**Домашнее задание № 10**

**Лабораторная работа по теме «Язык PL/pgSQL»**

**Автор: Сорокин Максим Евгеньевич**

**set** search\_path **to** lesson10;

**create** **table** student

(

id serial **primary** **key**,

**name** **text** **not** **null**,

total\_score **integer** **not** **null** **default** 0

);

**insert** **into** student(id, **name**, total\_score) **values** (10, 'Иван Иванов', 0);

**insert** **into** student(id, **name**, total\_score) **values** (20, 'Екатерина Андреева', 0);

**insert** **into** student(id, **name**, total\_score) **values** (30, 'Анна Потапова', 0);

**insert** **into** student(id, **name**, total\_score) **values** (40, 'Ильяс Мухаметшин', 0);

**insert** **into** student(id, **name**, total\_score) **values** (50, 'Сергей Петров', 0);

**create** **table** activity\_scores

(

id serial **primary** **key**,

student\_id **integer** **not** **null**,

activity\_type **text** **not** **null**,

score **integer** **not** **null** **default** 0,

**constraint** fk\_student\_id **foreign** **key** (student\_id) **references** student(id)

);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (10, 'Queries', 5);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (10, 'Queries', 5);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (20, 'Exam', 45);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (20, 'Certification', 50);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (30, 'Homework', 15);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (30, 'Queries', 5);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (30, 'Homework', 10);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (40, 'Exam', 45);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (40, 'Laboratory', 25);

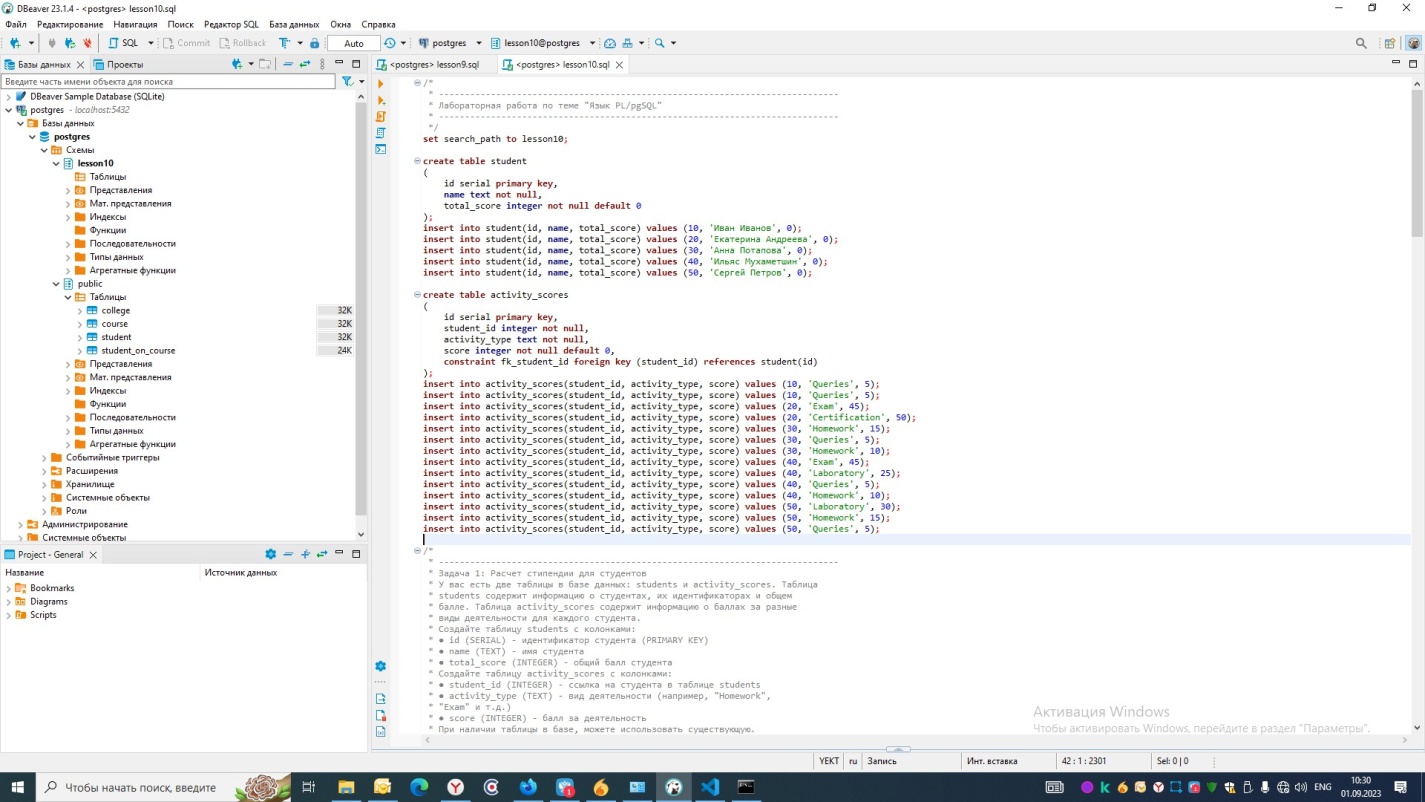
**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (40, 'Queries', 5);

