**Лабораторная работа по теме “Язык PL/pgSQL”**

**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

**Решение:**

**create** **schema** university\_scores;

**set** search\_path = "university\_scores, public";

**show** search\_path;

--Создаем таблицу students

**create** **table** **if** **not** **exists** university\_scores.students (

id SERIAL **PRIMARY** **KEY**,

**name** **VARCHAR**(200),

total\_score **INTEGER**

);

--Создаем таблицу activity\_scores

**create** **table** **if** **not** **exists** university\_scores.activity\_scores (

id SERIAL **PRIMARY** **KEY**,

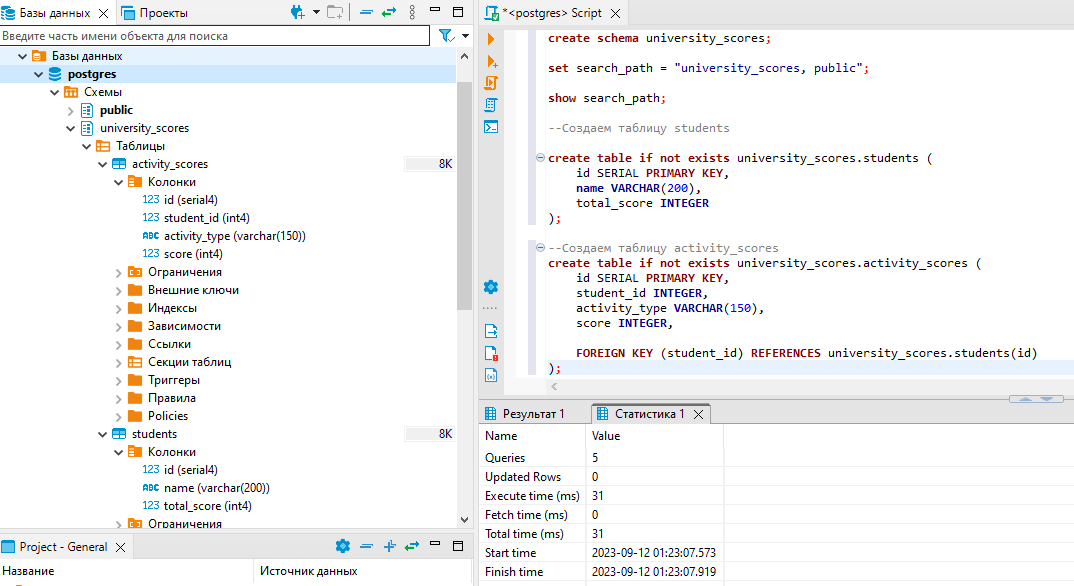
student\_id **INTEGER**,

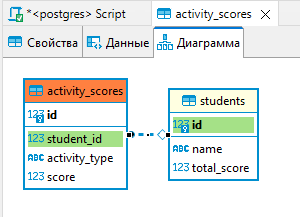
activity\_type **VARCHAR**(150),

score **INTEGER**,

**FOREIGN** **KEY** (student\_id) **REFERENCES** university\_scores.students(id)

);





**Задача 1: 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.**

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.

