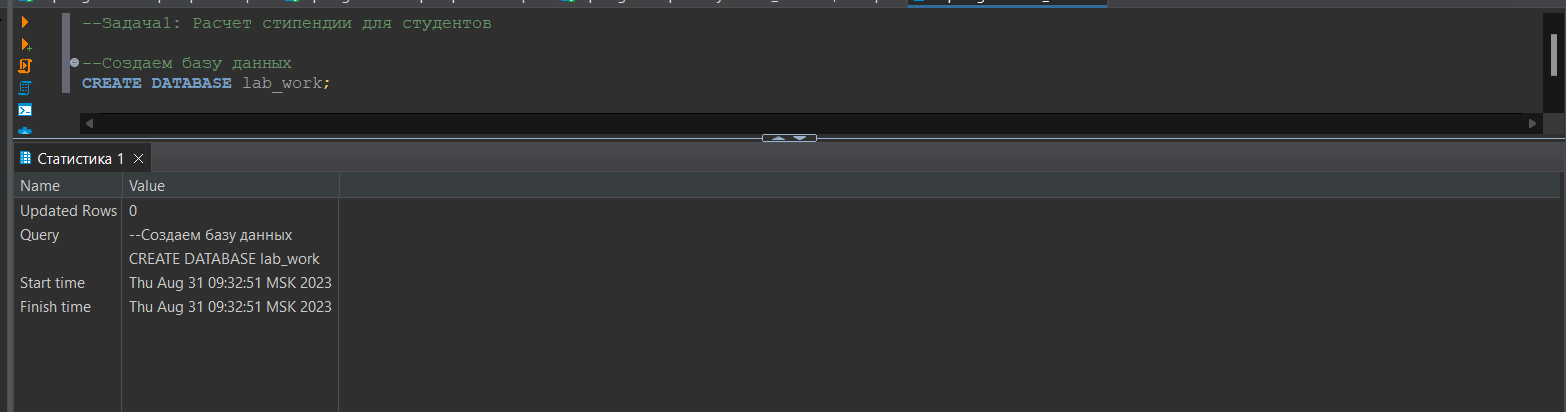


--Задача1: Расчет стипендии для студентов

--Создаем базу данных

**CREATE** **DATABASE** lab\_work;



--Создаем схему

**CREATE** **SCHEMA** lab\_work;



--Создаем таблицу students

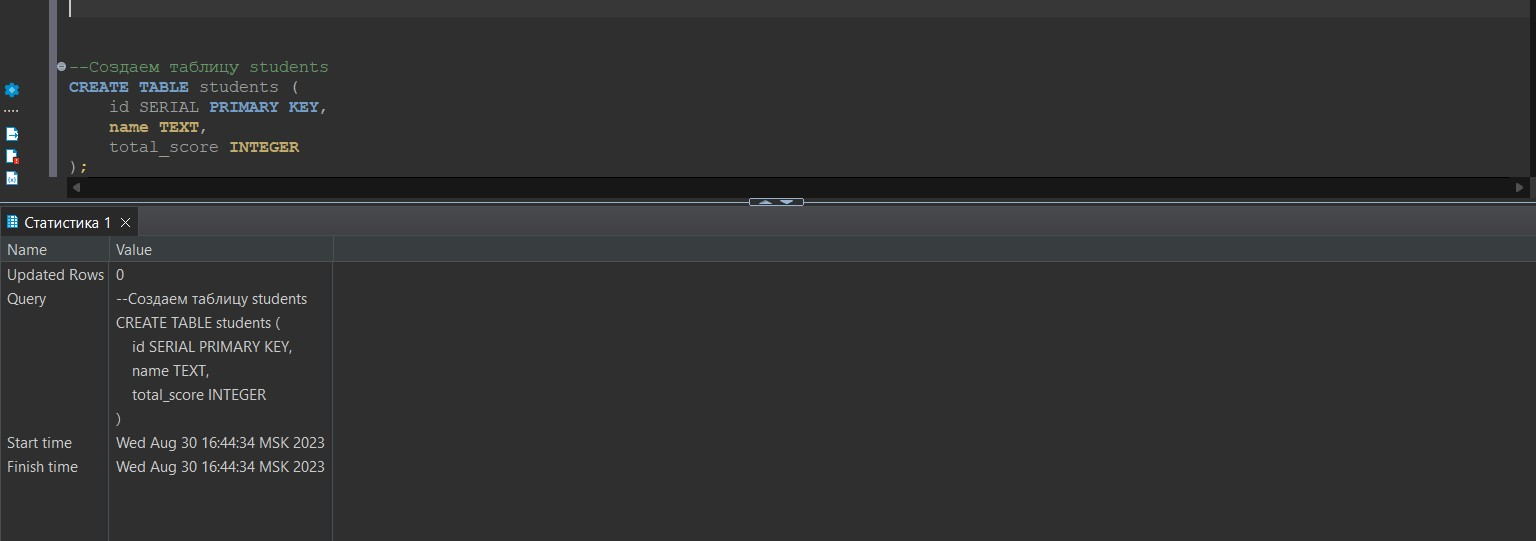
**CREATE** **TABLE** students (

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

**name** **TEXT**,

total\_score **INTEGER**

);