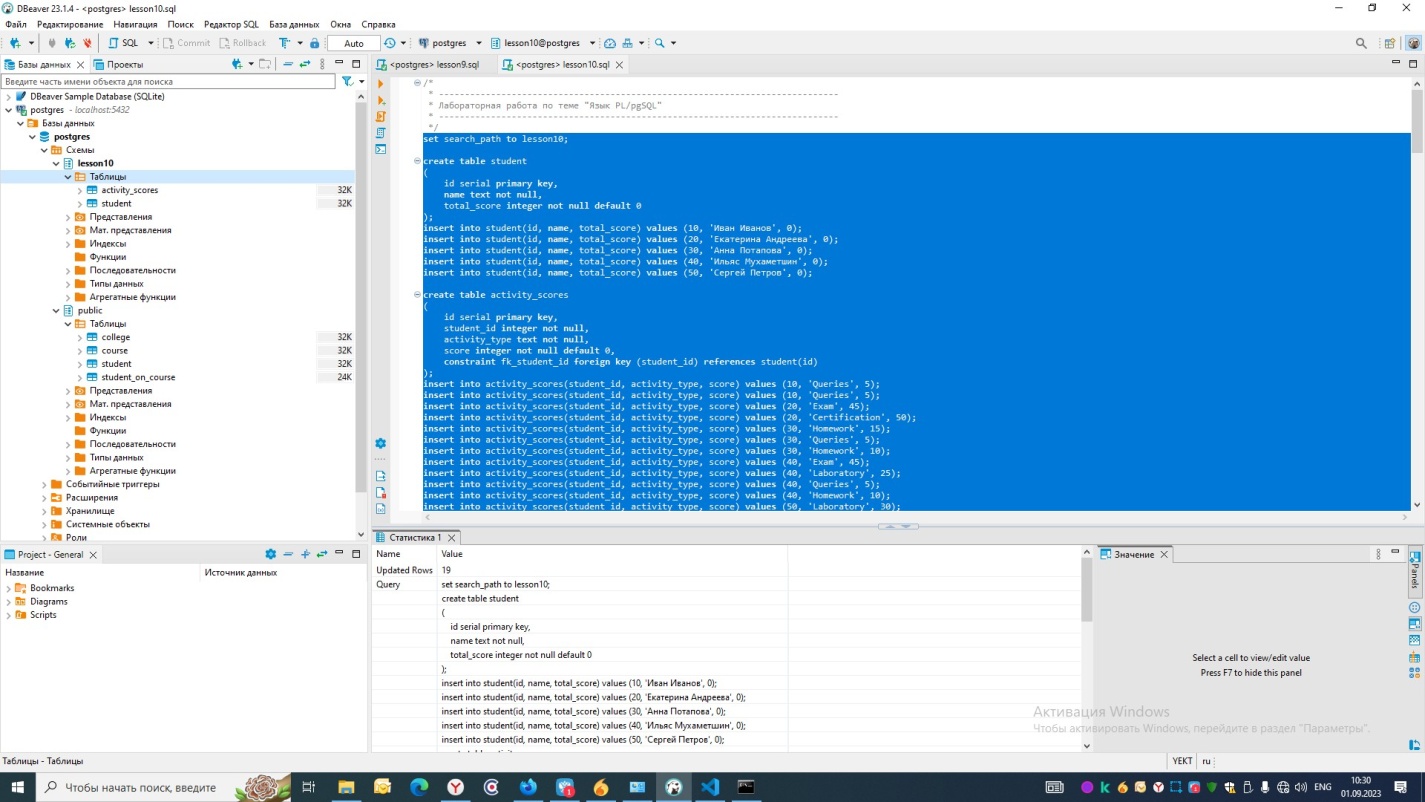
**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (40, 'Homework', 10);

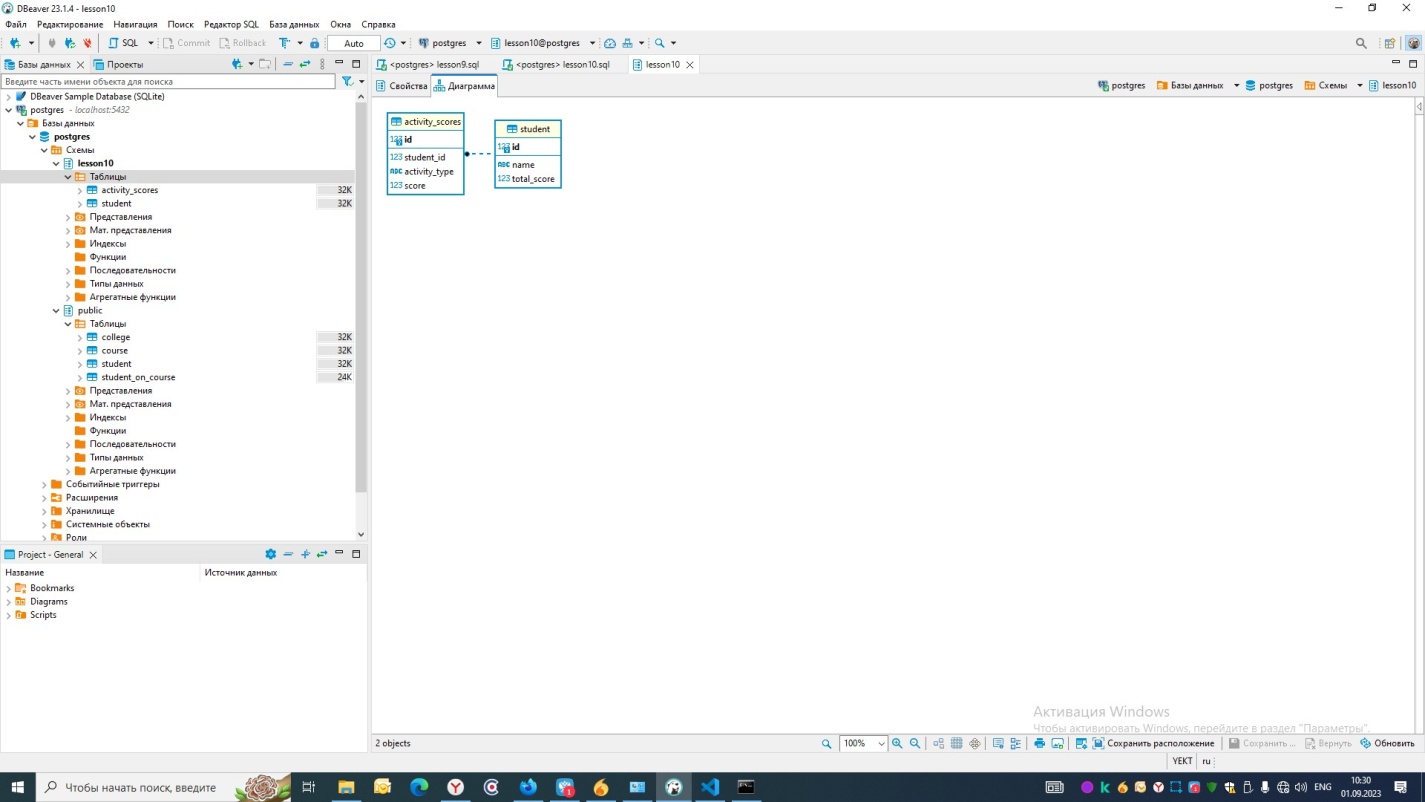
**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (50, 'Laboratory', 30);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (50, 'Homework', 15);

**insert** **into** activity\_scores(student\_id, activity\_type, score) **values** (50, 'Queries', 5);







**Задача 1: Расчет стипендии для студентов:**

У вас есть две таблицы в базе данных: students и activity\_scores. Таблица students содержит информацию о студентах, их идентификаторах и общем балле. Таблица activity\_scores содержит информацию о баллах за разные виды деятельности для каждого студента.

