en	Twit

12. Jednoduché indexy nad countami

Bez indexu:

```

154 ✓ EXPLAIN ANALYZE
155 SELECT * FROM conversations WHERE reply_count > 150
156 AND retweet_count >= 5000
157 ORDER BY quote_count;

```

Output pdt_tweets.public.conversations x Result 268 x

19 rows

QUERY PLAN

```

1 Gather Merge (cost=1135492.67..1136676.60 rows=9888 width=221) (actual time=10916.316..10952.164 rows=8364 loops=1)
2   Workers Planned: 4
3   Workers Launched: 4
4   -> Sort (cost=1134492.61..1134498.79 rows=2472 width=221) (actual time=10889.921..10890.060 rows=1673 loops=5)
5     Sort Key: quote_count
6     Sort Method: quicksort Memory: 1152kB
7     Worker 0: Sort Method: quicksort Memory: 623kB
8     Worker 1: Sort Method: quicksort Memory: 718kB
9     Worker 2: Sort Method: quicksort Memory: 526kB
10    Worker 3: Sort Method: quicksort Memory: 571kB
11    -> Parallel Seq Scan on conversations (cost=0.00..1134353.29 rows=2472 width=221) (actual time=195.649..10888.849 rows=1673 loops=5)
12      Filter: ((reply_count > 150) AND (retweet_count >= 5000))
13      Rows Removed by Filter: 6467729
14 Planning Time: 0.140 ms
15 JIT:
16   Functions: 10
17   Options: Inlining true, Optimization true, Expressions true, Deforming true
18 Timing: Generation 1.957 ms, Inlining 293.021 ms, Optimization 83.623 ms, Emission 49.434 ms, Total 428.035 ms
19 Execution Time: 10952.728 ms

```

S indexami:

```

154 ✓ EXPLAIN ANALYZE
155 SELECT * FROM conversations WHERE reply_count > 150
156 AND retweet_count >= 5000
157 ORDER BY quote_count;
158

```

Output pdt_tweets.public.conversations x Result 272 x

9 rows

QUERY PLAN

```

1 Sort (cost=21581.05..21605.78 rows=9889 width=221) (actual time=793.359..794.063 rows=8364 loops=1)
2   Sort Key: quote_count
3   Sort Method: quicksort Memory: 3594kB
4   -> Index Scan using reply_count_index on conversations (cost=0.44..20924.84 rows=9889 width=221) (actual time=5.117..790.413 rows=8364 loops=1)
5     Index Cond: (reply_count > 150)
6     Filter: (retweet_count >= 5000)
7     Rows Removed by Filter: 94248
8 Planning Time: 1.086 ms
9 Execution Time: 794.255 ms

```

Pri spustení s 3 jednoduchými indexami sa využije iba `reply_count_index`, pretože je efektívnejší, ako `retweet_count`, keďže `reply_count` je viac limitujúci - teda ostane mi menej dát. Quicksort použitý cez `quote_count` je tiež rýchlejší, keďže využije vlastnosti Btree - vľavo menšia, vpravo väčšia hodnota.

Najskôr sa vyfiltruje `retweet_count` (bez indexu) cez filter - tento filter kontroluje každý riadok, ktorý je vybraný ako vhodný na základe indexu nad `reply_countom`. Nakoniec sú výsledky usporiadané a to pomocou quicksortu nad `quote_countom`. Vďaka využitiu indexov je táto query stále násobne rýchlejšia, ako bezindexová.

