

VIEW ACTIVE PRODUCTS

Овој view прикажува листа на сите активни производи кои се достапни за продажба. За секој производ се прикажуваат основни информации како наслов, цена, валута, локација и главна слика (thumbnail). Дополнително се пресметува и поле low_stock кое означува дали производот има мала залиха (5 или помалку парчиња).

```
-- преглед на сите активни продукти
CREATE OR REPLACE VIEW vw_active_products AS
SELECT
  p.product_id,
  p.title,
  p.price,
  p.currency,

  CASE
    WHEN p.quantity <= 5 THEN 1
    ELSE 0
  END AS low_stock,

  ( SELECT pi.image_url
    FROM productimages pi
    WHERE pi.product_id = p.product_id
    ORDER BY pi.image_id
    LIMIT 1) AS thumbnail,

  --za filter
  u.user_id AS seller_id,
  c.category_id,
  p.location

FROM product p
JOIN category c ON c.category_id = p.category_id
JOIN appuser u ON u.user_id = p.seller_id
WHERE p.is_active = 1 AND p.quantity > 0 AND u.is_active = 1;
```

Оптимизација

Беше

38	Planning Time: 1.592 ms
39	JIT:
40	Functions: 47
41	Options: Inlining true, Optimization true, Expressions true, Deforming true
42	Timing: Generation 3.242 ms (Deform 1.586 ms), Inlining 18.835 ms, Optimization 283.236 ms, Emission 173.422 ms, Total 478.
43	Execution Time: 1156130.771 ms

Додадовме индекс

```
CREATE INDEX idx_productimages_product_id ON productimages(product_id);
```

27	Functions: 81
28	Options: Inlining false, Optimization false, Expressions true, Deforming true
29	Timing: Generation 7.262 ms (Deform 3.790 ms), Inlining 0.000 ms, Optimization 3.706 ms, Emission 65.569 ms, Total 76.537 ms
30	Execution Time: 24814.268 ms

За пребарување активни продукти според категорија:

```
EXPLAIN ANALYZE
```

```
SELECT * FROM vw_active_products
```

```
WHERE category_id = 60;
```

Додадовме индекс

```
CREATE INDEX idx_product_category_id ON product(category_id);
```

И добивме:

Timing: Generation 8.432 ms (Deform 4.451 ms), Inlining 0.000 ms, Optimization 4.277 ms, Emission 73.529 ms, Total 86.237 ms
Execution Time: 951.950 ms

PRODUCT DETAILS

Овој view обезбедува детален приказ на избран производ. Покрај основните информации за производот, вклучува податоци за продавачот, просечна оцена, број на рецензии и листа од сите слики поврзани со производот. Наменет е за страницата со детали за производ.

```
-- детален преглед на избран продукт

CREATE OR REPLACE VIEW vw_product_details AS

WITH product_rating_reviews AS (
    SELECT r.product_id,
           COALESCE(avg(r.rating), 0::numeric) AS average_rating,
           count(r.review_id) AS total_reviews
    FROM review r
    GROUP BY r.product_id
),

product_images AS (
    SELECT pi.product_id,
           array_agg(pi.image_url ORDER BY pi.image_id) AS images,
           min(pi.image_url::text) AS thumbnail
    FROM productimages pi
    GROUP BY pi.product_id
)

SELECT
    p.product_id,
    p.title,
    p.description,
    p.price,
    p.currency,
    p.created_at,
    p.location,
    CASE
        WHEN p.quantity <= 5 THEN 1
        ELSE 0
    END AS low_stock,

    c.name,

    u.username AS seller_username,
    u.first_name,
    u.last_name,

    ps.average_rating,
    ps.total_reviews,
```

```

- ps.average_rating,
  ps.total_reviews,
  i.images,
  i.thumbnail

- FROM product p
  JOIN category c ON c.category_id = p.category_id
  JOIN appuser u ON u.user_id = p.seller_id
  LEFT JOIN product_rating_reviews ps ON ps.product_id = p.product_id
  LEFT JOIN product_images i ON i.product_id = p.product_id;