Создайте таблицу students с колонками:

* id (SERIAL) - идентификатор студента (PRIMARY KEY)
* name (TEXT) - имя студента
* total\_score (INTEGER) - общий балл студента

Создайте таблицу activity\_scores с колонками:

* student\_id (INTEGER) - ссылка на студента в таблице students
* activity\_type (TEXT) - вид деятельности (например, "Homework", "Exam" и т.д.)
* score (INTEGER) - балл за деятельность

При наличии таблицы в базе, можете использовать существующую. Создайте функцию calculate\_scholarship, которая будет рассчитывать стипендию для студента. Стипендия зависит от общего балла студента:

* Если общий балл больше или равен 90, стипендия равна 1000.
* Если общий балл больше или равен 80, но меньше 90, стипендия равна 500.
* В остальных случаях, стипендия равна 0.

Создайте триггер update\_scholarship\_trigger, который будет автоматически вызывать функцию calculate\_scholarship при обновлении баллов за деятельность в таблице activity\_scores.

Протестируйте решение, вставив данные о студентах и их баллах за деятельность. Посмотрите, как автоматически обновляется стипендия каждого студента после добавления баллов.

**Решение:**

**drop** **function** calculate\_scholarship **cascade**;

/\* ----------------------------------------------------------------------------- \*/

**create** **or** **replace** **function**

calculate\_scholarship()

**returns**

**trigger** **as** **$$**

**declare**

user\_score **numeric**;

student\_name **text**;

**begin**

-- Получаем балл за деятельность

**select**

**sum**(score)

**into**

user\_score

**from**

activity\_scores

**where**

activity\_scores.student\_id=**new**.student\_id;

-- Имя студента

student\_name := (

**select**

**name**

**from**

student

**where**

id=**new**.student\_id

);

-- Условие расчета степендии студента

**if** user\_score >= 90 **then**

**raise** **notice** e'Студент: %\nБалл студента: %\nСтепендия: % руб.\n', student\_name, user\_score, 1000;

**elseif** user\_score >= 80 **and** user\_score < 90 **then**

**raise** **notice** e'Студент: %\nБалл студента: %\nСтепендия: % руб.\n', student\_name, user\_score, 500;

**else**

**raise** **notice** e'Студент: %\nБалл студента: %\nСтепендия: % руб.\n', student\_name, user\_score, 0;

**end** **if**;

**return** **new**;

**end**;

**$$** **language** plpgsql;

/\* ----------------------------------------------------------------------------- \*/

**create** **or** **replace** **trigger**

update\_scholarship\_trigger

**after** **update** **on**

activity\_scores

**for** **each** **row**

**execute** **function** calculate\_scholarship();

/\* ----------------------------------------------------------------------------- \*/

