|  |
| --- |
| https://lh6.googleusercontent.com/QcftzNtI05T0Y6fjdSh1Rr2rt8oqZ1IvnLvbn1jLJ7CCyteVir3k-xBLv4SL1wAgWJsRhmmJSR0UW-RP63_GQenE4vVWv05BRoZTsmIcBccVTnfxwmsnNMvjg599x9SqZd8E3dkd |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА - Российский технологический университет»**  **РТУ МИРЭА** |

Институт перспективных технологий и индустриального программирования (ИПТИП)

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКИМ РАБОТАМ** | |
| **по дисциплине** | |
| «Базы данных и анализ промышленных данных» | |
| **Практическая работа №6** | |
| Выполнил студент группы ЭФМО-02-23 | Мурадов Н.Н. |

Москва 2023

**СОДЕРЖАНИЕ**

[Задача 3](#_Toc153912351)

[Решение 4](#_Toc153912352)

[Результаты 9](#_Toc153912353)

Задача

1. Конвертирует стоимость объекта недвижимости в евро и долларах.

|  |  |
| --- | --- |
| **Вх. параметр** | **Вых. параметр** |
| курс, код объекта недвижимости | стоимость в евро |

1. Рассчитывает заработную плату риэлтора по формуле: N\*S+R, где N

– общая сумма проданных объектов недвижимости в месяц (подсчет осуществляется автоматически по данным таблицы «Продажи» с использованием агрегатной функции), S – коэффициент, R – премия.

|  |  |
| --- | --- |
| **Вх. параметр** | **Вых. параметр** |
| S, R, начальная и конечная дата, фамилия риэлтора | размер заработной платы |

1. Добавить таблицу «Заработная плата риэлтора», содержащая сведения: год, месяц, размер выплаты. Изменить тело функции из пункта 2 таким образом, чтобы рассчитанная заработная плата сохранялась в этой таблице.
2. Рассчитывает процент изменения продажной стоимости объекта недвижимости от первоначально заявленной и срок продажи (в месяцах)
3. Изменить функцию, созданную в пункте 5 таким образом, чтобы в зависимости от срока продажи выводилось сообщение

|  |  |
| --- | --- |
| **Срок** | **Сообщение** |
| менее 3 месяцев | Очень быстро |
| от 3 месяцев до 6 месяцев | Быстро |
| от 6 месяцев до 1 года | Долго |
| от 1 года | Очень долго |

1. Формирует список средних оценок по каждому критерию для объекта недвижимости.
2. Написать функцию, которая рассчитывает сумму ежемесячного платежа по ипотеке. **Входные параметры:** код объекта недвижимости, процентная ставка, срок, первоначальный взнос.
3. Написать функцию, которая рассчитывает сумму налога на недвижимость.
4. Добавить таблицу «Динамика цен», где будет хранится изменения стоимости. Таблица будет содержать следующие колонки: код объекта недвижимости, новая стоимость, дата изменения.

Написать функцию, которая будет возвращаться следующий результат:

|  |  |  |  |
| --- | --- | --- | --- |
| Дата | Новая стоимость | Изменение | % Изменения |
| 25.03.2022 | 1000000 | 0 | 0 |
| 28.03.2022 | 1500000 | 500000 | +50 |
| 01.04.2022 | 1300000 | -300000 | -13,33 |

% изменения рассчитывается по формуле:

**((V2-V1)/V1) × 100**. Где **V1** – старое значение, а **V2** – новое

Решение

Листинг кода:

1:

CREATE OR REPLACE FUNCTION getExchangesRate(bigint, float, float)

RETURNS TABLE("В рублях" BIGINT, "В долларах" FLOAT, "В евро" FLOAT) AS $$

SELECT PRICE, rnd(price/$2, 2), rnd(price/$3, 2) FROM REALTY\_OBJECT r WHERE ID = $1;

$$ language SQL;