--Вносим данные в таблицу students

**INSERT** **INTO** students (**name**, total\_score) **VALUES**

('Иванов Иван', 85),

('Петрова Анна', 92),

('Сидоров Павел', 78),

('Смирнова Екатерина', 67),

('Козлов Михаил', 95),

('Соколова Александра', 88),

('Григорьев Дмитрий', 76),

('Морозова Ольга', 91),

('Николаев Игорь', 82),

('Белова Елена', 79),

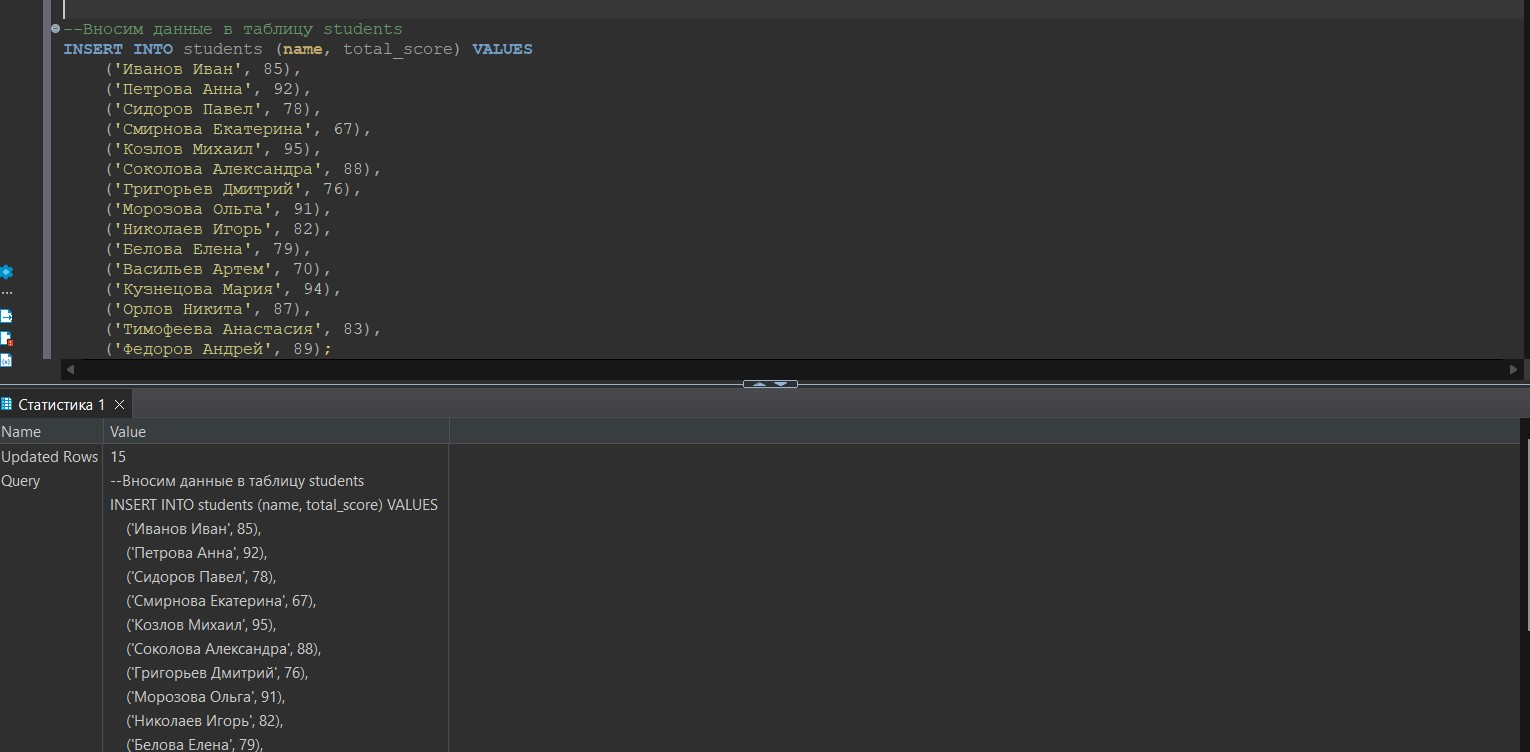
('Васильев Артем', 70),

('Кузнецова Мария', 94),

('Орлов Никита', 87),

('Тимофеева Анастасия', 83),

('Федоров Андрей', 89);



--Создаем таблицу activity\_scores

**CREATE** **TABLE** activity\_scores (

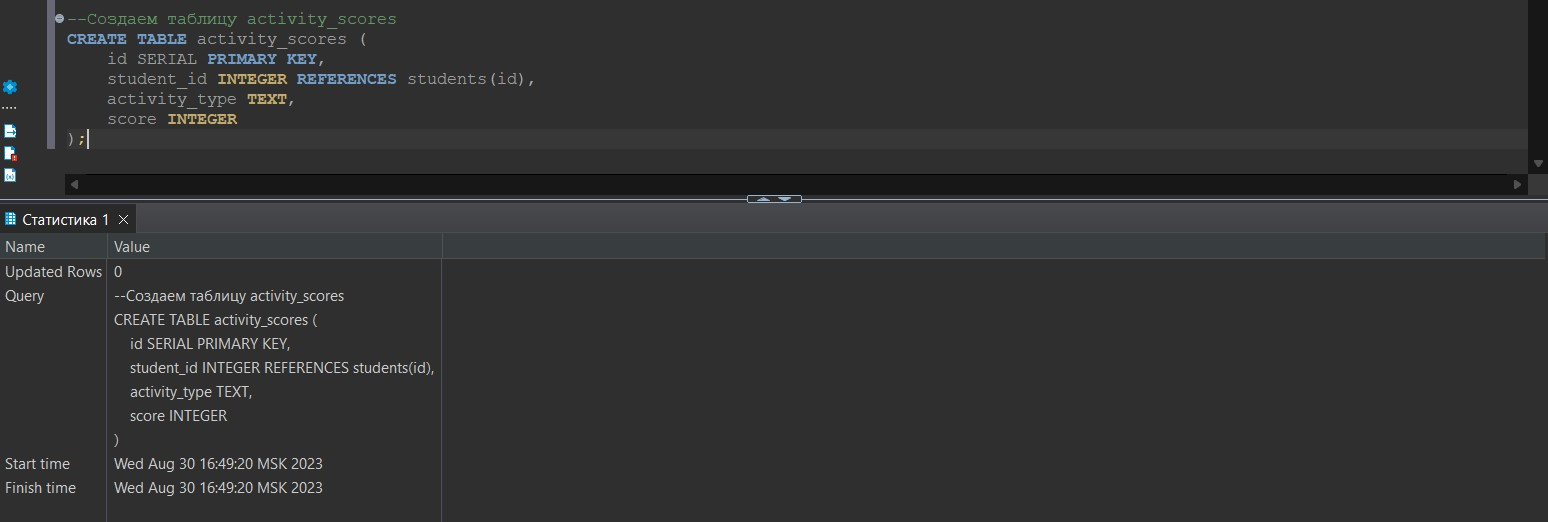
id SERIAL **PRIMARY** **KEY**,

student\_id **INTEGER** **REFERENCES** students(id),

activity\_type **TEXT**,

score **INTEGER**

);