```

Оптимизација

explain analyze

SELECT *

FROM vw_product_details

where product_id = 500;

6	-> Seq Scan on category c (cost=0.00..4.26 rows=226 width=16) (actual time=0.017..0.028 rows=113 loops=1)
7	-> Hash (cost=8.45..8.45 rows=1 width=96) (actual time=13.112..13.112 rows=1 loops=1)
8	Buckets: 1024 Batches: 1 Memory Usage: 9kB
9	-> Index Scan using product_pkey on product p (cost=0.43..8.45 rows=1 width=96) (actual time=13.103..13.105 rows=1 loops=1)
10	Index Cond: (product_id = 500)
11	-> Index Scan using appuser_pkey on appuser u (cost=0.42..8.44 rows=1 width=36) (actual time=0.031..0.031 rows=1 loops=1)
12	Index Cond: (user_id = p.seller_id)
13	-> GroupAggregate (cost=0.42..8.46 rows=1 width=44) (actual time=0.014..0.015 rows=0 loops=1)
14	-> Index Scan using idx_review_product_id on review r (cost=0.42..8.44 rows=1 width=12) (actual time=0.014..0.014 rows=0 loops=1)
15	Index Cond: (product_id = 500)
16	-> GroupAggregate (cost=10.41..10.45 rows=1 width=68) (actual time=0.042..0.042 rows=1 loops=1)
17	-> Sort (cost=10.41..10.42 rows=4 width=68) (actual time=0.026..0.027 rows=2 loops=1)
18	Sort Key: pi.image_id
19	Sort Method: quicksort Memory: 25kB
20	-> Index Scan using idx_productimages_product_id on productimages pi (cost=0.43..10.37 rows=4 width=68) (actual time=0.014..0.016 rows=2 loops=1)
21	Index Cond: (product_id = 500)
22	Planning Time: 0.860 ms
23	Execution Time: 13.376 ms

Претходно направив индекс за images по product_id, па така нема потреба од дополнителна оптимизација.

REVIEW

Овој view прикажува сите рецензии за производите. За секоја рецензија се прикажуваат оцената, коментарот, датумот на креирање и корисникот кој ја оставил рецензијата. Се користи за приказ на корисничките оценки и коментари за одреден продукт.

```
-- преглед на reviews за конкретен продукт
DROP VIEW vw_product_reviews

CREATE OR REPLACE VIEW vw_product_reviews AS
SELECT
  r.review_id,
  r.product_id,
  p.title,
  r.rating,
  r.comment,
  r.created_at,

  r.buyer_id,
  ub.username AS buyer_username,
  r.seller_id

FROM review r
JOIN product p ON p.product_id = r.product_id
JOIN appuser ub ON ub.user_id = r.buyer_id;
```

Оптимизација

explain analyze

SELECT *

FROM vw_product_reviews

where product_id = 240446;

Беше

	ABC QUERY PLAN
16	-> Seq Scan on review (cost=0.00..5449.04 rows=253204 width=4) (actual time=0.006..22.187 rows=253204)
17	-> Index Scan using product_pkey on product p (cost=0.43..2.11 rows=1 width=27) (actual time=0.200..0.200 rows=1 loops=1)
18	Index Cond: (product_id = r.product_id)
19	-> Index Scan using appuser_pkey on appuser ub (cost=0.42..0.50 rows=1 width=23) (actual time=0.003..0.003 rows=1 loops=1)
20	Index Cond: (user_id = r.buyer_id)
21	Planning Time: 0.737 ms
22	Execution Time: 2412.917 ms

Додадовме Index: **CREATE INDEX** idx_review_product_id **ON** review(product_id);