**Решение:**

**create** **or** **replace** **function** update\_total\_score () **returns** **TRIGGER** **AS** **$$**

**begin**

**if** TG\_OP = 'UPDATE' **or** TG\_OP = 'INSERT' **THEN**

**update** university\_scores.students

**set** total\_score = (**select** **sum**(score)

**from** university\_scores.activity\_scores

**where** activity\_scores.student\_id = **new**.student\_id)

**where** id = **new**.student\_id;

**end** **if**;

**return** **null**;

**end**;

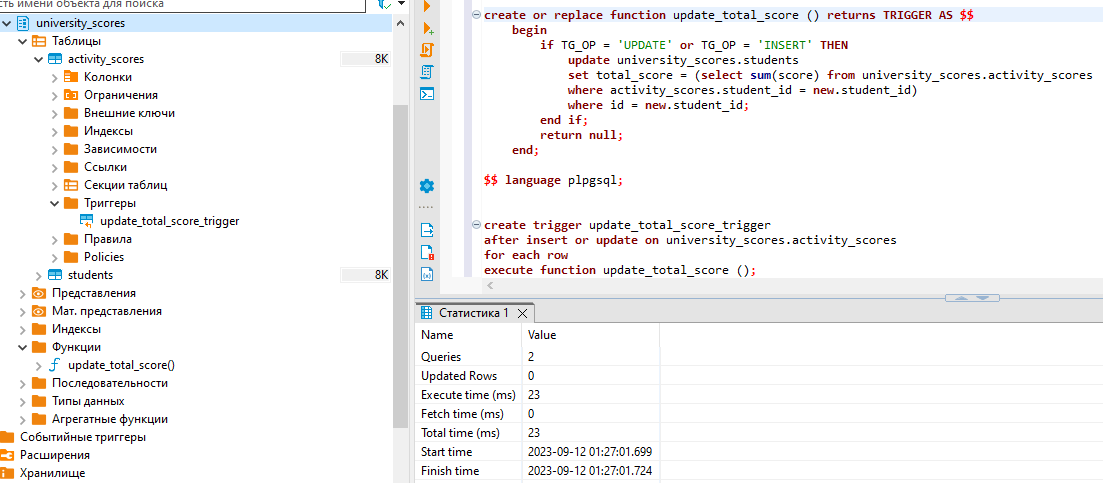
**$$** **language** plpgsql;

**create** **trigger** update\_total\_score\_trigger

**after** **insert** **or** **update** **on** university\_scores.activity\_scores

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

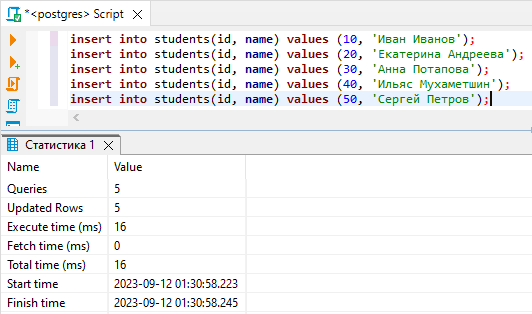
**execute** **function** update\_total\_score ();

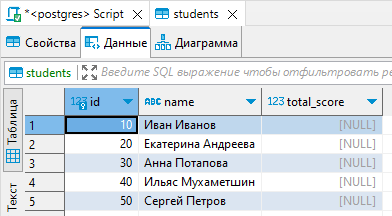


Предоставьте примеры использования:

* Вставьте несколько студентов в таблицу students.

**Решение:**

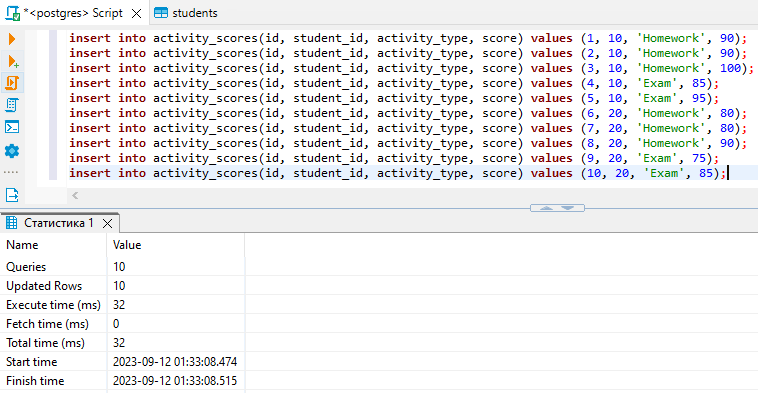


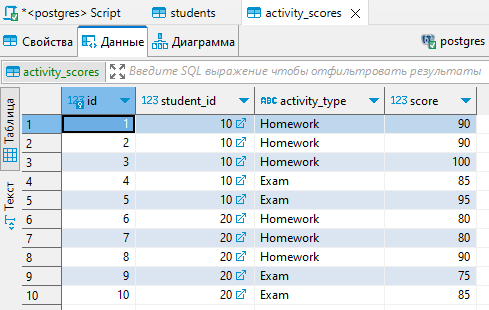


Предоставьте примеры использования:

* Вставьте записи о баллах за разные виды деятельности в таблицу activity\_scores.