**update**

activity\_scores

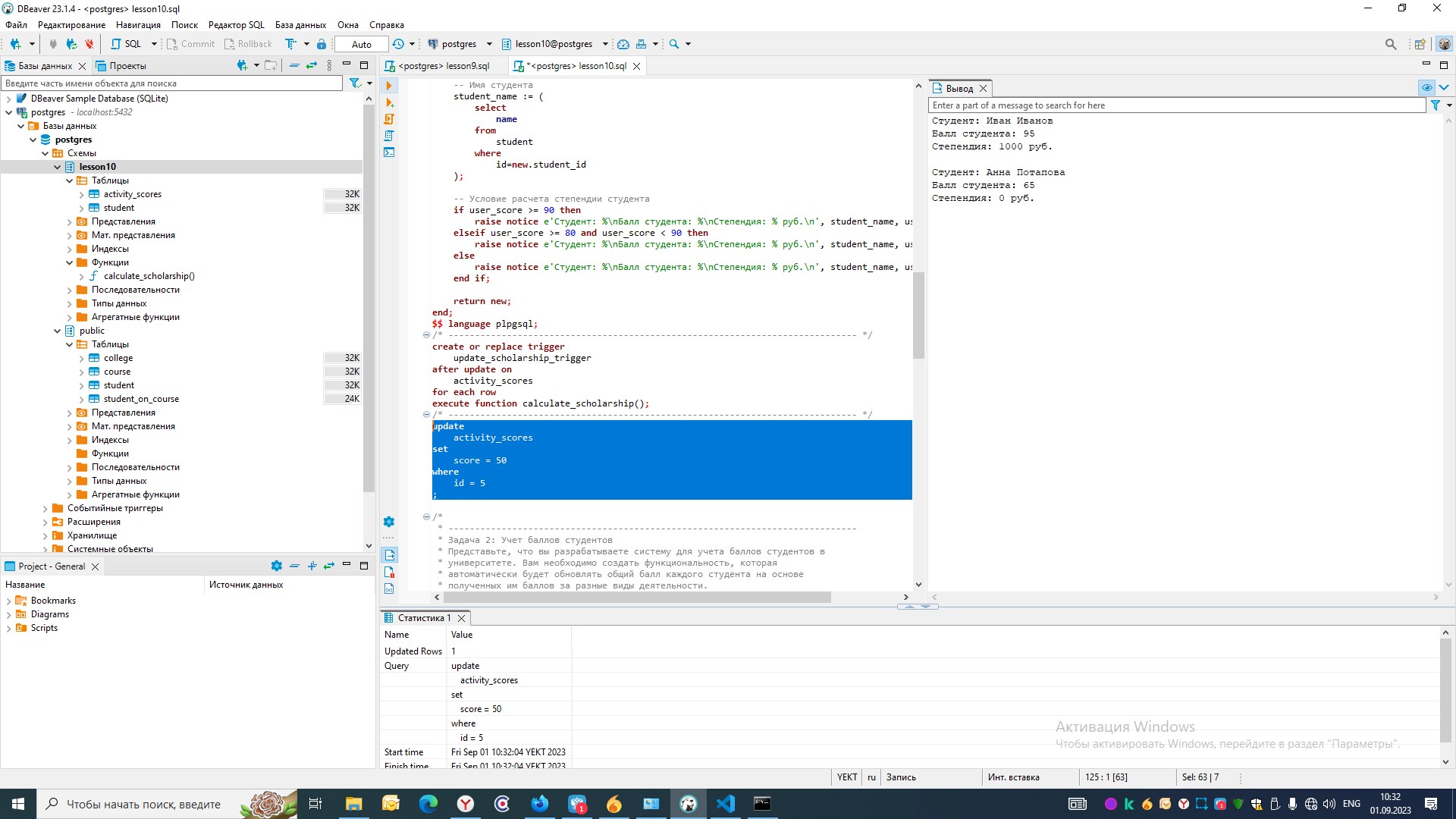
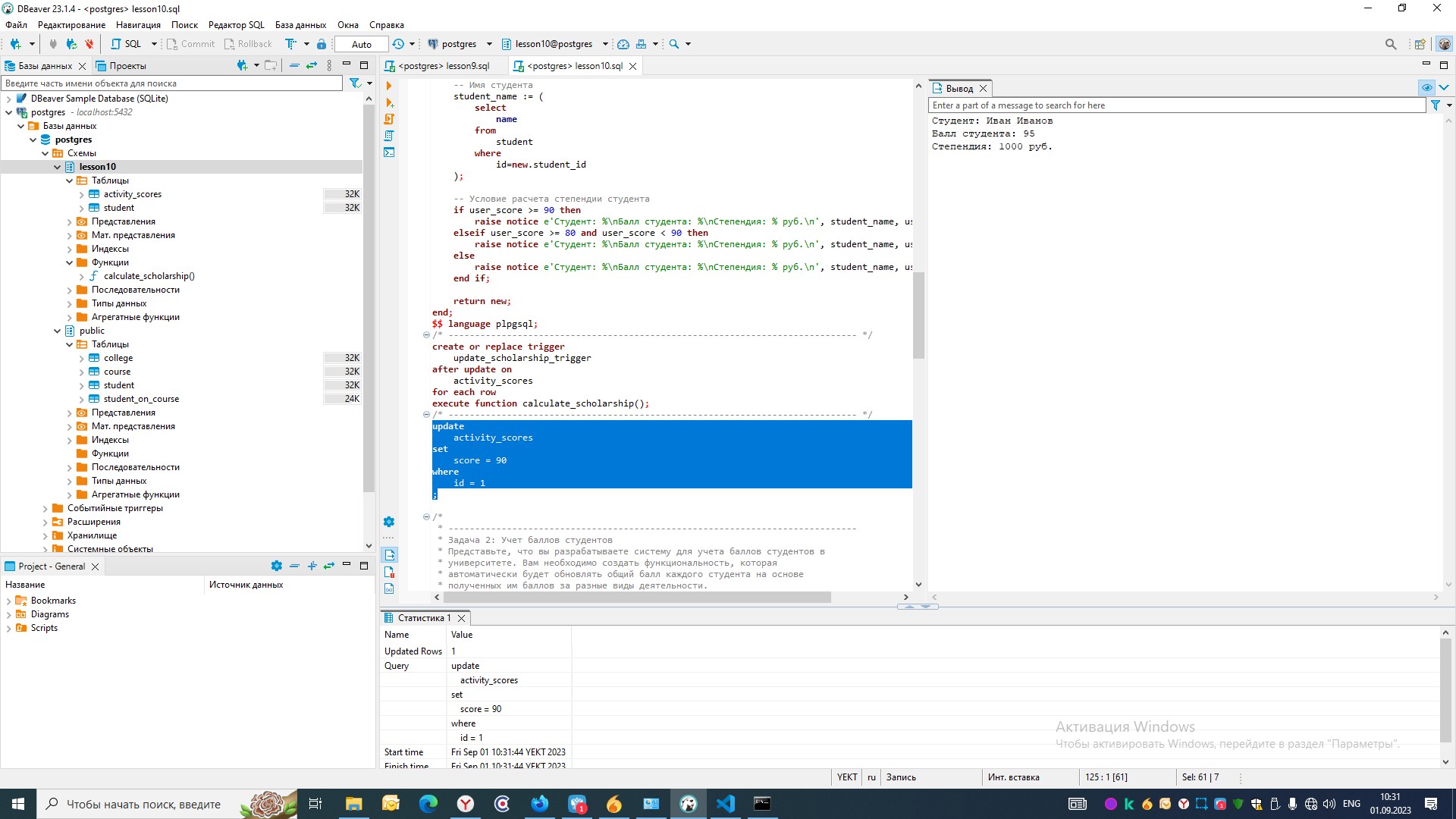