13	Index Cond: (product_id = review.product_id)
14	-> Index Scan using idx_review_product_id on review r (cost=0.42..0.44 rows=1 width=62) (actual time=0.006..0.034 rows=103 loops=1)
15	Index Cond: (product_id = p.product_id)
16	-> Index Scan using appuser_pkey on appuser ub (cost=0.42..0.50 rows=1 width=23) (actual time=0.003..0.003 rows=1 loops=10331)
17	Index Cond: (user_id = r.buyer_id)
18	Planning Time: 1.238 ms
19	Execution Time: 946.953 ms

FAVORITES

Овој view ја прикажува листата на омилене производи за секој корисник. За секој производ се прикажуваат основни информации, thumbnail слика и индикатор за ниска залиха. Се користи за функционалноста „Favorites“.

```
-- преглед на листата со омилене производи на корисникот.  
CREATE OR REPLACE VIEW vw_favorites AS  
SELECT  
    f.user_id,  
    p.product_id,  
    p.title,  
    p.price,  
    p.currency,  
    CASE  
        WHEN p.quantity <= 5 THEN 1  
        ELSE 0  
    END AS low_stock,  
    (SELECT pi.image_url  
     FROM productimages pi  
     WHERE pi.product_id = p.product_id  
     ORDER BY pi.image_id  
     LIMIT 1) AS thumbnail  
FROM favorites f  
JOIN product p ON f.product_id = p.product_id;  
  
explain analyze  
SELECT *  
FROM vw_favorites  
where user_id = 5;
```

Оптимизација

При самото креирање на табелата Favorites имавме ставено UNIQUE constraint:

```
uq_fav_user_product= UNIQUE (user_id, product_id)
```

+ го користи индексот креиран погоре idx_productimages_product_id

ABC QUERY PLAN	
1	Nested Loop (cost=0.86..193.21 rows=9 width=562) (actual time=1.006..325.849 rows=19 loops=1)
2	-> Index Only Scan using uq_fav_user_product on favorites f (cost=0.43..23.62 rows=9 width=8) (actual time=0.021..26.571 rows=19 loops=1)
3	Index Cond: (user_id = 11)
4	Heap Fetches: 15
5	-> Index Scan using product_pkey on product p (cost=0.43..8.45 rows=1 width=42) (actual time=15.052..15.052 rows=1 loops=19)
6	Index Cond: (product_id = f.product_id)
7	SubPlan 1
8	-> Limit (cost=10.39..10.39 rows=1 width=64) (actual time=0.694..0.694 rows=1 loops=19)
9	-> Sort (cost=10.39..10.40 rows=4 width=64) (actual time=0.693..0.693 rows=1 loops=19)
10	Sort Key: pi.image_id
11	Sort Method: top-N heapsort Memory: 25kB
12	-> Index Scan using idx_productimages_product_id on productimages pi (cost=0.43..10.37 rows=4 width=64) (actual time=0.017..0.685 rows=4 loops=19)
13	Index Cond: (product_id = p.product_id)
14	Planning Time: 0.435 ms
15	Execution Time: 325.904 ms

CART ITEMS

Овој view ги прикажува сите ставки кои се наоѓаат во кошничката на корисникот. За секој производ се прикажуваат количината, цената во моментот на додавање, тековната цена, пресметан subtotal и thumbnail слика. Се користи за приказ на содржината на кошничката.

```
-- преглед на сите ставки во кошничката на корисникот.  
CREATE OR REPLACE VIEW vw_cart_items AS  
SELECT  
    ci.cart_id, --filter  
    ci.cart_item_id,  
  
    ci.product_id,  
    p.title,  
    p.price AS current_price,  
    p.currency,  
    ci.quantity,  
    ci.price_at_time,  
  
    (ci.quantity * ci.price_at_time) AS subtotal,  
  
    (  
        SELECT image_url  
        FROM productimages pi  
        WHERE pi.product_id = p.product_id  
        ORDER BY image_id  
        LIMIT 1  
    ) AS thumbnail  
  
FROM cartitems ci  
JOIN product p ON p.product_id = ci.product_id
```