**Решение:**

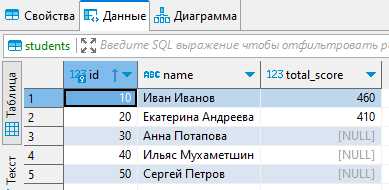




Предоставьте примеры использования:

* После вставки баллов, убедитесь, что общий балл каждого студента автоматически обновлен в таблице students.

**Решение:**



**Задача 2: Расчет стипендии для студентов**

У вас есть две таблицы в базе данных: students и activity\_scores. Таблица

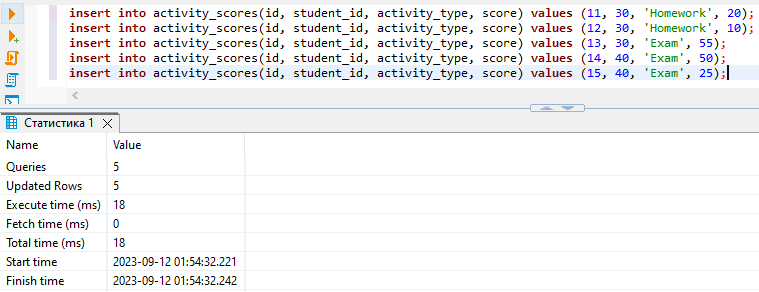
students содержит информацию о студентах, их идентификаторах и общем

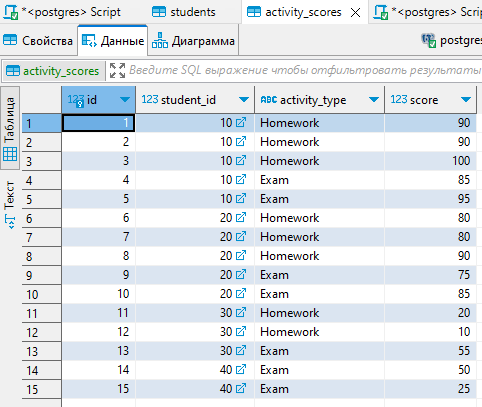
балле. Таблица activity\_scores содержит информацию о баллах за разные

виды деятельности для каждого студента.

**Решение:**

Дополним записями таблицу activity\_scores для наглядности





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.

**Решение:**

**CREATE** **OR** **REPLACE**

**FUNCTION** calculate\_scholarship(student\_id **INTEGER**)

**returns** **INTEGER** **as**

**$$**

**declare**

student\_total\_score **INTEGER**;

scholarship\_total **INTEGER**;

**begin**

**select** **sum**(score) **into** student\_total\_score **from** university\_scores.activity\_scores

**where** activity\_scores.student\_id = calculate\_scholarship.student\_id;

**if** student\_total\_score >= 90 **then**

scholarship\_total :=1000;

**elsif** student\_total\_score >= 80 **then**

scholarship\_total :=500;

**else**

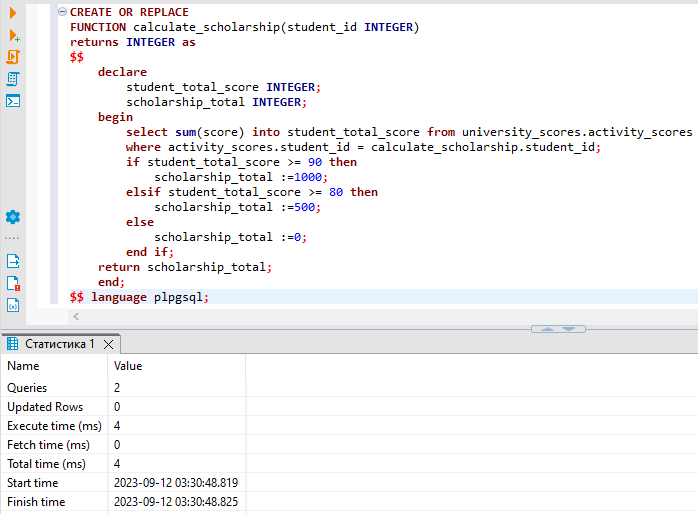
scholarship\_total :=0;

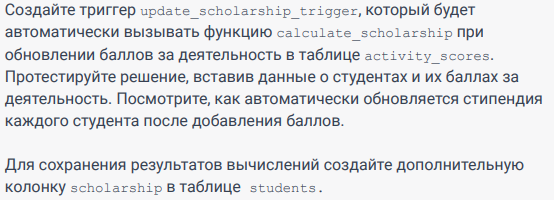
**end** **if**;