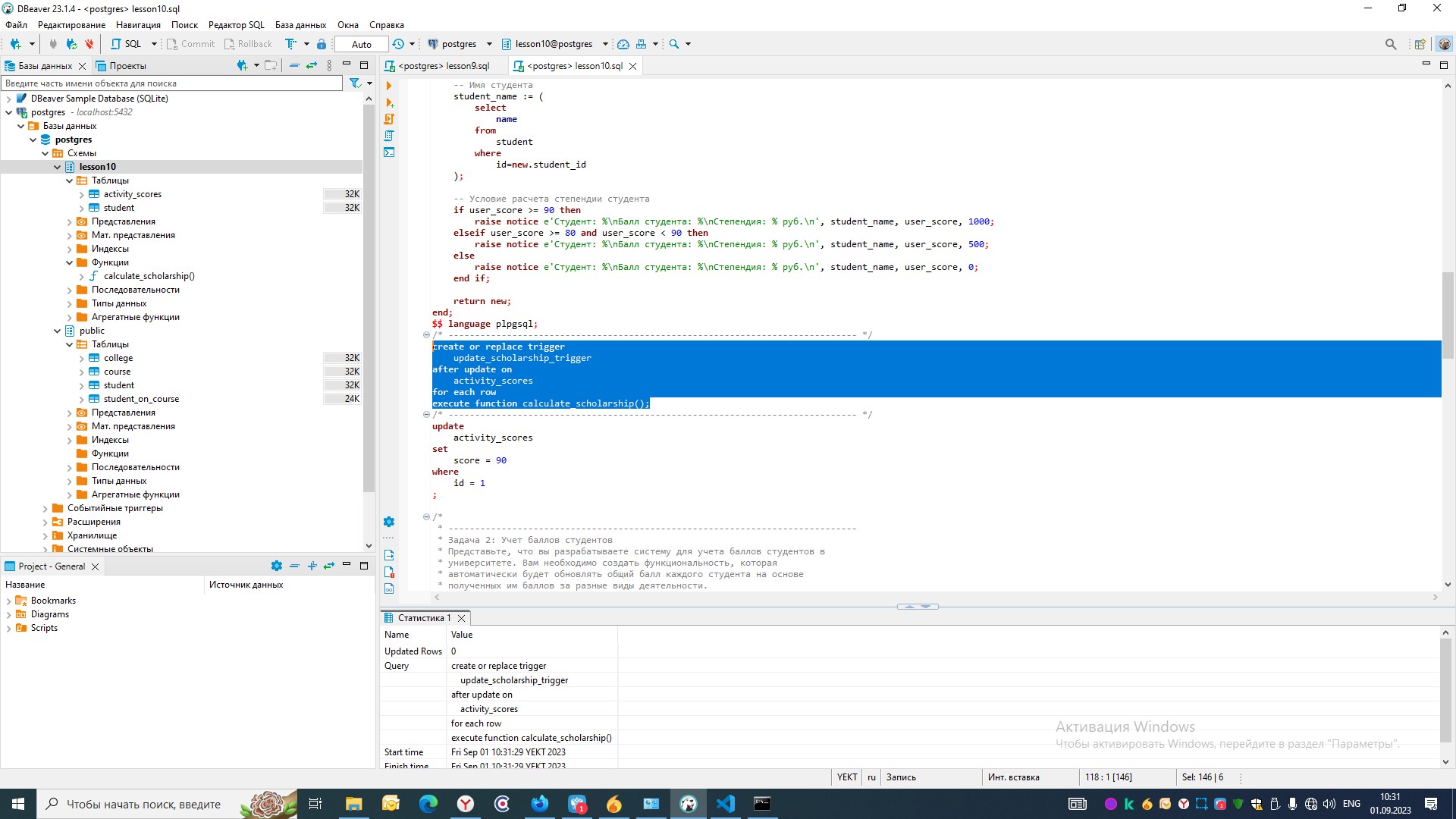
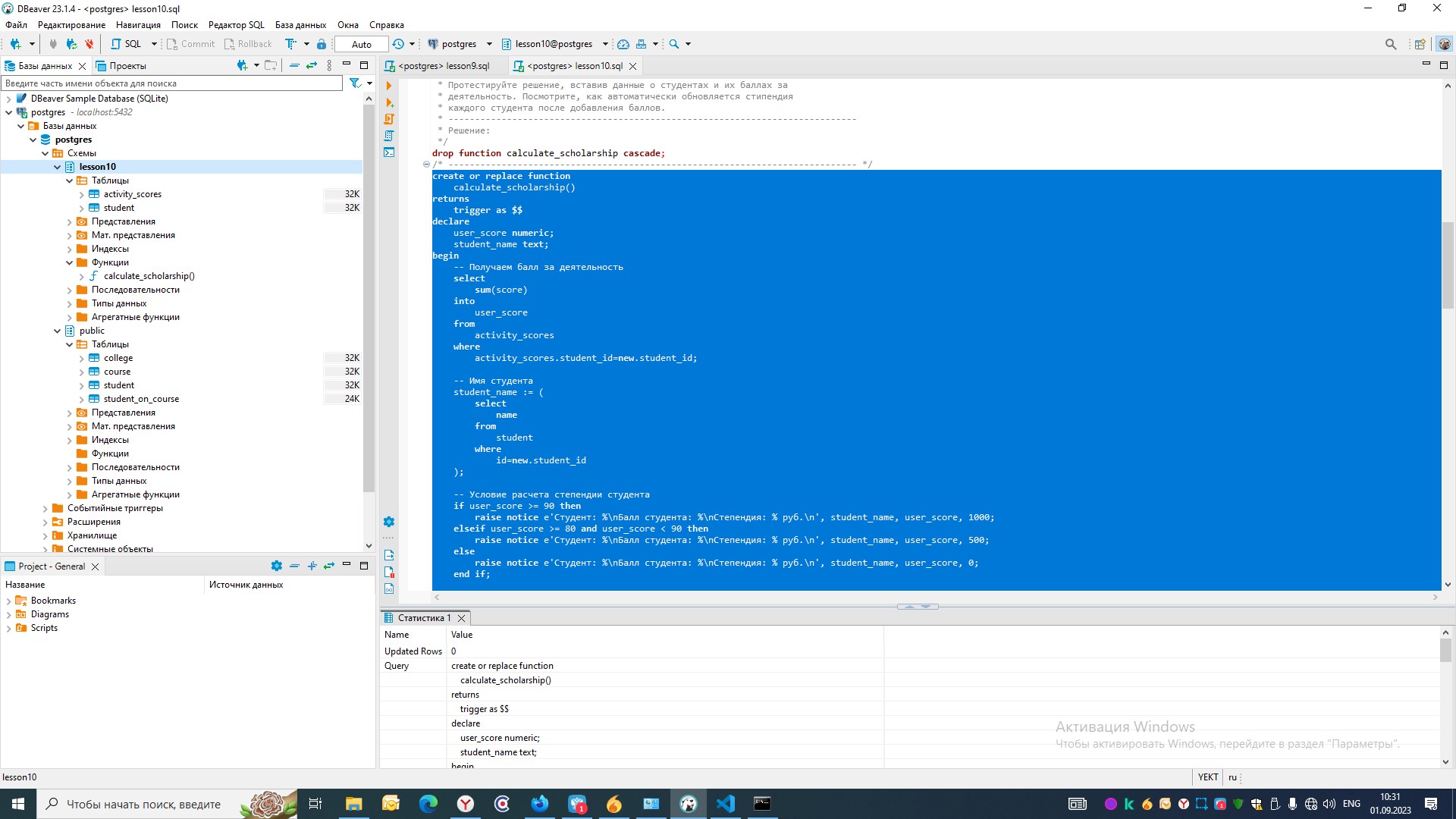
**set**

