Zadanie II.

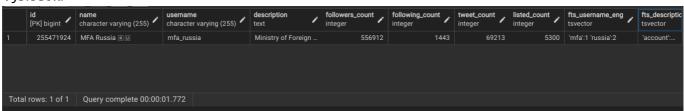
Vypracoval: Štefan Hajdú GitHub: https://github.com/StefanHajdu/PDT-22/tree/master/Assignment_2 Úloha 1: Query: select * from authors where username = 'mfa_russia';

Explain:

	QUERY PLAN text			
1	Gather (cost=1000.00147613.15 rows=1 width=126)			
2	Workers Planned: 2			
3	-> Parallel Seq Scan on authors (cost=0.00146613.05 rows=1 width=126)			
4	Filter: ((username)::text = 'mfa_russia'::text)			
5	JIT:			
6	Functions: 2			
7	Options: Inlining false, Optimization false, Expressions true, Deforming true			

Plánovač vybral paralelný sekvenčný scan. Dôvodom je to, že úloha je svojou charakteristikou paralelizovateľná, pretože môžeme pole autorov rozdeliť do menších častí, ktoré sa prehľadajú samostatne. Tiež máme v konfiguráku nastavený počet workerov na 2, čiže Postres má dovolené spawnovať workerov, ak potrebuje.

Výsledok:



Úloha 2:

Na selecte pracovali 2 workery (podľa hodnoty nastavenej v konfiguráku). Ich úlohou je prehľadať rôzne časti tabuľky (teraz sa tabuľka rozdelí na 2 nezávislé časti, každý worker prehľadá jednu)

	QUERY PLAN text	â
1	Seq Scan on authors (cost=0.00189598.73 rows=1 width=126) (actual time=36.978477.194 rows=1 loops=1)	
2	Filter: ((username)::text = 'mfa_russia'::text)	
3	Rows Removed by Filter: 5895179	
4	Planning Time: 0.050 ms	
5	JIT:	
6	Functions: 2	
7	Options: Inlining false, Optimization false, Expressions true, Deforming true	
8	Timing: Generation 0.201 ms, Inlining 0.000 ms, Optimization 0.132 ms, Emission 1.893 ms, Total 2.226 ms	
9	Execution Time: 477.438 ms	

Sekvenčný scan

	QUERY PLAN text
1	Gather (cost=1000.00160256.00 rows=1 width=126) (actual time=32.590277.121 rows=1 loops=1)
2	Workers Planned: 1
3	Workers Launched: 1
4	-> Parallel Seq Scan on authors (cost=0.00159255.90 rows=1 width=126) (actual time=138.306254.3
5	Filter: ((username)::text = 'mfa_russia'::text)
6	Rows Removed by Filter: 2947590
7	Planning Time: 0.056 ms
8	JIT:
9	Functions: 4
10	Options: Inlining false, Optimization false, Expressions true, Deforming true
11	Timing: Generation 0.596 ms, Inlining 0.000 ms, Optimization 0.385 ms, Emission 5.453 ms, Total 6.434
12	Execution Time: 277.372 ms

1 worker

	QUERY PLAN text
1	Gather (cost=1000.00147613.15 rows=1 width=126) (actual time=26.775189.564 rows=1 loops=1)
2	Workers Planned: 2
3	Workers Launched: 2
4	-> Parallel Seq Scan on authors (cost=0.00146613.05 rows=1 width=126) (actual time=119.114170.532 rows=0 loops
5	Filter: ((username)::text = 'mfa_russia'::text)
6	Rows Removed by Filter: 1965060
7	Planning Time: 0.064 ms
8	JIT:
9	Functions: 6
10	Options: Inlining false, Optimization false, Expressions true, Deforming true
11	Timing: Generation 0.661 ms, Inlining 0.000 ms, Optimization 0.511 ms, Emission 7.888 ms, Total 9.060 ms
12	Execution Time: 189.839 ms

2 workers

	QUERY PLAN text
1	Gather (cost=1000.00140679.98 rows=1 width=126) (actual time=153.173160.544 rows=1 loops=1)
2	Workers Planned: 3
3	Workers Launched: 3
4	-> Parallel Seq Scan on authors (cost=0.00139679.88 rows=1 width=126) (actual time=106.711138.666 rows=0 loops
5	Filter: ((username)::text = 'mfa_russia'::text)
6	Rows Removed by Filter: 1473795
7	Planning Time: 0.060 ms
8	JIT:
9	Functions: 8
10	Options: Inlining false, Optimization false, Expressions true, Deforming true
11	Timing: Generation 1.025 ms, Inlining 0.000 ms, Optimization 0.648 ms, Emission 10.756 ms, Total 12.429 ms
12	Execution Time: 160.783 ms

3 workers

	QUERY PLAN text
1	Gather (cost=1000.00135331.53 rows=1 width=126) (actual time=186.663194.670 rows=1 loops=1)
2	Workers Planned: 4
3	Workers Launched: 3
4	-> Parallel Seq Scan on authors (cost=0.00134331.43 rows=1 width=126) (actual time=130.894168.586 rows=0 loops
5	Filter: ((username)::text = 'mfa_russia'::text)
6	Rows Removed by Filter: 1473795
7	Planning Time: 0.055 ms
8	JIT:
9	Functions: 8
10	Options: Inlining false, Optimization false, Expressions true, Deforming true
11	Timing: Generation 1.342 ms, Inlining 0.000 ms, Optimization 0.808 ms, Emission 13.422 ms, Total 15.572 ms
12	Execution Time: 194.915 ms

4 workers

	QUERY PLAN text	â
1	Gather (cost=1000.00135331.53 rows=1 width=126) (actual time=179.117187.286 rows=1 loops=1)	
2	Workers Planned: 4	
3	Workers Launched: 3	
4	-> Parallel Seq Scan on authors (cost=0.00134331.43 rows=1 width=126) (actual time=126.654163.921 rows=0 loops=4)	
5	Filter: ((username)::text = 'mfa_russia'::text)	
6	Rows Removed by Filter: 1473795	
7	Planning Time: 0.054 ms	
8	JIT:	
9	Functions: 8	
10	Options: Inlining false, Optimization false, Expressions true, Deforming true	
11	Timing: Generation 1.027 ms, Inlining 0.000 ms, Optimization 0.664 ms, Emission 11.020 ms, Total 12.711 ms	
12	Execution Time: 187.488 ms	

6 workers

Čas na vykonanie selectu klesal do momentu kým sa nám neminuli voľné CPU (máme 4). Od nastavenia počtu workerov na 4, sa nám nezmenil počet spustených workerov. Počet workerov sme nastavovali pomocou:

```
set max_parallel_workers_per_gather to desired_number;
```

Ďaľším limitom je veľkosť tabuľky, Postgres má definové koľko workerov vie vytvoriť pre danú veľkosť tabuľky. Pre našu tabuľku autorov o veľkosti cca. 1GB je maximálny počet workerov 5 (https://www.2ndquadrant.com/en/blog/postgresql96-parallel-sequential-scan/), ale toľko CPU už nemáme.