DROP FUNCTION getexchangesrate(bigint,double precision,double precision);

SELECT \* FROM getExchangesRate(124, 88.81, 96.92);

2:

CREATE OR REPLACE FUNCTION getSalary(S float, R bigint, f text, beg TIMESTAMP, ends TIMESTAMP)

RETURNS TABLE(idrr BIGINT, last\_name character(15), y integer, m integer, cnt BIGINT

, arrids BIGINT[], arridr BIGINT[], sumobj text, "Сумма заработной платы" text) AS $$

BEGIN

DROP TABLE IF EXISTS x, x1;

create temporary table x as SELECT rR.ID idRR, rR.LAST\_NAME, to\_char(DATE\_OF\_SALE, 'yyyy')::INT y, to\_char(DATE\_OF\_SALE, 'mm')::INT m, s.id idS, s.REALTY\_OBJECT\_ID idR FROM SALE s, REALTOR rR WHERE REALTOR\_ID = rR.ID AND rR.LAST\_NAME = f AND DATE\_OF\_SALE BETWEEN beg AND ends;

create temporary table x1 as SELECT x.idRR, x.LAST\_NAME, x.y, x.m, COUNT(x.m), array\_agg(idS) arrids, array\_agg(idR), to\_char(SUM(PRICE), '999 999 999') sumObj FROM x, REALTY\_OBJECT r WHERE idR = r.ID GROUP BY x.idRR, x.LAST\_NAME, x.y, x.m;

return query(SELECT x1.\*, to\_char(replace(x1.sumObj, ' ', '')::INT\*S+R, '999 999.99') FROM x1);

END $$ language plpgsql;

SELECT \* FROM getSalary(0.002, 40000, 'Мякина', '2017-05-07', '2023-11-24');

DROP FUNCTION getSalary(S float, R bigint, f text, beg TIMESTAMP, ends TIMESTAMP);

3:

CREATE TABLE REALTORS\_SALARY (

ID BIGINT PRIMARY KEY,

REALTOR\_ID BIGINT REFERENCES REALTOR (ID),

YEAR BIGINT,

MONTH BIGINT,

VALUE FLOAT

);

CREATE OR REPLACE FUNCTION getSalary3(S float, R bigint, f text, beg TIMESTAMP, ends TIMESTAMP)

RETURNS TABLE(ID BIGINT, REALTOR\_ID BIGINT, YEAR BIGINT, MONTH BIGINT, VALUE FLOAT) AS $$

BEGIN

DROP TABLE IF EXISTS x, x1;

create temporary table x as SELECT rR.ID idRR, rR.LAST\_NAME, to\_char(DATE\_OF\_SALE, 'yyyy')::INT y, to\_char(DATE\_OF\_SALE, 'mm')::INT m, s.id idS, s.REALTY\_OBJECT\_ID idR FROM SALE s, REALTOR rR WHERE s.REALTOR\_ID = rR.ID AND rR.LAST\_NAME = f AND DATE\_OF\_SALE BETWEEN beg AND ends;

create temporary table x1 as SELECT x.idRR, x.LAST\_NAME, x.y, x.m, COUNT(x.m), array\_agg(idS) arrids, array\_agg(idR), to\_char(SUM(PRICE), '999 999 999') sumObj FROM x, REALTY\_OBJECT r WHERE idR = r.ID GROUP BY x.idRR, x.LAST\_NAME, x.y, x.m;

INSERT INTO REALTORS\_SALARY SELECT

ROW\_NUMBER() over() + 508, idRR, y, m, rnd(replace(sumObj, ' ', '')::INT\*S+R, 2)

from x1;

RETURN query(SELECT \* FROM REALTORS\_SALARY);

END $$ language plpgsql;

SELECT \* FROM getSalary3(0.002, 40000, 'Мякина', '2017-05-07', '2023-11-24');

DROP FUNCTION getSalary3(S float, R bigint, f text, beg TIMESTAMP, ends TIMESTAMP);