--Вносим данные в таблицу activity\_scores

**INSERT** **INTO** activity\_scores (student\_id, activity\_type, score) **VALUES**

(1, 'Homework', 90),

(1, 'Exam', 78),

(2, 'Homework', 95),

(2, 'Exam', 88),

(3, 'Homework', 70),

(3, 'Exam', 82),

(4, 'Homework', 60),

(4, 'Exam', 72),

(5, 'Homework', 87),

(5, 'Exam', 95),

(6, 'Homework', 75),

(6, 'Exam', 68),

(7, 'Homework', 92),

(7, 'Exam', 89),

(8, 'Homework', 80),

(8, 'Exam', 76),

(9, 'Homework', 65),

(9, 'Exam', 72),

(10, 'Homework', 88),

(10, 'Exam', 90),

(11, 'Homework', 78),

(11, 'Exam', 84),

(12, 'Homework', 92),

(12, 'Exam', 91),

(13, 'Homework', 85),

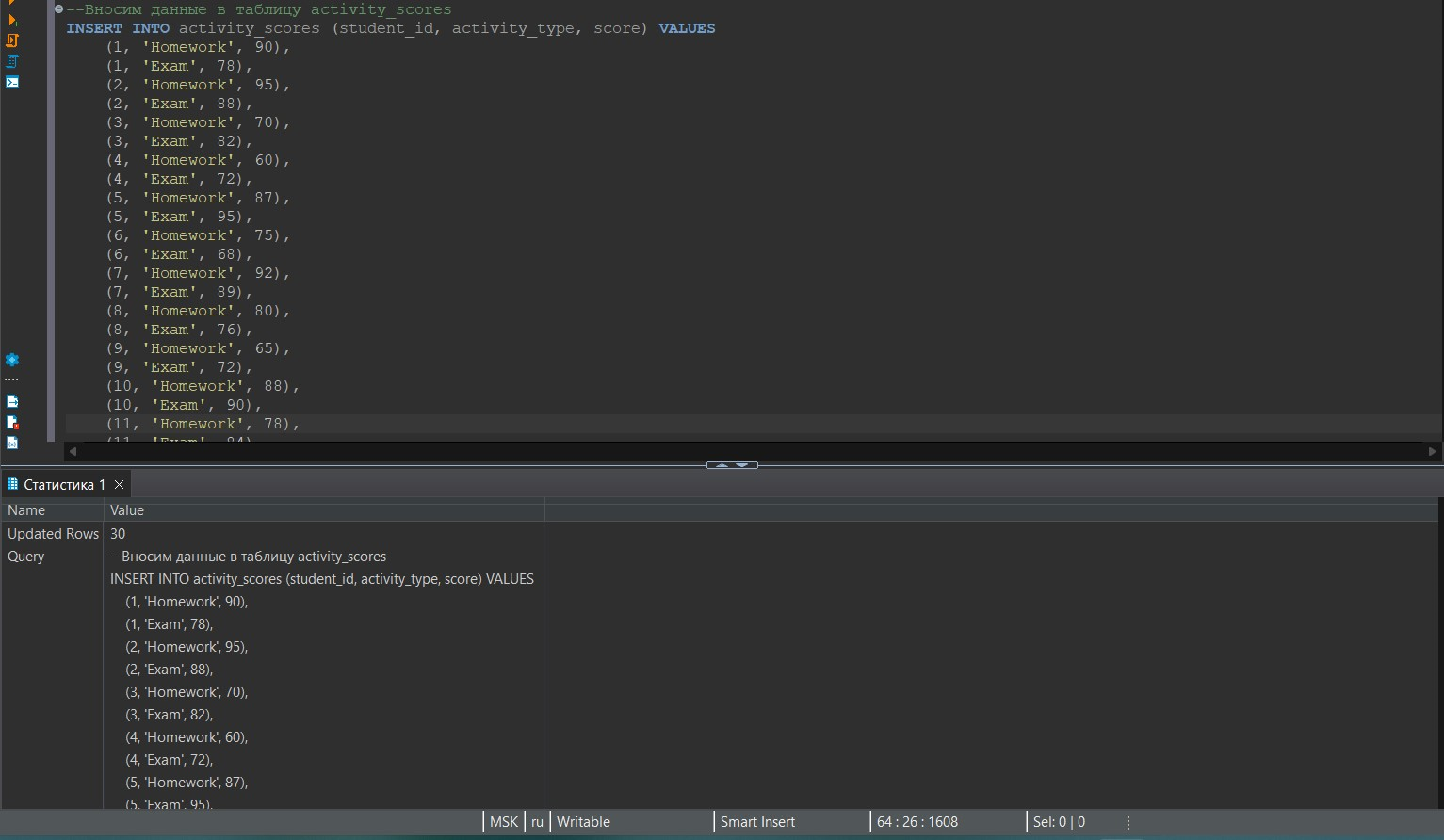
(13, 'Exam', 89),

(14, 'Homework', 70),

(14, 'Exam', 75),

(15, 'Homework', 95),

(15, 'Exam', 92);



--Создаем функцию calculate\_scholarship

**CREATE** **OR** **REPLACE** **FUNCTION** calculate\_scholarship(student\_id **INTEGER**)

**RETURNS** **INTEGER** **AS** **$$**

**DECLARE**

student\_total\_score **INTEGER**;

scholarship\_amount **INTEGER**;

**BEGIN**

-- Получение общего балла студента

**SELECT** total\_score **INTO** student\_total\_score **FROM** students **WHERE** id = student\_id;

- - Расчет стипендии в зависимости от общего балла

**IF** student\_total\_score >= 90 **THEN**

scholarship\_amount := 1000;

**ELSIF** student\_total\_score >= 80 **THEN**

scholarship\_amount := 500;

**ELSE**

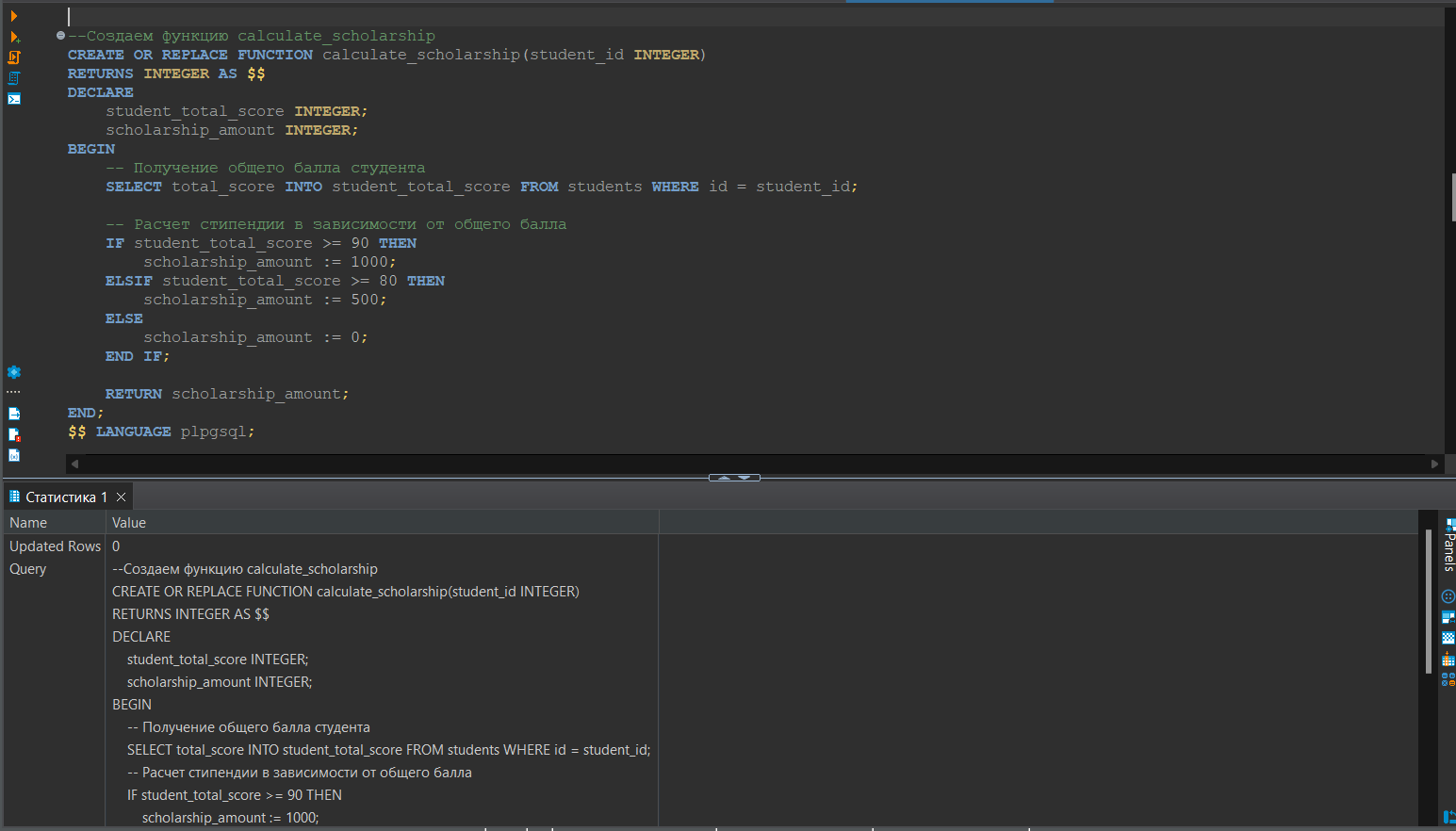
scholarship\_amount := 0;

