

Напредни бази на податоци

Фаза 4 - Индекси и оптимизација на прашалници

Проект: LibraryDB

Андреј Станоев 231103

Тодор Темелков 231089

Благоја Џорлев 231062

View1 vw_active_memberships

1. Примарен филтер за погледот е според id на membership, дополнително ќе се користи id на корисник и email на корисник.
2. Примарен случај на употреба ќе е пребарување на активни членови.
3. Иницијално време на извршување 512 ms.

```
explain analyze
select * from vw_active_memberships where id = 5757215;
```

```
QUERY PLAN
1  Nested Loop  (cost=1.00..25.10 rows=1 width=229) (actual time=0.247..0.251 rows=1 loops=1)
2    -> Nested Loop  (cost=0.85..16.90 rows=1 width=83) (actual time=0.201..0.204 rows=1 loops=1)
3      -> Index Scan using membership_pkey on membership m  (cost=0.43..8.45 rows=1 width=40) (actual time=0.169..0.171 rows=1 loops=1)
4        Index Cond: (id = 5757215)
5        Filter: ((cancelled_at IS NULL) AND (expires_at > now()))
6      -> Index Scan using "User_pkey" on "User" u  (cost=0.43..8.45 rows=1 width=51) (actual time=0.027..0.027 rows=1 loops=1)
7        Index Cond: (id = m.user_id)
8    -> Index Scan using membershipplan_pkey on membershipplan mp  (cost=0.15..8.17 rows=1 width=154) (actual time=0.044..0.044 rows=1 loops=1)
9      Index Cond: (id = m.membership_plan_id)
10 Planning Time: 0.551 ms
11 Execution Time: 0.296 ms
```

4. Ова е прифатливо време за извршување, нема потреба од додавање на индекси.

View2 vw_active_rentals

1. Примарен филтер за погледот е според id на rental, дополнително ќе се користи id на membership и име на издание.
2. Примарен случај на употреба ќе е пребарување на активно издадени книги.
3. Иницијално време на извршување 801 ms.

```
explain analyze
select * from vw_active_rentals where id = 6;
```

QUERY PLAN	
6	Index Cond: (id = 6)
7	Filter: (returned_at IS NULL)
8	-> Index Scan using resourcecopy_pkey on resourcecopy rc (cost=0.43..8.45 rows=1 width=27) (actual time=0.037..0.037 rows=1 width=27)
9	Index Cond: (id = r.resource_copy_id)
10	-> Index Scan using edition_pkey on edition e (cost=0.42..7.67 rows=1 width=75) (actual time=0.105..0.105 rows=1 width=75)
11	Index Cond: ((isbn)::text = (rc.edition_isbn)::text)
12	-> Index Scan using membership_pkey on membership m (cost=0.43..8.45 rows=1 width=16) (actual time=0.037..0.037 rows=1 width=16)
13	Index Cond: (id = r.membership_id)
14	-> Index Scan using "User_pkey" on "User" u (cost=0.43..7.66 rows=1 width=38) (actual time=0.043..0.043 rows=1 loops=1)
15	Index Cond: (id = m.user_id)
16	Planning Time: 2.612 ms
17	Execution Time: 0.537 ms

4. Прифатливо време, нема потреба од оптимизации.

View3 vw_available_editions_for_rent

1. Примарен филтер за погледот е според id на rental, дополнително ќе се користи id на membership и име на издание.
2. Примарен случај на употреба ќе е пребарување на книги кои што можат да се изнајмат.
3. Иницијално време на извршување 1s 424ms.

```

explain analyze
select * from vw_available_editions_for_rent where isbn = 'ISBN-100019-4-7382';