DROP TABLE REALTORS\_SALARY;

delete FROM REALTORS\_SALARY WHERE ID>508;

4:

CREATE OR REPLACE FUNCTION getInfo6\_4()

RETURNS TABLE(sP double precision, rP double precision, dR TIMESTAMP, dS TIMESTAMP, "%" FLOAT, getDiffMonth NUMERIC) AS $$

BEGIN

DROP TABLE IF EXISTS x;

create temporary table x as SELECT s.PRICE sP, r.PRICE rP, DATE\_OF\_ANNOUNCEMENT dR, DATE\_OF\_SALE dS FROM SALE s, REALTY\_OBJECT r WHERE REALTY\_OBJECT\_ID = r.ID;

RETURN query(SELECT x.\*, rnd((x.sP-x.rP)/x.rP \* 100, 2), getDiffMonth(x.dS, x.dR) FROM x);

END $$ language plpgsql;

SELECT \* FROM getInfo6\_4();

DROP FUNCTION getInfo6\_4();

5:

CREATE OR REPLACE FUNCTION getInfo6\_5()

RETURNS TABLE(sP double precision, rP double precision, dR TIMESTAMP, dS TIMESTAMP, "%" FLOAT, getDiffMonth NUMERIC, "Срок" text) AS $$

BEGIN

DROP TABLE IF EXISTS x, x1;

create temporary table x as SELECT s.PRICE sP, r.PRICE rP, DATE\_OF\_ANNOUNCEMENT dR, DATE\_OF\_SALE dS FROM SALE s, REALTY\_OBJECT r WHERE REALTY\_OBJECT\_ID = r.ID;

create temporary table x1 as SELECT x.\*, rnd((x.sP-x.rP)/x.rP \* 100, 2), getDiffMonth(x.dS, x.dR) dif FROM x;

RETURN query(SELECT \*, (CASE

WHEN x1.dif < 3 THEN 'Очень быстро'

WHEN x1.dif >= 3 AND x1.dif < 6 THEN 'Быстро'

WHEN x1.dif >= 6 AND x1.dif < 12 THEN 'Долго'

WHEN x1.dif >= 12 THEN 'Очень долго'

ELSE ''

END) FROM x1 limit 30);

END $$ language plpgsql;

SELECT \* FROM getInfo6\_5();

DROP FUNCTION getInfo6\_5();

6:

CREATE OR REPLACE FUNCTION getInfo6\_6(idRe BIGINT)

RETURNS TABLE("Критерий" character(30), "Средняя оценка" text) AS $$

BEGIN

DROP TABLE IF EXISTS x;

create temporary table x as SELECT NAME, RND(AVG(VALUE), 1) avg FROM Score s, REALTY\_OBJECT r, ASSESSMENT\_CRITERION asCr WHERE REALTY\_OBJECT\_ID = idRe AND REALTY\_OBJECT\_ID = r.ID AND ASSESSMENT\_CRITERION\_ID = asCr.ID GROUP BY NAME;

RETURN query(SELECT x.NAME, (x.avg || ' из 5') FROM x);

END $$ language plpgsql;

SELECT \* FROM getInfo6\_6(20);

DROP FUNCTION getInfo6\_6(idRe BIGINT);

7:

CREATE OR REPLACE FUNCTION getInfo6\_7(idRe BIGINT, per FLOAT, y BIGINT, fPay DOUBLE PRECISION)

RETURNS DOUBLE PRECISION AS $$

DECLARE sP DOUBLE PRECISION;

rez DOUBLE PRECISION;

i FLOAT;

n BIGINT;

BEGIN

sP := (SELECT s.PRICE sP FROM SALE s, REALTY\_OBJECT r WHERE r.ID = idRe AND REALTY\_OBJECT\_ID = r.ID limit 1);

sP := sP - fPay;

i := per / 100 / 12;

n := y \* 12;

rez := sP \* (i + i/(POWER(1+i, n) - 1));