**END** **IF**;

**RETURN** scholarship\_amount;

**END**;

**$$** **LANGUAGE** plpgsql;



--Создаем триггер update\_scholarship\_trigger

**CREATE** **OR** **REPLACE** **FUNCTION** update\_scholarship()

**RETURNS** **TRIGGER** **AS** **$$**

**BEGIN**

-- Вызываем функцию calculate\_scholarship для соответствующего студента

**SELECT** calculate\_scholarship(**NEW**.student\_id) **INTO** **NEW**.score;

**RETURN** **NEW**;

**END**;

**$$** **LANGUAGE** plpgsql;

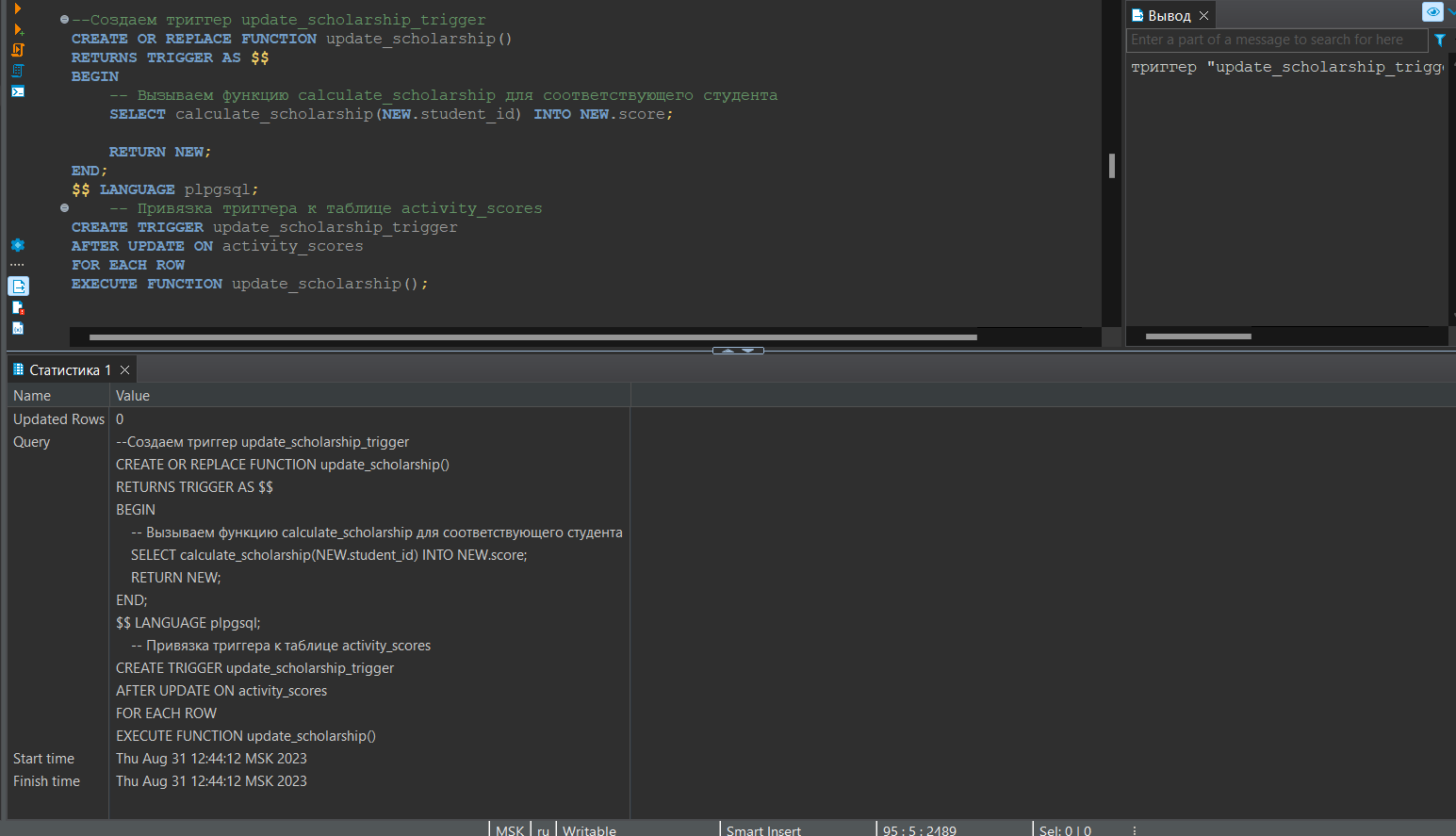
-- Привязка триггера к таблице activity\_scores