```

QUERY PLAN	
2	Filter: ((count(rc.id) - count(r_1.id)) > 0)
3	-> Nested Loop Left Join (cost=1002.14..207140.75 rows=294 width=91) (actual time=584.994..594.119 rows=100 loops=1)
4	Join Filter: (r.resource_copy_id = rc.id)
5	-> Nested Loop Left Join (cost=1002.14..207105.30 rows=294 width=91) (actual time=584.959..594.058 rows=100 loops=1)
6	Join Filter: (r_1.resource_copy_id = rc.id)
7	-> Nested Loop Left Join (cost=1000.42..206969.29 rows=294 width=83) (actual time=584.923..593.997 rows=100 loops=1)
8	-> Index Scan using edition_pkey on edition e (cost=0.42..8.44 rows=1 width=75) (actual time=0.081..0.087 rows=1 loops=1)
9	Index Cond: ((isbn)::text = 'ISBN-100019-4-7382'::text)
10	-> Gather (cost=1000.00..206957.91 rows=294 width=27) (actual time=584.758..593.813 rows=100 loops=1)
11	Workers Planned: 2
12	Workers Launched: 2
13	-> Parallel Seq Scan on resourcecopy rc (cost=0.00..205928.51 rows=122 width=27) (actual time=395.175..553.533 rows=33..)
14	Filter: ((edition_isbn)::text = 'ISBN-100019-4-7382'::text)
15	Rows Removed by Filter: 3333300
16	-> Materialize (cost=1.72..118.39 rows=4 width=16) (actual time=0.000..0.000 rows=0 loops=100)
17	-> Nested Loop (cost=1.72..118.37 rows=4 width=16) (actual time=0.020..0.023 rows=0 loops=1)
18	-> Nested Loop (cost=1.29..101.88 rows=4 width=24) (actual time=0.019..0.022 rows=0 loops=1)
19	-> Nested Loop (cost=0.86..68.10 rows=4 width=24) (actual time=0.018..0.020 rows=0 loops=1)
20	-> Nested Loop (cost=0.43..51.61 rows=4 width=43) (actual time=0.018..0.019 rows=0 loops=1)
21	-> Seq Scan on rental r_1 (cost=0.00..17.80 rows=4 width=24) (actual time=0.017..0.017 rows=0 loops=1)
22	Filter: (returned_at IS NULL)
23	-> Index Scan using resourcecopy_pkey on resourcecopy rc_1 (cost=0.43..8.45 rows=1 width=27) (never executed)
24	Index Cond: (id = r_1.resource_copy_id)
25	-> Index Only Scan using edition_pkey on edition e_1 (cost=0.42..4.12 rows=1 width=19) (never executed)
26	Index Cond: (isbn = (rc_1.edition_isbn)::text)
27	Heap Fetches: 0
28	-> Index Scan using membership_pkey on membership m (cost=0.43..8.45 rows=1 width=16) (never executed)
29	Index Cond: (id = r_1.membership_id)
30	-> Index Only Scan using "User_pkey" on "User" u (cost=0.43..7.66 rows=1 width=38) (never executed)
31	Index Cond: (id = m.user_id)

4. Овде премногу време трае делот за секвенцијално скенирање на копиите. Со тоа можеме да додадime индекс на isbn во рамки на табелата за копиите.