RETURN rnd(rez, 2);

END $$ language plpgsql;

SELECT \* FROM getInfo6\_7(20, 10.0, 20, 1000000);

DROP FUNCTION getInfo6\_7(idRe BIGINT, per FLOAT, y BIGINT, fPay DOUBLE PRECISION);

8:

CREATE OR REPLACE FUNCTION getInfo6\_8(idRe BIGINT, lastTax DOUBLE PRECISION)

RETURNS DOUBLE PRECISION AS $$

DECLARE sP DOUBLE PRECISION;

rez DOUBLE PRECISION;

rate FLOAT;

BEGIN

sP := (SELECT s.PRICE sP FROM SALE s, REALTY\_OBJECT r WHERE r.ID = idRe AND REALTY\_OBJECT\_ID = r.ID limit 1);

rate := (SELECT CASE

WHEN sP < 10000000 THEN 0.1

WHEN sP >= 10000000 AND sP < 20000000 THEN 0.15

WHEN sP >= 20000000 AND sP < 50000000 THEN 0.2

WHEN sP >= 50000000 AND sP < 300000000 THEN 0.3

END);

sP := sP / rand(3, 10);

rez := (sP \* rate - lastTax)\* 0.05 + lastTax;

RETURN rnd(rez, 2);

END $$ language plpgsql;

SELECT \* FROM getInfo6\_8(20, 15000);

DROP FUNCTION getInfo6\_8(idRe BIGINT, lastTax DOUBLE PRECISION);

9:

CREATE TABLE PRICE\_MOVEMENT (

ID BIGINT PRIMARY KEY,

REALTY\_OBJECT\_ID BIGINT REFERENCES REALTY\_OBJECT (ID),

NEW\_PRICE DOUBLE PRECISION,

LAGG DOUBLE PRECISION,

DATE\_OF\_CHANGE TIMESTAMP

);

CREATE OR REPLACE FUNCTION getInfo6\_9(idRe BIGINT)

RETURNS TABLE("Дата" TIMESTAMP, "Новая стоимость" DOUBLE PRECISION

, "Изменение" DOUBLE PRECISION, "% Изменения" FLOAT) AS $$

BEGIN

DROP TABLE x;

create temporary table x as SELECT r.ID rID, s.PRICE sP, s.DATE\_OF\_SALE dS, r.PRICE rP FROM SALE s, REALTY\_OBJECT r WHERE r.ID = idRe AND REALTY\_OBJECT\_ID = r.ID;

INSERT INTO PRICE\_MOVEMENT SELECT

ROW\_NUMBER() over() + 524, x.rID, x.sP, LAG(x.sP) OVER(ORDER BY x.dS), x.dS

from x;

RETURN query(SELECT DATE\_OF\_CHANGE, NEW\_PRICE, coalesce(NEW\_PRICE - LAGG, 0), coalesce(rnd((NEW\_PRICE-LAGG)/LAGG \* 100, 2), 0) FROM PRICE\_MOVEMENT ORDER BY DATE\_OF\_CHANGE);

END $$ language plpgsql;

SELECT \* FROM getInfo6\_9(20);

--SELECT \* FROM PRICE\_MOVEMENT;

delete FROM PRICE\_MOVEMENT WHERE ID>524;

DROP FUNCTION getInfo6\_9(idRe BIGINT);

delete FROM PRICE\_MOVEMENT WHERE ID>524;

DROP TABLE PRICE\_MOVEMENT;