13. Zložený index nad countami

```

154 CREATE INDEX quote_retweet_reply_index ON conversations
155 USING btree
156 (reply_count, retweet_count, quote_count);
157
158 ✓ EXPLAIN ANALYZE
159 SELECT * FROM conversations WHERE retweet_count >= 5000
160 AND reply_count >= 150
161 ORDER BY quote_count;
162

```

Output pdt_tweets.public.conversations Result 285

7 rows

QUERY PLAN

```

1 Sort (cost=7889.03..7834.17 rows=10056 width=221) (actual time=13.756..14.722 rows=8365 loops=1)
2   Sort Key: quote_count
3   Sort Method: quicksort  Memory: 3594kB
4   -> Index Scan using quote_retweet_reply_index on conversations (cost=0.56..7140.52 rows=10056 width=221) (actual time=0.029..11.411 rows=8365 loops=1)
5     Index Cond: ((reply_count >= 150) AND (retweet_count >= 5000))
6 Planning Time: 0.270 ms
7 Execution Time: 14.948 ms

```

Ako sa dalo predpokladať, zložený index beží rýchlejšie, ako tri separátne indexy. Select dokáže prehľadávať nad tabuľkou všetkých troch indexov, vďaka čomu je rýchlejší. Rozdiel v pláne je, že nie je potrebné opakovane aplikovať filter, ale aplikuje sa iba raz - v podobe filtrovania skrz index. To zabezpečí veľkú rýchlosť, keďže nie je potrebné vyhľadávať viac krát, ale stačí iba raz - a aj to v Btree.

Tento query plan vyberá vhodné záznamy rovno pomocou index BTree nad reply_countom a retweet_countom a teda nemá potrebu filtrovať ešte riadky dodatočne - netreba prechádzať každý riadok viacnásobne. Sortovanie potom rovnako prebehne cez quick_sort quote_countu.

Výsledky boli pre úlohu 12 a 13 rovnaké, akurát sme sa k nim dostali iným spôsobom a sú teda v inom poradí.

```

151 ✓ SELECT * FROM conversations WHERE retweet_count >= 5000
152 AND reply_count >= 150
153 ORDER BY quote_count;
154

```

Output -- S10 -----,----- x -- S10 -----,----- 2 x pdt_tweets.public.conversations x

1-500 of 501+ Tx: Auto DDL

	id	author_id	content	possib
1	1497786759710752768	1428688987259080704	\$50 700K • 5 Hours RT & Follow Me (🔔)	false
2	1498156335581573120	1428688987259080704	\$50 700K • 4 Hours RT & Follow Me (🔔)	false
3	444848423804866560	258809606	What an asshole. #Putin knows #angelamerkel is afraid of dogs yet brings...	false
4	1502965141998272513	1347140802577788928	\$100 • 1,4 JT 12 HOURS RT & Follow @DukeShiller (🔔)	false
5	1498836083416584195	1389658584196149251	\$50 700.000 IDR • End 8 hour RT Like & Follow me (🔔)	false
6	1504290852151390212	1347140802577788928	\$100 • 1,4 JT 12 HOURS RT & Follow me (🔔)	false
7	1497825913178066948	1347140802577788928	\$50 • 700K 4 HOURS RT + Follow @Missypromotes	false
8	1497807026092531721	1363765131172220933	\$75 1,05 JT in 5 HOURS RT & Follow @brxwlo1	false
9	1504084445384507394	1200355745600069635	\$150 in 24 hours 🌸 RT + Follow @LRI@CryptoRussDPDI + @cryptogod1101	false
10	1503936472994381825	1200355745600069635	\$150 in 24 hours 🌸 RT + Follow @Imaginary_Apes@metarpgnft@Tom...	false
11	1498174280890449921	1461324154381103112	\$200 2.800.000 IDR in 24 Hours RT & Follow @colinjordan + Li...	false
12	1498607064436068353	1449414811436023812	giveaway 100\$ 1,400,000 IDR -- rt follow (🔔) @flozinbtc ends in 8 ...	false
13	1502275400336314368	1449414811436023812	giveaway 100\$ 1,400,000 IDR --rt follow (🔔) @nemmogiveaway ends in...	false
14	1497827472431149056	1347140802577788928	\$80 • 1.12 JT 12 HOURS RT & Follow @aiyahgameverse	false
15	1504756243969490950	1461324154381103112	\$150 2.100.000 IDR in 12 Hours RT & Follow @HectorDAO_HEC	false
16	1497815959637151744	1449414811436023812	giveaway 100\$ 1,400,000 IDR -- rt & follow @FynnToTheMoon ends i...	false
17	1497839586822656001	1363765131172220933	\$75 1,05 JT in 5 HOURS RT & Follow @GlazeCrypto (🔔)	false
18	420419735072014336	116362700	Winston is the truth	false

Všetky hodnoty sa dajú nájsť v súbore z02_12_schon.csv.

