

Напредни бази на податоци

Фаза 4 – Индекси и оптимизација на прашалници

Проект: Eventix

Членови на тимот:

- Сандра Ацевска 231070 – координатор
- Филип Ѓоргиев 231122
- Ивана Костовска 232007

View 1: view_events_by_date

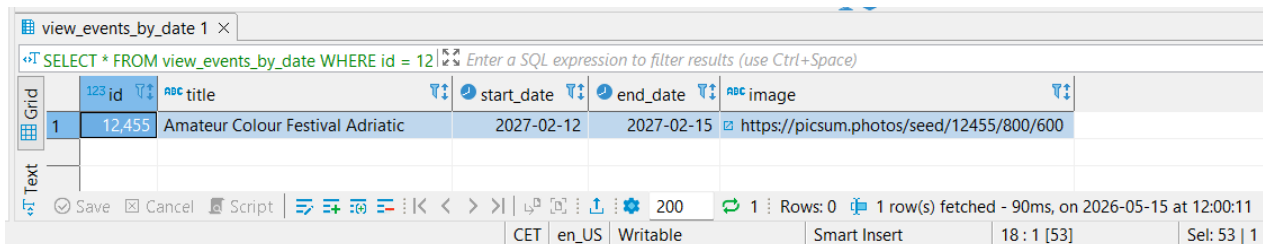
1. Примарен филтер: Погледот примарно се филтрира според event_id, а исто така се користи и start_date за филтрирање на идни настани.

2. Случај на употреба: Се користи на почетната страница на апликацијата за приказ на идни настани подредени по датум.

3. Иницијално време

```
SELECT * FROM view_events_by_date WHERE id = 12455;
```

-- Резултат: 90 ms



The screenshot shows a database client interface with a query editor and a results grid. The query executed is `SELECT * FROM view_events_by_date WHERE id = 12455;`. The results grid displays one row with the following data:

	id	title	start_date	end_date	image
1	12455	Amateur Colour Festival Adriatic	2027-02-12	2027-02-15	https://picsum.photos/seed/12455/800/600

The status bar at the bottom indicates that 1 row(s) were fetched in 90ms on 2026-05-15 at 12:00:11.

Ова е прифатливо време за апликацијата, и за истото не е потребно индексирање.

4. План на извршување (EXPLAIN ANALYZE)

```
EXPLAIN ANALYZE SELECT *
```

```
FROM view_events_by_date
```

```
WHERE id = 12455;
```

Results 1 ×	
EXPLAIN ANALYZE SELECT * FROM view_events_by_d	
Enter a SQL expression to filter results (use Ctrl+Space)	
Grid	ABC QUERY PLAN
Text	1 Sort (cost=16300.84..16300.85 rows=1 width=83) (actual time=35.081..42.071 rows=1 loops=1)
	2 Sort Key: e.start_date
	3 Sort Method: quicksort Memory: 25kB
	4 -> GroupAggregate (cost=1000.42..16300.83 rows=1 width=83) (actual time=35.071..42.061 rows=1 loops=1)
	5 -> Nested Loop Left Join (cost=1000.42..16300.80 rows=10 width=91) (actual time=1.527..42.029 rows=10 loops=1)
	6 -> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=51) (actual time=0.019..0.024 rows=1 loops=1)
	7 Index Cond: (id = 12455)
	8 Filter: (start_date >= CURRENT_DATE)
	9 -> Gather (cost=1000.00..16292.26 rows=10 width=48) (actual time=1.506..41.997 rows=10 loops=1)
	10 Workers Planned: 3
	11 Workers Launched: 3
	12 -> Parallel Seq Scan on event_image ei (cost=0.00..15291.26 rows=3 width=48) (actual time=5.408..29.224 rows=2 loops=4)
	13 Filter: (eventid = 12455)
	14 Rows Removed by Filter: 249998
Record	15 Planning Time: 0.285 ms
	16 Execution Time: 42.126 ms

Времето е прифатливо и затоа не се продолжува со индексирање.

5. Нема потреба од индексирање

6. Време на извршување на INSERT/UPDATE

Statistics 1 ×	
Name	Value
Updated Rows	1
Query	INSERT INTO event (title, description, start_date, end_date, categorizationid, subcategoryid) VALUES ('New Test Event ', 'Test opis', CURRENT_DATE + 10, CURRENT_DATE + 11, 1, 1)
Finish time	Fri May 15 12:10:56 CEST 2026
Save Cancel Script	
200 1 Rows: 1 1 row(s) updated - 44ms,	

Statistics 1 ×	
Name	Value
Updated Rows	1
Query	UPDATE event SET title = 'New Test Event Updated' WHERE id = (SELECT MAX(id) FROM event)
Finish time	Fri May 15 12:11:56 CEST 2026
Save Cancel Script	
200 1 Rows: 1 1 row(s) updated - 14ms,	

Времето на операциите INSERT и UPDATE останува исто бидејќи нема нови индекси.

View 2: view_event_details

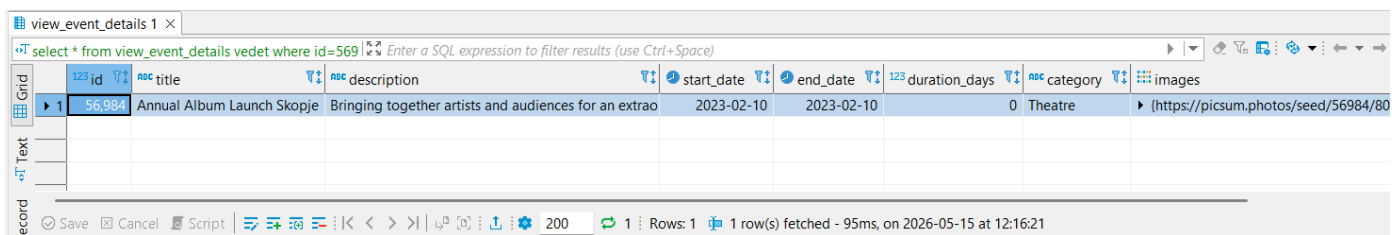
1. Примарен филтер: Погледот примарно се филтрира според event_id.

2. Случај на употреба: Се користи кога корисникот ќе кликне на конкретен настан за да ги види деталите - наслов, опис, датуми, категорија и слики.

3. Иницијално време

```
SELECT * FROM view_event_details WHERE id = 56984;
```

-- Резултат: 95ms

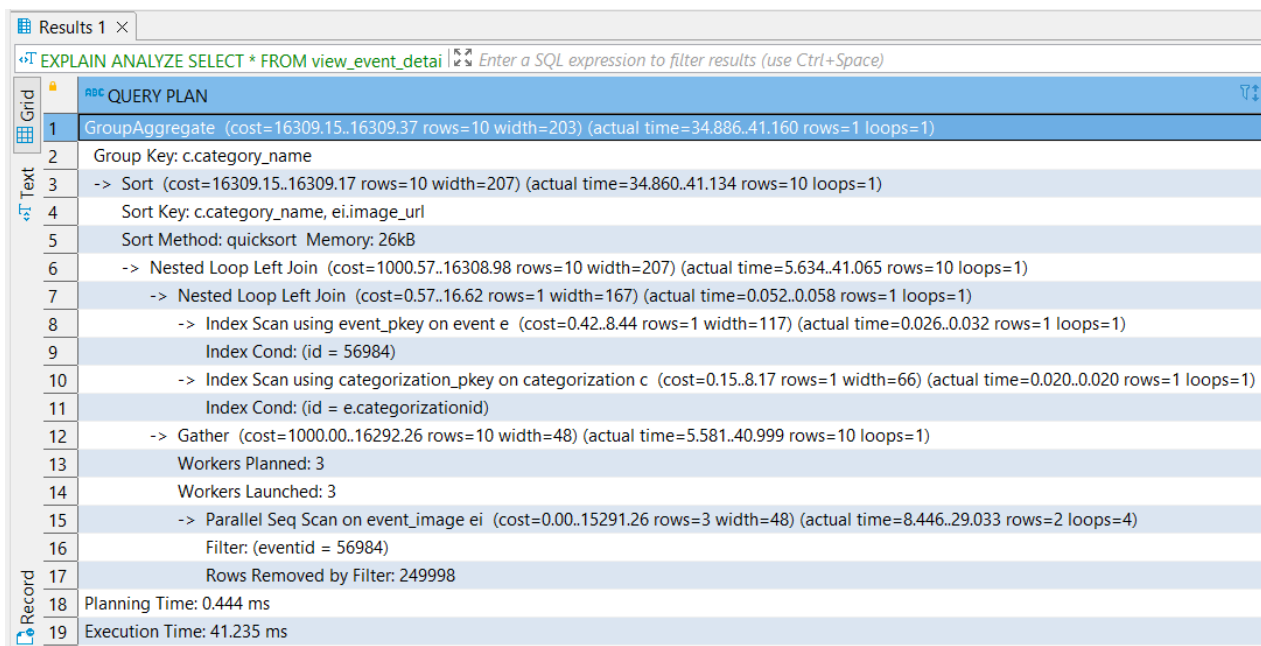


The screenshot shows a database interface with a query window containing the SQL statement: `select * from view_event_details vedet where id=569`. The results are displayed in a table with columns: id, title, description, start_date, end_date, duration_days, category, and images. The first row is highlighted, showing details for event_id 56984.

id	title	description	start_date	end_date	duration_days	category	images
56984	Annual Album Launch Skopje	Bringing together artists and audiences for an extrao	2023-02-10	2023-02-10	0	Theatre	(https://picsum.photos/seed/56984/80

4. План на извршување

```
EXPLAIN ANALYZE SELECT * FROM view_event_details WHERE id = 56984;
```



The screenshot shows the query plan for the SQL statement: `EXPLAIN ANALYZE SELECT * FROM view_event_details WHERE id = 56984`. The plan is displayed in a table with columns: id, text, and cost. The plan shows the execution flow from the GroupAggregate step down to the Parallel Seq Scan on event_image ei, which is the most expensive step.

id	text	cost
1	GroupAggregate (cost=16309.15..16309.37 rows=10 width=203) (actual time=34.886..41.160 rows=1 loops=1)	
2	Group Key: c.category_name	
3	-> Sort (cost=16309.15..16309.17 rows=10 width=207) (actual time=34.860..41.134 rows=10 loops=1)	
4	Sort Key: c.category_name, ei.image_url	
5	Sort Method: quicksort Memory: 26kB	
6	-> Nested Loop Left Join (cost=1000.57..16308.98 rows=10 width=207) (actual time=5.634..41.065 rows=10 loops=1)	
7	-> Nested Loop Left Join (cost=0.57..16.62 rows=1 width=167) (actual time=0.052..0.058 rows=1 loops=1)	
8	-> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=117) (actual time=0.026..0.032 rows=1 loops=1)	
9	Index Cond: (id = 56984)	
10	-> Index Scan using categorization_pkey on categorization c (cost=0.15..8.17 rows=1 width=66) (actual time=0.020..0.020 rows=1 loops=1)	
11	Index Cond: (id = e.categorizationid)	
12	-> Gather (cost=1000.00..16292.26 rows=10 width=48) (actual time=5.581..40.999 rows=10 loops=1)	
13	Workers Planned: 3	
14	Workers Launched: 3	
15	-> Parallel Seq Scan on event_image ei (cost=0.00..15291.26 rows=3 width=48) (actual time=8.446..29.033 rows=2 loops=4)	
16	Filter: (eventid = 56984)	
17	Rows Removed by Filter: 249998	
18	Planning Time: 0.444 ms	
19	Execution Time: 41.235 ms	

Најбавната операција е Parallel Seq Scan на табелата event_image која скенира ~250,000 редови. Иако постои Full Scan, времето е прифатливо па нема потреба од индексирање.