Результаты

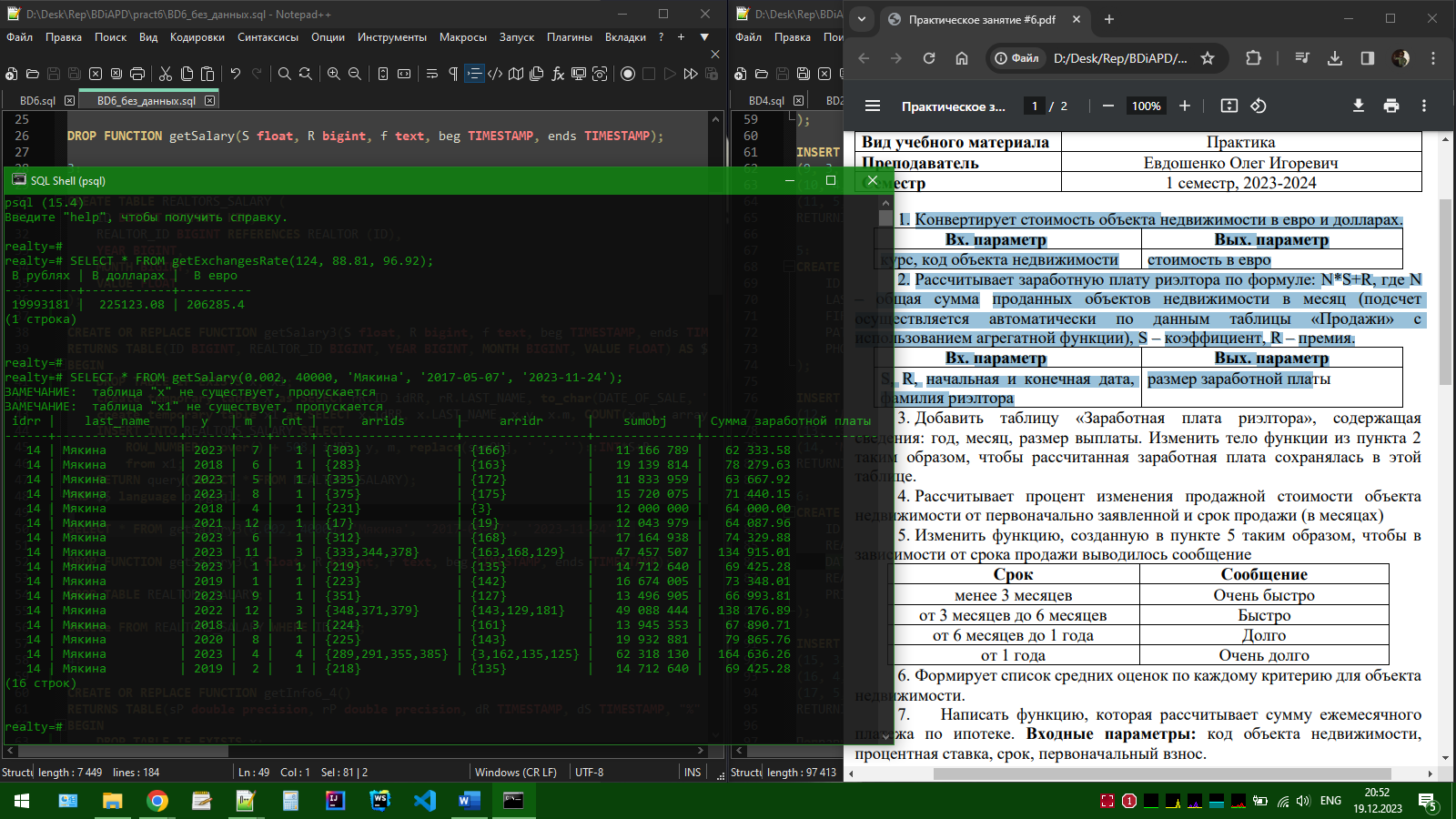


Рисунок 1 – Результат к 1 и 2 задаче