Оптимизација

idx_cartitems_cart_id — на cart_id — за брзо земање на items по кошничка

```
explain analyze
```

```
select *
```

```
from vw_cart_items
```

```
where cart_id =92846;
```

	abc QUERY PLAN
1	Nested Loop (cost=0.85..83.88 rows=4 width=605) (actual time=0.100..0.200 rows=5 loops=1)
2	-> Index Scan using idx_cartitems_cart_id on cartitems ci (cost=0.42..8.49 rows=4 width=23) (actual time=0.044..0.046 rows=5 loops=1)
3	Index Cond: (cart_id = 92846)
4	-> Index Scan using product_pkey on product p (cost=0.43..8.45 rows=1 width=38) (actual time=0.008..0.008 rows=1 loops=5)
5	Index Cond: (product_id = ci.product_id)
6	SubPlan 1
7	-> Limit (cost=10.39..10.39 rows=1 width=64) (actual time=0.018..0.018 rows=1 loops=5)
8	-> Sort (cost=10.39..10.40 rows=4 width=64) (actual time=0.018..0.018 rows=1 loops=5)
9	Sort Key: pi.image_id
10	Sort Method: top-N heapsort Memory: 25kB
11	-> Index Scan using idx_productimages_product_id on productimages pi (cost=0.43..10.37 rows=4 width=64) (actual time=0.018..0.018 rows=4 loops=5)
12	Index Cond: (product_id = p.product_id)
13	Planning Time: 0.703 ms
14	Execution Time: 0.250 ms

CART SUMMARY

Овој view обезбедува сумиран приказ на кошничката. Ги содржи вкупната вредност на кошничката, бројот на различни ставки и вкупната количина на производи. Се користи за брз преглед пред нарачка.

```
-- сумиран приказ на кошничката.  
drop view vw_cart_summary;  
  
CREATE OR REPLACE VIEW vw_cart_summary AS  
SELECT  
    c.cart_id,  
    c.created_at,  
  
    c.user_id, --filter  
  
    c.total_price AS cart_total,  
  
    COUNT(ci.cart_item_id) AS total_items,  
    SUM(ci.quantity) AS total_quantity  
  
FROM cart c  
JOIN cartitems ci ON ci.cart_id = c.cart_id  
  
GROUP BY c.cart_id;
```

Оптимизација

1	GroupAggregate (cost=4486.04..4486.08 rows=1 width=35) (actual time=31.601..31.603 rows=1 loops=1)
2	Group Key: c.cart_id
3	-> Sort (cost=4486.04..4486.05 rows=3 width=27) (actual time=31.587..31.588 rows=5 loops=1)
4	Sort Key: c.cart_id
5	Sort Method: quicksort Memory: 25kB
6	-> Nested Loop (cost=0.42..4486.02 rows=3 width=27) (actual time=1.126..31.577 rows=5 loops=1)
7	-> Seq Scan on cart c (cost=0.00..4477.49 rows=1 width=19) (actual time=1.108..31.554 rows=1 loops=1)
8	Filter: (user_id = 22)
9	Rows Removed by Filter: 169798
10	-> Index Scan using idx_cartitems_cart_id on cartitems ci (cost=0.42..8.49 rows=4 width=12) (actual time=0.013..0.015 rows=5 loops=1)
11	Index Cond: (cart_id = c.cart_id) -> Index Scan using idx_cartitems_cart_id on cartitems ci (cost=0.42..8.49 rows=4 width=12) (actual time=0.013..0.015 rows=5 loops=1)
12	Planning Time: 0.718 ms
13	Execution Time: 31.662 ms

Го користи претходно направениот индекс за cart_items по cart_id.

Нема потреба од дополнителна оптимизација.