5. Нема потреба од индексирање

6. Време на INSERT/UPDATE

Statistics 1	
Name	Value
Updated Rows	1
Query	INSERT INTO event (title, description, start_date, end_date, categorizationid, subcategoryid) VALUES ('Test Event Details', 'Test opis', CURRENT_DATE + 10, CURRENT_DATE + 11, 2, 1)
Finish time	Tue May 19 10:09:34 CEST 2026
<div> Save Cancel Script </div> <div> 200 1 Rows: 1 1 row(s) updated - 78ms </div>	

Statistics 1	
Name	Value
Updated Rows	1
Query	UPDATE event SET description = 'Test opis updated' WHERE title = 'Test Event Details'
Finish time	Tue May 19 10:10:26 CEST 2026
<div> Save Cancel Script </div> <div> 200 1 Rows: 1 1 row(s) updated - 32ms </div>	

Времето на операциите INSERT и UPDATE останува исто бидејќи нема нови индекси.

View 3: view_event_halls

- Примарен филтер:** Погледот примарно се филтрира според event_id.
- Случај на употреба:** Се користи кога корисникот сака да купи билет за настан и за истиот треба да ги види достапните сали и venues.

3. Иницијално време

SELECT *

FROM view_event_halls

WHERE event_id = 12356;

-- Резултат: 256 ms

view_event_halls 1	
<div> select * from VIEW_EVENT_HALLS VEH where VEH.E </div>	
Grid	Text
123 event_id	abc title
123 hall_id	abc hall_name
123 capacity	allowed_access
abc venue_name	abc city
1	12,356 Amateur Color Run 2026
2	12,356 Amateur Color Run 2026
14,747	Hall 1
15,141	Hall 3
209	[]
80	[v]
	Axis Plaza I
	Axis Titan I
	Columbia
	Fayetteville
<div> Save Cancel Script </div> <div> 200 5 Rows: 1 5 row(s) fetched - 256ms, on 2026-05-15 at 12:37:02 </div>	

Ова е прифатливо време за апликацијата.

4. План на извршување

EXPLAIN ANALYZE SELECT * FROM view_event_halls WHERE event_id = 12356;

Results 1 ×	
EXPLAIN ANALYZE SELECT * FROM view_event_halls	
abc QUERY PLAN	
1	Nested Loop (cost=1.42..46.37 rows=3 width=88) (actual time=0.051..0.098 rows=5 loops=1)
2	-> Nested Loop (cost=1.13..45.37 rows=3 width=71) (actual time=0.042..0.068 rows=5 loops=1)
3	-> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=43) (actual time=0.018..0.018 rows=1 loops=1)
4	Index Cond: (id = 12356)
5	-> Nested Loop (cost=0.71..36.90 rows=3 width=36) (actual time=0.023..0.047 rows=5 loops=1)
6	-> Index Scan using event_hall_pkey on event_hall eh (cost=0.42..11.97 rows=3 width=17) (actual time=0.010..0.014 rows=5 loops=1)
7	Index Cond: (eventid = 12356)
8	-> Index Scan using hall_pkey on hall h (cost=0.29..8.31 rows=1 width=27) (actual time=0.005..0.005 rows=1 loops=5)
9	Index Cond: (id = eh.hallid)
10	-> Index Scan using venue_pkey on venue v (cost=0.29..0.33 rows=1 width=33) (actual time=0.005..0.005 rows=1 loops=5)
11	Index Cond: (id = h.venueid)
12	Planning Time: 0.470 ms
13	Execution Time: 0.147 ms

Планот на извршување покажува дека сите операции користат Index Scan - нема Full Scan на ниту една табела. Времето е прифатливо па нема потреба од индексирање.

5. Нема потреба од индексирање

6. Време на INSERT/UPDATE

Statistics 1 ×	
Name	Value
Updated Rows	1
Query	INSERT INTO hall (hall_name, capacity, venueid) VALUES ('Test Hall', 100, 1)
Finish time	Fri May 15 12:46:03 CEST 2026

Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 65ms

Statistics 1 ×	
Name	Value
Updated Rows	1
Query	UPDATE hall SET capacity = 150 WHERE id = (SELECT MAX(id) FROM hall)
Finish time	Fri May 15 12:46:34 CEST 2026

Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 198ms

Времето на операциите INSERT и UPDATE останува исто бидејќи нема нови индекси.

View 4: view_event_ticket_availability

1. Примарен филтер: Погледот примарно се филтрира според event_id.

2. Случај на употреба: Се користи кога корисникот има изберео сала и може да ги види достапните типови на тикети, нивните цени и преостанатите количини.

3. Иницијално време

SELECT * FROM view_event_ticket_availability WHERE event_id = 12455;

-- Резултат: 49.649s

view_event_ticket_availability 1							
SELECT * FROM view_event_ticket_availability WHERE Enter a SQL expression to filter results (use Ctrl+Space)							
Grid	123 event_id	123 ticket_type_id	abc ticket_type	123 price	123 quantity_available	123 sold	123 remaining
1	12,455	1	STANDARD	780	425	126	299
2	12,455	4	GENERAL_ADMISSION	346	345	0	345
Save Cancel Script 200 Rows: 1 2 row(s) fetched - 49.649s, c							

Ова време не е прифатливо за апликацијата па затоа пристапуваме кон оптимизација.

4. План на извршување

EXPLAIN ANALYZE

SELECT * FROM view_event_ticket_availability WHERE event_id = 12455;

Results 1	
EXPLAIN ANALYZE SELECT * FROM view_event_ticket Enter a SQL expression to filter results (use Ctrl+Space)	
Grid Text Record	QUERY PLAN
	1 GroupAggregate (cost=55839.84..165517.71 rows=3 width=158) (actual time=396.037..403.829 rows=2 loops=1)
	2 Group Key: tt.id, ett.price, ett.quantity_available
	3 -> Incremental Sort (cost=55839.84..165517.64 rows=3 width=156) (actual time=395.922..403.728 rows=187 loops=1)
	4 Sort Key: tt.id, ett.price, ett.quantity_available, tt.id
	5 Presorted Key: tt.id
	6 Full-sort Groups: 2 Sort Method: quicksort Average Memory: 29kB Peak Memory: 29kB
	7 Pre-sorted Groups: 1 Sort Method: quicksort Average Memory: 38kB Peak Memory: 38kB
	8 -> Nested Loop Left Join (cost=1001.00..165517.50 rows=3 width=156) (actual time=88.192..403.380 rows=187 loops=1)
	9 Join Filter: (t.ticket_typeid = tt.id)
	10 Rows Removed by Join Filter: 186
	11 -> Nested Loop (cost=1.00..36.46 rows=3 width=142) (actual time=18.900..18.939 rows=2 loops=1)
	12 -> Nested Loop (cost=0.84..20.45 rows=3 width=24) (actual time=18.788..18.803 rows=2 loops=1)
	13 -> Index Scan using event_ticket_type_pkey on event_ticket_type ett (cost=0.42..11.97 rows=3 width=24) (actual time=0.072..0.078 rows=2 loops=1)
	14 Index Cond: (eventid = 12455)
	15 -> Materialize (cost=0.42..8.44 rows=1 width=8) (actual time=0.032..0.036 rows=1 loops=2)
	16 -> Index Only Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=8) (actual time=0.061..0.066 rows=1 loops=1)
	17 Index Cond: (id = 12455)
	18 Heap Fetches: 0
	19 -> Memoize (cost=0.16..6.84 rows=1 width=126) (actual time=0.058..0.058 rows=1 loops=2)
	20 Cache Key: ett.ticket_typeid
	21 Cache Mode: logical
	22 Hits: 0 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 1kB
	23 -> Index Scan using ticket_type_pkey on ticket_type tt (cost=0.15..6.83 rows=1 width=126) (actual time=0.037..0.037 rows=1 loops=2)
	24 Index Cond: (id = ett.ticket_typeid)
	25 -> Materialize (cost=1000.00..165466.33 rows=346 width=30) (actual time=34.638..192.156 rows=186 loops=2)
	26 -> Gather (cost=1000.00..165464.60 rows=346 width=30) (actual time=69.270..384.196 rows=186 loops=1)
	27 Workers Planned: 4
	28 Workers Launched: 4
	29 -> Parallel Seq Scan on ticket t (cost=0.00..164430.00 rows=86 width=30) (actual time=35.506..335.954 rows=37 loops=5)
	30 Filter: (eventid = 12455)
	31 Rows Removed by Filter: 1999963
	32 Planning Time: 0.699 ms
	33 JIT:
	34 Functions: 43
	35 Options: Inlining false, Optimization false, Expressions true, Deforming true
	36 Timing: Generation 3.934 ms (Deform 2.017 ms), Inlining 0.000 ms, Optimization 2.590 ms, Emission 43.150 ms, Total 49.675 ms
	37 Execution Time: 405.761 ms

Најбавната операција е Parallel Seq Scan на табелата ticket која скенира ~2,000,000 редови за да најде само 186 тикети за тој настан. Оваа операција може да се подобри со индекс на колоната event_id.