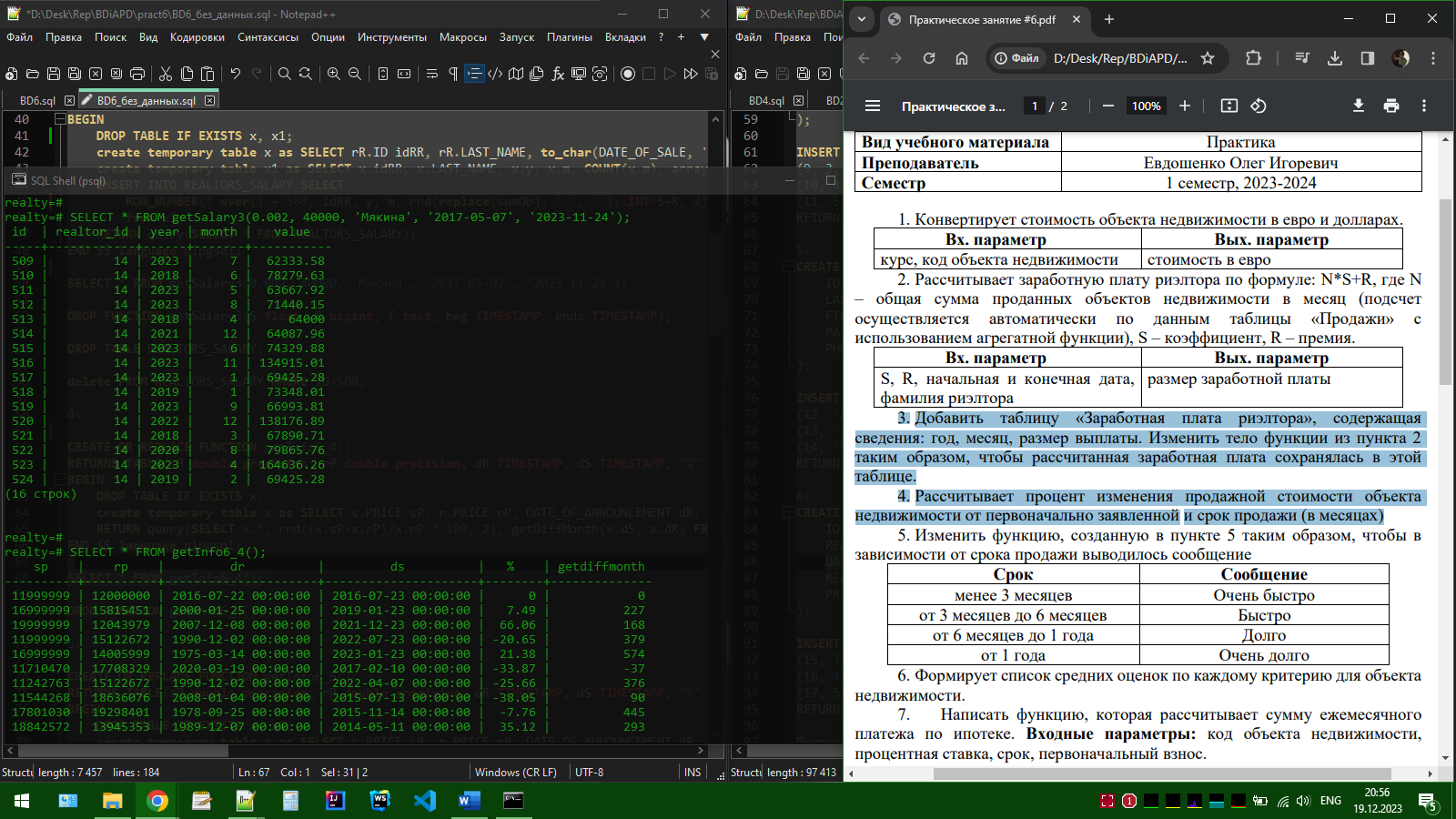


Рисунок 2 – Результат к 3 и 4 задаче

