



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

---

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

**ОТЧЕТ**

по лабораторной работе № 4

**Название:** Оптимизация процессов в PostgreSQL

**Дисциплина:** Базы данных

Студент

ИУ6-33Б  
(Группа)

19.11.2023

(Подпись, дата)

В.К. Залыгин

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

М.А. Скворцова

(И.О. Фамилия)

Москва, 2023

## Цель

Сформировать у студента понимание методов упрощения работы аналитика с БД.

## Ход работы

### Часть 1. Функции

```
1 create or replace function compute_sum_module(modu integer, lim integer)
2 returns integer as
3 $$
4 declare
5     curs1 cursor for select payments.id, payments.price from payments;
6     id uuid;
7     price integer;
8     res integer;
9 begin
10     res = 0;
11     while lim > 0 loop
12         fetch curs1 into id, price;
13         if id % modu = 0 then
14             res = res + price;
15         end if;
16         lim = lim - 1;
17     end loop;
18     return res;
19 exception when others
20 then
21     return -1;
22 end
23 $$
24 language 'plpgsql'
```

Рисунок 1. Функция вычисления суммы стоимостей услуг с ограничением по количеству, id которых имеет определенную кратность.

### Часть 2. Анализ и оптимизация запросов

Будем исследовать и оптимизировать работы 2 запросов.

```
1 -- Какие квартиры имеют номер < 300
2 select *
3 from apartments
4 where cast(full_address->>'apartment_number' as integer) < 300;
5
6 --Вывести все платежи по квартирам, которые находятся на Льва Толстого, где цена платежа больше 3000
7 select *
8 from payments
9 join apartments on apartments.id = payments.apartment_id
10 where apartments.full_address->>'street' = 'улица Льва Толстого' and payments.price > 3000;
11
```

Рисунок 2. Запросы для оптимизации

```

1  explain
2  select *
3  from apartments
4  where cast(full_address->>'apartment_number' as integer) < 300;
5
6  explain (ANALYZE)
7  select *
8  from apartments
9  where cast(full_address->>'apartment_number' as integer) < 300;
10
11 QUERY PLAN
12 -----+
13 Seq Scan on apartments (cost=0.00..43700.00 rows=333333 width=158) |
14   Filter: (((full_address ->> 'apartment_number'::text))::integer < 300)|
15
16 QUERY PLAN
17 -----+
18 Seq Scan on apartments (cost=0.00..43700.00 rows=333333 width=158) (actual time=0.062..327.899 rows=299709 loops=1)|
19   Filter: (((full_address ->> 'apartment_number'::text))::integer < 300) |
20   Rows Removed by Filter: 700292 |
21 Planning Time: 0.040 ms |
22 Execution Time: 339.606 ms ||

```

Рисунок 3. План и выполнение запроса 1 до создания индекса

```

24 create index on apartments (full_address);
25
26 explain
27 select *
28 from apartments
29 where cast(full_address->>'apartment_number' as integer) < 300;
30
31 explain (ANALYZE)
32 select *
33 from apartments
34 where cast(full_address->>'apartment_number' as integer) < 300;
35
36 QUERY PLAN
37 -----+
38 Seq Scan on apartments (cost=0.00..43700.02 rows=333334 width=158) |
39   Filter: (((full_address ->> 'apartment_number'::text))::integer < 300)|
40
41 QUERY PLAN
42 -----+
43 Seq Scan on apartments (cost=0.00..43700.02 rows=333334 width=158) (actual time=0.031..324.752 rows=299709 loops=1)|
44   Filter: (((full_address ->> 'apartment_number'::text))::integer < 300) |
45   Rows Removed by Filter: 700292 |
46 Planning Time: 0.043 ms |
47 Execution Time: 336.084 ms |

```

Рисунок 4. План и выполнение запроса 1 после создания индекса

```

1  explain
2  select *
3  from payments
4  join apartments on apartments.id = payments.apartment_id
5  where apartments.full_address->>'street' = 'улица Льва Толстого' and payments.price > 3000;
6
7  explain (ANALYZE)
8  select *
9  from payments
10 join apartments on apartments.id = payments.apartment_id
11 where apartments.full_address->>'street' = 'улица Льва Толстого' and payments.price > 3000;
12
13 QUERY PLAN
14 -----
15 Gather  (cost=30976.04..190505.86 rows=8704 width=311)
16   Workers Planned: 2
17   -> Parallel Hash Join  (cost=29976.04..188635.46 rows=3627 width=311)
18     Hash Cond: (payments.apartment_id = apartments.id)
19     -> Parallel Seq Scan on payments  (cost=0.00..156755.51 rows=725294 width=153)
20         Filter: (price > 3000)
21     -> Parallel Hash  (cost=29950.01..29950.01 rows=2083 width=158)
22         -> Parallel Seq Scan on apartments  (cost=0.00..29950.01 rows=2083 width=158)
23             Filter: ((full_address ->> 'street'::text) = 'улица Льва Толстого'::text)
24
25 JIT:
26   Functions: 14
27   Options: Inlining false, Optimization false, Expressions true, Deforming true
28
29 QUERY PLAN
30 -----
31 Gather  (cost=30976.04..190505.86 rows=8704 width=311) (actual time=243.628..1135.171 rows=215853 loops=1)
32   Workers Planned: 2
33   Workers Launched: 2
34   -> Parallel Hash Join  (cost=29976.04..188635.46 rows=3627 width=311) (actual time=209.860..1044.985 rows=71951 loops=3)
35     Hash Cond: (payments.apartment_id = apartments.id)
36     -> Parallel Seq Scan on payments  (cost=0.00..156755.51 rows=725294 width=153) (actual time=0.563..627.313 rows=722785 loops=3)
37         Filter: (price > 3000)
38         Rows Removed by Filter: 943882
39     -> Parallel Hash  (cost=29950.01..29950.01 rows=2083 width=158) (actual time=208.639..208.639 rows=33136 loops=3)
40         Buckets: 131072 (originally 8192) Batches: 1 (originally 1) Memory Usage: 20736kB
41         -> Parallel Seq Scan on apartments  (cost=0.00..29950.01 rows=2083 width=158) (actual time=25.005..176.026 rows=33136 loops=3)
42             Filter: ((full_address ->> 'street'::text) = 'улица Льва Толстого'::text)
43             Rows Removed by Filter: 300198
44
45 Planning Time: 0.235 ms
46 JIT:
47   Functions: 42
48   Options: Inlining false, Optimization false, Expressions true, Deforming true
49   Timing: Generation 3.057 ms, Inlining 0.000 ms, Optimization 11.432 ms, Emission 63.633 ms, Total 78.122 ms
50 Execution Time: 1147.079 ms

```

Рисунок 5. План и выполнение запроса 2 до индекса

```

50 create index on payments (price);
51
52 explain
53 select *
54 from payments
55 join apartments on apartments.id = payments.apartment_id
56 where apartments.full_address->>'street' = 'улица Льва Толстого' and payments.price > 3000;
57
58 explain (ANALYZE)
59 select *
60 from payments
61 join apartments on apartments.id = payments.apartment_id
62 where apartments.full_address->>'street' = 'улица Льва Толстого' and payments.price > 3000;
63
64 QUERY PLAN
65 -----
66 Gather  (cost=30976.04..193082.40 rows=9530 width=311)
67   Workers Planned: 2
68   -> Parallel Hash Join  (cost=29976.04..191129.40 rows=3971 width=311)
69     Hash Cond: (payments.apartment_id = apartments.id)
70     -> Parallel Seq Scan on payments  (cost=0.00..159068.67 rows=794165 width=153)
71         Filter: (price > 3000)
72     -> Parallel Hash  (cost=29950.01..29950.01 rows=2083 width=158)
73         -> Parallel Seq Scan on apartments  (cost=0.00..29950.01 rows=2083 width=158)
74             Filter: ((full_address ->> 'street'::text) = 'улица Льва Толстого'::text)
75
76 JIT:
77   Functions: 14
78   Options: Inlining false, Optimization false, Expressions true, Deforming true
79
80 QUERY PLAN
81 -----
82 Gather  (cost=30976.04..193082.40 rows=9530 width=311) (actual time=191.520..715.632 rows=215853 loops=1)
83   Workers Planned: 2
84   Workers Launched: 2
85   -> Parallel Hash Join  (cost=29976.04..191129.40 rows=3971 width=311) (actual time=176.612..662.595 rows=71951 loops=3)
86     Hash Cond: (payments.apartment_id = apartments.id)
87     -> Parallel Seq Scan on payments  (cost=0.00..159068.67 rows=794165 width=153) (actual time=0.060..263.736 rows=722785 loops=3)
88         Filter: (price > 3000)
89         Rows Removed by Filter: 943882
90     -> Parallel Hash  (cost=29950.01..29950.01 rows=2083 width=158) (actual time=175.970..175.971 rows=33136 loops=3)
91         Buckets: 131072 (originally 8192) Batches: 1 (originally 1) Memory Usage: 20736kB
92         -> Parallel Seq Scan on apartments  (cost=0.00..29950.01 rows=2083 width=158) (actual time=13.873..152.083 rows=33136 loops=3)
93             Filter: ((full_address ->> 'street'::text) = 'улица Льва Толстого'::text)
94             Rows Removed by Filter: 300198
95
96 Planning Time: 0.286 ms
97 JIT:
98   Functions: 42
99   Options: Inlining false, Optimization false, Expressions true, Deforming true
100  Timing: Generation 2.980 ms, Inlining 0.000 ms, Optimization 1.699 ms, Emission 39.975 ms, Total 44.655 ms
101 Execution Time: 728.135 ms

```

Рисунок 6. План и выполнение запроса 2 после добавления индекса

## **Вывод**

Было сформировано понимание методов упрощения работы аналитика с БД.