INSERT / UPDATE пред индексирање

Name	Value
Updated Rows	1
Query	INSERT INTO ticket (code, status, ticket_typeid, user_orderid, seatid, app_userid, eventid, hallid) VALUES ('TKT-TEST-009', 'ACTIVE', 1, (SELECT MIN(id) FROM user_order), NULL, (SELECT MIN(id) FROM app_user WHERE id != 1), 12455, (SELECT MIN(hallid) FROM event_hall WHERE eventid = 12455))
Finish time	Fri May 15 13:30:32 CEST 2026

Save Cancel Script 200 1 Rows: 2 1 row(s) updated - 10ms,

Name	Value
Updated Rows	1
Query	-- UPDATE pred indeks UPDATE ticket SET status = 'CANCELLED' WHERE code = 'TKT-TEST-009'
Finish time	Fri May 15 13:31:43 CEST 2026

Save Cancel Script 200 1 Rows: 2 1 row(s) updated - 17ms,

5. Индекси и ново време

CREATE INDEX idx_ticket_eventid ON ticket(eventid);

SELECT * FROM view_event_ticket_availability WHERE event_id = 12455;

-- Резултат: 16ms

Grid	event_id	ticket_type_id	ticket_type	price	quantity_available	sold	remaining
1	12,455	1	STANDARD	780	425	126	299
2	12,455	4	GENERAL_ADMISSION	346	345	0	345

view_event_ticket_availability 1 x SELECT * FROM view_event_ticket_availability WHERE Enter a SQL expression to filter results (use Ctrl+Space) 200 2 Rows: 1 2 row(s) fetched - 16ms, on

План на извршување по додавање на индекс

EXPLAIN ANALYZE

SELECT * FROM view_event_ticket_availability WHERE event_id = 12455;

Results 1 ×	
EXPLAIN ANALYZE SELECT * FROM view_event_ticket	
QUERY PLAN	
1	GroupAggregate (cost=1380.38..1380.46 rows=3 width=158) (actual time=2.300..2.306 rows=2 loops=1)
2	Group Key: tt.id, ett.price, ett.quantity_available
3	-> Sort (cost=1380.38..1380.38 rows=3 width=156) (actual time=2.243..2.256 rows=187 loops=1)
4	Sort Key: tt.id, ett.price, ett.quantity_available, tt.id
5	Sort Method: quicksort Memory: 38kB
6	-> Hash Right Join (cost=43.62..1380.35 rows=3 width=156) (actual time=0.278..2.111 rows=187 loops=1)
7	Hash Cond: (t.ticket_typeid = tt.id)
8	-> Bitmap Heap Scan on ticket t (cost=7.12..1342.53 rows=346 width=30) (actual time=0.083..1.783 rows=186 loops=1)
9	Recheck Cond: (eventid = 12455)
10	Heap Blocks: exact=134
11	-> Bitmap Index Scan on idx_ticket_eventid (cost=0.00..7.03 rows=346 width=0) (actual time=0.050..0.050 rows=186 loops=1)
12	Index Cond: (eventid = 12455)
13	-> Hash (cost=36.46..36.46 rows=3 width=142) (actual time=0.185..0.187 rows=2 loops=1)
14	Buckets: 1024 Batches: 1 Memory Usage: 9kB
15	-> Nested Loop (cost=1.00..36.46 rows=3 width=142) (actual time=0.171..0.182 rows=2 loops=1)
16	-> Nested Loop (cost=0.84..20.45 rows=3 width=24) (actual time=0.134..0.140 rows=2 loops=1)
17	-> Index Scan using event_ticket_type_pkey on event_ticket_type ett (cost=0.42..11.97 rows=3 width=24) (actual time=0.075..0.076 rows=2 loops=1)
18	Index Cond: (eventid = 12455)
19	-> Materialize (cost=0.42..8.44 rows=1 width=8) (actual time=0.027..0.028 rows=1 loops=2)
20	-> Index Only Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=8) (actual time=0.051..0.052 rows=1 loops=1)
21	Index Cond: (id = 12455)
22	Heap Fetches: 0
23	-> Memoize (cost=0.16..6.84 rows=1 width=126) (actual time=0.019..0.020 rows=1 loops=2)
24	Cache Key: ett.ticket_typeid
25	Cache Mode: logical
26	Hits: 0 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 1kB
27	-> Index Scan using ticket_type_pkey on ticket_type tt (cost=0.15..6.83 rows=1 width=126) (actual time=0.016..0.016 rows=1 loops=2)
28	Index Cond: (id = ett.ticket_typeid)
29	Planning Time: 0.567 ms
30	Execution Time: 2.395 ms

Времето изминато во извршување со индекс изнесува 16ms и тоа е прифатливо. Наместо Parallel Seq Scan кој скенираше ~2,000,000 редови, сега се користи Bitmap Index Scan кој директно ги наоѓа потребните редови.

6. INSERT/UPDATE по индексирање

Statistics 1 ×	
Name	Value
Updated Rows	1
Query	INSERT INTO ticket (code, status, ticket_typeid, user_orderid, seatid, app_userid, eventid, hallid) VALUES ('TKT-TEST-007', 'ACTIVE', 1, (SELECT MIN(id) FROM user_order), NULL, (SELECT MIN(id) FROM app_user WHERE id != 1), 12455, (SELECT MIN(hallid) FROM event_hall WHERE eventid = 12455))
Finish time	Fri May 15 13:33:39 CEST 2026
Save Cancel Script 200 1 Rows: 2 1 row(s) updated - 15ms,	
Statistics 1 ×	
Name	Value
Updated Rows	1
Query	-- UPDATE по индекси UPDATE ticket SET status = 'CANCELLED' WHERE code = 'TKT-TEST-007'
Finish time	Fri May 15 13:34:27 CEST 2026
Save Cancel Script 200 1 Rows: 2 1 row(s) updated - 20ms,	

Разликата меѓу времињата пред и по индексирање е минимална и прифатлива.

View 5: view_ticket_status

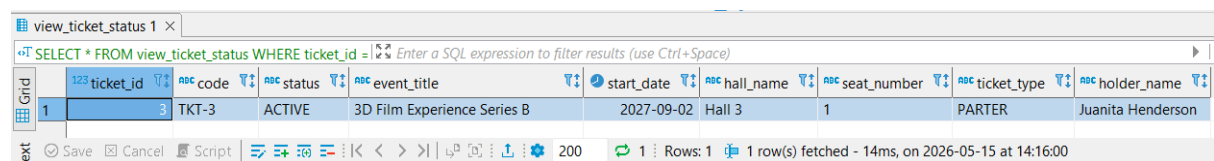
1. Примарен филтер: Погледот примарно се филтрира според ticket_id.

2. Случај на употреба: Се користи кога корисникот сака да ги види деталите за својот тикет – статус, настан, сала, седиште и тип на тикет.

3. Иницијално време

```
SELECT * FROM view_ticket_status WHERE ticket_id = 3;
```

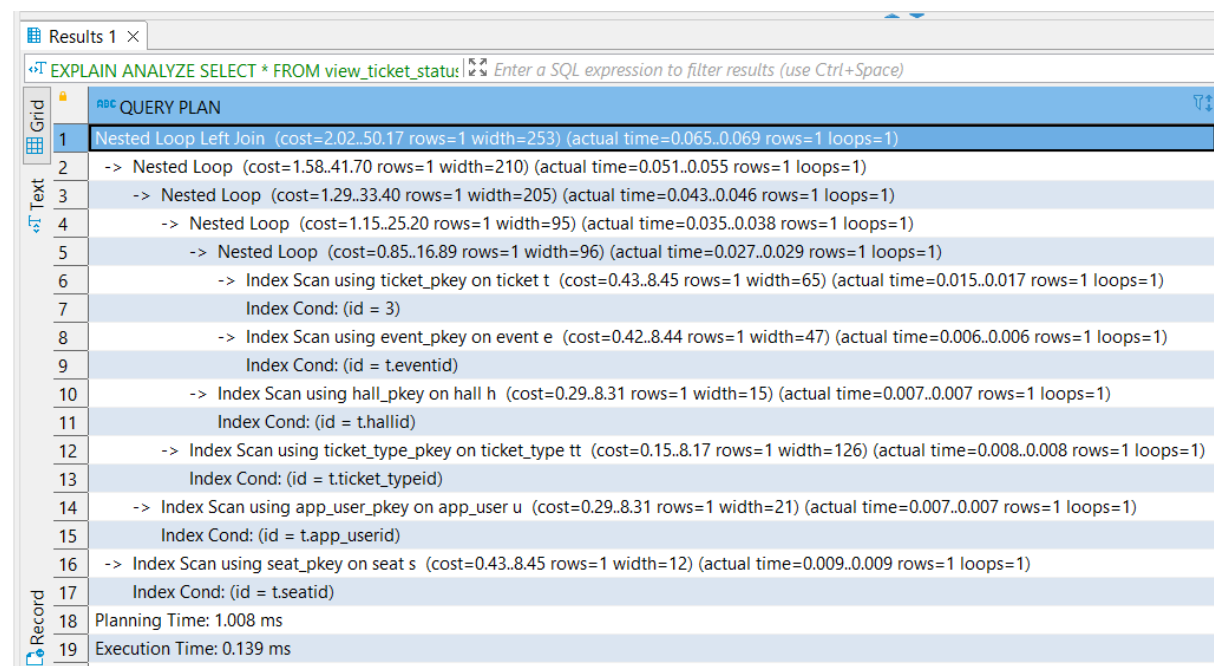
-- Резултат: 14ms



ticket_id	code	status	event_title	start_date	hall_name	seat_number	ticket_type	holder_name
3	TKT-3	ACTIVE	3D Film Experience Series B	2027-09-02	Hall 3	1	PARTER	Juanita Henderson

4. План на извршување

```
EXPLAIN ANALYZE SELECT * FROM view_ticket_status WHERE ticket_id = 3;
```



QUERY PLAN
1 Nested Loop Left Join (cost=2.02..50.17 rows=1 width=253) (actual time=0.065..0.069 rows=1 loops=1)
2 -> Nested Loop (cost=1.58..41.70 rows=1 width=210) (actual time=0.051..0.055 rows=1 loops=1)
3 -> Nested Loop (cost=1.29..33.40 rows=1 width=205) (actual time=0.043..0.046 rows=1 loops=1)
4 -> Nested Loop (cost=1.15..25.20 rows=1 width=95) (actual time=0.035..0.038 rows=1 loops=1)
5 -> Nested Loop (cost=0.85..16.89 rows=1 width=96) (actual time=0.027..0.029 rows=1 loops=1)
6 -> Index Scan using ticket_pkey on ticket t (cost=0.43..8.45 rows=1 width=65) (actual time=0.015..0.017 rows=1 loops=1)
7 Index Cond: (id = 3)
8 -> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=47) (actual time=0.006..0.006 rows=1 loops=1)
9 Index Cond: (id = t.eventid)
10 -> Index Scan using hall_pkey on hall h (cost=0.29..8.31 rows=1 width=15) (actual time=0.007..0.007 rows=1 loops=1)
11 Index Cond: (id = t.hallid)
12 -> Index Scan using ticket_type_pkey on ticket_type tt (cost=0.15..8.17 rows=1 width=126) (actual time=0.008..0.008 rows=1 loops=1)
13 Index Cond: (id = t.ticket_typeid)
14 -> Index Scan using app_user_pkey on app_user u (cost=0.29..8.31 rows=1 width=21) (actual time=0.007..0.007 rows=1 loops=1)
15 Index Cond: (id = t.app_userid)
16 -> Index Scan using seat_pkey on seat s (cost=0.43..8.45 rows=1 width=12) (actual time=0.009..0.009 rows=1 loops=1)
17 Index Cond: (id = t.seatid)
18 Planning Time: 1.008 ms
19 Execution Time: 0.139 ms

Планот на извршување покажува дека сите операции користат Index Scan – нема Full Scan на ниту една табела. Времето е прифатливо па нема потреба од индексирање.