5. Времето на извршување со додаден индекс изнесува 644ms.

```

-> Bitmap Heap Scan on resourcecopy rc (cost=6.71..1147.83 rows=294 width=27) (actual time=0.097..0.147 rows=100 loops=1)
    Recheck Cond: ((edition_isbn)::text = 'ISBN-100019-4-7382')::text)
    Heap Blocks: exact=2
-> Bitmap Index Scan on idx_resourcecopy_edition_isbn (cost=0.00..6.64 rows=294 width=0) (actual time=0.055..0.055 rows=100 loops=1)
    Index Cond: ((edition_isbn)::text = 'ISBN-100019-4-7382')::text)
-> Hash (cost=113.15..113.15 rows=4 width=16) (actual time=0.008..0.011 rows=0 loops=1)
    Buckets: 1024 Batches: 1 Memory Usage: 8kB
-> Nested Loop (cost=5.85..113.15 rows=4 width=16) (actual time=0.008..0.009 rows=0 loops=1)
    -> Nested Loop (cost=5.42..96.66 rows=4 width=24) (actual time=0.007..0.009 rows=0 loops=1)
        -> Nested Loop (cost=4.99..62.88 rows=4 width=24) (actual time=0.007..0.008 rows=0 loops=1)
            -> Nested Loop (cost=4.57..46.39 rows=4 width=43) (actual time=0.007..0.008 rows=0 loops=1)
                -> Bitmap Heap Scan on rental r_1 (cost=4.13..12.58 rows=4 width=24) (actual time=0.007..0.007 rows=0 loops=1)
                    Recheck Cond: (returned_at IS NULL)
                -> Bitmap Index Scan on idx_rental_active_resource_copy (cost=0.00..4.13 rows=4 width=0) (actual time=0.007..0.007 rows=0 loops=1)
                    Index Cond: (id = r_1.resource_copy_id)
            -> Index Scan using resourcecopy_pkey on resourcecopy rc_1 (cost=0.43..8.45 rows=1 width=27) (never executed)
                Index Cond: (id = r_1.resource_copy_id)
        -> Index Only Scan using edition_pkey on edition e_1 (cost=0.42..4.12 rows=1 width=19) (never executed)
            Index Cond: (isbn = (rc_1.edition_isbn)::text)
            Heap Fetches: 0
    -> Index Scan using membership_pkey on membership m (cost=0.43..8.45 rows=1 width=16) (never executed)
        Index Cond: (id = r_1.membership_id)
    -> Index Only Scan using "User_pkey" on "User" u (cost=0.43..4.12 rows=1 width=8) (never executed)
        Index Cond: (id = m.user_id)
        Heap Fetches: 0
-> Hash (cost=12.58..12.58 rows=4 width=8) (actual time=0.007..0.008 rows=0 loops=1)
    Buckets: 1024 Batches: 1 Memory Usage: 8kB
-> Bitmap Heap Scan on rental r (cost=4.13..12.58 rows=4 width=8) (actual time=0.007..0.007 rows=0 loops=1)
    Recheck Cond: (returned_at IS NULL)
-> Bitmap Index Scan on idx_rental_active_resource_copy (cost=0.00..4.13 rows=4 width=0) (actual time=0.002..0.003 rows=0 loops=1)

Planning Time: 3.342 ms
Execution Time: 0.477 ms

```

View4 vw_edition_copy_count

1. Примарен филтер за погледот **vw_edition_copy_count** ќе биде според isbn на едисијата. Може да се користи и според името на едисијата.
2. Примарен случај на употреба на овој поглед е кога корисник или пак некој вработен би сакал да погледне од која едисија колку копии има.
3. Иницијално време на извршување на оваа квери е 547ms. Ова е прифатливо време, нема потреба од индекси.

```

1 ✓ select *
2 from vw_edition_copy_count as v
3 where v.isbn = 'ISBN-100002-5-74658';

```

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup Cost	Actual Startup Time	Raw Desc
Aggregate		1	1	1159.96	0.161	7.14	0.16	Planning Time = 0.346; Trig...
↳ Nested Loops (Nested Loo		294	200	1159.21	0.142	7.14	0.084	Strategy = Sorted/Partial M...
↳ Index Scan	table: edition; index: edition_pkey;	1	1	8.44	0.027	0.42	0.026	Parent Relationship = Outer...
↳ Bitmap Index Scan (Btm	table: resourcecopy;	294	200	1147.83	0.087	6.71	0.053	Parent Relationship = Inner...
↳ Bitmap Index Scan	index: idx_resourcecopy_edition_isbn;	294	200	6.64	0.039	0.0	0.039	Parent Relationship = Outer...

View5 vw_edition_ratings

1. Примарен филтер за погледот **vw_edition_ratings** ќе биде според isbn на едисијата. Може да се користи и според името на едисијата.