14. Index pre Putin a New World Order cez GiST a GIN

GIN:

```

187 CREATE INDEX conversations_gin_content_index ON conversations
188     USING gin(to_tsvector('english', content));
189 ✓ EXPLAIN ANALYZE
190 SELECT *
191 FROM conversations
192 WHERE to_tsvector('english', content) @@ to_tsquery('english', 'Putin & New<->World<->Order')
193 AND possibly_sensitive;

```

Output | COUNT(*)bigint | plainto_tsquery(url):tsquery | Result 437

10 rows

QUERY PLAN

```

1 Bitmap Heap Scan on conversations (cost=27.50..28.86 rows=1 width=221) (actual time=43.485..71.689 rows=7 loops=1)
2   Recheck Cond: (to_tsvector('english':regconfig, content) @@ 'putin' & 'new' <-> 'world' <-> 'order')::tsquery)
3   Rows Removed by Index Recheck: 88
4   Filter: possibly_sensitive
5   Rows Removed by Filter: 841
6   Heap Blocks: exact=935
7   -> Bitmap Index Scan on conversations_gin_content_index (cost=0.00..27.50 rows=1 width=0) (actual time=41.392..41.392 rows=936 loops=1)
8     Index Cond: (to_tsvector('english':regconfig, content) @@ 'putin' & 'new' <-> 'world' <-> 'order')::tsquery)
9 Planning Time: 0.129 ms
10 Execution Time: 71.712 ms

```

GIN výsledky:

```

190 SELECT *
191 FROM conversations
192 WHERE to_tsvector('english', content) @@ to_tsquery('english', 'Putin & New<->World<->Order')
193 AND possibly_sensitive;

```

Output | COUNT(*)bigint | plainto_tsquery(url):tsquery | pdt_tweets.public.conversations

7 rows

id	author_id	content	possibly_sensitive	language	source	retweet_count
1	1497395218659745882	47853235 The experience @davidaherst has had in Russia qualifies him for explaini...	• true	en	Twitter for Android	0
2	1497653688928028931	1178895780188884355 #Putin and the New World Order from Damascus to Kiev-@WhiteHouse spokesma...	• true	en	Twitter for Android	3
3	1498019842183544838	834833178867234816 RT @veteranstoday: Putin: New World Order Worships Satan-https://t.co/fd1...	• true	en	Twitter for Android	0
4	1498219488969437184	736549887981282384 "We fight for the New World Order"-We got it...#UkraineRussiaWar #noAspo...	• true	en	Twitter for iPhone	1
5	1498483618806449980	435312888 #Putin Has Banned #Rothschild And His New World Order Banking Cartel Fami...	• true	en	Twitter for Android	0
6	15017455780045743104	3247724967 #Putin is gonna Destroy The New World Order! 🇷🇺 https://t.co/vvha9YiwX4	• true	en	Twitter for iPhone	0
7	1498341252927946758	1586482726 Would Putin take responsibility for this? 6 year old Ukrainian girl kille...	• true	en	Twitter Web App	138

GiST:

```

192 ✓ EXPLAIN ANALYZE
193 SELECT *
194 FROM conversations
195 WHERE to_tsvector('english', content) @@ to_tsquery('english', 'Putin & New<->World<->Order')
196 AND possibly_sensitive;
197

```

Output | COUNT(*)bigint | plainto_tsquery(url):tsquery | Result 440

7 rows

QUERY PLAN

```

1 Index Scan using conversations_gist_content_index on conversations (cost=0.42..2.64 rows=1 width=221) (actual time=99.852..1380.460 rows=7 loops=1)
2   Index Cond: (to_tsvector('english':regconfig, content) @@ 'putin' & 'new' <-> 'world' <-> 'order')::tsquery)
3   Rows Removed by Index Recheck: 88
4   Filter: possibly_sensitive
5   Rows Removed by Filter: 841
6 Planning Time: 9.063 ms
7 Execution Time: 1380.485 ms