Úloha 3:

Query:

```
create index idx_authors_username on authors using BTREE (username);
select * from authors where username = 'mfa_russia';
```

Explain Analyze:

Nebolo použitých viac workerov. Zrýlechenie vyplýva z toho, že vytvorením indexu sa zmenila dátová štruktúra, v ktorej sa vyhľadáva. Teraz sa používa stromová štruktúra BTREE, ktorá má logaritmickú

zložitosť, na rozdiel od sekvenčného scanu, ktorý je lineárny.



3 workers



BTREE index

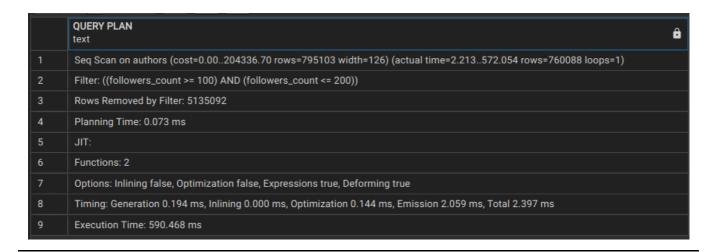
Použitím indexu sme vylepšili čas približne o polovicu v porovnaní s sekvenčným scanom s 3 workermi.

Úloha 4:

Query:

```
select * from authors where followers_count between 100 and 200; select * from authors where followers_count between 100 and 120;
```

Explain Analyze:



between 100 and 200

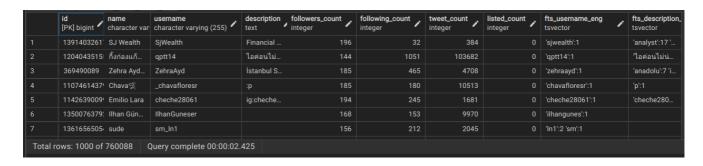
	QUERY PLAN text
1	Seq Scan on authors (cost=0.00204336.70 rows=795103 width=126) (actual time=2.213572.054 rows=760088 loops=1)
2	Filter: ((followers_count >= 100) AND (followers_count <= 200))
3	Rows Removed by Filter: 5135092
4	Planning Time: 0.073 ms
5	JIT:
6	Functions: 2
7	Options: Inlining false, Optimization false, Expressions true, Deforming true
8	Timing: Generation 0.194 ms, Inlining 0.000 ms, Optimization 0.144 ms, Emission 2.059 ms, Total 2.397 ms
9	Execution Time: 590.468 ms

	QUERY PLAN text
1	Gather (cost=1000.00166605.66 rows=211716 width=126) (actual time=3.565201.193 rows=199937 loops=1)
2	Workers Planned: 3
3	Workers Launched: 3
4	-> Parallel Seq Scan on authors (cost=0.00144434.06 rows=68295 width=126) (actual time=2.983174.706 rows=499
5	Filter: ((followers_count >= 100) AND (followers_count <= 120))
6	Rows Removed by Filter: 1423811
7	Planning Time: 0.090 ms
8	JIT:
9	Functions: 8
10	Options: Inlining false, Optimization false, Expressions true, Deforming true
11	Timing: Generation 1.196 ms, Inlining 0.000 ms, Optimization 0.630 ms, Emission 11.228 ms, Total 13.054 ms
12	Execution Time: 208.104 ms

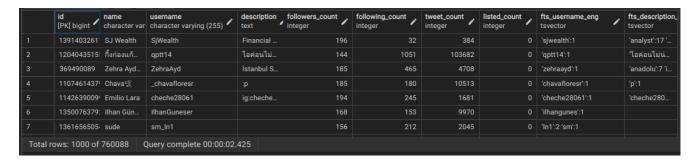
between 100 and 120

Ako vidíme rozdiel je v tom, že ak hľadáme vačší interval, tak sa plánovač uprednostní obyčajný sekvenčný namiesto paralelného. Toto môže byť zapríčinené tým, že ako sa zvyšuje interval, tým sa zvyšuje aj cena gather operácie (zozbieranie výsledkov od workeror do master procesu). Pretože paralelizácia nie je len o tom, že viac workerov => menší čas. Pri paralelizácií dochádza aj k rozdeleniu úlohy medzi workerov a komunikácie výsledkov do master procesu, ktoré tiež vyžadujú čas a výkon.

Výsledok:



between 100 and 200



	id [PK] bigint	name character var	username character var	description text	followers_count integer	following_count integer	tweet_count integer	listed_count integer	fts_username_eng tsvector	fts_descriptic tsvector
1	1420186765	Тимофеич	BFJ4KmS		112	596	8257		'bfj4kmsbiffiqle':1	
2	1167033716	Anichu	Anichu_u	19. Auron	102	183	3498		'anichu':1 'u':2	'19':1 '201
3	816767174	Æ Able	aeAble05	The limits	120	126	4265		'aeable05':1	'endur':9 'l
4	55795174	mrafieq	mrafieq		105	177	241		'mrafieq':1	
5	1493830171	hasan zor	hzorlu83		102	449	5859		'hzorlu83':1	
6	1106117393	888	Muskaan	Free-spirit	113	365	9995		'muskaanjain27':1	'free':2 'fr
7	1324025570	Nielen	niele_n	Creating	111	248	815		'n':2 'niel':1	'creat':1 's
Total r	Total rows: 1000 of 199937									

between 100 and 120

Úloha 5:

Query:

```
select * from authors where followers_count between 100 and 200; select * from authors where followers_count between 100 and 120;
```

Index:

```
create index idx_follow_interval on authors using BTREE (followers_count)
where (followers_count >= 100) and (followers_count <= 200);</pre>
```

Explain Analyze:

```
DUERY PLAN
text

Bitmap Heap Scan on authors (cost=4715.74..132551.29 rows=795103 width=126) (actual time=76.917..236.513 rows=760088 loops=1)

Recheck Cond: ((followers_count >= 100) AND (followers_count <= 200))

Heap Blocks: exact=115383

-> Bitmap Index Scan on idx_follow_interval (cost=0.00..4516.97 rows=795103 width=0) (actual time=54.997..54.997 rows=760088 loops=1)

Planning Time: 0.183 ms

JIT:

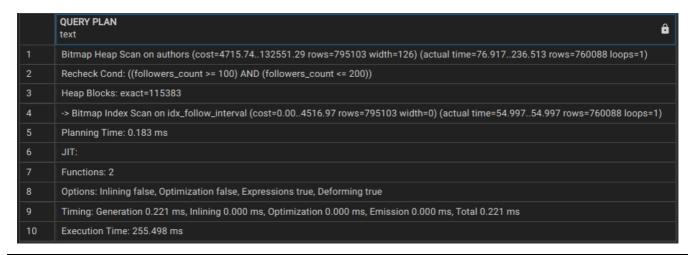
Functions: 2

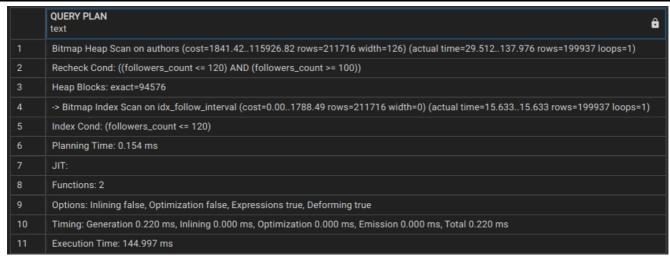
Options: Inlining false, Optimization false, Expressions true, Deforming true

Timing: Generation 0.221 ms, Inlining 0.000 ms, Optimization 0.000 ms, Emission 0.000 ms, Total 0.221 ms

Execution Time: 255.498 ms
```

between 100 and 200





between 100 and 120

Bitmap scany majú zmysel, ak je výstup príliš malý na sekvenčný, ale príliš veľký na index scan. Pretože oproti veľkému index scanu zmenšuje počet I/O operácií.

- Teda najprv sa prebehne celý index a zapamätá si (pomocou bitmapy) na akej stránke je hľadaný riadok uložený. Toto robí **Bitmap Index Scan**
- Potom pomocou vytorenej bitmapy vie, ktoré stránky obsahujú hľadané riadky a sekvenčne tieto stránky prehľadá. Toto robí Bitmap Heap Scan
- Recheck Condition je potrebný, aby sa dali lokalizovať hľadané riadky pri prehľadávaní stránky

Úloha 6:

Query:

```
create index idx_follow_interval on authors using BTREE (followers_count)
where (followers_count >= 100) and (followers_count <= 200);
create index idx_authors_name on authors using BTREE (name);
create index idx_authors_follow_cnt on authors using BTREE
(followers_count);
create index idx_authors_desc on authors using BTREE (description);
insert into authors values (456168618, 'StefanHajdu', 'stevexo', 'james bond fan', 1212, 1516, 22, 565);</pre>
```

```
drop index idx_follow_interval;
drop index idx_authors_name;
drop index idx_authors_follow_cnt;
drop index idx_authors_desc;
insert into authors values (400008618, 'StefanHajdu', 'stevexo', 'james bond fan', 1212, 1516, 22, 565);
```

Porovnanie času:

```
INSERT 0 1
Query returned successfully in 152 msec.
```

insert do tabuľky so 4 indexami

```
INSERT 0 1
Query returned successfully in 110 msec.
```

insert do tabuľky bez indexov

Očakávali sme, že insert do tabuľky s indexami bude o niekoľko rádov pomalší, kvôli aktualizácií indexov. Keďže teraz musí byť miesto vloženia určené indexom, nie je možné vložiť záznam na prvé voľné miesto. Ale nakoniec trvajú približne rovnako, niekedy dokonca rýchlejšie insertneme do tabuľky s indexami.

Úloha 7:

Index:

```
create index idx_conv_content on conversations using BTREE (content);
create index idx_conv_retweet on conversations using BTREE (retweet_count);
```

Porovnanie času:

```
Query returned successfully in 24 secs 165 msec.

Total rows: 11 of 11 Query complete 00:00:24.165
```

vytvorenie indexu pre retweet_count

```
Query returned successfully in 24 secs 165 msec.

Total rows: 11 of 11 Query complete 00:00:24.165
```

```
Query returned successfully in 2 min 28 secs.

Total rows: 11 of 11 Query complete 00:02:28.433
```

vytvorenie indexu pre content

Hlavným faktorom ovplyňujúcim rýchlosť vytvárania indexu je počet záznamov. V našom prípade je, ale počet záznamov rovnaký.

Pri vytváraní BTREE indexu dochádza k porovnaniu hodnôt. Dlhý text ako content sa porovnáva pomalšie ako číselné hodnoty, lebo text je nutné porovnať alfa-numericky, pričom sa prechádza po znakoch kým sa nenájde rozdiel.

Textové reťazce sú väčšie ako obyčajné integery do počtu bytov, tak aj index vytvorený nad nimi je väčší, teda je potrebných viac blokov v indexe, čo navyšuje aj počet vykonaných I/O operácií.