2. Примарен случај на употреба на овој поглед е кога корисник или пак некој вработен би сакал да виде просечната оцена за некоја едиција.
3. Иницијалното извршување на овој поглед е 1,6 s. Ова не е прифатливо време за апликацијата па поради тоа ќе пробаме да оптимизираме со индекси.

```
6 ✓
7
8 select *
9 from vw_edition_ratings as v
10 where v.isbn = 'ISBN-100000-8-85075';
```

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup Cost	Actual Startup Time
Select		1	1	156461.79	252.846	1000.42	244.806
Aggregate		1	1	156461.76	252.787	1000.42	46.458
Nested Loops (Nested Loop)		4	4	8.44	0.034	0.42	0.029
Index Scan	table: edition; index: edition_pkey;	1	1	156453.27	252.738	1000.0	46.419
Unknown (Gather)		4	4	155452.88	211.093	0.0	100.483
Full Scan (Seq Scan)	table: review;	1	1				

4. Бидејќи во табелата review имавме full scan, а во дефиницијата на погледот имаме join по review.resource_edition_isbn го создадив овој индекс и времето се намали до 600ms

```
create index review_resource_edition_isbn on review(resource_edition_isbn);
```

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup Cost	Actual Startup Time
Select		1	1	29.01	0.067	0.85	0.066
Aggregate		1	1	28.98	0.055	0.85	0.047
Nested Loops (Nested Loop)		4	4	8.44	0.027	0.42	0.026
Index Scan	table: edition; index: edition_pkey;	1	1	20.5	0.023	0.43	0.017
Index Scan	table: review; index: review_resource_edition_isbn;	4	4				

View6 vw_upcoming_events

1. Примарен филтер за погледот **vw_upcoming_events** ќе биде по id на библиотека. А уште ќе се користи и според име на настан и тип на настан.
2. Примарен случај на употреба на овој поглед е кога корисник ќе сака да погледне кои се идни настани што ќе се случуваат во библиотеката.
3. Иницијално време на извршување на погледот е 1.8s. Ова не е прифатливо време за апликацијата па затоа ќе пробаме да оптимизираме со индекси.

```
select *
from vw_future_events_for_lib as v
where v.library_id = 1;
```

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup Cost	Actual Startup Time
↳ Select							
↳ Nested Loops (Nested Loop)		16353	16671	23456.11	98.138	21309.01	78.497
↳ Unknown (Gather Merge)		16353	16671	23243.53	91.967	21308.86	78.467
↳ Sort		5275	4168	20322.01	49.473	20308.82	48.949
↳ Hash Join		5275	4168	19982.7	46.115	23.38	0.332
↳ Hash Join		5275	4168	19952.78	44.789	7.54	0.276
↳ Full Scan (Seq Scan)	table: event;	5275	4168	19931.16	43.001	0.0	0.035
↳ Transformation (Hash)		246	246	4.46	0.121	4.46	0.12
↳ Full Scan (Seq Scan)	table: room;	246	246	4.46	0.066	0.0	0.029
↳ Transformation (Hash)		260	6	12.6	0.028	12.6	0.027
↳ Full Scan (Seq Scan)	table: eventtype;	260	6	12.6	0.02	0.0	0.019
↳ Temporary (Materialize)		1	1	8.17	0.0	0.15	0.0
↳ Index Scan	table: library; index: library_pkey;	1	1	8.17	0.024	0.15	0.022

4. Бидејќи во погледот имаме дефинирато where >= now() а и спојуваме според ид на библиотека решив на создадам индекс каков што е одма подоле на сликата и сега погледот се извршува за 600ms.

```
create index idx_event_library_id_start_time on event(library_id, start_time);
```

Operation	Params	Rows	Actual Rows	Total Cost
↳ Select				
↳ Nested Loops (Nested Loop)		16353	16670	19020.98
↳ Unknown (Gather Merge)		16353	16670	18808.4
↳ Sort		5275	4168	15886.89
↳ Hash Join		5275	4168	15547.57
↳ Hash Join		5275	4168	15517.65
↳ Bitmap Index Scan (Bitmap)	table: event;	5275	4168	15496.04
↳ Bitmap Index Scan	index: idx_event_library_id_start_time;	16353	16670	419.96
↳ Transformation (Hash)		246	246	4.46
↳ Full Scan (Seq Scan)	table: room;	246	246	4.46
↳ Transformation (Hash)		260	6	12.6
↳ Full Scan (Seq Scan)	table: eventtype;	260	6	12.6
↳ Temporary (Materialize)		1	1	8.17
↳ Index Scan	table: library; index: library_pkey;	1	1	8.17

5. Не додавам индекси за event_room_id и event_type_id бидејќи тие две табели имаат малце редови.