```

GIST výsledky:

```

193 SELECT *
194 FROM conversations
195 WHERE to_tsvector('english', content) @@ to_tsquery('english', 'Putin & New->World->Order')
196 AND possibly_sensitive;
197

```

id	author_id	content	possibly_sensitive	language	source	retweet_count
1	1498341252927946758	1506482726 Would Putin take responsibility for this? 6 year old Ukrainian girl killed...	true	en	Twitter Web App	138
2	15017455708645743104	3247724967 #Putin is gonna Destroy The New World Order! https://t.co/Vvha9VixK4	true	en	Twitter for iPhone	0
3	1498483618804649990	435312888 #Putin Has Banned #Rothschild And His New World Order Banking Cartel Fam!	true	en	Twitter for Android	0
4	1498819842183544838	834033178067234816 RT @veteranstoday: Putin: New World Order Worships Satanhttps://t.co/fDl...	true	en	Twitter for Android	8
5	1497653608028028931	1178085700188004355 #Putin and the New World Order from Damascus to Kiev--#WhiteHouse spokesaa...	true	en	Twitter for Android	3
6	1497395218659745882	47853235 The experience @davidahearst has had in Russia qualifies him for explaini...	true	en	Twitter for Android	0
7	1498219488969437184	736549807901282304 "We fight for the New World Order"--We got it...#UkraineRussiaWar #noAspoc...	true	en	Twitter for iPhone	1

Veľkosť indexov:

```

198 SELECT
199 (SELECT pg_size_pretty(pg_table_size('conversations_gist_content_index')) AS gist_size,
200 (SELECT pg_size_pretty(pg_table_size('conversations_gin_content_index')) AS gin_size;

```

gist_size	gin_size
3148 MB	2281 MB

Podarilo sa mi spraviť GIN aj GIST index nad contentom. Následne som pomocou `ts_vector` a `ts_query` vytvoril dopyt nad dátami. Výsledky boli rovnaké, dostal som všetky riadky vyhovujúce zadaniu. GIST index je väčší a mal by byť aj efektívnejší. Ak necháme na výber databázy (s tým, že existujú oba indexy) vyberie si GIST. Ten je síce pomalší ako GIN, ale za to lepšie pracuje s pamäťou. V prípade, že by som mal HDD disk tak si myslím, že by GIST bol jednoznačne lepší (vďaka práci s pamäťou a bottle neckom disku). Aktuálne ale ostáva GIN tým, kto zbehne rýchlejšie.

GIN spustí sken, vyberie možné hodnoty a tie následne prefiltruje cez senzitivitu a `recheck_condition` na overenie si údajov vo vybratých blokoch - pôvodne vybral 936 hodnôt, ktoré `recheck_condition` a senzitivita zredukovali na finálnych 7. Na rozdiel od neho GIST rovno vyberá správne hodnoty. Vyhľadá všetky riadky, ktoré vyhovujú jeho požiadavke na obsah daných slov a potom ich dodatočne prejde filtrom senzitivity.

15. Linky darujme.sk

```

283 CREATE INDEX links_gin_trigram_url_index ON links
284 USING gin(url gin_trgm_ops);
285 EXPLAIN ANALYZE
286 SELECT * FROM links WHERE url LIKE '%darujme.sk%'
287
Output | a x | COUNT(*)bigint x | plainto_tsquery(url):tsquery x | Result 430 x
7 rows
QUERY PLAN
1 Bitmap Heap Scan on links (cost=67.82..1267.41 rows=1087 width=360) (actual time=10.384..10.391 rows=5 loops=1)
2 Recheck Cond: ((url)::text ~ '%darujme.sk% '::text)
3 Heap Blocks: exact=5
4 -> Bitmap Index Scan on links_gin_trigram_url_index (cost=0.00..67.55 rows=1087 width=0) (actual time=10.356..10.356 rows=5 loops=1)
5 Index Cond: ((url)::text ~ '%darujme.sk% '::text)
6 Planning Time: 0.107 ms
7 Execution Time: 10.410 ms