Úloha 8

Query:

```
create extension pgstattuple;
create extension pageinspect;

select tree_level, root_block_no, index_size from
pgstatindex('idx_conv_content');
select tree_level, root_block_no, index_size from
pgstatindex('idx_conv_retweet_cnt');
select tree_level, root_block_no, index_size from
pgstatindex('idx_authors_name');
select tree_level, root_block_no, index_size from
pgstatindex('idx_authors_follow_cnt');

select avg_item_size, page_size from bt_page_stats('idx_conv_content',
1000);
select avg_item_size, page_size from bt_page_stats('idx_conv_retweet_cnt',
1000);
select avg_item_size, page_size from bt_page_stats('idx_authors_name',
```

```
1000);
select avg_item_size, page_size from
bt_page_stats('idx_authors_follow_cnt', 1000);
```

	tree_level integer	root_block_no bigint	index_size bigint
1	5	184604	2533449728

	avg_item_size integer	page_size integer
1	199	8192

idx_conv_content

	tree_level integer	root_block_no abigint	index_size bigint
1	2	209	225804288

	avg_item_size integer	page_size integer
1	729	8192

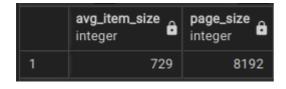
idx_conv_retweet



	avg_item_size integer	page_size integer
1	33	8192

idx_authors_name





idx_authors_follow_cnt

Ukázalo sa, že indexy nad textom vytvoria mohutnejší strom s väčším množstvom root uzlov aj vačšou hĺbkou. Čo dáva zmysel, keďže text je zložitejší typ na porovnanie ako obyčaný integer. Pre priemernú veľkosť itemu platilo, že v zložitejšom strome bola menšia ako v jednoduchšom. Pretože veľkosť indexu je rozdelená do viacerých uzlov.

Úloha 9

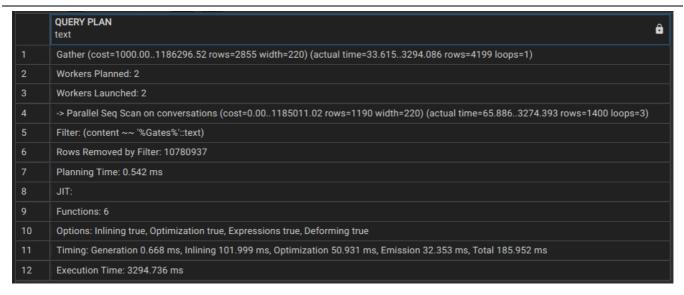
Query:

```
create index idx_conv_content on conversations using BTREE (content);
select * from conversations where content like '%Gates%';
```

Explain Analyze:

	QUERY PLAN text
1	Gather (cost=1000.001186296.52 rows=2855 width=220) (actual time=30.4863220.597 rows=4199 loops=1)
2	Workers Planned: 2
3	Workers Launched: 2
4	-> Parallel Seq Scan on conversations (cost=0.001185011.02 rows=1190 width=220) (actual time=59.2133205.582 rows=1400 loops
5	Filter: (content ~~ '%Gates%'::text)
6	Rows Removed by Filter: 10780937
7	Planning Time: 0.577 ms
8	JIT:
9	Functions: 6
10	Options: Inlining true, Optimization true, Expressions true, Deforming true
11	Timing: Generation 0.562 ms, Inlining 98.401 ms, Optimization 49.080 ms, Emission 25.432 ms, Total 173.476 ms
12	Execution Time: 3221.155 ms

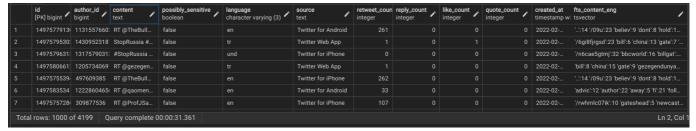
Hľadaj 'Gates' bez indexu



Hľadaj 'Gates' s BTREE indexom

Rozdiel v plánoch nie je žiaden, plánovač vyberie paralelný scan aj keď je vytvorený index. Čo dáva zmysel, kedže BTREE index nevie vyhľadávať podreťazce, iba prefixy, lebo text usporiada alfa-numericky.

Výsledok:



Úloha 10

Query:

select * from conversations where content like 'There are no excuses%' and possibly_sensitive=true;

Explain Analyze:



Vytvorený BTREE index sa nepoužil. Pretože plánovač zvažuje BTREE index, keď sa má vykonať porovnanie (=, <, >, <=, =>). Nie keď chceme vyhľadávať podreťazce pomocou LIKE.

Výsledok:



Ak bolo query nasledovné:

select * from conversations where content='There are no excuses' and
possibly_sensitive=true;

Explain Analyze:



Potom by sa použil index scan, ale takto sa text vyhľadávať nedá.

Zefektívniť vyhľadávanie môžeme vytvorením GIN indexu nad typom trigram. Hoci toto by bol trochu overkill.

Alebo vytvoriť BTREE nad inou sadou operátorov (varchar_pattern_ops), čím sa umožní vyhľadávanie prefixov pomocou LIKE:

```
create index idx_content_prefix ON conversations using BTREE (content
varchar_pattern_ops);
```

	QUERY PLAN text	â
1	Index Scan using test_index on conversations (cost=0.813.04 rows=31 width=252)	
2	Index Cond: ((content ~>=~ 'There are no excuses'::text) AND (content ~<~ 'There are no excuset'::text))	
3	Filter: (possibly_sensitive AND (content ~~ 'There are no excuses%'::text))	

Úloha 11

Query:

```
create index idx_conv_content_url on conversations using BTREE
(lower(right(content, length('https://t.co/pkFwLXZlEm'))));
select * from conversations where
lower(right(content, length('https://t.co/pkFwLXZlEm'))) =
lower('https://t.co/pkFwLXZlEm');
```

Explain Analyze:



BTREE sa aktivuje ak hľadáme podľa porovnania, preto zaindexujeme posledných N znakov (N je dĺžka reťazca 'https://t.co/pkFwLXZlEm'). Ak chceme aby výsledok nebol závislý od veľkosti znakov vstupu prevedieme indexovaný text na lowercase. Následne budeme podľa rovnakej podmienky aj vyhľadávať, teda hľadáme také záznamy, ktoré majú posledných N znakov **rovných** 'https://t.co/pkfwlxzlem'.