View7 vw_overdue_rentals

1. Како примарен филтер би го земале id од табелата resource исто така може да се пребарува и според email.
2. Примарен случај на употреба би било корисникот да добие информација која книга до кога треба да ја врати.
3. Инцијалното време на извршување е 485ms што е прифатливо односно нема потреба за додавање на индекси.

```
select * from vw_overdue_rentals
where id = 535211;
```

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup Cost	Actual Startup Time
Select		1	1	40.68	0.289	2.15	0.285
Nested Loops (Nested Lc		1	1	33.02	0.235	1.72	0.231
Nested Loops (Nestec		1	1	24.58	0.169	1.29	0.166
Nested Loops (Nes		1	1	16.91	0.15	0.87	0.147
Nested Loops (I		1	1	8.46	0.076	0.43	0.075
Index Scan	table: rental; index: rental_pkey;	1	1	8.45	0.069	0.43	0.069
Index Scan	table: resourcecopy; index: resourcecopy_pkey;	1	1	7.67	0.017	0.42	0.017
Index Scan	table: edition; index: edition_pkey;	1	1	8.45	0.063	0.43	0.063
Index Scan	table: membership; index: membership_pkey;	1	1	7.66	0.052	0.43	0.052
Index Scan	table: User; index: User_pkey;	1	1				

consoles > advdb_202526l_prj_libdb > console 1 [advdb_202526l_prj_libdb]

View8 vw_sorted_by_rentals_rented

- 1.Како примарен филтер ќе го искористиме ISBN од табелата edition дополнително можеме да пребраме преку name од табелата edition.
- 2.Примарен случај на употреба би бил да добие прегледност корисникот во тоа кој се најпопуларни изнајмени книги.
- 3.Иницијално време на извршување е 1.746ms.

```
select * from vw_sorted_by_rentals_rented_editions
where isbn = 'ISBN-155380-8-87703';
```

- 4.Овде најголем проблем ни претставува секвенцијалниот скен во табелата rental затоа додаваме индекс во табелата rental на колоната resource_copy_id.

Operation	Params	Rows	Actual Ro...	Total C...	Actual Total T...	Startup C...	Actual Startup ...
Select		1	1	156606.89	15189.246	2151.93	15180.353
Aggregate		294	107	156606.15	15189.176	2151.93	15078.102
Nested Loops (N		1	1	8.44	0.111	0.42	0.106
Index Scan	table: edition; index: edition_p..	294	107	156594.76	15189.038	2151.51	15077.986
Unknown (Gathe		122	36	155565.36	15147.283	1151.51	14618.499
Hash Join		4166...	3333333	143476.42	14782.638	0	100.407
Full Sc table: rental;		294	100	1147.83	12.088	1147.83	12.086
Transform:		294	100	1147.83	12.038	6.71	12.01
Bitmap table: resourcecopy;							

5. Времето на извршување по додадениот индекс изнесува 391ms.

Operation	Params	Rows	Actual Ro...	Total C...	Actual Total ...	Startup C...	Actual Startup ...
↳ Select							
↳ Aggregate		1	1	5984.85	0.591	7.57	0.589
↳ Nested Loop		294	107	5984.11	0.571	7.57	0.16
↳ Index Scan table: edition; index: edition_pkey;		1	1	8.44	0.061	0.42	0.06
↳ Nested Loop		294	107	5972.72	0.494	7.15	0.097
↳ Bitmap table: resourcecopy;		294	100	1147.83	0.098	6.71	0.068
↳ Bitmap index: idx_resourcecopy_edition_isb...		294	100	6.64	0.047	0	0.047
↳ Index table: rental; index: idx_rental_resour...		3	1	16.38	0.003	0.43	0.003