**CREATE** **TRIGGER** update\_scholarship\_trigger

**AFTER** **UPDATE** **ON** activity\_scores

**FOR** **EACH** **ROW**

**EXECUTE** **FUNCTION** update\_scholarship();



--Рассмотрим данные о баллах и стипендии студента

**SELECT**

s.id **AS** student\_id,

s.**name** **AS** student\_name,

s.total\_score **AS** student\_total\_score,

**CASE**

**WHEN** s.total\_score >= 90 **THEN** 1000

**WHEN** s.total\_score >= 80 **THEN** 500

**ELSE** 0

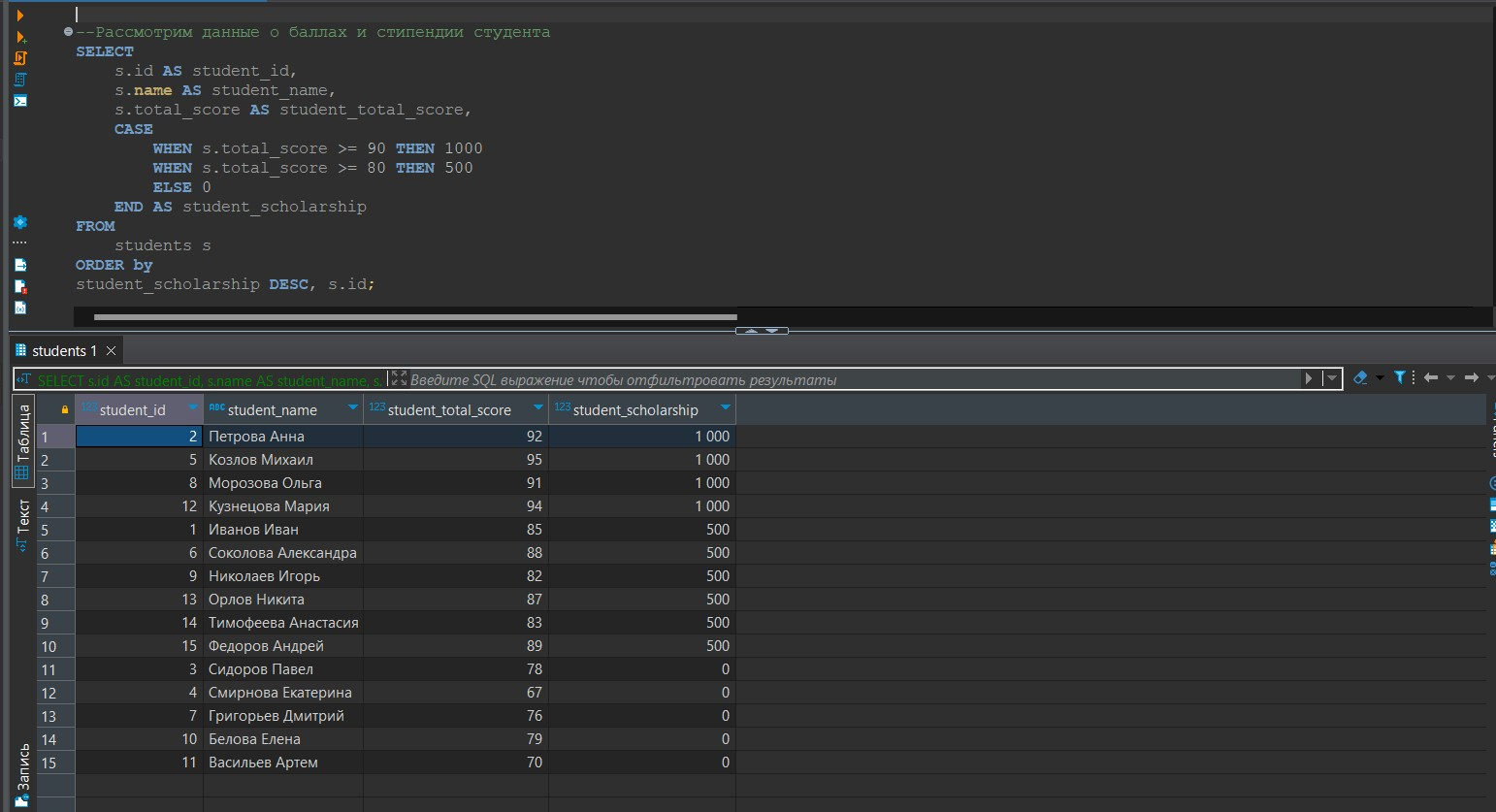
**END** **AS** student\_scholarship

**FROM**

students s

**ORDER** **by**

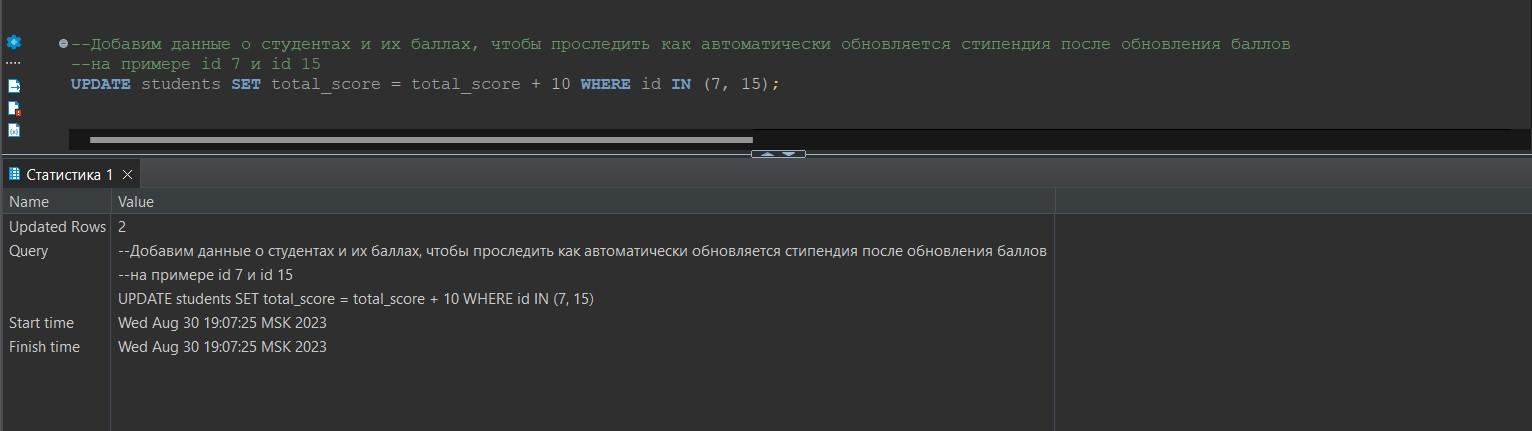
student\_scholarship **DESC**, s.id;



--Добавим данные о студентах и их баллах, чтобы проследить как автоматически обновляется стипендия после обновления баллов