ORDER SUMMARY

Овој view обезбедува целосен преглед на една нарачка. Покрај информациите за нарачката, содржи податоци за купувачот, адресата за достава, курирската служба, tracking бројот и статистика за бројот на производи во нарачката.

```
-- сумиран приказ на нарачка
CREATE OR REPLACE VIEW v_order_summary AS
SELECT
  o.order_id,
  o.created_at AS order_date,
  o.status AS order_status,
  o.total_price,

  u.first_name || ' ' || u.last_name AS buyer_name,
  ua.street || ' ' || ua.house_number AS street_address,
  ua.city,

  c.name AS carrier_name,
  s.tracking_number,

  (SELECT COUNT(*) FROM orderitems oi WHERE oi.order_id = o.order_id) AS total_items,
  (SELECT SUM(quantity) FROM orderitems oi WHERE oi.order_id = o.order_id) AS total_quantity

FROM "order" o
JOIN appuser u ON u.user_id = o.buyer_id
JOIN shipment s ON s.order_id = o.order_id
JOIN carriers c ON c.carrier_id = s.carrier_id
JOIN useraddress ua ON ua.user_address_id = s.user_address_id;
```

Оптимизација

EXPLAIN ANALYZE

```
SELECT * FROM v_order_summary  
WHERE order_id = 80;
```

22	SubPlan 2
23	-> Aggregate (cost=6039.56..6039.57 rows=1 width=8) (actual time=22.496..22.497 rows=1 loops=1)
24	-> Seq Scan on orderitems oi_1 (cost=0.00..6039.55 rows=4 width=4) (actual time=8.675..22.490 rows=2 loops=1)
25	Filter: (order_id = o.order_id)
26	Rows Removed by Filter: 304202
27	Planning Time: 2733.237 ms
28	Execution Time: 20687.802 ms

Додадовме индекс

```
CREATE INDEX idx_orderitems_order_id ON orderitems(order_id);
```

24	Heap Blocks: exact=2
25	-> Bitmap Index Scan on idx_orderitems_order_id (cost=0.00..4.45 rows=4 width=0) (actual time=0.011..0.011 rows=2)
26	Index Cond: (order_id = o.order_id)
27	Planning Time: 1.411 ms
28	Execution Time: 650.855 ms

ORDER ITEMS

Овој view ги прикажува сите производи кои припаѓаат на одредена нарачка. За секоја ставка се прикажуваат количината, цената при купување, моменталната цена и пресметаниот subtotal. Наменет е за деталниот приказ на нарачки.

```

CREATE OR REPLACE VIEW v_order_items AS
SELECT
  oi.order_id, -- filter
  oi.order_item_id,

  oi.product_id,
  p.title,
  p.price AS current_price,
  p.currency,
  oi.quantity,
  oi.price_at_time,

  (oi.quantity * oi.price_at_time) AS subtotal,

  (
    SELECT image_url
    FROM productimages pi
    WHERE pi.product_id = p.product_id
    ORDER BY image_id
    LIMIT 1
  ) AS thumbnail

FROM orderitems oi
JOIN product p ON p.product_id = oi.product_id;

```

Оптимизација

претходно направивме `idx_orderitems_order_id` — на `order_id` — за брзо земање на items по нарачка

Нема потреба од дополнителна оптимизација.

13	Sort Method: quicksort Memory: 25kB
14	-> Index Scan using idx_productimages_product_id on productimages pi (cost=0.43..10.37 rows=4 width=64) (ac
15	Index Cond: (product_id = p.product_id)
16	Planning Time: 2829.747 ms
17	Execution Time: 333.912 ms