score = 50

**where**

id = 5

;



**Задача 2: Учет баллов студентов:**

Представьте, что вы разрабатываете систему для учета баллов студентов в университете. Вам необходимо создать функциональность, которая автоматически будет обновлять общий балл каждого студента на основе полученных им баллов за разные виды деятельности.

Создайте таблицу students, содержащую следующие поля:

* id (SERIAL, PRIMARY KEY)
* name (TEXT)
* total\_score (INTEGER)

Создайте таблицу activity\_scores, содержащую следующие поля:

* student\_id (INTEGER, FOREIGN KEY к полю id таблицы students)
* activity\_type (TEXT)
* score (INTEGER)

При наличии таблицы в базе, можете использовать существующую.

Напишите функцию update\_total\_score(student\_id INTEGER):

* Эта функция должна пересчитывать общий балл студента на основе баллов за разные виды деятельности в таблице activity\_scores.
* Используйте цикл для итерации по всем записям в activity\_scores для заданного student\_id.
* Обновите поле total\_score для соответствующего студента в таблице students суммой всех баллов за разные виды деятельности.

Напишите триггер, который будет автоматически вызывать функцию update\_total\_score при вставке новых записей в таблицу activity\_scores.

Предоставьте примеры использования:

* Вставьте несколько студентов в таблицу students.
* Вставьте записи о баллах за разные виды деятельности в таблицу activity\_scores.
* После вставки баллов, убедитесь, что общий балл каждого студента автоматически обновлен в таблице students.

**Решение:**

**drop** **function** update\_total\_score **cascade**;

/\* ----------------------------------------------------------------------------- \*/

**create** **or** **replace** **function**

update\_total\_score()

**returns**

**trigger** **as** **$$**

**declare**

scores\_array **integer**[];

i **integer**;

scores\_sum **integer**;

student\_name **text**;

**begin**

-- Значение суммы по умолчанию

scores\_sum := 0;

-- Имя студента

student\_name := (

**select**

**name**

**from**

student

**where**

id=**new**.student\_id

);

-- Массив из баллов за деятельность для студента

scores\_array := **array**(

**select**

score

**from**

activity\_scores

**where**

student\_id=**new**.student\_id

);

-- Перебор массива из баллов за деятельность и сложение в общую сумму

**foreach** i **in** **array** scores\_array **loop**

scores\_sum=scores\_sum+i;

**end** **loop**;

-- Обновление общего балла у студента

**update**

student

**set**

total\_score=scores\_sum

**where**

id=**new**.student\_id

;

-- Информационный вывод

**raise** **notice** e'Студент: %\nОбщий балл: %\n', student\_name, scores\_sum;

**return** **new**;

**end**;

**$$** **language** plpgsql;

/\* ----------------------------------------------------------------------------- \*/

**create** **or** **replace** **trigger**

insert\_total\_score\_trigger

**after** **insert** **on**

activity\_scores

**for** **each** **row**

**execute** **function** update\_total\_score();

/\* ----------------------------------------------------------------------------- \*/

**insert** **into** activity\_scores (

student\_id,

activity\_type,

score

)

**values** (

20,

'Exam',

30

);

/\* ----------------------------------------------------------------------------- \*/

**select**

\*

**from**

student

;

/\* ----------------------------------------------------------------------------- \*/