--на примере id 7 и id 15

**UPDATE** students **SET** total\_score = total\_score + 10 **WHERE** id **IN** (7, 15);



--Рассмотрим обновленные данные о баллах и стипендии студента

**SELECT**

s.id **AS** student\_id,

s.**name** **AS** student\_name,

s.total\_score **AS** student\_total\_score,

**CASE**

**WHEN** s.total\_score >= 90 **THEN** 1000

**WHEN** s.total\_score >= 80 **THEN** 500

**ELSE** 0

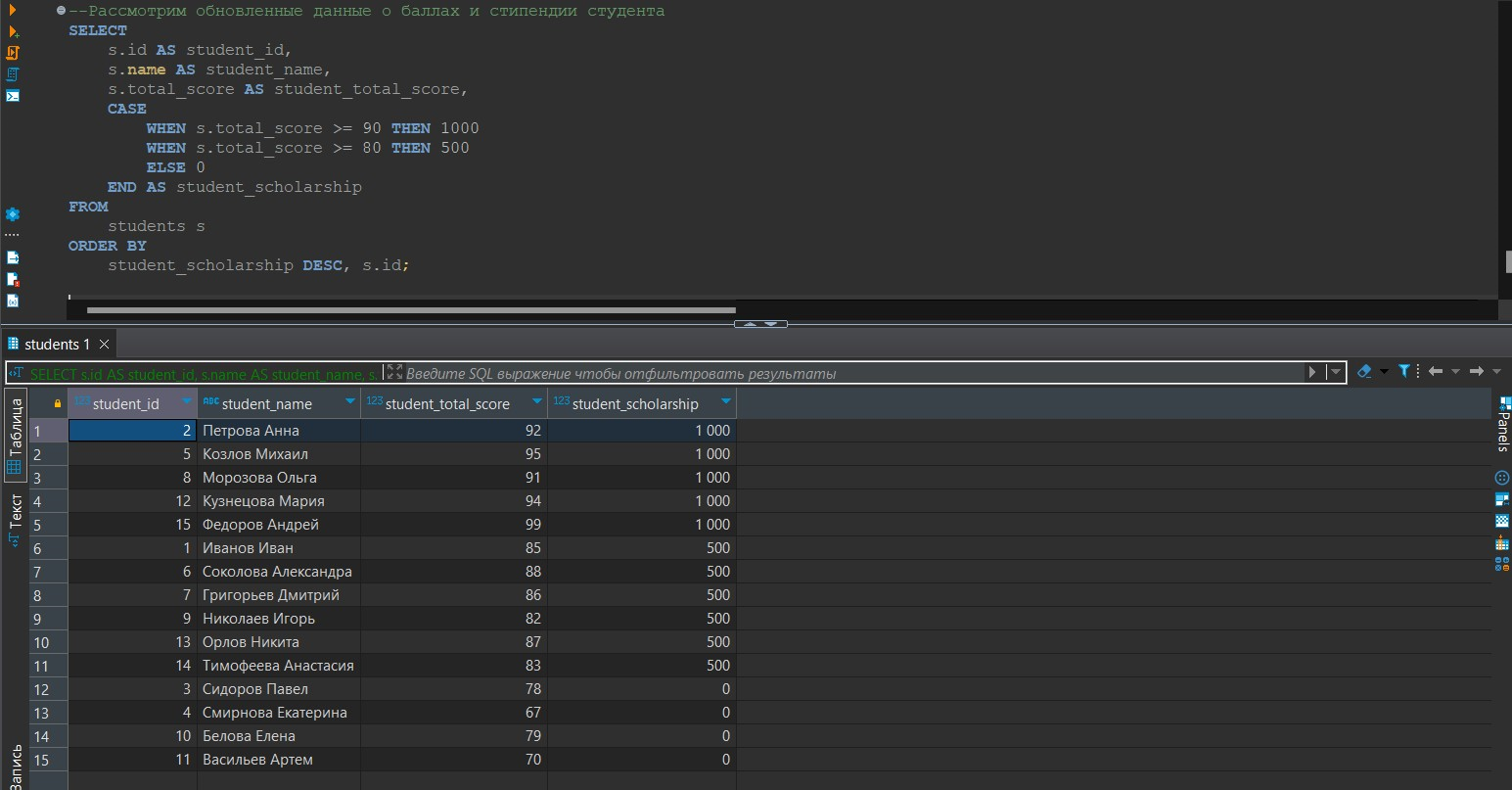
**END** **AS** student\_scholarship

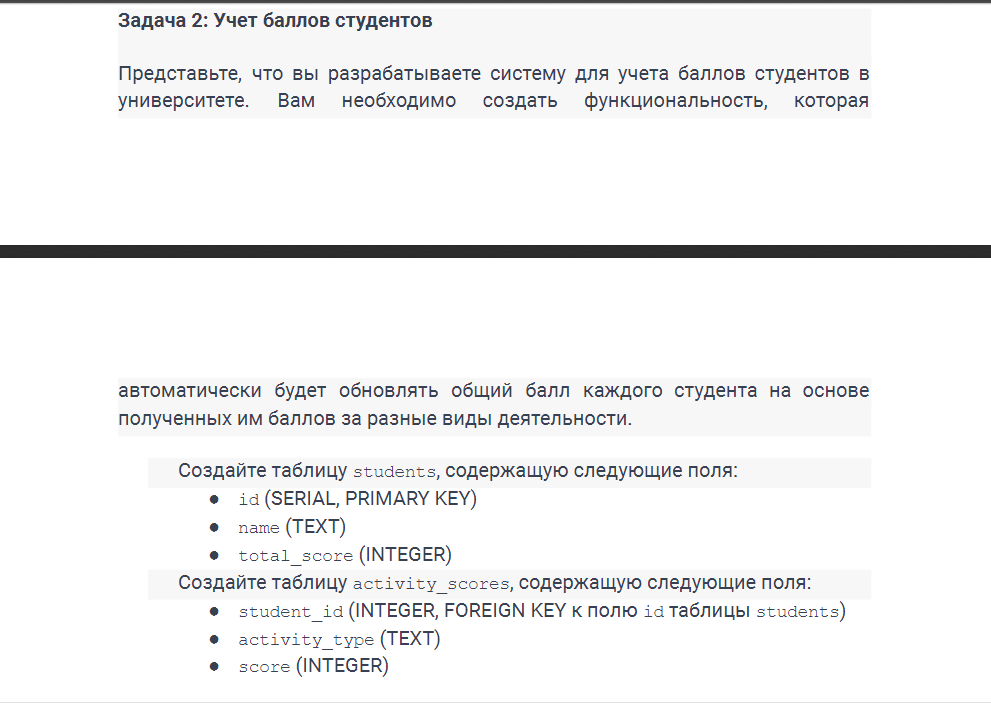
**FROM**

students s

**ORDER** **BY**

student\_scholarship **DESC**, s.id;