5. Нема потреба од индексирање

6. Време на INSERT/UPDATE

Name	Value
Updated Rows	1
Query	INSERT INTO ticket (code, status, ticket_typeid, user_orderid, seatid, app_userid, eventid, hallid) VALUES ('TKT-TEST-004', 'ACTIVE', 1, (SELECT MIN(id) FROM user_order), NULL, (SELECT MIN(id) FROM app_user WHERE id != 1), 12455, (SELECT MIN(hallid) FROM event_hall WHERE eventid = 12455))
Finish time	Fri May 15 14:30:16 CEST 2026

Save Cancel Script 200 1 Rows: 19 1 row(s) updated - 16ms,

Name	Value
Updated Rows	1
Query	UPDATE ticket SET status = 'USED' WHERE code = 'TKT-TEST-004'
Finish time	Fri May 15 14:30:40 CEST 2026

Save Cancel Script 200 1 Rows: 19 1 row(s) updated - 0ms,

Времето на операциите INSERT и UPDATE останува исто бидејќи нема нови индекси.

View 6: view_event_reviews

- Примарен филтер:** Погледот примарно се филтрира според event_id.
- Случај на употреба:** Се користи за приказ на рецензии и оценки за минат настан.
- Иницијално време:**

SELECT * FROM view_event_reviews WHERE event_id = 12345;

-- Резултат: 2m 54s

Grid	event_id	event_title	start_date	review_id	user_name	rating	review_comment
7	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	362,327	Mable Bloom	2	Forgettable.
8	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	363,137	Al Jones	5	The production quality was exceptional and clearly reflected a significant investment.
9	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	363,931	Clair Palmer	5	The attention to detail was remarkable. You could tell how much effort went in.
10	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	364,717	Anonymous	4	Flawless organization, incredible atmosphere, top-tier performances.
11	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	365,569	Virginia Cassidy	4	Thoroughly enjoyed the evening. Will attend again next year.
12	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	366,350	Maudie Pate	3	Middle of the road experience. Some aspects were great, others less so.
13	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	367,201	Pasquale Welch	2	Felt like the organizers cut corners in too many places.
14	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	367,965	Arlene Terry	5	[NULL]

Save Cancel Script 200 82 Rows: 1 82 row(s) fetched - 2m 54s (2ms fetch), on 2026-05-15 at 14:40:

Ова не е прифатливо време за апликацијата.

4. План на извршување

EXPLAIN ANALYZE SELECT * FROM view_event_reviews WHERE event_id = 12345;

Results 1 ×	
EXPLAIN ANALYZE SELECT * FROM view_event_review Enter a SQL expression to filter results (use Ctrl+Space)	
QUERY PLAN	
1	Nested Loop Left Join (cost=1000.71..170568.83 rows=145 width=146) (actual time=51.367..365.627 rows=82 loops=1)
2	-> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=47) (actual time=0.016..0.022 rows=1 loops=1)
3	Index Cond: (id = 12345)
4	-> Gather (cost=1000.29..170558.22 rows=145 width=88) (actual time=51.340..365.556 rows=82 loops=1)
5	Workers Planned: 4
6	Workers Launched: 4
7	-> Nested Loop Left Join (cost=0.29..169543.72 rows=36 width=88) (actual time=19.347..322.440 rows=16 loops=5)
8	-> Parallel Seq Scan on review r (cost=0.00..169307.21 rows=36 width=83) (actual time=19.267..322.265 rows=16 loops=5)
9	Filter: (eventid = 12345)
10	Rows Removed by Filter: 2289917
11	-> Index Scan using app_user_pkey on app_user u (cost=0.29..6.57 rows=1 width=21) (actual time=0.006..0.006 rows=1 loops=82)
12	Index Cond: (id = r.app_userid)
13	Planning Time: 0.540 ms
14	JIT:
15	Functions: 44
16	Options: Inlining false, Optimization false, Expressions true, Deforming true
17	Timing: Generation 4.260 ms (Deform 2.422 ms), Inlining 0.000 ms, Optimization 2.386 ms, Emission 44.274 ms, Total 50.920 ms
18	Execution Time: 366.636 ms

Најбавната операција е Parallel Seq Scan на табелата review која скенира ~ 2 милиони редови за да најде само 82 рецензии за тој настан. Оваа операција може да се подобри со индекс на колоната EVENTid.

INSERT / UPDATE пред индексирање

Statistics 1 ×	
Name	Value
Updated Rows	1
Query	INSERT INTO review (rating, review_comment, app_userid, eventid) VALUES (5, 'Test review komentar', 28547, 33)
Finish time	Fri May 15 15:03:17 CEST 2026
Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 777ms,	

Statistics 1 ×	
Name	Value
Updated Rows	1
Query	UPDATE review SET rating = 4 WHERE review_comment = 'Test review komentar'
Finish time	Fri May 15 15:04:42 CEST 2026
Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 1.483s	

5. Индекси и ново време

```
CREATE INDEX idx_review_eventid ON REVIEW(EVENTid);
```

```
SELECT * FROM view_event_reviews WHERE event_id = 12345;
```

-- Резултат: 19ms

view_event_reviews 1 x							
SELECT * FROM view_event_reviews WHERE event_id = 123 Enter a SQL expression to filter results (use Ctrl+Space)							
	event_id	event_title	start_date	review_id	user_name	rating	review_comment
7	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	362,327	Mable Bloom	2	Forgettable.
8	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	363,137	Al Jones	5	The production quality was exceptional and clearly reflected a significant investment.
9	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	363,931	Clair Palmer	5	The attention to detail was remarkable. You could tell how much effort went in.
10	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	364,717	Anonymous	4	Flawless organization, incredible atmosphere, top-tier performances.
11	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	365,569	Virginia Cassidy	4	Thoroughly enjoyed the evening. Will attend again next year.
12	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	366,350	Maudie Pate	3	Middle of the road experience. Some aspects were great, others less so.
13	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	367,201	Pasquale Welch	2	Felt like the organizers cut corners in too many places.
14	12,345	Amateur Collector Showcase Vol. 2	2024-10-26	367,965	Arlene Terry	5	[NULL]

Времето изминато во извршување со индекс изнесува 19ms и тоа е прифатливо.

EXPLAIN ANALYZE по индексирање

Results 1 x	
EXPLAIN ANALYZE SELECT * FROM view_event_review Enter a SQL expression to filter results (use Ctrl+Space)	
ABC QUERY PLAN	
1	Nested Loop Left Join (cost=1.14..998.31 rows=145 width=146) (actual time=0.043..0.388 rows=82 loops=1)
2	-> Nested Loop Left Join (cost=0.85..44.99 rows=145 width=122) (actual time=0.031..0.186 rows=82 loops=1)
3	-> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=47) (actual time=0.015..0.016 rows=1 loops=1)
4	Index Cond: (id = 12345)
5	-> Index Scan using idx_review_eventid on review r (cost=0.43..35.11 rows=145 width=83) (actual time=0.012..0.150 rows=82 loops=1)
6	Index Cond: (eventid = 12345)
7	-> Index Scan using app_user_pkey on app_user u (cost=0.29..6.57 rows=1 width=21) (actual time=0.002..0.002 rows=1 loops=82)
8	Index Cond: (id = r.app_userid)
9	Planning Time: 0.463 ms
10	Execution Time: 0.435 ms

Наместо Parallel Seq Scan сега се користи Index Scan on idx_review_eventid - директно ги наоѓа потребните редови.

6. Време на INSERT/UPDATE по индексирање

Statistics 1 x	
Name	Value
Updated Rows	1
Query	INSERT INTO review (rating, review_comment, app_userid, eventid) VALUES (5, 'Test review komentar 2', 28547, 33)
Finish time	Fri May 15 15:07:17 CEST 2026

Statistics 1 x	
Name	Value
Updated Rows	1
Query	UPDATE review SET rating = 3 WHERE review_comment = 'Test review komentar 2'
Finish time	Fri May 15 15:07:39 CEST 2026

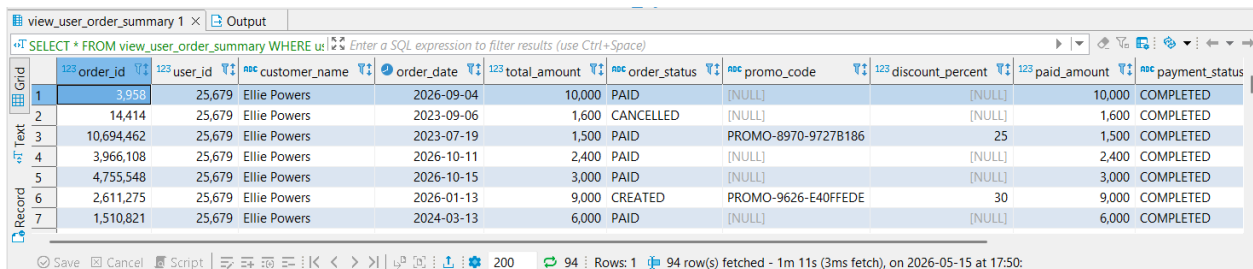
Разликата меѓу времињата пред и по индексот е минимална и прифатлива.

View 7: view_user_order_summary

- Примарен филтер:** Погледот примарно се филтрира според `user_id`, а може дополнително да се филтрира и по `order_date`.
- Случај на употреба:** Се користи за приказ на целосната историја на нарачки по корисник – нарачки, плаќања, тикети и промо кодови.
- Иницијално време:**

```
SELECT * FROM view_user_order_summary WHERE user_id = 25679;
```