Použité funkcie:

- lower(): zmena vstupu na lowercase, podmienka nezávislosti vstupu
- right(): extrahovanie posledných N znakov z textu
- length(): vráti dĺžku reťazca

Výsledok:

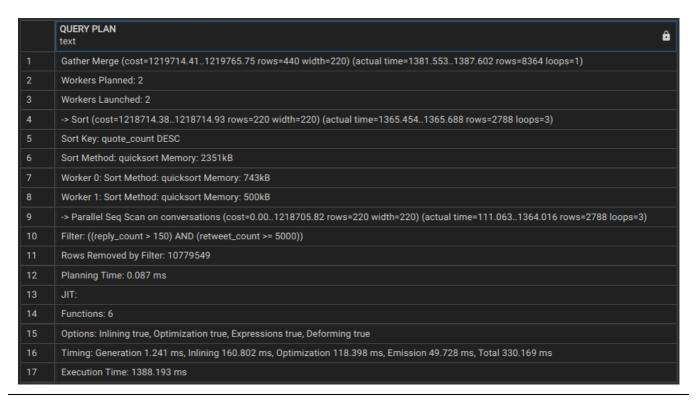


Úloha 12

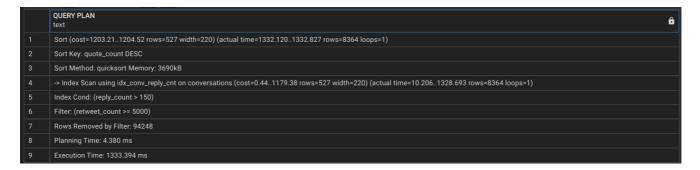
Query:

```
create index idx_conv_reply_cnt on conversations using BTREE (reply_count);
create index idx_conv_retweet_cnt on conversations using BTREE
(retweet_count);
create index idx_conv_quote_cnt on conversations using BTREE (quote_count);
select * from conversations where reply_count > 150 and retweet_count >=
5000
order by quote_count desc;
```

Explain Analyze:



Bez indexov



	id [PK] bigint	author_id bigint	content text	possibly_sensitive boolean	language character varying (3)	source text	retweet_count integer	reply_count integer	like_count integer	quote_count integer	created_at timestamp w	fts_content tsvector
	10815446307541	137857547	ZOZOTOWN新春	false		Twitter for iPhone	3749726	324127	1184675	270748	2019-01	'/ckqfppb
	13251267334823	30354991	We did it, @JoeBi	false	en	Twitter Media Studio	532141	97559	3158150	175888	2020-11	'/ocgeylsj.
	13219419861742	116362700	Just had a great	false	en	Twitter for iPhone	88105	62888	392446	143977	2020-10	'/q9c5k1y.
4	14967161689205	7325210585	https://t.co/laqF	false	ZXX	Twitter for iPhone	295836	28046	1798353	117743	2022-02	'/iaqfbpay
5	15066622308923	181561712	Harry's House. M	false	en	Twitter for iPhone	302581	52047	1156788	117176	2022-03	'/gvnjscc
6	15075529589882	19081001	https://t.co/ffPH	false	ZXX	Twitter Web App	153729	28634	1002220	117102	2022-03	'/ffphhukr
7	14968775417720	105574103	Dear Mister Presi	false	en	Twitter for iPhone	19445	38659	93222	113224	2022-02	'/lbdfbhv
8	15031471417950	1108050829	These past two	false	en	Twitter for iPhone	132964	25925	621599	93141	2022-03	'/u0yhrkv.
9	10815980513226	7431997145	O jewa ke eng ?	false		Twitter for iPhone	36878	67112	108601	88148	2019-01	'eng':4 'je.
10	15016315127432	1445650784	We're giving awa	false	en	Twitter Web App	31755	46807	47944	84261	2022-03	'50':5 '500
11	14598996869454	17919972	SURPRISE! NEW	false	en	Twitter for iPhone	97003	25631	506712	80616	2021-11	'/2qh7kev
12	14965112157193	1115826452	Greg Abbot has o	false	en	Twitter Web App	49916	7636	73362	79136	2022-02	'/ao4fdyn
13	14919793667245	1430086309	Bull100X token in	false	en	Twitter Web App	209259	178706	222850	71923	2022-02	'/9kqik7bi
14	15018288912451	335141638	D-day!! \n#PTD	false	en	Twitter for Android	617440	62204	2097110	59955	2022-03	'/flozxss3
15	14955567773999	1494275085		false	en	Twitter Web App	156676	110718	161846	59898	2022-02	'/m03d3h
16	14857142656758	17919972	@DamonAlbarn I	false	en	Twitter for iPhone	112853	22185	741492	57078	2022-01	'/t6gyxbu
17	14762193653812	335141638	생일 축하한다 태형	false	ko	Twitter for Android	602177	61190	2093583	55591	2021-12	'격리중이
18	14930001782691	7507512064:	Enter a new dime	false	en	Twitter Media Studio	115475	4398	344442	54587	2022-02	'/fizgdqto
19	15044268441996	12044602	I love the Russian	true	en	Twitter for iPhone	425291	44540	1323423	52660	2022-03	'/6gyvrhg

S indexami

Ukázalo sa, že sa používa jediný index scan, a to nad stĺpcom reply_count. Ktorým sa získajú záznamy s reply_count > 150. Ďaľšia časť podmienky už nie je riešená cez index, ale cez klasický filter. Pravdepodobne prvý index scan zredukuje množinu až tak, že ďaľší index scan nie je nutný. Pre toto konkrétne query nie je potrebné držať indexy pre retweet_count a quote_count.