-- Создаем функцию update\_total\_score

**CREATE** **OR** **REPLACE** **FUNCTION** update\_total\_score()

**RETURNS** **TRIGGER** **AS** **$$**

**BEGIN**

**UPDATE** students

**SET** total\_score = (

**SELECT** **COALESCE**(**SUM**(score), 0)

**FROM** activity\_scores

**WHERE** student\_id = **NEW**.student\_id

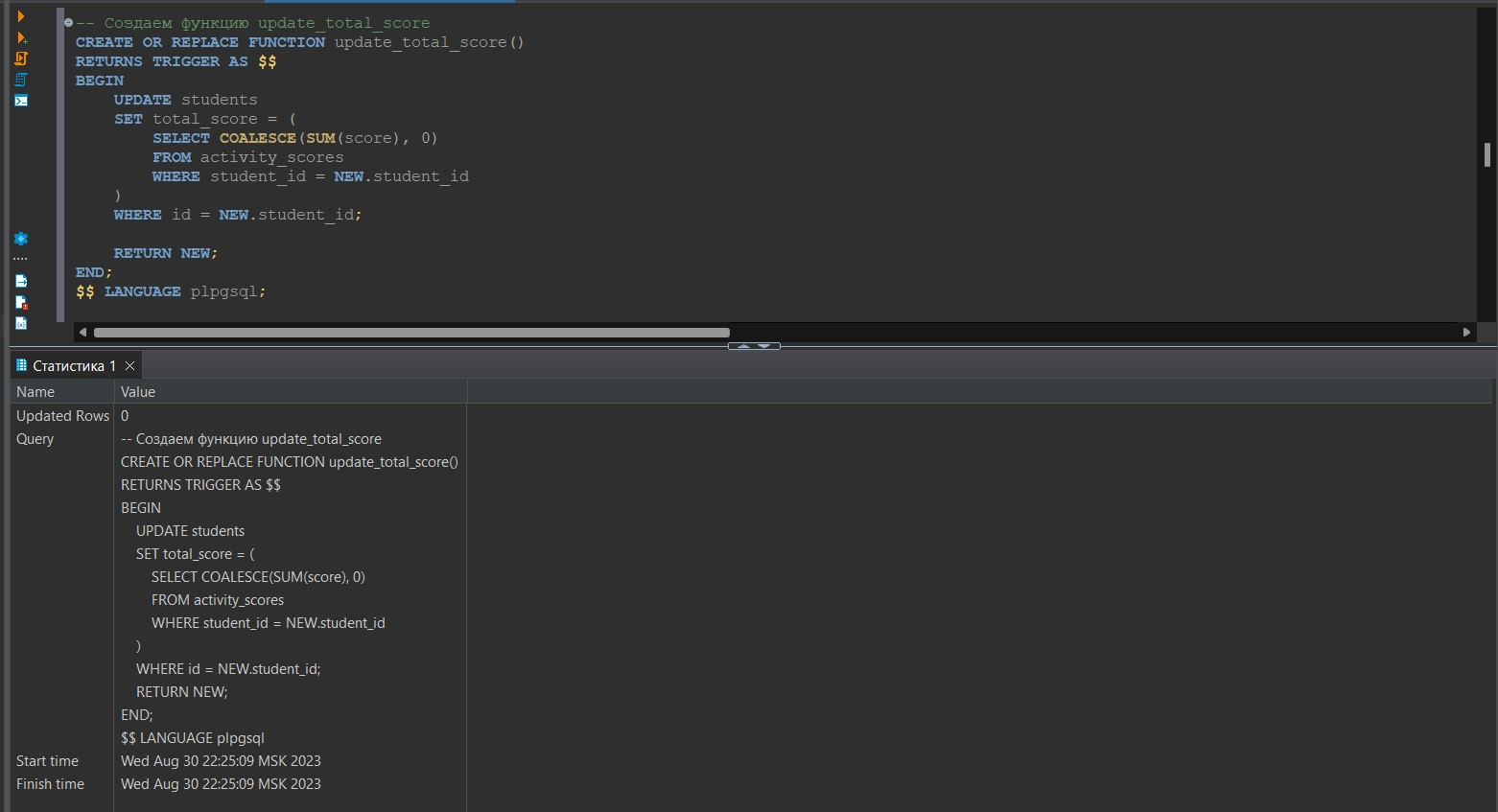
)

**WHERE** id = **NEW**.student\_id;

**RETURN** **NEW**;

**END**;

**$$** **LANGUAGE** plpgsql;



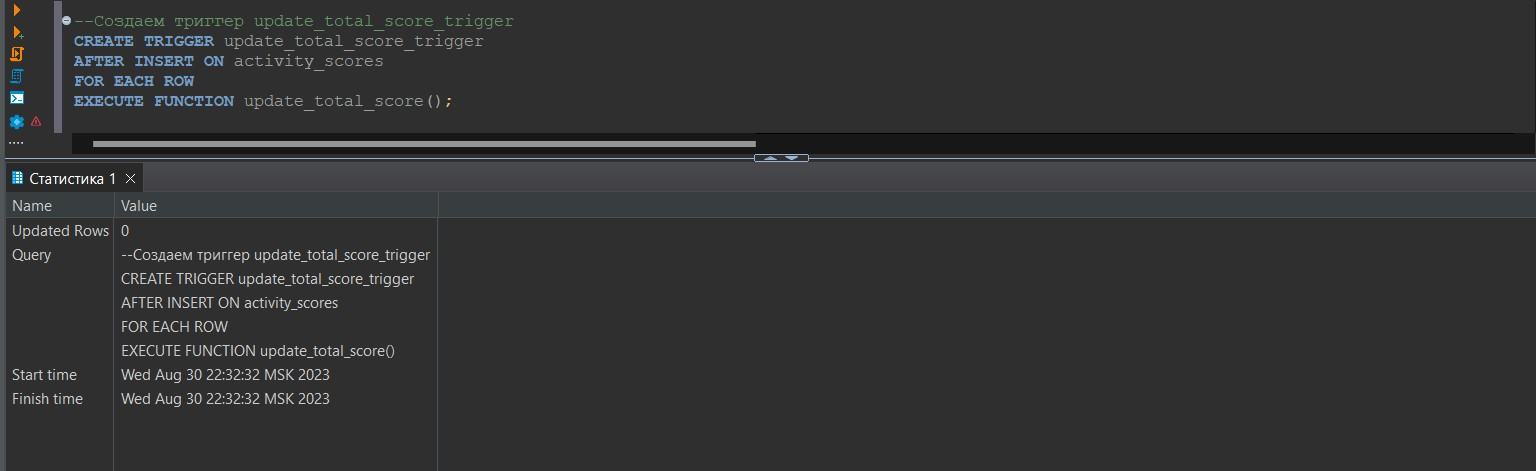
--Создаем триггер update\_total\_score\_trigger