-- Резултат: 1m 11s



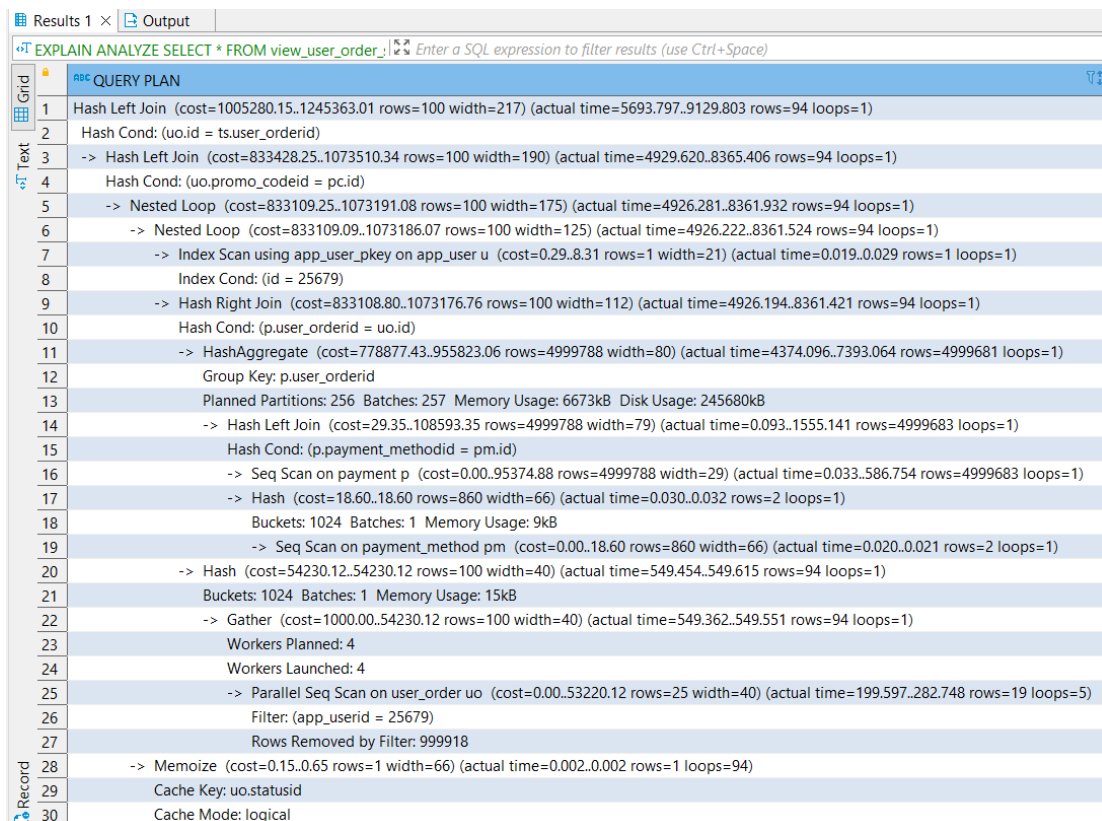
	order_id	user_id	customer_name	order_date	total_amount	order_status	promo_code	discount_percent	paid_amount	payment_status
1	3,958	25,679	Ellie Powers	2026-09-04	10,000	PAID	[NULL]	[NULL]	10,000	COMPLETED
2	14,414	25,679	Ellie Powers	2023-09-06	1,600	CANCELLED	[NULL]	[NULL]	1,600	COMPLETED
3	10,694,462	25,679	Ellie Powers	2023-07-19	1,500	PAID	PROMO-8970-9727B186	25	1,500	COMPLETED
4	3,966,108	25,679	Ellie Powers	2026-10-11	2,400	PAID	[NULL]	[NULL]	2,400	COMPLETED
5	4,755,548	25,679	Ellie Powers	2026-10-15	3,000	PAID	[NULL]	[NULL]	3,000	COMPLETED
6	2,611,275	25,679	Ellie Powers	2026-01-13	9,000	CREATED	PROMO-9626-E40FFEDE	30	9,000	COMPLETED
7	1,510,821	25,679	Ellie Powers	2024-03-13	6,000	PAID	[NULL]	[NULL]	6,000	COMPLETED

Ова не е прифатливо време за апликацијата.

4. План на извршување

EXPLAIN ANALYZE

```
SELECT * FROM view_user_order_summary WHERE user_id = 25679;
```



	QUERY PLAN
1	Hash Left Join (cost=1005280.15..1245363.01 rows=100 width=217) (actual time=5693.797..9129.803 rows=94 loops=1)
2	Hash Cond: (uo.id = ts.user_orderid)
3	-> Hash Left Join (cost=833428.25..1073510.34 rows=100 width=190) (actual time=4929.620..8365.406 rows=94 loops=1)
4	Hash Cond: (uo.promo_codeid = pc.id)
5	-> Nested Loop (cost=833109.25..1073191.08 rows=100 width=175) (actual time=4926.281..8361.932 rows=94 loops=1)
6	-> Nested Loop (cost=833109.09..1073186.07 rows=100 width=125) (actual time=4926.222..8361.524 rows=94 loops=1)
7	-> Index Scan using app_user_pkey on app_user u (cost=0.29..8.31 rows=1 width=21) (actual time=0.019..0.029 rows=1 loops=1)
8	Index Cond: (id = 25679)
9	-> Hash Right Join (cost=833108.80..1073176.76 rows=100 width=112) (actual time=4926.194..8361.421 rows=94 loops=1)
10	Hash Cond: (p.user_orderid = uo.id)
11	-> HashAggregate (cost=778877.43..955823.06 rows=4999788 width=80) (actual time=4374.096..7393.064 rows=4999681 loops=1)
12	Group Key: p.user_orderid
13	Planned Partitions: 256 Batches: 257 Memory Usage: 6673kB Disk Usage: 245680kB
14	-> Hash Left Join (cost=29.35..108593.35 rows=4999788 width=79) (actual time=0.093..1555.141 rows=4999683 loops=1)
15	Hash Cond: (p.payment_methodid = pm.id)
16	-> Seq Scan on payment p (cost=0.00..95374.88 rows=4999788 width=29) (actual time=0.033..586.754 rows=4999683 loops=1)
17	-> Hash (cost=18.60..18.60 rows=860 width=66) (actual time=0.030..0.032 rows=2 loops=1)
18	Buckets: 1024 Batches: 1 Memory Usage: 9kB
19	-> Seq Scan on payment_method pm (cost=0.00..18.60 rows=860 width=66) (actual time=0.020..0.021 rows=2 loops=1)
20	-> Hash (cost=54230.12..54230.12 rows=100 width=40) (actual time=549.454..549.615 rows=94 loops=1)
21	Buckets: 1024 Batches: 1 Memory Usage: 15kB
22	-> Gather (cost=1000.00..54230.12 rows=100 width=40) (actual time=549.362..549.551 rows=94 loops=1)
23	Workers Planned: 4
24	Workers Launched: 4
25	-> Parallel Seq Scan on user_order uo (cost=0.00..53220.12 rows=25 width=40) (actual time=199.597..282.748 rows=19 loops=5)
26	Filter: (app_userid = 25679)
27	Rows Removed by Filter: 999918
28	-> Memoize (cost=0.15..0.65 rows=1 width=66) (actual time=0.002..0.002 rows=1 loops=94)
29	Cache Key: uo.statusid
30	Cache Mode: logical

31	Hits: 91 Misses: 3 Evictions: 0 Overflows: 0 Memory Usage: 1kB
32	-> Index Scan using status_pkey on status s (cost=0.14..0.64 rows=1 width=66) (actual time=0.022..0.022 rows=1 loops=3)
33	Index Cond: (id = uo.statusid)
34	-> Hash (cost=194.00..194.00 rows=10000 width=31) (actual time=3.284..3.284 rows=10000 loops=1)
35	Buckets: 16384 Batches: 1 Memory Usage: 754kB
36	-> Seq Scan on promo_code pc (cost=0.00..194.00 rows=10000 width=31) (actual time=0.025..1.279 rows=10000 loops=1)
37	-> Hash (cost=171848.21..171848.21 rows=295 width=16) (actual time=764.151..764.237 rows=302 loops=1)
38	Buckets: 1024 Batches: 1 Memory Usage: 23kB
39	-> Subquery Scan on ts (cost=171695.13..171848.21 rows=295 width=16) (actual time=763.587..764.138 rows=302 loops=1)
40	-> Finalize GroupAggregate (cost=171695.13..171845.26 rows=295 width=16) (actual time=763.580..764.097 rows=302 loops=1)
41	Group Key: ticket.user_orderid
42	-> Gather Merge (cost=171695.13..171836.41 rows=1180 width=16) (actual time=763.545..763.918 rows=1264 loops=1)
43	Workers Planned: 4
44	Workers Launched: 4
45	-> Sort (cost=170695.07..170695.81 rows=295 width=16) (actual time=729.960..729.976 rows=253 loops=5)
46	Sort Key: ticket.user_orderid
47	Sort Method: quicksort Memory: 32kB
48	Worker 0: Sort Method: quicksort Memory: 32kB
49	Worker 1: Sort Method: quicksort Memory: 33kB
50	Worker 2: Sort Method: quicksort Memory: 32kB
51	Worker 3: Sort Method: quicksort Memory: 32kB
52	-> Partial HashAggregate (cost=170680.02..170682.97 rows=295 width=16) (actual time=729.793..729.845 rows=253 loops=5)
53	Group Key: ticket.user_orderid
54	Batches: 1 Memory Usage: 61kB
55	Worker 0: Batches: 1 Memory Usage: 61kB
56	Worker 1: Batches: 1 Memory Usage: 61kB
57	Worker 2: Batches: 1 Memory Usage: 61kB
58	Worker 3: Batches: 1 Memory Usage: 45kB
59	-> Parallel Seq Scan on ticket (cost=0.00..158180.01 rows=2500001 width=8) (actual time=0.050..251.989 rows=2000002 loops=5)
60	Planning Time: 1.424 ms
61	JIT:
62	Functions: 91
63	Options: Inlining true, Optimization true, Expressions true, Deforming true
64	Timing: Generation 7.799 ms (Deform 3.072 ms), Inlining 885.564 ms, Optimization 576.982 ms, Emission 405.442 ms, Total 1875.787 ms
65	Execution Time: 9790.351 ms

Најбавните операции се:

- Seq Scan on payment – ги скенира сите ~5,000,000 плаќања и ги агрегира сите по user_orderid пред да ги филтрира
- Parallel Seq Scan on ticket – ги скенира сите ~10,000,000 тикети и ги агрегира сите по user_orderid пред да ги филтрира
- Parallel Seq Scan on user_order – ги скенира ~5,000,000 нарачки за да најде само 94 нарачки по корисник

Проблемот е во структурата на оригиналниот поглед со користење на subqueries кои ги агрегираат сите редови од TICKET и PAYMENT пред да ги филтрираат.

Бидејќи станува збор за аналитички прашалник, индекси не можат да се употребат за подобрување на перформансите, па затоа пристапуваме кон преуредување на прашалникот.

INSERT/UPDATE пред преуредување

Name	Value
Updated Rows	1
Query	INSERT INTO user_order (order_date, total_amount, app_userid, statusid) VALUES (CURRENT_DATE, 1000, 25679, 1)
Finish time	Fri May 15 17:57:37 CEST 2026

Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 66ms

Name	Value
Updated Rows	1
Query	UPDATE user_order SET total_amount = 1500 WHERE app_userid = 25679 AND order_date = CURRENT_DATE
Finish time	Fri May 15 17:58:15 CEST 2026

Save
Cancel
Script
SQL Editor
200
1
Rows: 1
1 row(s) updated - 478ms

5. Преуредување на прашалникот

Оригинална форма на прашалникот:

```

CREATE OR REPLACE VIEW view_user_order_summary AS
SELECT
    uo.id AS order_id,
    u.id AS user_id,
    u.first_name || ' ' || u.last_name AS customer_name,
    uo.order_date,
    uo.total_amount,
    s.status_name AS order_status,
    pc.code AS promo_code,
    pc.discount_percent,
    ps.paid_amount,
    ps.payment_status,
    ps.payment_method,
    ts.ticket_count
FROM USER_ORDER uo
JOIN APP_USER u ON u.id = uo.APP_USERid
JOIN STATUS s ON s.id = uo.STATUSid
LEFT JOIN PROMO_CODE pc ON pc.id = uo.PROMO_CODEid
LEFT JOIN (
    SELECT user_orderid, COUNT(*) AS ticket_count
    FROM ticket
    GROUP BY user_orderid
) ts ON ts.user_orderid = uo.id
LEFT JOIN (
    SELECT
        p.user_orderid,
        SUM(p.amount) FILTER (WHERE p.status = 'COMPLETED') AS paid_amount,
        MAX(p.status) AS payment_status,
        MAX(pm.method_name) AS payment_method
    FROM payment p
    LEFT JOIN payment_method pm ON pm.id = p.payment_methodid
    GROUP BY p.user_orderid
) ps ON ps.user_orderid = uo.id;

SELECT * FROM view_user_order_summary WHERE user_id = 25679;
-- Результат 1мин 11с