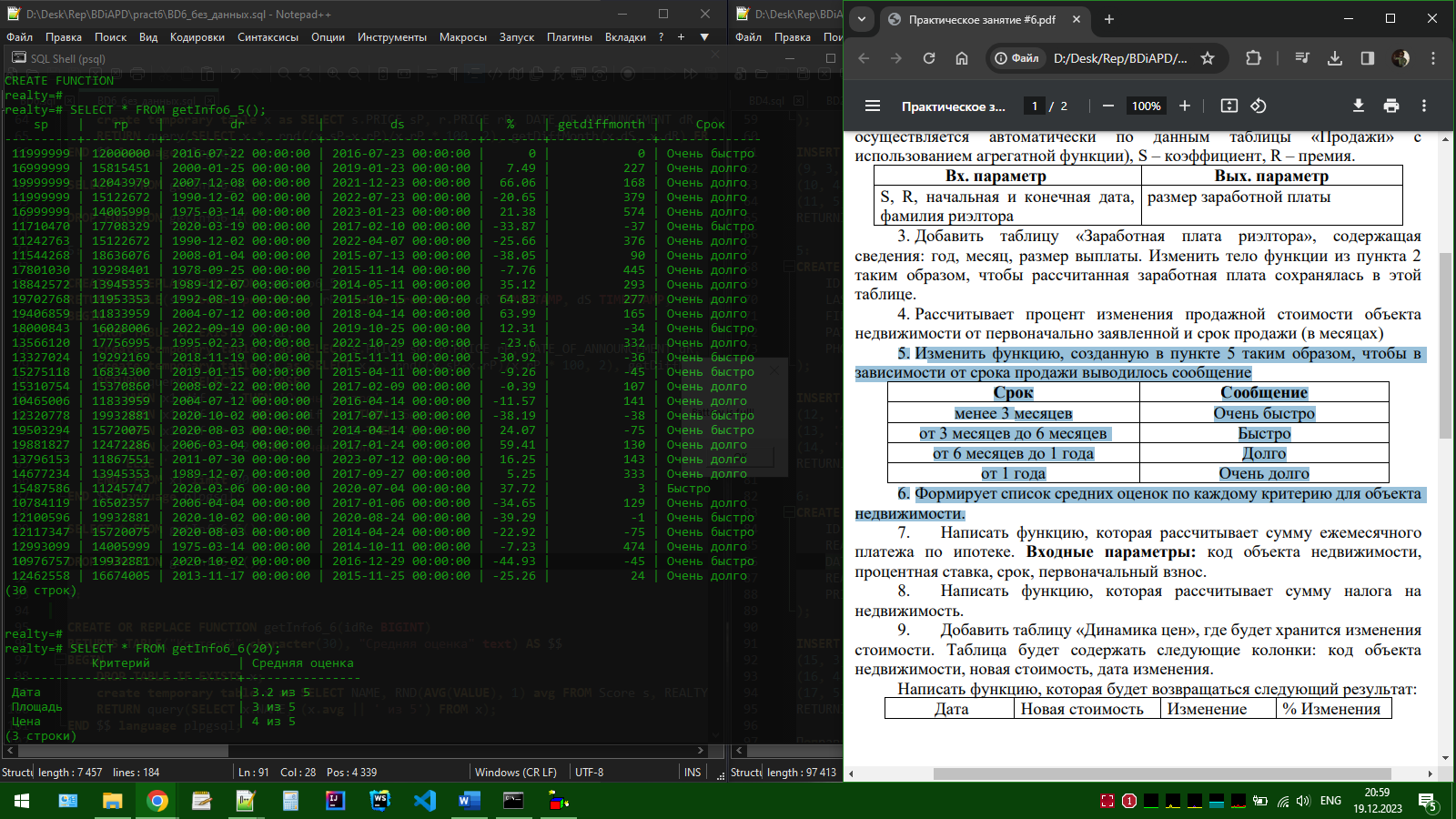


Рисунок 3 – Результат к 5 и 6 задаче

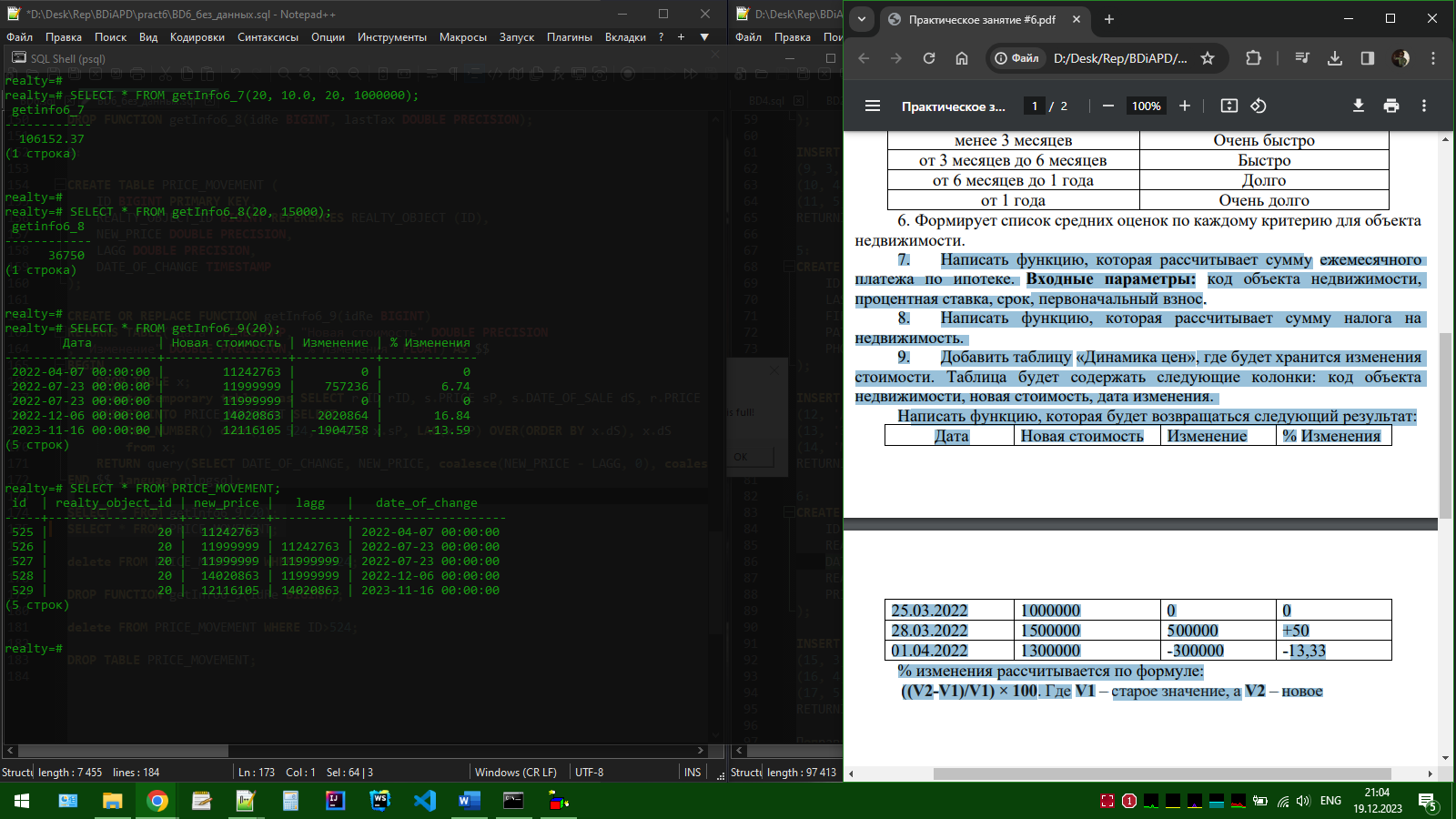


Рисунок 4 – Результат к 7, 8 и 9 задаче