**return** scholarship\_total;

**end**;

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





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

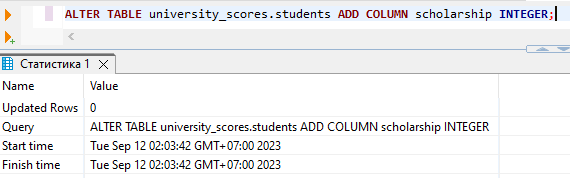
Протестируйте решение, вставив данные о студентах и их баллах за деятельность. Посмотрите, как автоматически обновляется стипендия каждого студента после добавления баллов. Для сохранения результатов вычислений создайте дополнительную

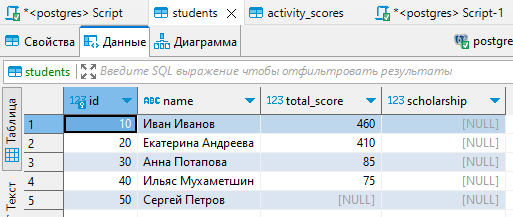
колонку scholarship в таблице students.

**Решение:**

Создаем дополнительную колонку scholarship в таблице students.

**ALTER** **TABLE** university\_scores.students **ADD** **COLUMN** scholarship **INTEGER**;





Создадим триггер update\_scholarship\_trigger с дополнительной функцией calculate\_scholarship\_wrapper.

**create** **or** **replace** **function** update\_scholarship() **returns** **TRIGGER** **AS** **$$**

**begin**

**if** TG\_OP = 'UPDATE' **or** TG\_OP = 'INSERT' **then**

**update** university\_scores.students

**set** scholarship = (**select** university\_scores.calculate\_scholarship(**new**.student\_id)) **where** id = **new**.student\_id;

**end** **if**;

**return** **null**;

**end**;

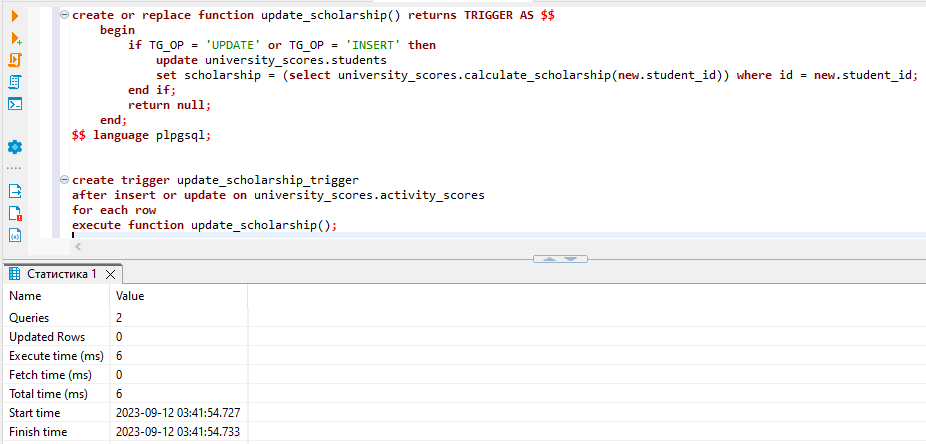
**$$** **language** plpgsql;

**create** **trigger** update\_scholarship\_trigger

**after** **insert** **or** **update** **on** university\_scores.activity\_scores

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

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



Выполним запрос, обновляющий данные об оценках студентов, чтобы таблица activity\_scores приняла следующий вид:

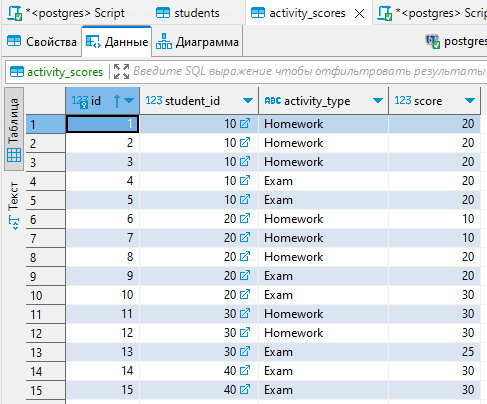


Таблица students:

