Базы данных для индустриальных задач 2 Практическая работа №1

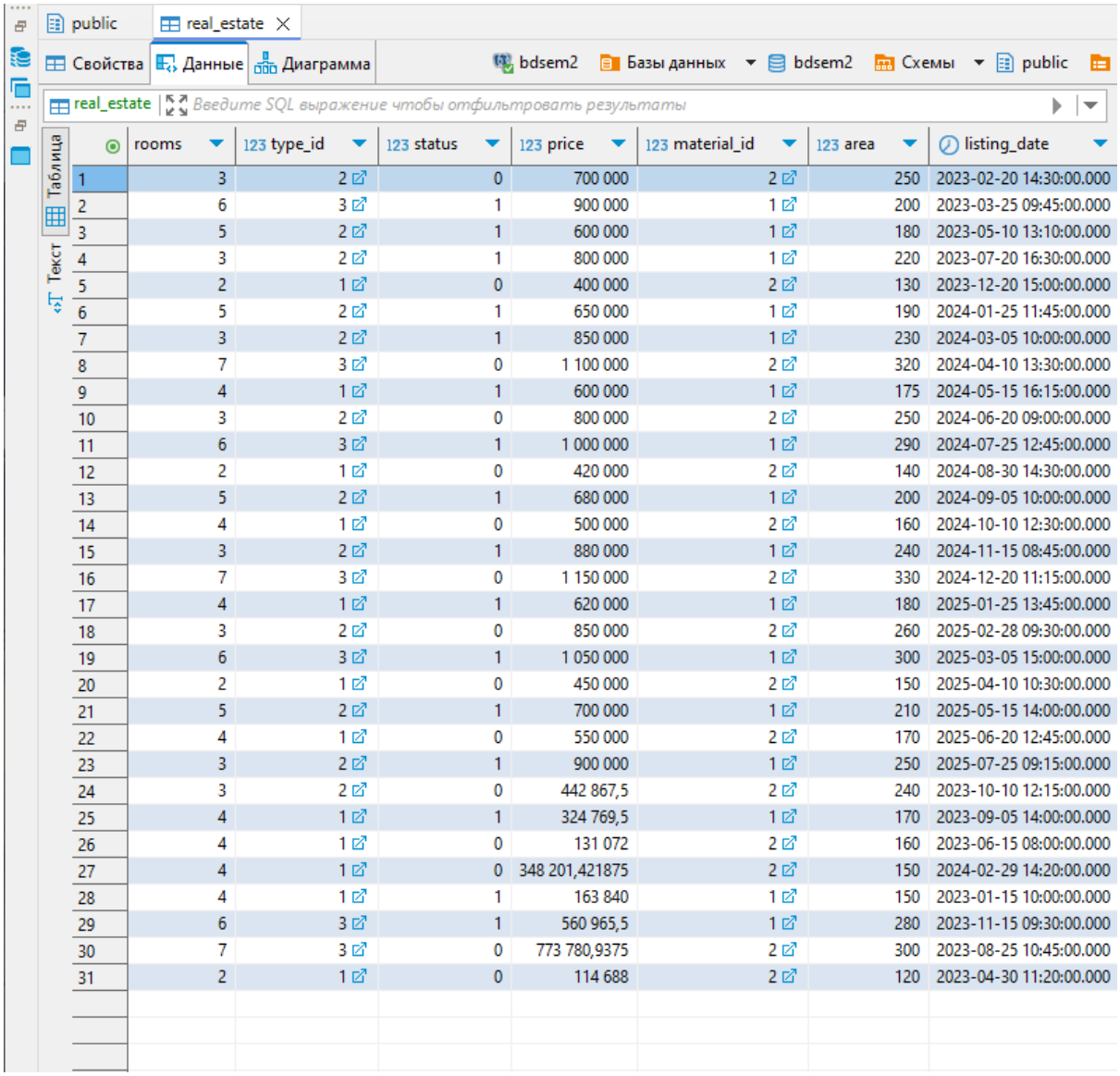
Муравьев Иван ЭФБО-02-22

Задание 1

Функция:

```
CREATE OR REPLACE FUNCTION public.adjust_building_prices()
RETURNS void AS $$
DECLARE
    v_real_estate_id INTEGER;
    v_listing_date TIMESTAMP;
    v_avg_rating NUMERIC;
    v_current_price NUMERIC;
BEGIN
    FOR v_real_estate_id, v_listing_date, v_avg_rating, v_current_price IN
        SELECT re.real_estate_id, re.listing_date, AVG(ev.score), re.price
        FROM real_estate re
        JOIN evaluation ev ON re.real_estate_id = ev.real_estate_id
            GROUP BY re.real_estate_id, re.listing_date, re.price
    LOOP
        IF age(v_listing_date::DATE) > INTERVAL '6 months' AND v_avg_rating < 6 THEN
            UPDATE real_estate
            SET price = v_current_price * 0.95
            WHERE real_estate.real_estate_id = v_real_estate_id;
        END IF;

        IF age(v_listing_date::DATE) > INTERVAL '9 months' AND v_avg_rating < 5 THEN
            UPDATE real_estate
            SET price = v_current_price * 0.90
            WHERE real_estate.real_estate_id = v_real_estate_id;
        END IF;

        IF age(v_listing_date::DATE) > INTERVAL '12 months' AND v_avg_rating < 4 THEN
            UPDATE real_estate
            SET price = v_current_price * 0.80
            WHERE real_estate.real_estate_id = v_real_estate_id;
        END IF;
    END LOOP;
END;
$$ LANGUAGE plpgsql;

SELECT adjust_building_prices();
```

Данные до выполнения функции:



public | real_estate

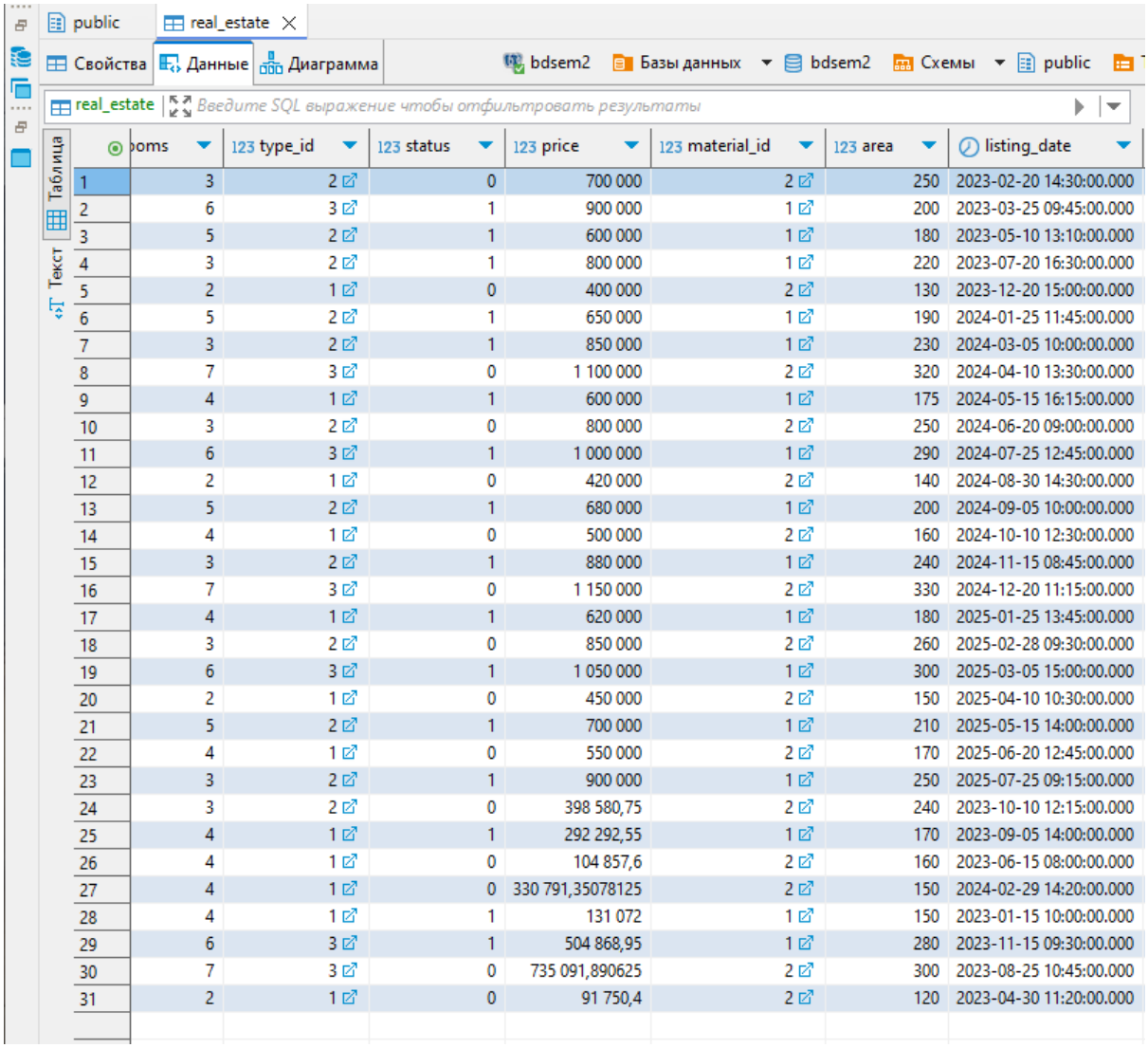
Свойства | Данные | Диаграмма | bdsem2 | Базы данных | bdsem2 | Схемы | public

real_estate | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст

	rooms	type_id	status	price	material_id	area	listing_date
1	3	2	0	700 000	2	250	2023-02-20 14:30:00.000
2	6	3	1	900 000	1	200	2023-03-25 09:45:00.000
3	5	2	1	600 000	1	180	2023-05-10 13:10:00.000
4	3	2	1	800 000	1	220	2023-07-20 16:30:00.000
5	2	1	0	400 000	2	130	2023-12-20 15:00:00.000
6	5	2	1	650 000	1	190	2024-01-25 11:45:00.000
7	3	2	1	850 000	1	230	2024-03-05 10:00:00.000
8	7	3	0	1 100 000	2	320	2024-04-10 13:30:00.000
9	4	1	1	600 000	1	175	2024-05-15 16:15:00.000
10	3	2	0	800 000	2	250	2024-06-20 09:00:00.000
11	6	3	1	1 000 000	1	290	2024-07-25 12:45:00.000
12	2	1	0	420 000	2	140	2024-08-30 14:30:00.000
13	5	2	1	680 000	1	200	2024-09-05 10:00:00.000
14	4	1	0	500 000	2	160	2024-10-10 12:30:00.000
15	3	2	1	880 000	1	240	2024-11-15 08:45:00.000
16	7	3	0	1 150 000	2	330	2024-12-20 11:15:00.000
17	4	1	1	620 000	1	180	2025-01-25 13:45:00.000
18	3	2	0	850 000	2	260	2025-02-28 09:30:00.000
19	6	3	1	1 050 000	1	300	2025-03-05 15:00:00.000
20	2	1	0	450 000	2	150	2025-04-10 10:30:00.000
21	5	2	1	700 000	1	210	2025-05-15 14:00:00.000
22	4	1	0	550 000	2	170	2025-06-20 12:45:00.000
23	3	2	1	900 000	1	250	2025-07-25 09:15:00.000
24	3	2	0	442 867,5	2	240	2023-10-10 12:15:00.000
25	4	1	1	324 769,5	1	170	2023-09-05 14:00:00.000
26	4	1	0	131 072	2	160	2023-06-15 08:00:00.000
27	4	1	0	348 201,421875	2	150	2024-02-29 14:20:00.000
28	4	1	1	163 840	1	150	2023-01-15 10:00:00.000
29	6	3	1	560 965,5	1	280	2023-11-15 09:30:00.000
30	7	3	0	773 780,9375	2	300	2023-08-25 10:45:00.000
31	2	1	0	114 688	2	120	2023-04-30 11:20:00.000

Данные после выполнения функции:



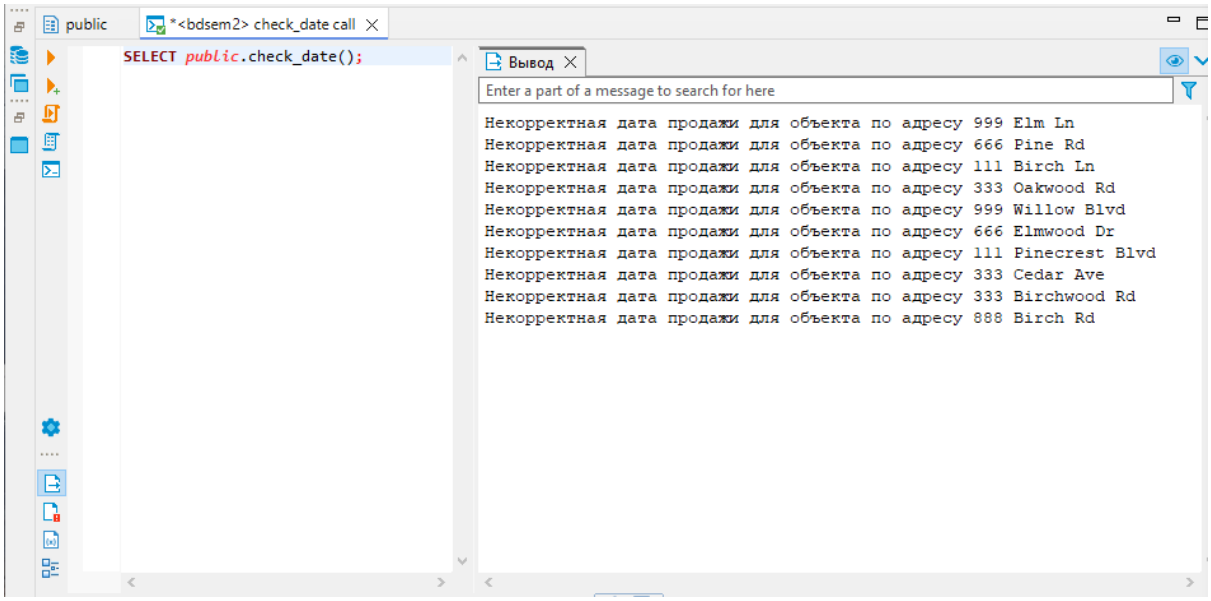
	oms	type_id	status	price	material_id	area	listing_date
1	3	2	0	700 000	2	250	2023-02-20 14:30:00.000
2	6	3	1	900 000	1	200	2023-03-25 09:45:00.000
3	5	2	1	600 000	1	180	2023-05-10 13:10:00.000
4	3	2	1	800 000	1	220	2023-07-20 16:30:00.000
5	2	1	0	400 000	2	130	2023-12-20 15:00:00.000
6	5	2	1	650 000	1	190	2024-01-25 11:45:00.000
7	3	2	1	850 000	1	230	2024-03-05 10:00:00.000
8	7	3	0	1 100 000	2	320	2024-04-10 13:30:00.000
9	4	1	1	600 000	1	175	2024-05-15 16:15:00.000
10	3	2	0	800 000	2	250	2024-06-20 09:00:00.000
11	6	3	1	1 000 000	1	290	2024-07-25 12:45:00.000
12	2	1	0	420 000	2	140	2024-08-30 14:30:00.000
13	5	2	1	680 000	1	200	2024-09-05 10:00:00.000
14	4	1	0	500 000	2	160	2024-10-10 12:30:00.000
15	3	2	1	880 000	1	240	2024-11-15 08:45:00.000
16	7	3	0	1 150 000	2	330	2024-12-20 11:15:00.000
17	4	1	1	620 000	1	180	2025-01-25 13:45:00.000
18	3	2	0	850 000	2	260	2025-02-28 09:30:00.000
19	6	3	1	1 050 000	1	300	2025-03-05 15:00:00.000
20	2	1	0	450 000	2	150	2025-04-10 10:30:00.000
21	5	2	1	700 000	1	210	2025-05-15 14:00:00.000
22	4	1	0	550 000	2	170	2025-06-20 12:45:00.000
23	3	2	1	900 000	1	250	2025-07-25 09:15:00.000
24	3	2	0	398 580,75	2	240	2023-10-10 12:15:00.000
25	4	1	1	292 292,55	1	170	2023-09-05 14:00:00.000
26	4	1	0	104 857,6	2	160	2023-06-15 08:00:00.000
27	4	1	0	330 791,35078125	2	150	2024-02-29 14:20:00.000
28	4	1	1	131 072	1	150	2023-01-15 10:00:00.000
29	6	3	1	504 868,95	1	280	2023-11-15 09:30:00.000
30	7	3	0	735 091,890625	2	300	2023-08-25 10:45:00.000
31	2	1	0	91 750,4	2	120	2023-04-30 11:20:00.000

Задание 2

Функция:

```sql
create or replace function public.check_date()
returns void as $$
declare
    v_invalid_address text;
begin
    for v_invalid_address in
        select re.address
        from real_estate re
        join sale s on re.real_estate_id = s.real_estate_id
        where s.sale_date < re.listing_date
    loop
        raise notice 'Некорректная дата продажи для объекта по адресу %', v_invalid_address;
    end loop;

    if not found then
        raise notice 'Некорректных дат продажи объектов нет';
    end if;
end;
$$ language plpgsql;

select public.check_date();
```

Вывод функции:



Задание 3

Скрипт для создания нового отношения:

<bdsem2> Insert_Price_dynamic_Script <bdsem2> Create_Price_Dynamic_Script

```sql
create table public.price_dynamic
(
    real_estate_id serial primary key,
    change_date timestamp,
    new_price real
);
```

Скрипт для добавления записей в новое отношение:

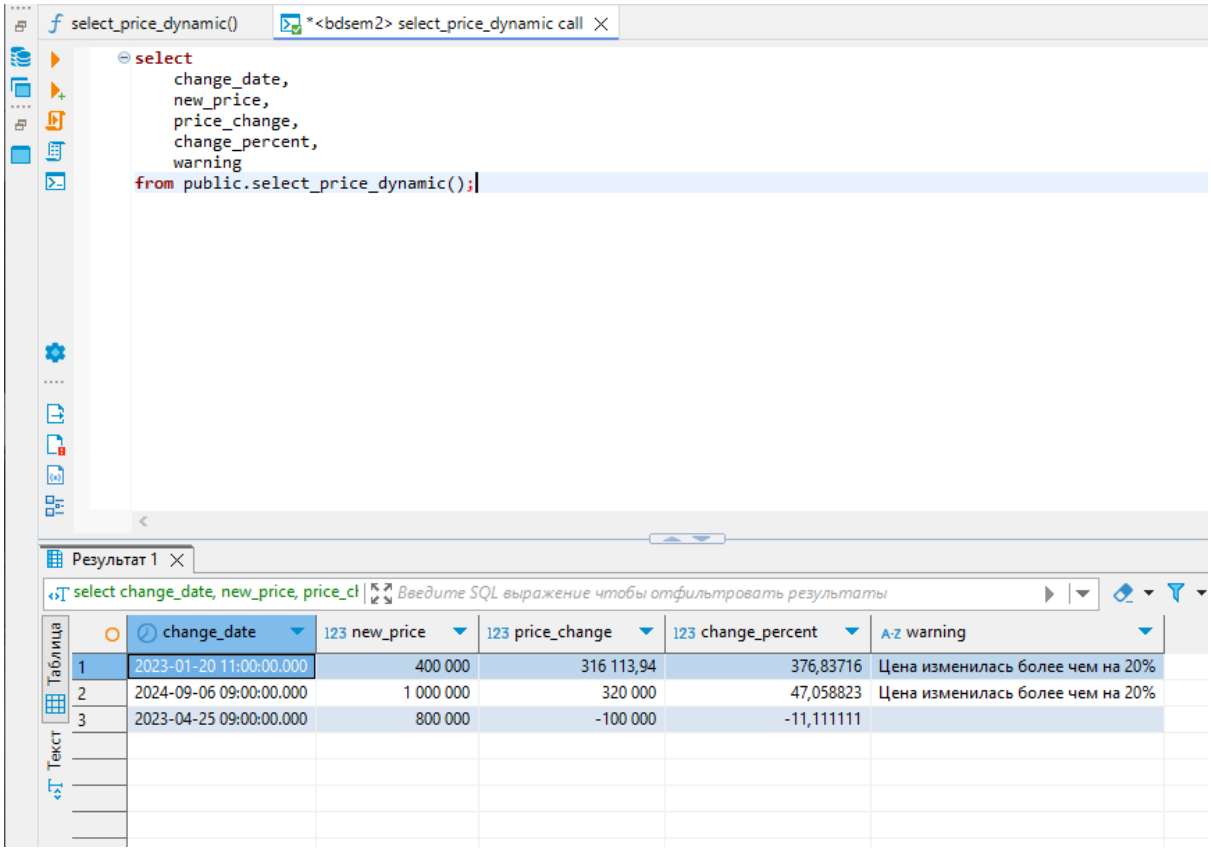
<bdsem2> Insert_Price_dynamic_Script

```sql
insert into public.price_dynamic (real_estate_id, new_price, change_date)
values
(1, 400000, '2023-01-20 11:00:00.000'),
(21, 1000000, '2024-09-06 09:00:00.000'),
(3, 800000, '2023-04-25 09:00:00.000');
```

Функция:

```sql
create or replace function select_price_dynamic()
RETURNS table
(
    change_date timestamp,
    new_price real,
    price_change real,
    change_percent real,
    warning text
) as $$
begin
    return QUERY
    select pd.change_date,
           pd.new_price,
           pd.new_price - re.price::real as price_change,
           (((pd.new_price - re.price) / re.price) * 100)::real as change_percent,
        case
           when ((pd.new_price - re.price) / re.price) * 100 > 20 then 'Цена изменилась более чем на 20%'
           else ''
       END AS warning
    from price_dynamic pd
    join real_estate re on pd.real_estate_id = re.real_estate_id;
end;
$$ language plpgsql;

select * from select_price_dynamic();
```

Вывод функции:

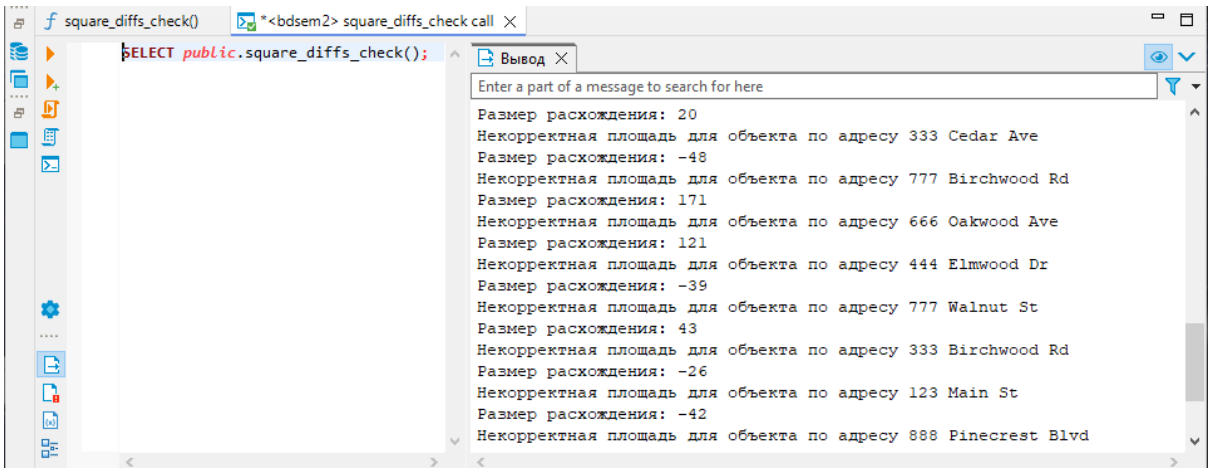


Задание 4

Функция:

```sql
create or replace function public.square_diffs_check()
returns void as $$
declare
    invalid_address text;
    area_difference real;
begin
    for invalid_address, area_difference in
        select re.address,
               re.area - (
                select sum(ps.area)
                from property_structure ps
                where ps.real_estate_id = re.real_estate_id
               ) as area_difference
        from real_estate re
        where re.area <> (
            select sum(ps.area)
            from property_structure ps
            where ps.real_estate_id = re.real_estate_id
        )
    loop
        raise notice 'Некорректная площадь для объекта по адресу %', invalid_address;
        raise notice 'Размер расхождения: %', area_difference;
    end loop;
    if not found then
        raise notice 'Некорректных площадей продажи объектов нет';
    end if;
end;

$$ language plpgsql;

select public.square_diffs_check();
```

Вывод функции:

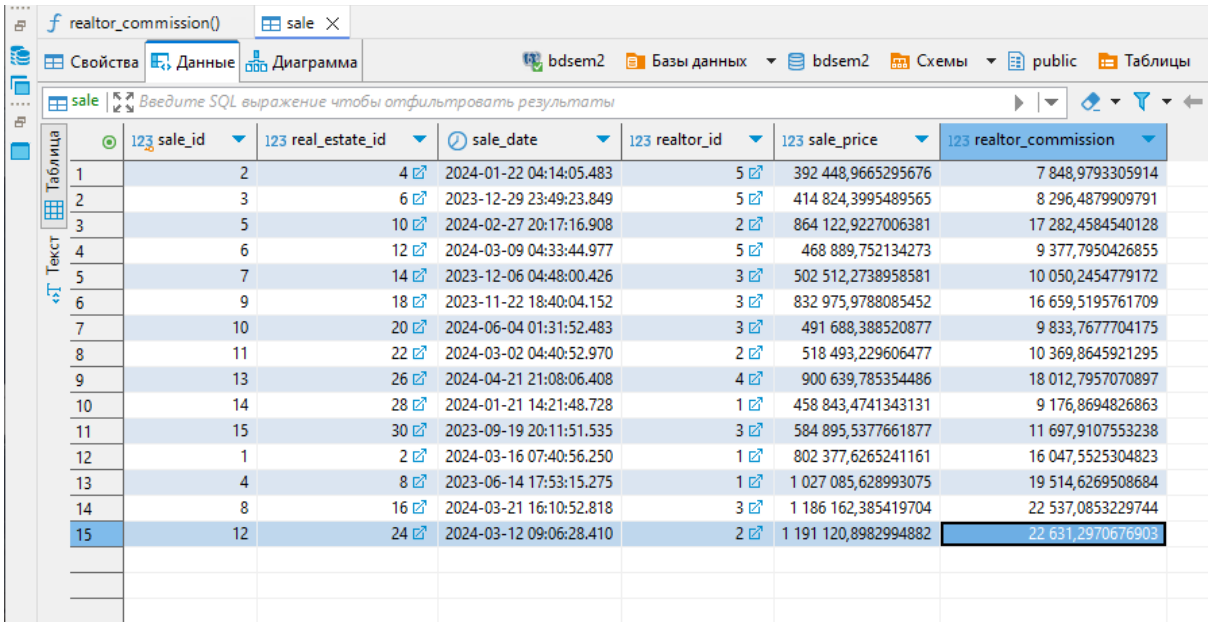


Задание 5

Функция:

```sql
create or replace function public.realtor_commission()
returns void as $$
declare
    sales_price double precision;
begin
    for sales_price in
        select s.sale_price
        from sale s
        loop
            update sale
            set realtor_commission = sale_price * 0.02
            where sale_price < 1000000;

            update sale
            set realtor_commission = sale_price * 0.019
            where sale_price > 1000000 and sale_price < 3000000;

            update sale
            set realtor_commission = sale_price * 0.017
            where sale_price > 3000000;
        end loop;
end;

$$ language plpgsql;

select * from public.realtor_commission();
```

Данные в таблице после выполнения функции:



realtor_commission() | sale

Свойства | Данные | Диаграмма | bdsem2 | Базы данных | bdsem2 | Схемы | public | Таблицы

sale | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст

	sale_id	real_estate_id	sale_date	realtor_id	sale_price	realtor_commission
1	2	4	2024-01-22 04:14:05.483	5	392 448,9665295676	7 848,9793305914
2	3	6	2023-12-29 23:49:23.849	5	414 824,3995489565	8 296,4879909791
3	5	10	2024-02-27 20:17:16.908	2	864 122,9227006381	17 282,4584540128
4	6	12	2024-03-09 04:33:44.977	5	468 889,752134273	9 377,7950426855
5	7	14	2023-12-06 04:48:00.426	3	502 512,2738958581	10 050,2454779172
6	9	18	2023-11-22 18:40:04.152	3	832 975,9788085452	16 659,5195761709
7	10	20	2024-06-04 01:31:52.483	3	491 688,388520877	9 833,7677704175
8	11	22	2024-03-02 04:40:52.970	2	518 493,229606477	10 369,8645921295
9	13	26	2024-04-21 21:08:06.408	4	900 639,785354486	18 012,7957070897
10	14	28	2024-01-21 14:21:48.728	1	458 843,4741343131	9 176,8694826863
11	15	30	2023-09-19 20:11:51.535	3	584 895,5377661877	11 697,9107553238
12	1	2	2024-03-16 07:40:56.250	1	802 377,6265241161	16 047,5525304823
13	4	8	2023-06-14 17:53:15.275	1	1 027 085,628993075	19 514,6269508684
14	8	16	2024-03-21 16:10:52.818	3	1 186 162,385419704	22 537,0853229744
15	12	24	2024-03-12 09:06:28.410	2	1 191 120,8982994882	22 631,2970676903