```

Преуреден прашалник:

```

CREATE OR REPLACE VIEW view_user_order_summary AS
SELECT
    uo.id                                AS order_id,
    u.id                                AS user_id,
    u.first_name || ' ' || u.last_name AS customer_name,
    uo.order_date,
    uo.total_amount,
    s.status_name                        AS order_status,
    pc.code                             AS promo_code,
    pc.discount_percent,
    SUM(p.amount) FILTER (WHERE p.status = 'COMPLETED') AS paid_amount,
    MAX(p.status)                        AS payment_status,
    MAX(pm.method_name)                  AS payment_method,
    COUNT(DISTINCT t.id)                 AS ticket_count
FROM USER_ORDER uo
JOIN APP_USER u      ON u.id = uo.APP_USERId
JOIN STATUS s        ON s.id = uo.STATUSId
LEFT JOIN PROMO_CODE pc ON pc.id = uo.PROMO_CODEId
LEFT JOIN PAYMENT p   ON p.user_orderid = uo.id
LEFT JOIN PAYMENT_METHOD pm ON pm.id = p.payment_methodid
LEFT JOIN TICKET t    ON t.user_orderid = uo.id
GROUP BY uo.id, u.id, u.first_name, u.last_name,
         uo.order_date, uo.total_amount, s.status_name,
         pc.code, pc.discount_percent;

SELECT * FROM view_user_order_summary WHERE user_id = 25679;
-- Результат 3.135с

```


Со преуредувањето PostgreSQL прво ги наоѓа нарачките за тој корисник па потоа ги агрегира само нивните тикети и плаќања - наместо сите.

--Резултат: 3.135s

view_user_order_summary 1 × Output										
SELECT * FROM view_user_order_summary WHERE u... Enter a SQL expression to filter results (use Ctrl+Space)										
	order_id	user_id	customer_name	order_date	total_amount	order_status	promo_code	discount_percent	paid_amount	payment_status
1	3,958	25,679	Ellie Powers	2026-09-04	10,000	PAID	[NULL]	[NULL]	10,000	COMPLETED
2	14,414	25,679	Ellie Powers	2023-09-06	1,600	CANCELLED	[NULL]	[NULL]	1,600	COMPLETED
3	107,661	25,679	Ellie Powers	2025-02-09	3,000	PAID	PROMO-3011-8950C0B3	50	3,000	COMPLETED
4	325,867	25,679	Ellie Powers	2026-06-25	3,600	CREATED	PROMO-9387-E75D3912	25	[NULL]	PENDING
5	483,028	25,679	Ellie Powers	2026-10-15	9,000	CREATED	[NULL]	[NULL]	[NULL]	PENDING

Save Cancel Script 200 95 Rows: 1 95 row(s) fetched - 3.135s, on 2026-05-15 at 18:29:45
CET en_US Writable Smart Insert 417 : 1 : 12983 Sel: 0 | 0

EXPLAIN ANALYZE по преуредување

Results 1 × Output	
EXPLAIN ANALYZE SELECT * FROM view_user_order... Enter a SQL expression to filter results (use Ctrl+Space)	
Grid Text Record	QUERY PLAN
	1 Subquery Scan on view_user_order_summary (cost=281741.77..281785.74 rows=200 width=217) (actual time=920.848..930.331 rows=95 loops=1)
	2 -> GroupAggregate (cost=281741.77..281783.74 rows=200 width=230) (actual time=920.840..930.312 rows=95 loops=1)
	3 Group Key: uo.id, s.status_name, pc.code, pc.discount_percent
	4 -> Nested Loop (cost=281741.77..281776.24 rows=200 width=197) (actual time=920.804..930.178 rows=95 loops=1)
	5 -> Gather Merge (cost=281741.48..281765.43 rows=200 width=184) (actual time=920.730..930.054 rows=95 loops=1)
	6 Workers Planned: 4
	7 Workers Launched: 4
	8 -> Sort (cost=280741.42..280741.55 rows=50 width=184) (actual time=882.919..882.933 rows=19 loops=5)
	9 Sort Key: uo.id, s.status_name, pc.code, pc.discount_percent, tid
	10 Sort Method: quicksort Memory: 26kB
	11 Worker 0: Sort Method: quicksort Memory: 26kB
	12 Worker 1: Sort Method: quicksort Memory: 26kB
	13 Worker 2: Sort Method: quicksort Memory: 27kB
	14 Worker 3: Sort Method: quicksort Memory: 26kB
	15 -> Nested Loop Left Join (cost=218174.54..280740.01 rows=50 width=184) (actual time=647.783..882.803 rows=19 loops=5)
	16 -> Parallel Hash Right Join (cost=218174.38..280738.27 rows=50 width=134) (actual time=647.708..882.647 rows=19 loops=5)
	17 Hash Cond: (p.user_orderid = uo.id)
	18 -> Parallel Seq Scan on payment p (cost=0.00..57876.47 rows=1249947 width=29) (actual time=0.037..103.457 rows=999937 loops=5)
	19 -> Parallel Hash (cost=218173.75..218173.75 rows=50 width=113) (actual time=643.933..643.940 rows=19 loops=5)
	20 Buckets: 1024 Batches: 1 Memory Usage: 40kB
	21 -> Nested Loop Left Join (cost=53220.76..218173.75 rows=50 width=113) (actual time=643.333..643.398 rows=19 loops=5)
	22 -> Nested Loop (cost=53220.48..217966.63 rows=50 width=98) (actual time=643.322..643.349 rows=19 loops=5)
	23 -> Parallel Hash Right Join (cost=53220.32..217962.85 rows=50 width=48) (actual time=643.305..643.318 rows=19 loops=5)
	24 Hash Cond: (t.user_orderid = uo.id)
	25 -> Parallel Seq Scan on ticket t (cost=0.00..158180.01 rows=2500001 width=16) (actual time=0.065..250.768 rows=2000002 loops=5)
	26 -> Parallel Hash (cost=53220.01..53220.01 rows=25 width=40) (actual time=123.675..123.676 rows=19 loops=5)
	27 Buckets: 1024 Batches: 1 Memory Usage: 168kB
	28 -> Parallel Seq Scan on user_order uo (cost=0.00..53220.01 rows=25 width=40) (actual time=36.295..123.511 rows=19 loops=5)
	29 Filter: (app_userid = 25679)
	30 Rows Removed by Filter: 999918
	31 -> Memoize (cost=0.15..0.65 rows=1 width=66) (actual time=0.001..0.001 rows=1 loops=95)
	32 Cache Key: uo.statusid
	33 Cache Mode: logical
	34 Worker 1: Hits: 92 Misses: 3 Evictions: 0 Overflows: 0 Memory Usage: 1kB
	35 -> Index Scan using status_pkey on status s (cost=0.14..0.64 rows=1 width=66) (actual time=0.020..0.020 rows=1 loops=3)
	36 Index Cond: (id = uo.statusid)
	37 -> Index Scan using promo_code_pkey on promo_code pc (cost=0.29..4.14 rows=1 width=31) (actual time=0.002..0.002 rows=0 loops=95)
	38 Index Cond: (id = uo.promo_codeid)
	39 -> Memoize (cost=0.16..0.18 rows=1 width=66) (actual time=0.006..0.006 rows=1 loops=95)
	40 Cache Key: p.payment_methodid
	41 Cache Mode: logical
	42 Hits: 16 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 1kB
	43 Worker 0: Hits: 18 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 1kB
	44 Worker 1: Hits: 13 Misses: 3 Evictions: 0 Overflows: 0 Memory Usage: 1kB
	45 Worker 2: Hits: 23 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 1kB
	46 Worker 3: Hits: 14 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 1kB
	47 -> Index Scan using payment_method_pkey on payment_method pm (cost=0.15..0.17 rows=1 width=66) (actual time=0.022..0.022 rows=1 loops=1)
	48 Index Cond: (id = p.payment_methodid)
	49 -> Materialize (cost=0.29..8.31 rows=1 width=21) (actual time=0.001..0.001 rows=1 loops=95)
	50 -> Index Scan using app_user_pkey on app_user u (cost=0.29..8.31 rows=1 width=21) (actual time=0.057..0.059 rows=1 loops=1)
	51 Index Cond: (id = 25679)
	52 Planning Time: 2.230 ms
	53 JIT:
	54 Functions: 240
	55 Options: Inlining false, Optimization false, Expressions true, Deforming true
	56 Timing: Generation 18.722 ms (Deform 9.436 ms), Inlining 0.000 ms, Optimization 6.846 ms, Emission 153.214 ms, Total 178.783 ms
	57 Execution Time: 935.174 ms

И по преуредувањето сè уште постојат Parallel Seq Scan на ticket и payment бидејќи PostgreSQL мора да ги провери сите редови за да ги поврзе со нарачките на корисникот. Ова е карактеристика на аналитички прашалници кои користат агрегатни функции врз големи табели.

Времето по преуредување е ~3 секунди, кое е подобрување во споредба со оригиналното, но сè уште над прифатливото. Бидејќи погледот користи агрегатни функции врз табели со милиони редови, не можат да се употребат индекси за дополнително подобрување на перформансите. Ова е очекувано однесување за аналитички прашалници од ваков тип.

6. Време на INSERT/UPDATE

Name	Value
Updated Rows	1
Query	INSERT INTO user_order (order_date, total_amount, app_userid, statusid) VALUES (CURRENT_DATE, 1000, 25679, 1)
Finish time	Fri May 15 18:46:41 CEST 2026

Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 15ms

Name	Value
Updated Rows	2
Query	UPDATE user_order SET total_amount = 1500 WHERE app_userid = 25679 AND order_date = CURRENT_DATE
Finish time	Fri May 15 18:47:01 CEST 2026

Save Cancel Script 200 1 Rows: 1 2 row(s) updated - 624ms

Времето на операциите INSERT и UPDATE останува исто бидејќи нема нови индекси.

View 8: view_user_subscriptions_feed

1. Примарен филтер: Погледот примарно се филтрира според user_id, а исто така може да се пребарува и според category и start_date.

2. Случај на употреба: Се користи за приказ на идни настани само од категориите и подкатегиите на кои е претплатен корисникот. Перформансите се важни бидејќи овој поглед се вчитува при секое отворање на персонализираниот feed на корисникот.