Úloha 13

Query:

```
create index idx_conv_metrics on conversations using btree (reply_count,
retweet_count, quote_count);
select * from conversations where reply_count > 150 and retweet_count >=
5000
order by quote_count desc;
```

Explain Analyze:



	id [PK] bigint	author_id bigint	content text
ı	1081544630754103296	137857547	ZOZOTOWN新春セールが史上最速で取扱高100億円を先ほど突破!!日頃の感謝を込め、僕個人から100名様に100万円【総額1億円のお年玉】を現金でプレゼントします。応募方法は、僕を
	1325126733482385409	30354991	We did it, @JoeBiden. https://t.co/oCgeylsjB4
3	1321941986174226432	116362700	Just had a great meeting with @realdonaldtrump @potus besides what he's done so far with criminal reform, the platinum plan is going to give the community real ownership. He listened to
	1496716168920547331	732521058507620356	https://t.co/laqFbpayqz
	1506662230892314634	181561712	Harry's House. May 20th. https://t.co/gvNJsccE25
	1507552958988255234	19081001	https://t.co/ffPHhUKRT4
	1496877541772062727	105574103	Dear Mister President Vladimir Putin https://t.co/LbDFBHVWJf
	1503147141795045378	1108050829393707008	These past two months I've realized my place is still on the field and not in the stands. That time will come. But it's not now. I love my teammates, and I love my supportive family. They mak
	1081598051322679297	743199714502402048	0 jewa ke eng ?
0	1501631512743231489	1445650784	We're giving away \$50 to 500 of our followers to help at the pump. RT with your \$cashtag for a chance to win #GasCash.
	1459899686945447936	17919972	SURPRISE! NEW MUSIC VIDEO TOMORROW at 10am ET 🐒 I finally got to work with the brilliant, brave, & amp; wickedly funny @blakelively on her directorial debut. Join us as we raise a toas
2	1496511215719399431	1115826452715511809	Greg Abbot has officially directed Family and Protective Services to begin investigating all trans children in Texas and prosecuting their parents as child abusers.\n\nHe has also instructed
	1491979366724554757	1430086309557669891	Bull100X token info:\n\nContract address: 0x72aDD749DdCFE9b749f50CBFf76B59A2e6c80Af5\nName: BULL Token\nSymbol: BULL\nDecimals: 18\nPancakeswap add LP time: April 13, 2
4	1501828891245105155	335141638	D-day!! \n#PTD_ON_STAGE_SEOUL\n#우리함께라면_허락은필요없어 https://t.co/FLozxSs3dA
5	1495556777399955456	1494275085632344069	#OFFICIAL PRESALE OPENING ★\n\n⊙BUY NOW: https://t.co/VTgZnZX51c\n\n∳ Contract: 0xE833dC9b5A21daBC3ed8559858Caa92279aC54Dc \n≣Start Time: Feb 21st, 2022\n⊙End
6	1485714265675812866	17919972	@DamonAlbarn I was such a big fan of yours until I saw this. I write ALL of my own songs. Your hot take is completely false and SO damaging. You don't have to like my songs but it's really
	1476219365381279746	335141638	생일 축하한다 태형아 격리중이라 마음만 보낸드아~#태형생일ㅊㅋ #슈가형이야 #행복한하루되렴
8	1493000178269179906	750751206427860992	Enter a new dimension of Strange. Watch the official trailer for Marvel Studios' #DoctorStrange in the Multiverse of Madness. Only in theaters May 6. https://t.co/FizGdQtoRv
	1504426844199669762	12044602	llove the Russian people. That is why I have to tell you the truth. Please watch and share. https://t.co/6gyVRhgpFV

Zložený index

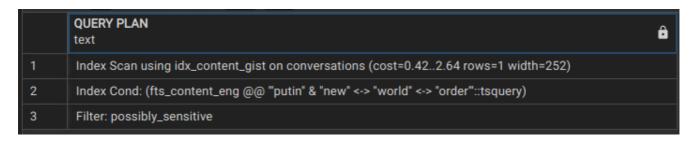
Separátne indexy by mali teoreticky pomalšie z toho dôvodu, že treba prejť viac rôznych indexov osobitne, teda máme viac I/O operácií ako pri zloženom indexe. V našom prípade, ale dôvodom zrýchelnia bolo, že vyhľadávanie v query je teraz vykonané pomocou jediného index scanu. Nie je už potrebný index scan na prvom indexe z ľava a potom filter na zvyšku. Preto je pre toto query výhodnejší zložený index. Použitie zloženého indexu na query, ho vylepší približne o 0.5 sekundy.

Úloha 14

Query:

```
-- create ts_vector column
alter table conversations
```

Explain Analyze + Výsledky:





Vytvorený GiST aj GIN



	QUERY PLAN text	â
1	Index Scan using idx_content_gist on conversations (cost=0.422.64 rows=1 width=252)	
2	Index Cond: (fts_content_eng @@ "putin" & "new" <-> "world" <-> "order"::tsquery)	
3	Filter: possibly_sensitive	

/fdi0gfcatn':11 /fqxxtyeh3t':31 /vvha9yiwx4':1
/vvha9yiwx4':1
/i2rdsiufg9':48
/1vbwuu4ctn':1
/ufijywbis0':37

Vytvorený iba GiST



	id [PK] bigint	author_id /	content text	possibly_sensitive boolean	language character varying (3)	source text	retweet_count integer	reply_count integer	like_count integer	quote_count /	created_at timestamp with time zone	fts_content_eng tsvector
	14976536080	1178005700°	#Putin an	true	en	Twitter fo					2022-02-26 20:23:39+01	'/i2rdsiufg9':48 '
	14980198421	8340331780	RT @vete	true	en	Twitter fo					2022-02-27 20:38:56+01	'/fdi0gfcatn':11 '
	14982194889	7365498079	"We fight	true	en	Twitter fo					2022-02-28 09:52:16+01	'/fqxxtyeh3t':31 '
	14984836188	435312888	#Putin Ha	true	en	Twitter fo					2022-03-01 03:21:49+01	'/1vbwuu4ctn':1
	15017455700	3247724967	#Putin is	true	en	Twitter fo					2022-03-10 03:23:39+01	'/vvha9yiwx4':1
	14973952186	47853235	The exper	true	en	Twitter fo					2022-02-26 03:16:55+01	'/ufijywbis0':37 '
Tota	l rows: 6 of 6	Query cor	nplete 00:00	:00.139								

Vytvorený iba GIN

Vyhľadávanie nad GIN je efektívnejšie. Aj napriek tomu, že riešené cez Bitmap Heap Scan a nie cez priamy index scan ako GIST. Ukázalo sa, že až je 10-krát rýchlejšie v porovnaní s GIST-om. Tiež sa nám GIN vytvoril rýchlejšie ako GIST (8 min vs 30 min).