```

V tejto úlohe som použil trigramy v kombinácii s GIN indexom. Keďže hľadám darujme.sk v linkoch (a linky sú jeden dlhý string), potrebujem použiť LIKE. Na zaindexovanie dopytov používajúcich LIKE je riešením použiť práve spomínané trigramy. Tie umožňujú vyhľadávanie v texte pomocou indexov. Vďaka tomu môžem vyhľadávať v stringu a zároveň použiť indexy na zrýchlenie takéhoto dopytu - trigramy mi pomôžu rozbiť si text a hľadať v ňom.

Výsledok:

```

289 SELECT * FROM links WHERE url LIKE '%darujme.sk%';

```

	id	conversation_id	url	title	description
1	1590896	1497299831383076868	https://charita.darujme.sk/ukrajina/	Pomoc Ukrajine	To, čoho sme sa mes
2	4305004	1498345164833800192	https://clovekvohrozeni.darujme.sk/pomoc-uk...	<null>	<null>
3	9177211	1505620764636196870	https://redcross.darujme.sk/pomahame-ukrajine	Pomáhame Ukrajine	Prioritou Červeného
4	10673188	1497548409334640644	https://redcross.darujme.sk/pomahame-ukraji...	<null>	<null>
5	11036938	1501509972080877570	https://zvieraciombudsman.darujme.sk/animal...	<null>	<null>

16. Query pomocou FTS

Ako prvé som si vytvoril GIN indexy nad stĺpcami, ktoré budem používať. Vybral som si GIN, nakoľko som spätne skúšal aj GIST a ako rýchlejší mi vyšiel GIN (pri GISTE okolo 3 sekúnd, pri GINe okolo 500ms).

V postgrese NIE JE možné vytvoriť index nad viacerými tabuľkami súčasne. Bolo by možné vytvoriť jeden index nad stĺpcami tabuľky autorov, ale, podobne ako GIST, ani to nezvýšilo efektívnosť.

```
CREATE INDEX authors_description_gin_index ON authors
  USING gin(to_tsvector('english', description));
CREATE INDEX authors_username_gin_index ON authors
  USING gin(to_tsvector('english', username));
CREATE INDEX conversations_content_index ON conversations
  USING gin(to_tsvector('english', content));
```

Následne som vyskúšal niekoľko prístupov k získaniu sediacich dát. Nasledujúca query vráti 7923 riadkov dát, ktoré verím, že sú správne:

```
SELECT content.content, username.username, description.description, content.retweet_count
FROM
  (SELECT id, author_id, content, retweet_count
   FROM conversations
   WHERE (to_tsvector('english', content)
          @@ to_tsquery('english', 'Володимир | Президент')) AS content
  )
FULL OUTER JOIN (SELECT id, username
                  FROM authors
                  WHERE (to_tsvector('english', username)
                         @@ to_tsquery('english', 'Володимир | Президент'))
                  AS username
  ON username.id = content.author_id
)
FULL OUTER JOIN (SELECT id, description
                  FROM authors
                  WHERE to_tsvector('english', description)
                         @@ to_tsquery('english', 'Володимир | Президент'))
  AS description
  ON description.id = content.author_id
ORDER BY retweet_count DESC NULLS LAST;
```

V tejto query vykonám výber vhodných hodnôt z každého stĺpca samostatne - pomocou tsvectorov a tsqueries operácií nad jednotlivými tabuľkami. Samostatné vyberanie vo vnorenej query prebehne za pomoci indexu a následne môžem takto vybrané hodnoty spojiť pomocou JOINov a ANDov medzi nimi. Na koniec zoradím hodnoty podľa retweet countov. Výsledné dáta prikladám aj v csv súbore - *z02_16_schon.csv*.