3. Иницијално време:

```
SELECT * FROM view_user_subscriptions_feed WHERE user_id = 24589;
```

-- Резултат: 37ms

view_user_subscriptions_feed 1 × Output									
select * from view_user_subscriptions_feed vusf where Enter a SQL expression to filter results (use Ctrl+Space)									
	asc user_id	asc email	asc event_id	asc event_title	asc start_date	asc category	asc subcategory	asc subscription_type	
1	24,589	Esperanza.Shapiro1302@gmail.com	211	3D Printing Conference City Center	2026-05-16	Exhibitions	Art	CATEGORY	
2	24,589	Esperanza.Shapiro1302@gmail.com	212	3D Printing Conference Collectors Edition	2026-05-27	Exhibitions	Art	CATEGORY	
3	24,589	Esperanza.Shapiro1302@gmail.com	213	3D Printing Conference Debut	2026-06-07	Exhibitions	Art	CATEGORY	
4	24,589	Esperanza.Shapiro1302@gmail.com	214	3D Printing Conference Deluxe Edition	2026-06-18	Exhibitions	Science	CATEGORY	
5	24,589	Esperanza.Shapiro1302@gmail.com	215	3D Printing Conference Downtown	2026-06-29	Exhibitions	Science	CATEGORY	
6	24,589	Esperanza.Shapiro1302@gmail.com	216	3D Printing Conference Edition I	2026-07-10	Exhibitions	Art	CATEGORY	

Ова е прифатливо време за апликацијата.

4. План на извршување

EXPLAIN ANALYZE SELECT * FROM view_user_subscriptions_feed WHERE user_id = 24589;

Results 1 × Output	
EXPLAIN ANALYZE SELECT * FROM view_user_subscri Enter a SQL expression to filter results (use Ctrl+Space)	
Grid	QUERY PLAN
1	Unique (cost=8791.63..8797.54 rows=263 width=564) (actual time=24.236..26.368 rows=5680 loops=1)
2	Planning Time: 2.092 ms
3	Execution Time: 26.775 ms
4	-> Sort (cost=8791.63..8792.28 rows=263 width=564) (actual time=24.234..24.654 rows=5680 loops=1)
5	Sort Method: quicksort Memory: 966kB
6	Sort Key: u.id, u.email, e.id, e.title, e.start_date, c.category_name, sub.subcategory_name, ('SUBCATEGORY':text)
7	-> Gather (cost=1028.90..8781.05 rows=263 width=564) (actual time=3.506..16.761 rows=5680 loops=1)
8	Workers Planned: 2
9	Workers Launched: 2
10	-> Parallel Append (cost=28.90..7754.75 rows=109 width=564) (actual time=1.070..10.094 rows=1893 loops=3)
11	-> Nested Loop Left Join (cost=28.90..3878.88 rows=57 width=290) (actual time=0.999..9.882 rows=1893 loops=3)
12	-> Hash Join (cost=30.00..3875.33 rows=98 width=290) (actual time=0.209..0.212 rows=0 loops=1)
13	Hash Cond: (e.subcategoryid = sub.id)
14	-> Parallel Seq Scan on event e (cost=0.00..3783.35 rows=16315 width=55) (actual time=0.023..0.023 rows=1 loops=1)
15	-> Memoize (cost=0.16..0.18 rows=1 width=126) (actual time=0.000..0.000 rows=1 loops=5680)
16	-> Hash Join (cost=28.75..3873.85 rows=57 width=148) (actual time=0.967..8.947 rows=1893 loops=3)
17	-> Hash (cost=29.96..29.96 rows=3 width=227) (actual time=0.164..0.166 rows=0 loops=1)
18	Worker 1: Hits: 1521 Misses: 4 Evictions: 0 Overflows: 0 Memory Usage: 1kB
19	Worker 0: Hits: 1566 Misses: 4 Evictions: 0 Overflows: 0 Memory Usage: 1kB
20	Hits: 2581 Misses: 4 Evictions: 0 Overflows: 0 Memory Usage: 1kB
21	Hash Cond: (e_1.categorizationid = c_1.id)
22	Filter: (start_date >= CURRENT_DATE)
23	Cache Mode: logical
24	Cache Key: e_1.subcategoryid
25	Buckets: 1024 Batches: 1 Memory Usage: 8kB
26	-> Parallel Seq Scan on event e_1 (cost=0.00..3783.35 rows=16315 width=63) (actual time=0.019..7.371 rows=9159 loops=3)
27	-> Nested Loop (cost=4.95..29.96 rows=3 width=227) (actual time=0.163..0.165 rows=0 loops=1)
28	-> Index Scan using subcategory_pkey on subcategory sub_1 (cost=0.15..0.17 rows=1 width=126) (actual time=0.008..0.008 rows=1 loops=12)
29	-> Hash (cost=28.71..28.71 rows=3 width=109) (actual time=0.123..0.124 rows=1 loops=3)
30	Rows Removed by Filter: 24175
31	Index Cond: (id = e_1.subcategoryid)

32	Filter: (start_date >= CURRENT_DATE)
33	Buckets: 1024 Batches: 1 Memory Usage: 9kB
34	-> Nested Loop (cost=4.80..29.17 rows=3 width=177) (actual time=0.163..0.164 rows=0 loops=1)
35	-> Nested Loop (cost=0.74..28.71 rows=3 width=109) (actual time=0.115..0.119 rows=1 loops=3)
36	-> Index Scan using categorization_pkey on categorization c (cost=0.15..0.26 rows=1 width=66) (never executed)
37	Index Cond: (id = sub.categorizationid)
38	-> Nested Loop (cost=0.58..12.69 rows=3 width=43) (actual time=0.092..0.095 rows=1 loops=3)
39	-> Memoize (cost=0.16..6.84 rows=1 width=66) (actual time=0.021..0.021 rows=1 loops=3)
40	-> Index Scan using app_user_pkey on app_user u (cost=0.29..8.31 rows=1 width=35) (actual time=0.052..0.053 rows=1 loops=1)
41	-> Hash Join (cost=4.51..20.83 rows=3 width=150) (actual time=0.108..0.109 rows=0 loops=1)
42	Worker 1: Hits: 0 Misses: 1 Evictions: 0 Overflows: 0 Memory Usage: 1kB
43	Worker 0: Hits: 0 Misses: 1 Evictions: 0 Overflows: 0 Memory Usage: 1kB
44	Index Cond: (id = 24589)
45	Hits: 0 Misses: 1 Evictions: 0 Overflows: 0 Memory Usage: 1kB
46	Hash Cond: (sub.id = ucs.subcategoryid)
47	Cache Mode: logical
48	Cache Key: ucs.categorizationid
49	-> Seq Scan on subcategory sub (cost=0.00..15.00 rows=500 width=134) (actual time=0.032..0.033 rows=1 loops=1)
50	-> Materialize (cost=0.29..8.31 rows=1 width=35) (actual time=0.025..0.027 rows=1 loops=3)
51	-> Index Scan using categorization_pkey on categorization c_1 (cost=0.15..6.83 rows=1 width=66) (actual time=0.017..0.017 rows=1 loops=3)
52	-> Index Only Scan using user_category_subscription_pkey on user_category_subscription ucs (cost=0.29..4.35 rows=3 width=16) (actual time=0.044..0.044 rows=0 loops=1)
53	-> Hash (cost=4.47..4.47 rows=3 width=16) (actual time=0.044..0.044 rows=0 loops=1)
54	Index Cond: (id = ucs.categorizationid)
55	Index Cond: (app_userid = 24589)
56	Heap Fetches: 0
57	Buckets: 1024 Batches: 1 Memory Usage: 8kB
58	-> Index Scan using app_user_pkey on app_user u_1 (cost=0.29..8.31 rows=1 width=35) (actual time=0.018..0.020 rows=1 loops=3)
59	-> Index Only Scan using user_subcategory_subscription_pkey on user_subcategory_subscription uss (cost=0.42..4.47 rows=3 width=16) (actual time=0.044..0.044 rows=0 loops=1)
60	Index Cond: (id = 24589)
61	Index Cond: (app_userid = 24589)
62	Heap Fetches: 0

Планот на извршување покажува дека се користат Index Scan на primary key индекси и Hash Join операции. Единствениот Seq Scan е на табелата subcategory која содржи само 25 редови, тоа е прифатливо и нема потреба од индексирање.

5. Нема потреба од индексирање

6. Време на INSERT/UPDATE

```
INSERT INTO user_category_subscription (app_userid, categorizationid)
VALUES (24589, 2);
```

Statistics 1 ×

Output

Name	Value	
Updated Rows	1	
Query	INSERT INTO user_category_subscription (app_userid, categorizationid) VALUES (24589, 2)	
Finish time	Fri May 15 19:20:56 CEST 2026	

Save

Cancel

Script

<

```
UPDATE user_category_subscription SET categorizationid = 3
```

```
WHERE app_userid = 24589 AND categorizationid = 2;
```

Name	Value
Updated Rows	1
Query	UPDATE user_category_subscription SET categorizationid = 3 WHERE app_userid = 24589 AND categorizationid = 2
Finish time	Fri May 15 19:21:29 CEST 2026

Save Cancel Script 200 1 Rows: 62 1 row(s) updated - 0ms

Времето на операциите INSERT и UPDATE останува исто бидејќи нема нови индекси.

View 9: view_event_sales_report

- Примарен филтер:** Погледот примарно се филтрира според event_id.
- Случај на употреба:** Се користи за приказ на финансиски извештај по настан - приходи, рефундации и пополнетост на салите. Погледот го користат администраторите и организаторите на настани.
- Иницијално време:**

SELECT * FROM view_event_sales_report WHERE event_id = 12345;

-- Резултат 2.538s

event_id	title	start_date	category	active_tickets	cancelled_tickets	total_revenue	total_refunded	net_revenue	total_capacity	occupancy_percent
12345	Amateur Collector Showcase Vol. 2	2024-10-26	Theatre	555	27	2,138,700	883,800	1,254,900	67,830	0.82

Save Cancel Script 200 1 Rows: 1 1 row(s) fetched - 2.538s, on 2026-05-15 at 20:23:20

Времето е блиску до прифатливо, но може дополнително да се направи обид за оптимизација.