View9 vw_user_activiy

1. Како примарен филтер ќе го искористиме user_id дополнително можеме да пребаруваме според email.
2. Примарен случај на употреба би било корисникот да добие преглед во своите податоци во библиотеките како колку пати изнајмил колку пати оставил оцена на книга и на колку настани учествувал.
3. Иницијалното време на извршување претставува 4.899ms.

```
select * from vw_user_activity
where user_id = 1
```

4. Пребарувањето трае премногу време поради тоа што имаме 4 различни секвенцијални скенирања на табелите.

rental, membership, review, eventregistration за да го решиме овој проблем можеме да додадеме индекс на user_id во табелите membership, review и eventregistration, а во табелата rental да додадеме индекс на membership_id.

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup ...	Actual S...
↳ Select							
↳ Aggregate		1	1	475547.68	252480.534	475539.27	252470.06
↳ Sort		840	20	475541.37	252480.436	475539.27	252469.9...
↳ Nested Loops (Nested Loop)		840	20	475498.48	252480.41	25223.23	118328.013
↳ Nested Loops (Nested Loop)		70	4	342433.41	251767.88	24223.23	117615.833
↳ Nested Loops (Nested Loop)		7	1	177646.21	101.377	23223.23	100.995
↳ Index Scan table: User; index: User_pkey;		1	1	8.45	0.061	0.43	0.053
↳ Unknown (Gather)		7	0	177637.7	101.231	23222.8	100.858
↳ Hash Join		3	0	176637	68.869	22222.8	68.864
↳ Full Scan (Seq Sc table: rental;		4166667	0	143476.67	0	0	0
↳ Transformation (Ha:		1	0	22222.79	68.348	22222.79	68.346
↳ Full Scan (Seq table: membership;		1	0	22222.79	68.253	0	68.253
↳ Temporary (Materialize)		10	4	164786.35	251666.485	1000	117514.826
↳ Unknown (Gather)		10	4	164786.3	251666.458	1000	117514.81
↳ Full Scan (Seq Scan table: review;		4	1	163785.3	251874.003	0	191962.368
↳ Temporary (Materialize)		12	5	133054.59	178.111	1000	178.032
↳ Unknown (Gather)		12	5	133054.53	712.433	1000	712.12
↳ Full Scan (Seq Scan) table: eventregistration;		5	2	132053.33	561.81	0	542.692

5. Со додавање на овие индекси времето паѓа на прифатливи 350ms.

Output		Query Plan		advdb_2025261_prj_lli...blic.vw_user_activity			
Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup Cost	Actual Startup Time
⌵ Select							
⌵ Aggregate		1	1	167.43	0.487	159.02	0.485
⌵ Sort		840	35	161.12	0.455	159.02	0.45
⌵ ⌵ Nested Loops (Nested Loo		840	35	118.22	0.429	2.15	0.328
⌵ ⌵ Nested Loops (Nested L		70	7	99.05	0.32	1.72	0.233
⌵ ⌵ Nested Loops (Neste		7	7	53.55	0.242	1.29	0.161
⌵ ⌵ Nested Loops (Ne		1	1	16.9	0.12	0.85	0.118
⌵ ⌵ ⌵ Index Scan	table: User; index: User_pkey;	1	1	8.45	0.064	0.43	0.063
⌵ ⌵ ⌵ Index Scan	table: membership; index: idx_membership_user_id;	1	1	8.45	0.051	0.43	0.05
⌵ ⌵ ⌵ Index Scan	table: rental; index: idx_rental_membership_id;	8	7	36.57	0.116	0.43	0.041
⌵ ⌵ ⌵ Temporary (Materiali		10	1	44.64	0.01	0.43	0.01
⌵ ⌵ ⌵ Index Scan	table: review; index: idx_review_user_id;	10	1	44.59	0.067	0.43	0.066
⌵ ⌵ Temporary (Materialize)		12	5	8.71	0.014	0.43	0.013
⌵ ⌵ ⌵ Index Scan	table: eventregistration; index: idx_eventregistration_user_id;	12	5	8.64	0.094	0.43	0.091