SELLER DASHBOARD

Овој view претставува dashboard за продавачите и прикажува агрегирани статистики за нивните производи. Вклучува вкупен број производи, број на продадени артикли, вкупен приход, просечна оцена од рецензии и активниот пакет на продавачот. Се користи за анализа на продажбата и перформансите на продавачите.

```
-- dashboard приказ со статистики за продуктите на одреден продавач
CREATE OR REPLACE VIEW v_seller_dashboard AS
WITH active_package AS (
    SELECT DISTINCT ON (seller_id)
        seller_id,
        package_id
    FROM UserPackages
    WHERE CURRENT_TIMESTAMP BETWEEN start_date AND end_date
    ORDER BY seller_id, end_date DESC
)
SELECT
    u.user_id AS seller_id,
    u.first_name,
    u.last_name,
    COUNT(DISTINCT pr.product_id) AS total_products,
    SUM(oi.quantity) AS total_sales,
    SUM(oi.quantity * oi.price_at_time) AS total_revenue,
    ROUND(AVG(r.rating), 2) AS avg_rating,
    pck.name AS package_name
FROM AppUser u
LEFT JOIN Product pr ON pr.seller_id = u.user_id
LEFT JOIN OrderItems oi ON oi.product_id = pr.product_id
LEFT JOIN Review r ON r.product_id = pr.product_id
LEFT JOIN active_package ap ON ap.seller_id = u.user_id
LEFT JOIN Package pck ON pck.package_id = ap.package_id
WHERE u.is_verified = 1
GROUP BY u.user_id, u.first_name, u.last_name, pck.name;

EXPLAIN ANALYZE
SELECT *
FROM v_seller_dashboard
WHERE seller_id = 120824;
```

Оптимизација

Времето беше лошо бидејќи прави Parallel Seq Scan на **userpackages**, **product**, **orderitems**

ABC QUERY PLAN	
6	-> Nested Loop Left Join (cost=105285.05..110019.39 rows=44 width=552) (actual time=472.513..515.113 rows=52 loops=1)
7	-> Nested Loop Left Join (cost=5991.42..6007.90 rows=1 width=533) (actual time=69.281..69.378 rows=1 loops=1)
8	-> Nested Loop Left Join (cost=5991.28..5999.43 rows=1 width=21) (actual time=69.258..69.354 rows=1 loops=1)
9	-> Index Scan using appuser_pkey on appuser u (cost=0.42..8.44 rows=1 width=17) (actual time=29.737..29.737 rows=1 loops=1)
10	Index Cond: (user_id = 120824)
11	Filter: (is_verified = 1)
12	-> Limit (cost=5990.86..5990.97 rows=1 width=16) (actual time=39.477..39.567 rows=0 loops=1)
13	-> Gather Merge (cost=5990.86..5991.20 rows=3 width=16) (actual time=39.476..39.565 rows=0 loops=1)
14	Workers Planned: 1
15	Workers Launched: 1
16	-> Sort (cost=4990.85..4990.85 rows=2 width=16) (actual time=9.364..9.365 rows=0 loops=2)
17	Sort Key: userpackages.end_date DESC
18	Sort Method: quicksort Memory: 25kB
19	Worker 0: Sort Method: quicksort Memory: 25kB
20	-> Parallel Seq Scan on userpackages (cost=0.00..4990.84 rows=2 width=16) (actual time=9.257..9.257 rows=0 loops=2)
21	Filter: ((seller_id = 120824) AND (CURRENT_TIMESTAMP >= start_date) AND (CURRENT_TIMESTAMP <= end_date))
22	Rows Removed by Filter: 121185
23	-> Index Scan using package_pkey on package pck (cost=0.14..8.16 rows=1 width=520) (actual time=0.005..0.005 rows=1 loops=1)
24	Index Cond: (package_id = userpackages.package_id)
25	-> Gather (cost=99293.64..104011.05 rows=44 width=23) (actual time=403.224..445.714 rows=52 loops=1)
26	Workers Planned: 1
27	Workers Launched: 1
28	-> Nested Loop Left Join (cost=98293.64..103006.65 rows=26 width=23) (actual time=388.936..411.717 rows=26 loops=1)
29	-> Parallel Hash Right Join (cost=98293.22..102789.38 rows=26 width=19) (actual time=388.867..411.350 rows=26 loops=1)
30	Hash Cond: (oi.product_id = pr.product_id)
31	-> Parallel Seq Scan on orderitems oi (cost=0.00..4026.44 rows=178944 width=15) (actual time=0.008..0.008 rows=26 loops=1)
32	-> Parallel Hash (cost=98293.08..98293.08 rows=11 width=8) (actual time=377.184..377.185 rows=22 loops=1)
33	Buckets: 1024 Batches: 1 Memory Usage: 72kB
34	-> Parallel Seq Scan on product pr (cost=0.00..98293.08 rows=11 width=8) (actual time=59.429..59.429 rows=11 loops=1)
35	Filter: (seller_id = 120824)

Решение: Материјализирано view

ABC QUERY PLAN	
1	Seq Scan on mv_seller_dashboard_mat (cost=0.00..1406.90 rows=1 width=50) (actual time=3.038..5.085 rows=1 loops=1)
2	Filter: (seller_id = 120824)
3	Rows Removed by Filter: 59831
4	Planning Time: 0.252 ms
5	Execution Time: 5.106 ms

USER PROFILE

Овој view обезбедува комплетен приказ на кориснички профил. Покрај основните податоци за корисникот, содржи примарна адреса на живеење и вкупен број направени нарачки. Се користи за страницата со профил на корисникот.

```
-- преглед на кориснички профил|
CREATE OR REPLACE VIEW v_user_profile as
WITH primary_location AS (
    SELECT user_id, city || ', ' || country AS location
    FROM UserAddress
    WHERE is_primary = 1
),
user_orders AS (
    SELECT buyer_id, COUNT(*) AS total_orders
    FROM "order"
    GROUP BY buyer_id
)
SELECT
    u.user_id,
    u.first_name,
    u.last_name,
    u.email,
    u.phone_number,
    pl.location AS primary_location,
    uo.total_orders,
    u.created_at
FROM AppUser u
LEFT JOIN primary_location pl ON pl.user_id = u.user_id
LEFT JOIN user_orders uo ON uo.buyer_id = u.user_id;
```

Оптимизација

Првично

1	Nested Loop Left Join (cost=1000.42..11206.82 rows=1 width=107) (actual time=2747.501..2747.625 rows=1 loops=1)
2	-> Nested Loop Left Join (cost=1000.42..8812.05 rows=1 width=81) (actual time=2501.392..2501.514 rows=1 loops=1)
3	-> Index Scan using appuser_pkey on appuser u (cost=0.42..8.44 rows=1 width=67) (actual time=0.032..0.037 rows=1 loops=1)
4	Index Cond: (user_id = 600)
5	-> Gather (cost=1000.00..8803.60 rows=1 width=18) (actual time=2501.345..2501.461 rows=1 loops=1)
6	Workers Planned: 2
7	Workers Launched: 2
8	-> Parallel Seq Scan on useraddress (cost=0.00..7803.50 rows=1 width=18) (actual time=1707.226..2492.091 rows=0 loops=3)
9	Filter: ((is_primary = 1) AND (user_id = 600))
10	Rows Removed by Filter: 166533
11	-> GroupAggregate (cost=0.00..2394.75 rows=1 width=12) (actual time=246.099..246.100 rows=1 loops=1)
12	-> Seq Scan on "order" (cost=0.00..2394.74 rows=1 width=4) (actual time=117.940..246.090 rows=1 loops=1)
13	Filter: (buyer_id = 600)
14	Rows Removed by Filter: 119102
15	Planning Time: 102.717 ms
16	Execution Time: 2747.816 ms

Решение: Материјализирано view

1	Seq Scan on mv_user_profile_mat (cost=0.00..5508.00 rows=1 width=90) (actual time=10.041..25.556 rows=1 loops=1)
2	Filter: (user_id = 1)
3	Rows Removed by Filter: 199999
4	Planning Time: 0.251 ms
5	Execution Time: 25.580 ms