4. План на извршување

EXPLAIN ANALYZE SELECT * FROM view_event_sales_report WHERE event_id = 12345;

Results 1	Output
EXPLAIN ANALYZE SELECT * FROM view_event_sales_report Enter a SQL expression to filter results (use Ctrl+Space)	
Grid	QUERY PLAN
1	GroupAggregate (cost=230144.26..230178.30 rows=200 width=185) (actual time=650.129..657.092 rows=1 loops=1)
2	Group Key: c.category_name
3	-> Sort (cost=230144.26..230146.85 rows=1038 width=149) (actual time=649.786..656.791 rows=768 loops=1)
4	Sort Key: c.category_name
5	Sort Method: quicksort Memory: 117kB
6	-> Nested Loop Left Join (cost=167435.08..230092.25 rows=1038 width=149) (actual time=648.142..656.430 rows=768 loops=1)
7	-> Nested Loop Left Join (cost=1.28..46.05 rows=3 width=109) (actual time=36.989..37.024 rows=3 loops=1)
8	-> Nested Loop Left Join (cost=0.99..21.12 rows=3 width=113) (actual time=36.967..36.976 rows=3 loops=1)
9	-> Nested Loop Left Join (cost=0.57..16.62 rows=1 width=105) (actual time=36.939..36.942 rows=1 loops=1)
10	-> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=55) (actual time=0.031..0.033 rows=1 loops=1)
11	Index Cond: (id = 12345)
12	-> Index Scan using categorization_pkey on categorization c (cost=0.15..8.17 rows=1 width=66) (actual time=0.025..0.025 rows=1 loops=1)
13	Index Cond: (id = e.categorizationid)
14	-> Index Only Scan using event_hall_pkey on event_hall eh (cost=0.42..4.47 rows=3 width=16) (actual time=0.020..0.024 rows=3 loops=1)
15	Index Cond: (eventid = 12345)
16	Heap Fetches: 0
17	-> Index Scan using hall_pkey on hall h (cost=0.29..8.31 rows=1 width=12) (actual time=0.009..0.009 rows=1 loops=3)
18	Index Cond: (id = eh.hallid)
19	-> Materialize (cost=167433.80..230034.09 rows=346 width=48) (actual time=203.706..206.406 rows=256 loops=3)
20	-> Gather (cost=167433.80..230032.36 rows=346 width=48) (actual time=611.112..619.133 rows=256 loops=1)
21	Workers Planned: 4
22	Workers Launched: 4
23	-> Hash Left Join (cost=166433.80..228997.76 rows=86 width=48) (actual time=564.333..574.745 rows=51 loops=5)
24	Hash Cond: (p.id = rf.paymentid)
25	-> Parallel Hash Right Join (cost=164812.80..227376.43 rows=86 width=43) (actual time=559.304..569.697 rows=51 loops=5)
26	Hash Cond: (p.user_orderid = uo.id)
27	-> Parallel Seq Scan on payment p (cost=0.00..57876.21 rows=1249921 width=29) (actual time=0.037..104.419 rows=999937 loops=5)
28	-> Parallel Hash (cost=164811.72..164811.72 rows=86 width=30) (actual time=327.433..327.435 rows=51 loops=5)
29	Buckets: 1024 Batches: 1 Memory Usage: 168kB
30	-> Nested Loop Left Join (cost=0.43..164811.72 rows=86 width=30) (actual time=31.208..327.213 rows=51 loops=5)
31	-> Parallel Seq Scan on ticket t (cost=0.00..164430.02 rows=86 width=30) (actual time=31.090..326.793 rows=51 loops=5)
32	Filter: (eventid = 12345)
33	Rows Removed by Filter: 1999952
34	-> Index Only Scan using user_order_pkey on user_order uo (cost=0.43..4.44 rows=1 width=8) (actual time=0.006..0.006 rows=1 loops=1)
35	Index Cond: (id = t.user_orderid)
36	Heap Fetches: 224
37	-> Hash (cost=996.00..996.00 rows=50000 width=21) (actual time=24.604..24.605 rows=50000 loops=1)
38	Buckets: 65536 Batches: 1 Memory Usage: 3247kB
39	-> Seq Scan on refund rf (cost=0.00..996.00 rows=50000 width=21) (actual time=0.090..10.240 rows=50000 loops=1)
40	Planning Time: 2.048 ms
41	JIT:
42	Functions: 137
43	Options: Inlining false, Optimization false, Expressions true, Deforming true
44	Timing: Generation 12.832 ms (Deform 7.007 ms), Inlining 0.000 ms, Optimization 5.323 ms, Emission 110.444 ms, Total 128.599 ms
45	Execution Time: 661.941 ms

Најбавните операции се:

- **Parallel Seq Scan on ticket** кој скенира ~2,000,000 редови за да најде само 192 тикети за тој настан
- **Parallel Seq Scan on payment** кој ги скенира сите ~5,000,000 плаќања

Погледот го користи веќе постоечкиот индекс `idx_ticket_eventid` креиран при оптимизацијата на погледот број 4 `view_event_ticket_availability`, но `Seq Scan` на `payment` сè уште останува, поради што обид за оптимизација може да се направи со додавање индекс на `payment (user_orderid)`.

INSERT / UPDATE пред индексирање

Name	Value
Updated Rows	1
Query	INSERT INTO ticket (code, status, ticket_typeid, user_orderid, seatid, app_userid, eventid, hallid) VALUES ('TKT-TEST-070', 'ACTIVE', 1, (SELECT MIN(id) FROM user_order), NULL, (SELECT MIN(id) FROM app_user WHERE id != 1), 12345, (SELECT MIN(hallid) FROM event_hall WHERE eventid = 12345))
Finish time	Fri May 15 20:23:51 CEST 2026

Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 10ms,

Name	Value
Updated Rows	1
Query	UPDATE ticket SET status = 'CANCELLED' WHERE code = 'TKT-TEST-070'
Finish time	Fri May 15 20:24:25 CEST 2026

Save Cancel Script 200 1 Rows: 1 1 row(s) updated - 13ms,

5. Индексирање

CREATE INDEX idx_payment_userorderid ON payment(user_orderid);

SELECT * FROM view_event_sales_report WHERE event_id = 12345;

-- Резултат: 1.630s

event_id	title	start_date	category	active_tickets	cancelled_tickets	total_revenue	total_refunded	net_revenue	total_capacity	occupancy_percent
12345	Amateur Collector Showcase Vol.	2024-10-26	Theatre	555	24	2,123,700	868,800	1,254,900	67,564	0.82

Save Cancel Script 200 1 Rows: 1 1 row(s) fetched - 1.630s, on 2026-05-15 at 20:09:55

Времето изминато во извршување со индекс изнесува 1.630 секунди и тоа е прифатливо.

План на извршување по индексирање

EXPLAIN ANALYZE SELECT * FROM view_event_sales_report WHERE event_id = 12455;

Results 1 × Output	
EXPLAIN ANALYZE SELECT * FROM view_event_sales_report \ Enter a SQL expression to filter results (use Ctrl+Space)	
QUERY PLAN	
1	GroupAggregate (cost=167623.17..167657.22 rows=200 width=185) (actual time=414.495..422.733 rows=1 loops=1)
2	Group Key: c.category_name
3	-> Sort (cost=167623.17..167625.77 rows=1038 width=149) (actual time=414.123..422.406 rows=762 loops=1)
4	Sort Key: c.category_name
5	Sort Method: quicksort Memory: 117kB
6	-> Nested Loop Left Join (cost=2623.14..167571.17 rows=1038 width=149) (actual time=53.603..422.038 rows=762 loops=1)
7	-> Nested Loop Left Join (cost=1.28..46.05 rows=3 width=109) (actual time=30.165..30.207 rows=3 loops=1)
8	-> Nested Loop Left Join (cost=0.99..21.12 rows=3 width=113) (actual time=30.143..30.155 rows=3 loops=1)
9	-> Nested Loop Left Join (cost=0.57..16.62 rows=1 width=105) (actual time=30.117..30.121 rows=1 loops=1)
10	-> Index Scan using event_pkey on event e (cost=0.42..8.44 rows=1 width=55) (actual time=0.019..0.021 rows=1 loops=1)
11	Index Cond: (id = 12345)
12	-> Index Scan using categorization_pkey on categorization c (cost=0.15..8.17 rows=1 width=66) (actual time=0.023..0.023 rows=1 loops=1)
13	Index Cond: (id = e.categorizationid)
14	-> Index Only Scan using event_hall_pkey on event_hall eh (cost=0.42..4.47 rows=3 width=16) (actual time=0.018..0.024 rows=3 loops=1)
15	Index Cond: (eventid = 12345)
16	Heap Fetches: 0
17	-> Index Scan using hall_pkey on hall h (cost=0.29..8.31 rows=1 width=12) (actual time=0.010..0.010 rows=1 loops=3)
18	Index Cond: (id = eh.hallid)
19	-> Materialize (cost=2621.86..167513.01 rows=346 width=48) (actual time=7.810..130.550 rows=254 loops=3)
20	-> Gather (cost=2621.86..167511.28 rows=346 width=48) (actual time=23.425..391.471 rows=254 loops=1)
21	Workers Planned: 4
22	Workers Launched: 4
23	-> Hash Left Join (cost=1621.87..166476.68 rows=86 width=48) (actual time=40.561..346.175 rows=51 loops=5)
24	Hash Cond: (p.id = rf.paymentid)
25	-> Nested Loop Left Join (cost=0.86..164855.35 rows=86 width=43) (actual time=17.495..323.055 rows=51 loops=5)
26	-> Nested Loop Left Join (cost=0.43..164811.72 rows=86 width=30) (actual time=17.449..322.770 rows=50 loops=5)
27	-> Parallel Seq Scan on ticket t (cost=0.00..164430.02 rows=86 width=30) (actual time=17.355..322.364 rows=50 loops=5)
28	Filter: (eventid = 12345)
29	Rows Removed by Filter: 1999952
30	-> Index Only Scan using user_order_pkey on user_order uo (cost=0.43..4.44 rows=1 width=8) (actual time=0.005..0.005 rows=1 loops=251)
31	Index Cond: (id = t.user_orderid)
32	Heap Fetches: 222
33	-> Index Scan using idx_payment_userorderid on payment p (cost=0.43..0.50 rows=1 width=29) (actual time=0.004..0.004 rows=1 loops=251)
34	Index Cond: (user_orderid = uo.id)
35	-> Hash (cost=996.00..996.00 rows=50000 width=21) (actual time=22.525..22.526 rows=50000 loops=5)
36	Buckets: 65536 Batches: 1 Memory Usage: 3247kB
37	-> Seq Scan on refund rf (cost=0.00..996.00 rows=50000 width=21) (actual time=0.042..8.722 rows=50000 loops=5)
38	Planning Time: 1.822 ms
39	JIT:
40	Functions: 117
41	Options: Inlining false, Optimization false, Expressions true, Deforming true
42	Timing: Generation 9.974 ms (Deform 4.717 ms), Inlining 0.000 ms, Optimization 4.180 ms, Emission 88.219 ms, Total 102.373 ms
43	Execution Time: 426.256 ms

По додавање на `idx_payment_userorderid`, PostgreSQL сега користи Index Scan on payment наместо Seq Scan со што директно ги наоѓа плаќањата по `user_orderid`. Seq Scan на refund е прифатлив бидејќи табелата содржи само 50,000 редови. Seq Scan на ticket останува, но е паралелен и прифатлив.

6. Време на INSERT/UPDATE по индексирање

Statistics 1 × Output	
Name	Value
Updated Rows	1
Query	INSERT INTO ticket (code, status, ticket_typeid, user_orderid, seatid, app_userid, eventid, hallid) VALUES ('TKT-TEST-060', 'ACTIVE', 1, (SELECT MIN(id) FROM user_order), NULL, (SELECT MIN(id) FROM app_user WHERE id != 1), 12345, (SELECT MIN(hallid) FROM event_hall WHERE eventid = 12345))
Finish time	Fri May 15 20:16:16 CEST 2026
Save Cancel Script 200 1 Rows: 43 1 row(s) updated - 9ms,	

Name	Value
Updated Rows	1
Query	UPDATE ticket SET status = 'CANCELLED' WHERE code = 'TKT-TEST-060'
Finish time	Fri May 15 20:16:46 CEST 2026

Save Cancel Script

Navigation icons

200

1 Rows: 43 1 row(s) updated - 16ms