Prvých pár výsledkov query:

	content	username	description	retweet_count
1	✖ Володимир Кличко звернувся до демократичного світу із закликом нег...	<null>	<null>	7351
2	Президент України Володимир Зеленський відвідав у госпіталі поранених...	<null>	<null>	5848
3	🗣️Президент України Володимир Зеленський:«Всі ми тут - захищаємо наш...	<null>	<null>	5346
4	В бою загинув льотчик-винищувач Олександр Оксанченко. «Він був одним ...	<null>	<null>	2799
5	✖ Володимир Кличко звернувся до громадян Німеччини, Австрії, Швейцарії...	<null>	<null>	2628
6	На митинг в Санкт-Петербурзі вийшла 84-літня Ірина Анохіна, яка п...	<null>	<null>	2569
7	Володимир Ковальський.«У 2016-му році під час виконання бойового завд...	<null>	<null>	2408
8	Президент України @ZelenskyUa поспілкувався зі звільненим із полону ...	<null>	<null>	2200
9	Итого на сегодня в России:«- уничтожены почти все независимые СМИ»- в...	<null>	<null>	2118
10	“Президент Зеленський оголосив про формування нового підрозділу - «Ін...	<null>	<null>	2054
11	This is a rhetorical question. I know Putin is in his Ural bunker. Th...	<null>	<null>	1676
12	☛ «Ми вдома і захищаємо Україну. Ніякої зброї ми складати не збираєм...	<null>	<null>	1580
13	Лукашенко, після розмови із Зеленським, жаліється, що "президент Укра...	<null>	<null>	1539
14	Президент Швейцарії заявив, що Швейцарія більше не нейтральна стран...	<null>	<null>	1530
15	Головнокомандувачу ЗС України генерал-лейтенанту Валерію Залужному пр...	<null>	<null>	1489
16	Повертаючись до України зі США, у Варшаві мене прийняв Президент Поль...	<null>	<null>	1446
17	✎Сергій Васіч обороняє позиції на Київщині. «Він знищив ворожу техні...	<null>	<null>	1400
18	Президент Молдови потребувала от РФ немедленно вывести войска из терр...	<null>	<null>	1341
19	Президент України @ZelenskyUa :«Торік я попереджав лідерів @NATO: як...	<null>	<null>	1306
20	"Я поранений, залишилося недовго!" Заступник мера Нової Каховки встиг...	<null>	<null>	1253
21	Во время Второй мировой войны четыре брата из одной еврейской семьи з...	<null>	<null>	1246
22	!!! Майже 9000 вбитих 🇷🇺 окупантів за один тиждень війни!«“Кожен ок...	<null>	<null>	1217
23	Пора уже всем россиянам с мозгами перестать быть «вне политики». ««Не...	<null>	<null>	1212
24	Володимир Андрійченко поїхав забирати поранених та вбитих побратимів ...	<null>	<null>	1134
25	!!!! Президент України Володимир Зеленський підписав заяву на чле...	<null>	<null>	1109
26	Кстати, а теперь ведь не кажется конспирологией, что президент Качиньс...	<null>	<null>	1092