Úloha 15

Query:

```
create index idx_urls_trgm on links using gin (url gin_trgm_ops);
select * from links where url like '%darujme.sk%';
```

Explain Analyze + Výsledky:



	id [PK] bigint	conversation_id , bigint	url character varying (2048)	title text	,	description text
1	1590851	149729983138307	https://charita.darujme.sk/ukrajina/	Pomoc L	J	To, čoho sme sa mesiace obávali, sa stalo skutočnosťou. Ruská federácia na
2	4304873	149834516483380	https://clovekvohrozeni.darujme.sk/pomoc-ukrajina			
3	9176935	150562076463619	https://redcross.darujme.sk/pomahame-ukrajine	Pomáha.		Prioritou Červeného kríža je zmierňovať utrpenie ľudí. Pracovníci a dobrovoľní
4	10672874	149754840933464	https://redcross.darujme.sk/pomahame-ukrajine/			
5	11036619	150150997208087	https://zvieraciombudsman.darujme.sk/animal-support-at-ukrainian-slovak-b			

Vytvorený GiST aj GIN

Správnou voľbou je vytvorenie GIN indexu nad trigramami vytvorenými z url. V tejto úlohe sa nepozeráme na text z pohľadu FTS, teda nepotrebujeme tokeny. Potrebujeme aby sme vedeli rýchlo vyhľadávať podreťazce pomocou LIKE, na to sú trigrami najvhodnejší typ.

Úloha 16

Indexy:

```
alter table authors
    add column fts_username_eng tsvector
        generated always as (to_tsvector('english', coalesce(username,'')))
stored;
alter table authors
    add column fts_description_eng tsvector
        generated always as (to_tsvector('english',
coalesce(description, ''))) stored;
alter table conversations
    add column fts_content_eng tsvector
        generated always as (to_tsvector('english', coalesce(content,'')))
stored;
create index idx_content_gin on conversations using gin (fts_content_eng);
create index idx_username_gin on authors using gin (fts_username_eng);
create index idx_description_gin on authors using gin
(fts_description_eng);
create index idx_author_id on authors using btree(id);
create index idx_conv_id on conversations using btree(author_id);
```

Query OR:

```
select
    authors.username, authors.description, conversations.content,
conversations.retweet_count
from
    conversations
inner join
    authors
on
    conversations.author_id = authors.id
where
    conversations.fts_content_eng @@ to_tsquery('english', 'Володимир &
Президент') ог
    authors.fts_username_eng @@ to_tsquery('english', 'Володимир &
Президент') or
    authors.fts_description_eng @@ to_tsquery('english', 'Володимир &
Президент')
order by
    retweet_count desc;
```

Explain:

```
      QUERY PLAN text

      1
      Gather Merge (cost=3354505.58..3354741.49 rows=2022 width=248)

      2
      Workers Planned: 2

      3
      -> Sort (cost=3355505.55..3553508.08 rows=1011 width=248)

      4
      Sort Key: conversations retweet_count DESC

      5
      -> Parallel Hash Join (cost=316153.33..3353455.10 rows=1011 width=248)

      6
      Hash Cond: (conversations author_id = authors.id)

      7
      Join Filter: ((conversations, fts_content_eng @@ "володимир" & "президент":tsquery) OR (to_tsvector(english::regconfig, (authors.username)::text) @@ "володимир" & "президент":tsquery) OR (to_tsvector(english::regconfig, (auth
```

JOIN OR

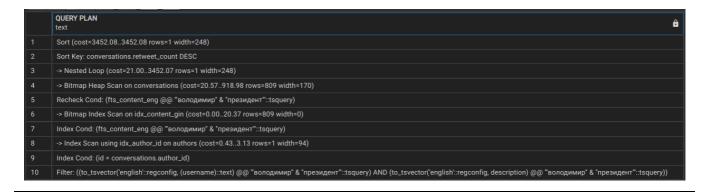
Problémom s OR podmienkov je, že plánovač sa ju snaží riešiť cez hash join, teda sa prechádzajú všetky riadky z jednej aj z druhej tabuľky a testujú sa voči podmienke. Tým, že OR nie je binárny ako AND, tak je potrebné vyskúšať viacero možností. Nemôžeme podmienku zamietnuť pri prvom porušení, lebo stále môže byť splnená v iných stĺpcoch. Preto sa plánovaču môže javiť hash join a pechádzanie cez všetky riadky ako vhodnejší spôsob.

Query AND:

```
select
   authors.username, authors.description, conversations.content,
conversations.retweet_count
from
   conversations
inner join
   authors
on
```

```
conversations.author_id = authors.id
where
    conversations.fts_content_eng @@ to_tsquery('english', 'Володимир &
Президент') and
    authors.fts_username_eng @@ to_tsquery('english', 'Володимир &
Президент') and
    authors.fts_description_eng @@ to_tsquery('english', 'Володимир &
Президент')
order by
    retweet_count desc;
```

Explain:



JOIN AND

Pri AND je situácia iné, pretože podmienka nám výrazne redukuje výsledok. Čiže sa už oplatí hľadať platnosť podmienky v indexe najprv pre prvý stĺpec a potom pre ďalšie. Nemusíme prechádzať cez všetky. Výsledky, pre ktoré podmienka platí (je ich určite výrazne menej v celom stĺci) sa potom ľahko spoja cez nested loop.

Výsledné query:

```
select
        authors.username, authors.description, conversations.content,
conversations.retweet count
    from
        conversations
    inner join
        authors
    on
        conversations.author id = authors.id
    where
        conversations.fts_content_eng @@ to_tsquery('english', 'Володимир &
Президент')
union
    select
        authors.username, authors.description, conversations.content,
conversations.retweet count
        conversations
    inner join
        authors
```

```
conversations.author_id = authors.id
    where
        authors.fts_description_eng @@ to_tsquery('english', 'Володимир &
Президент')
union
    select
        authors.username, authors.description, conversations.content,
conversations.retweet_count
        conversations
    inner join
        authors
    on
        conversations.author_id = authors.id
    where
        authors.fts_username_eng @@ to_tsquery('english', 'Володимир &
Президент')
order by
    retweet_count desc;
```

Explain + Výsledok:

	QUERY PLAN text
1	Sort (cost=15566.0515572.11 rows=2423 width=584)
2	Sort Key: conversations.retweet_count DESC
3	-> HashAggregate (cost=15405.6115429.84 rows=2423 width=584)
4	Group Key: authors.username, authors.description, conversations.content, conversations.retweet_count
5	-> Append (cost=21.0015381.38 rows=2423 width=584)
6	-> Nested Loop (cost=21.003040.83 rows=809 width=248)
7	-> Bitmap Heap Scan on conversations (cost=20.57918.98 rows=809 width=170)
8	Recheck Cond: (fts_content_eng @@ "володимир" & "президент"::tsquery)
9	-> Bitmap Index Scan on idx_content_gin (cost=0.0020.37 rows=809 width=0)
10	Index Cond: (fts_content_eng @@ "володимир" & "президент"::tsquery)
11	-> Index Scan using idx_author_id on authors (cost=0.432.62 rows=1 width=94)
12	Index Cond: (id = conversations.author_id)
13	-> Gather (cost=1013.686152.10 rows=807 width=248)
14	Workers Planned: 2
15	-> Nested Loop (cost=13.685071.40 rows=336 width=248)
16	-> Parallel Bitmap Heap Scan on authors authors_1 (cost=13.24175.33 rows=61 width=94)
17	Recheck Cond: (fts_description_eng @@ "володимир" & "президент"::tsquery)
18	-> Bitmap Index Scan on idx_description_gin (cost=0.0013.21 rows=147 width=0)
19	Index Cond: (fts_description_eng @@ "володимир" & "президент"::tsquery)
20	-> Index Scan using idx_conv_id on conversations conversations_1 (cost=0.4479.56 rows=70 width=170)
21	Index Cond: (author_id = authors_1.id)
22	-> Gather (cost=1013.686152.10 rows=807 width=248)
23	Workers Planned: 2
24	-> Nested Loop (cost=13.685071.40 rows=336 width=248)
25	-> Parallel Bitmap Heap Scan on authors authors_2 (cost=13.24175.33 rows=61 width=94)
26	Recheck Cond: (fts_username_eng @@ "володимир" & "президент"::tsquery)
27	-> Bitmap Index Scan on idx_username_gin (cost=0.0013.21 rows=147 width=0)
28	Index Cond: (fts_username_eng @@ "володимир" & "президент"::tsquery)
29	-> Index Scan using idx_conv_id on conversations conversations_2 (cost=0.4479.56 rows=70 width=170)
30	Index Cond: (author_id = authors_2.id)

	username character varying (255)	description text	content retweet_count integer	â
1	DefenceU	Official page of the Ministry	Президент України	5848
2	DefenceU	Official page of the Ministry	☞Президент Украї	5346
3	DefenceU	Official page of the Ministry	Головнокомандув	1489
4	UkrArmyBlog	СЛАВА УКРАЇНІ ТА ЇЇ ВОЇНА	!!! Майже 9000 вб	1217
5	verkhovna_rada	Офіційна сторінка Верховн	!!!!Президент Укр	1109
6	verkhovna_rada	Офіційна сторінка Верховн	!! Сьогодні, 1 берез	649
7	UkrArmyBlog	СЛАВА УКРАЇНІ ТА ЇЇ ВОЇНА	Президент ША Во	609
8	ua_industrial	My name is Pavlo. Here the w	Президент Володи	472
9	UkrArmyBlog	СЛАВА УКРАЇНІ ТА ЇЇ ВОЇНА	Президент України	433
10	ltpzdc	Тут атланти-терикони підпи	Президент у відпо	423
11	APUkraine	Офіс Президента України /	Україна звертаєть	347
12	APUkraine	Офіс Президента України /	Україна сьогодні в	341
13	APUkraine	Офіс Президента України /	Президент Володи	337
14	APUkraine	Офіс Президента України /	Президент України	303
15	GeneralStaffUA	Офіційний акаунт Генераль	Президент України	294
16	APUkraine	Офіс Президента України /	Президент Володи	287
17	APUkraine	Офіс Президента України /	Сприяння наданн	274
18	GeneralStaffUA	Офіційний акаунт Генераль	 ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐	258
19	Armia_fm	Перше в Україні військове	Президент Володи	246
20	APUkraine	Офіс Президента України /	Президент Володи	219
21	UKRINFORM	Мультимедійна платформа	Буча, Ірпінь, Охтир	208
22	10r1_sss24	ARMY / MOA / ENGENE \n\	RT @DefenceU: Пре	205
23	ShtfLiving	Ψ ♂ Prepper or Become a So	RT @DefenceU: Пре	205
24	erennegger	veni, vidi, dici	RT @DefenceU: Пре	205
25	AlexBor62550895		RT @DefenceU: Пре	205
26	GShelia1951	Liberal/Atheist/Antifa/Radica	RT @DefenceU: Пре	205
27	VovaAnt		RT @DefenceU: Пре	205
28	robotoasterdev	One in Eight\n\nbg autorstwa	RT @DefenceU: Пре	205
29	Ashkel333	Слава Украине!!!\пЖыве Бе	RT @DefenceU: Пре	205
30	anyazgxlb		RT @DefenceU: Пре	205
31	Fran_DagoGames	nada	RT @DefenceU: Пре	205
32	Oles84179873		RT @DefenceU: Пре	205
33	pokesourdPiR2		RT @DefenceU: Пре	205
34	SvenKauber	EE FI IM Full Stack Dev	RT @DefenceU: Пре	205
Total	rows: 1031 of 1031 Qu	uery complete 00:00:00.113		

Najlepšie query

Vytvorenie indexov nad PK (authors.id) a FK (conversations.authors_id) nám zapríčiní zmenu join algoritmu z hash join na nested loop, ktorý si už pomáha použitím indexov pri matchovaní PK a FK. Čo zlepší query približne o 0.5 sekundy.

Keďže plánovač nevie optimalizovať OR podmienku pri joine, tak sme spustili vyhľadávnie ako samostatné query pre každý stĺpec. Použili sme viacero malých joinov, ktoré je výhodnejšie porovnať cez index. Každé so samostatných vyhľadaní plánovač vie optimalizovať pomocou indexu. A výsledok sme vytvorili zjednotením výstupov pomocou union operácie.