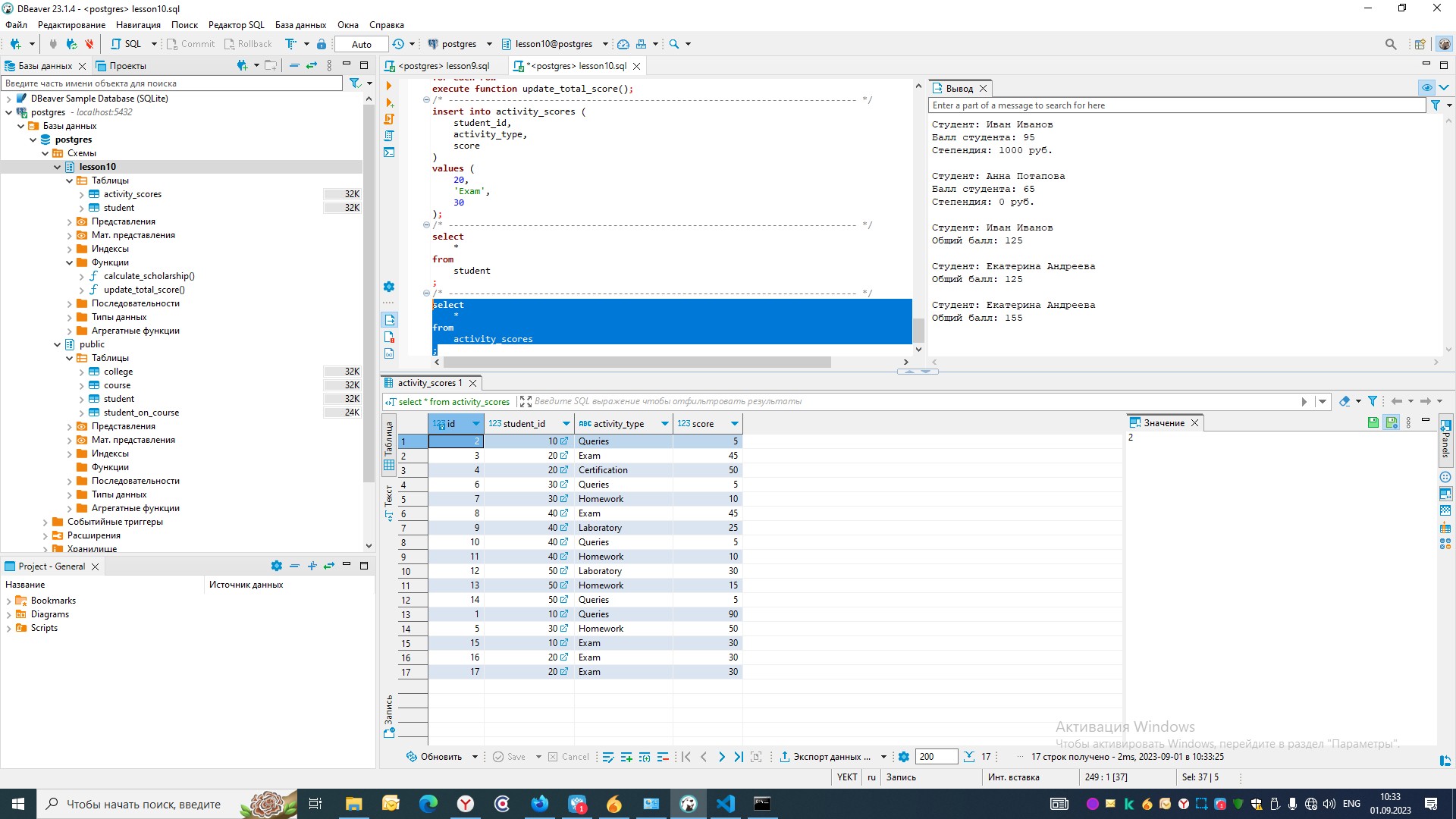
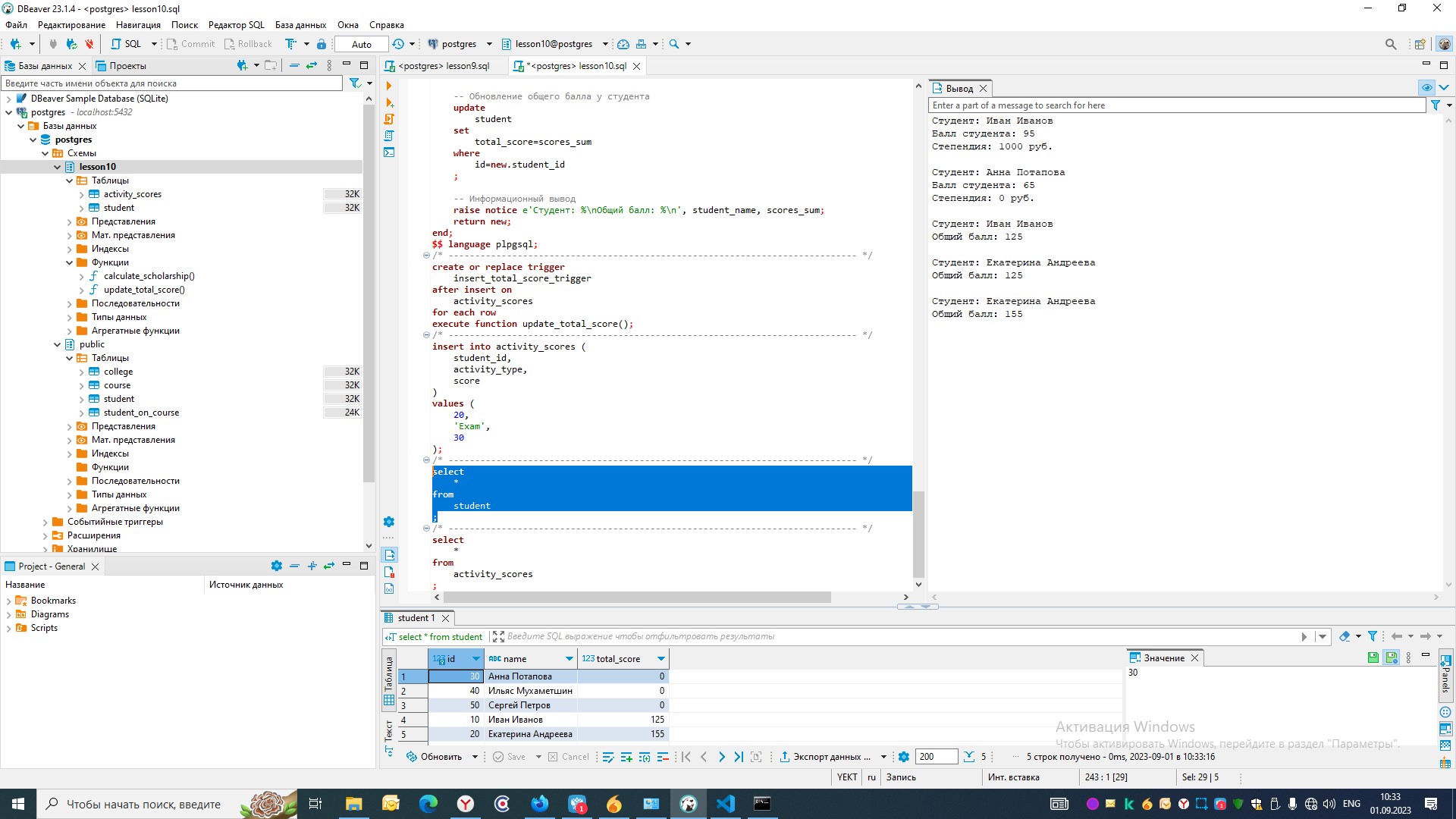
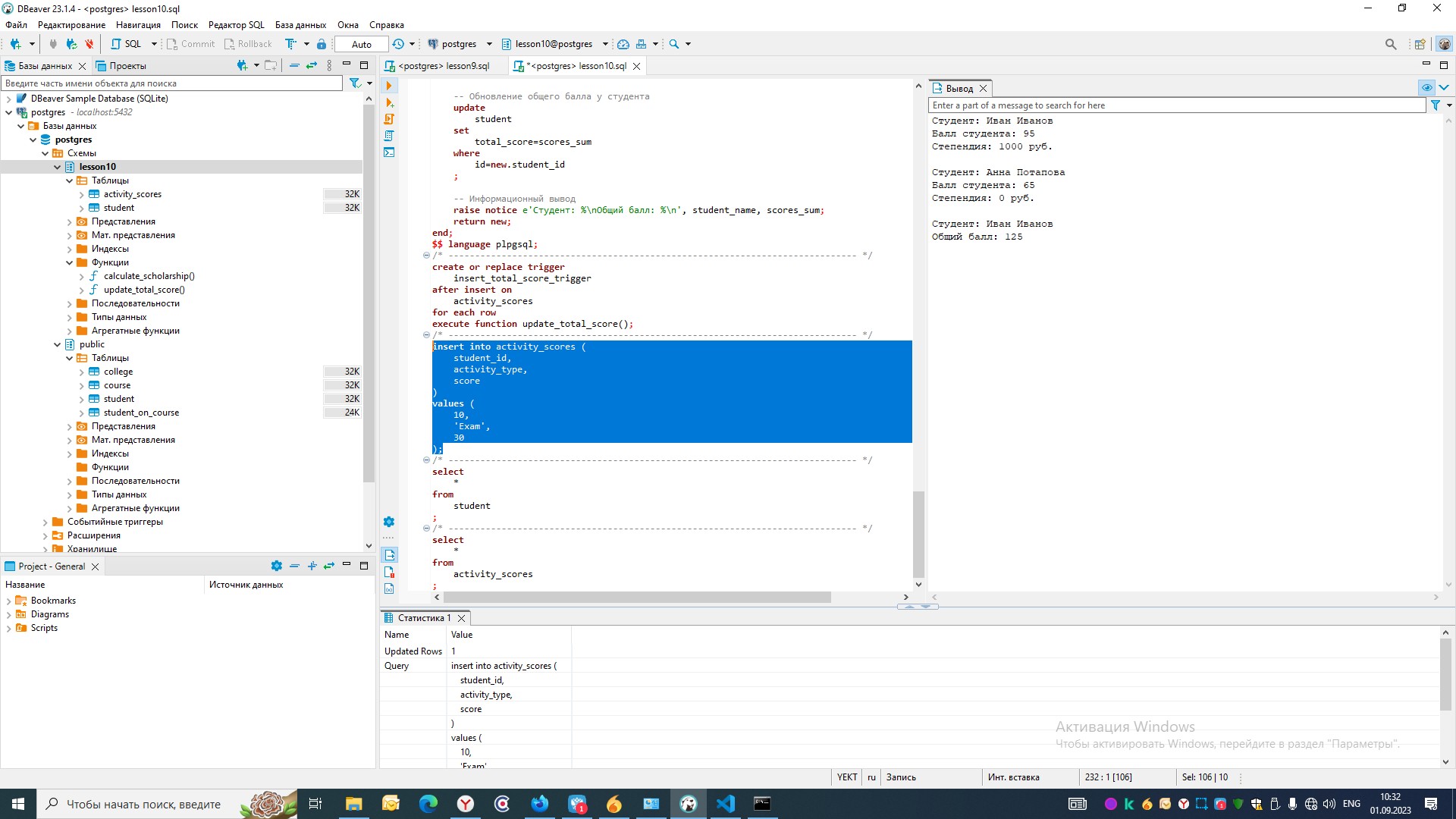
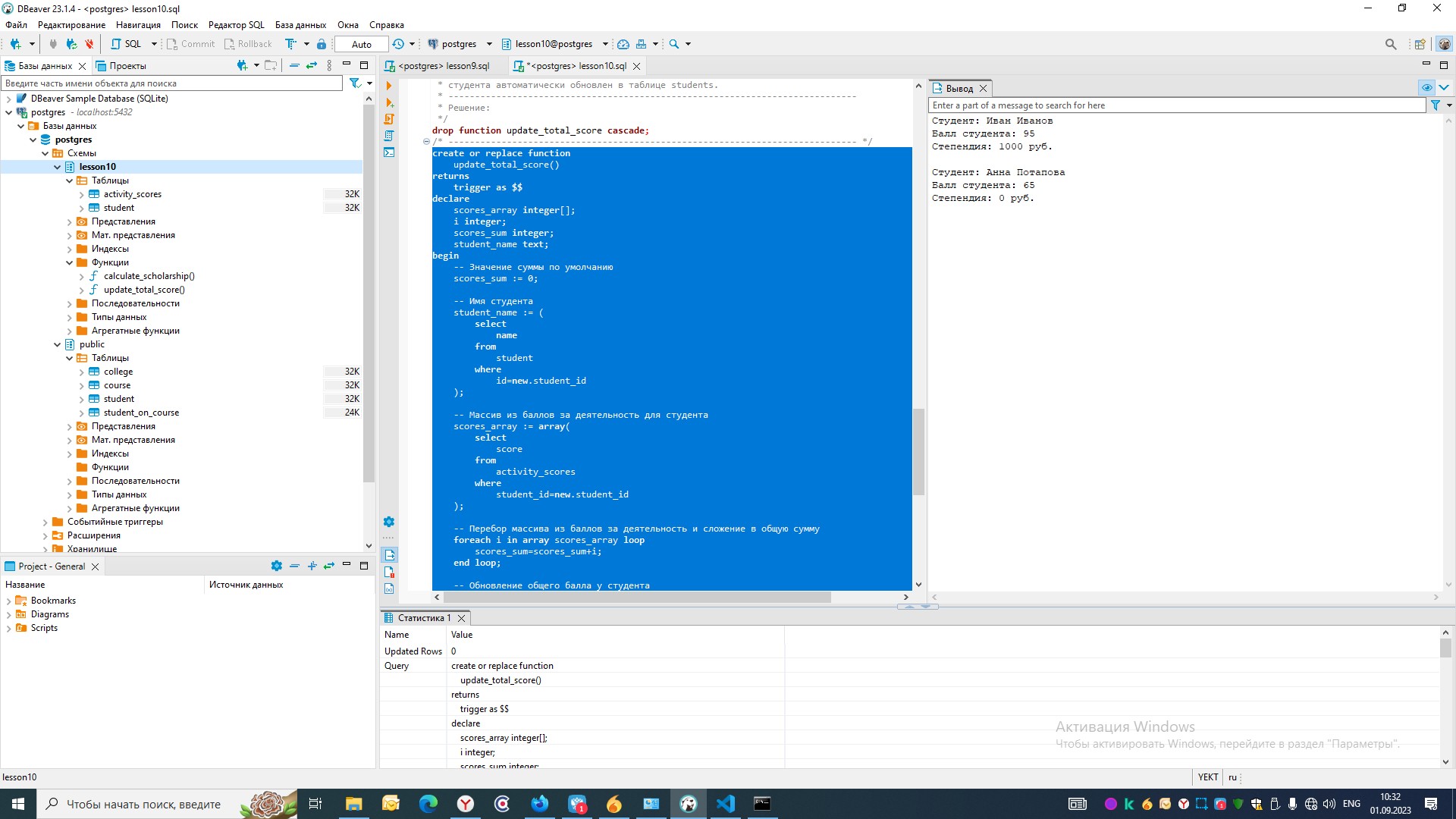
**select**

\*

**from**

activity\_scores

;



К решению приложены файлы:

«lesson10.sql» для PostgreSQL

Ссылка на GitHub:

<https://github.com/SotGE/innopolis2023/tree/main/lesson10>