Pomocou EXPLAIN ANALYZE som si overil aj použitie indexov / skenov. Ako je možné vidieť, použili sa indexy, joiny prebehli pomocou hash full joinu. Nebolo potrebné vykonať žiadny sekvenčný sken nad dátami z dôvodov uvedených vyššie.

```

QUERY PLAN
1  Sort (cost=1422838.36..1423645.01 rows=322661 width=248) (actual time=517.331..518.812 rows=7923 loops=1)
2    Sort Key: conversations.retweet_count DESC
3    Sort Method: quicksort  Memory: 3810kB
4    -> Hash Join (cost=125555.95..1371351.72 rows=322661 width=248) (actual time=281.309..514.271 rows=7923 loops=1)
5      Hash Cond: (conversations.author_id = authors_1.id)
6      -> Hash Full Join (cost=64198.34..1291957.13 rows=322661 width=182) (actual time=52.532..273.598 rows=7777 loops=1)
7        Hash Cond: (conversations.author_id = authors.id)
8        -> Gather (cost=3644.73..1230556.53 rows=322661 width=171) (actual time=28.244..230.878 rows=7777 loops=1)
9          Workers Planned: 8
10         Workers Launched: 8
11         -> Parallel Bitmap Heap Scan on conversations (cost=2644.73..1197290.43 rows=40333 width=171) (actual time=165.640..169.031 rows=864 loops=9)
12           Recheck Cond: (to_tsvector('english':regconfig, content) @@ 'волондмр' | 'президент':tsquery)
13           Heap Blocks: exact=7639
14           -> Bitmap Index Scan on conversations_gin_content_index (cost=0.00..2564.06 rows=322661 width=0) (actual time=26.161..26.161 rows=7777 loops=1)
15             Index Cond: (to_tsvector('english':regconfig, content) @@ 'волондмр' | 'президент':tsquery)
16         -> Hash (cost=59818.56..59818.56 rows=58804 width=19) (actual time=24.256..24.362 rows=0 loops=1)
17           Buckets: 65536 Batches: 1 Memory Usage: 512kB
18           -> Gather (cost=1490.93..59818.56 rows=58804 width=19) (actual time=24.255..24.360 rows=0 loops=1)
19             Workers Planned: 4
20             Workers Launched: 4
21             -> Parallel Bitmap Heap Scan on authors (cost=490.93..52938.16 rows=14701 width=19) (actual time=0.058..0.058 rows=0 loops=5)
22               Recheck Cond: (to_tsvector('english':regconfig, (username)::text) @@ 'волондмр' | 'президент':tsquery)
23               -> Bitmap Index Scan on authors_username_gin_index (cost=0.00..476.23 rows=58804 width=0) (actual time=0.013..0.014 rows=0 loops=1)
24                 Index Cond: (to_tsvector('english':regconfig, (username)::text) @@ 'волондмр' | 'президент':tsquery)
25         -> Hash (cost=59818.56..59818.56 rows=58804 width=82) (actual time=228.633..228.746 rows=155 loops=1)
26           Buckets: 65536 Batches: 2 Memory Usage: 529kB
27           -> Gather (cost=1490.93..59818.56 rows=58804 width=82) (actual time=228.457..228.591 rows=155 loops=1)
28             Workers Planned: 4
29             Workers Launched: 4
30             -> Parallel Bitmap Heap Scan on authors authors_1 (cost=490.93..52938.16 rows=14701 width=82) (actual time=177.939..178.089 rows=31 loops=5)
31               Recheck Cond: (to_tsvector('english':regconfig, description) @@ 'волондмр' | 'президент':tsquery)
32               Heap Blocks: exact=1
33               -> Bitmap Index Scan on authors_description_gin_index (cost=0.00..476.23 rows=58804 width=0) (actual time=0.270..0.270 rows=155 loops=1)
34                 Index Cond: (to_tsvector('english':regconfig, description) @@ 'волондмр' | 'президент':tsquery)
35 Planning Time: 0.259 ms

```

```
35 Planning Time: 0.259 ms
36 JIT:
37   Functions: 88
38   Options: Inlining true, Optimization true, Expressions true, Deforming true
39   Timing: Generation 8.510 ms, Inlining 1384.396 ms, Optimization 619.058 ms, Emission 305.446 ms, Total 2317.410 ms
40 Execution Time: 520.310 ms
```

Na zopakovanie dôvodov uvedených v texte: OR podmienka nie je možná, pretože sa nepoužívajú indexy a joiny sú možné, pretože výber dát prebehol pred joinovaním.

Záver

Celé pozorovanie bolo uskutočnené v PostgreSQL verzii 14. SQL súbor s dopytmi, rovnako ako aj spomínané dátové súbory sa nachádzajú priložené pri tomto dokumente, prípadne sú zverejnené na githube spomenutom na prvej strane tohto dokumentu. Zadanie som vypracoval samostatne, myšlienky sú moje vlastné. Verím, že sa vám dobre čítalo a všetko bolo zrozumiteľné 😊.