**CREATE** **TRIGGER** update\_total\_score\_trigger

**AFTER** **INSERT** **ON** activity\_scores

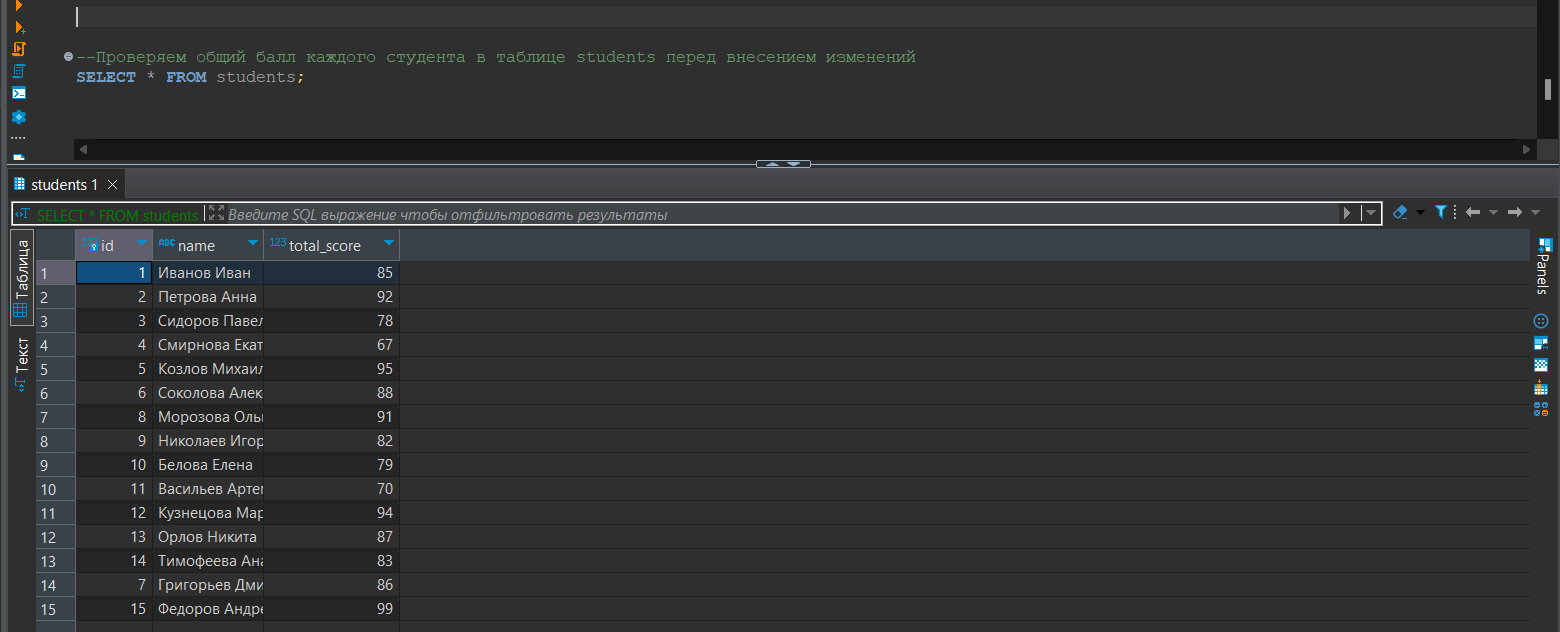
**FOR** **EACH** **ROW**

**EXECUTE** **FUNCTION** update\_total\_score();



--Проверяем общий балл каждого студента в таблице students перед внесением изменений

**SELECT** \* **FROM** students;



--Вносим дополнительные данные в таблицу students

**INSERT** **INTO** students (**name**, total\_score) **VALUES**

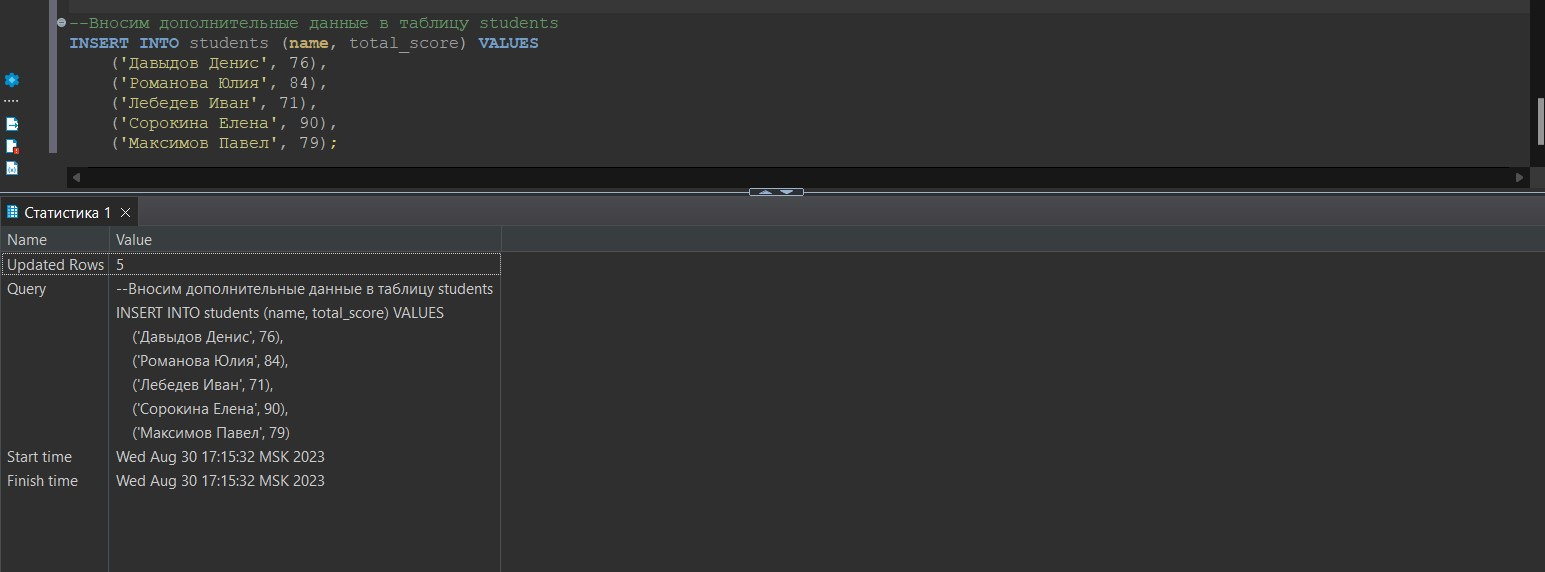
('Давыдов Денис', 76),

('Романова Юлия', 84),

('Лебедев Иван', 71),

('Сорокина Елена', 90),

('Максимов Павел', 79);



--Вносим дополнительные данные в таблицу activity\_scores

**INSERT** **INTO** activity\_scores (student\_id, activity\_type, score) **VALUES**

(16, 'Homework', 70),

(16, 'Exam', 80),

(17, 'Homework', 85),

(17, 'Exam', 92),

(18, 'Homework', 62),

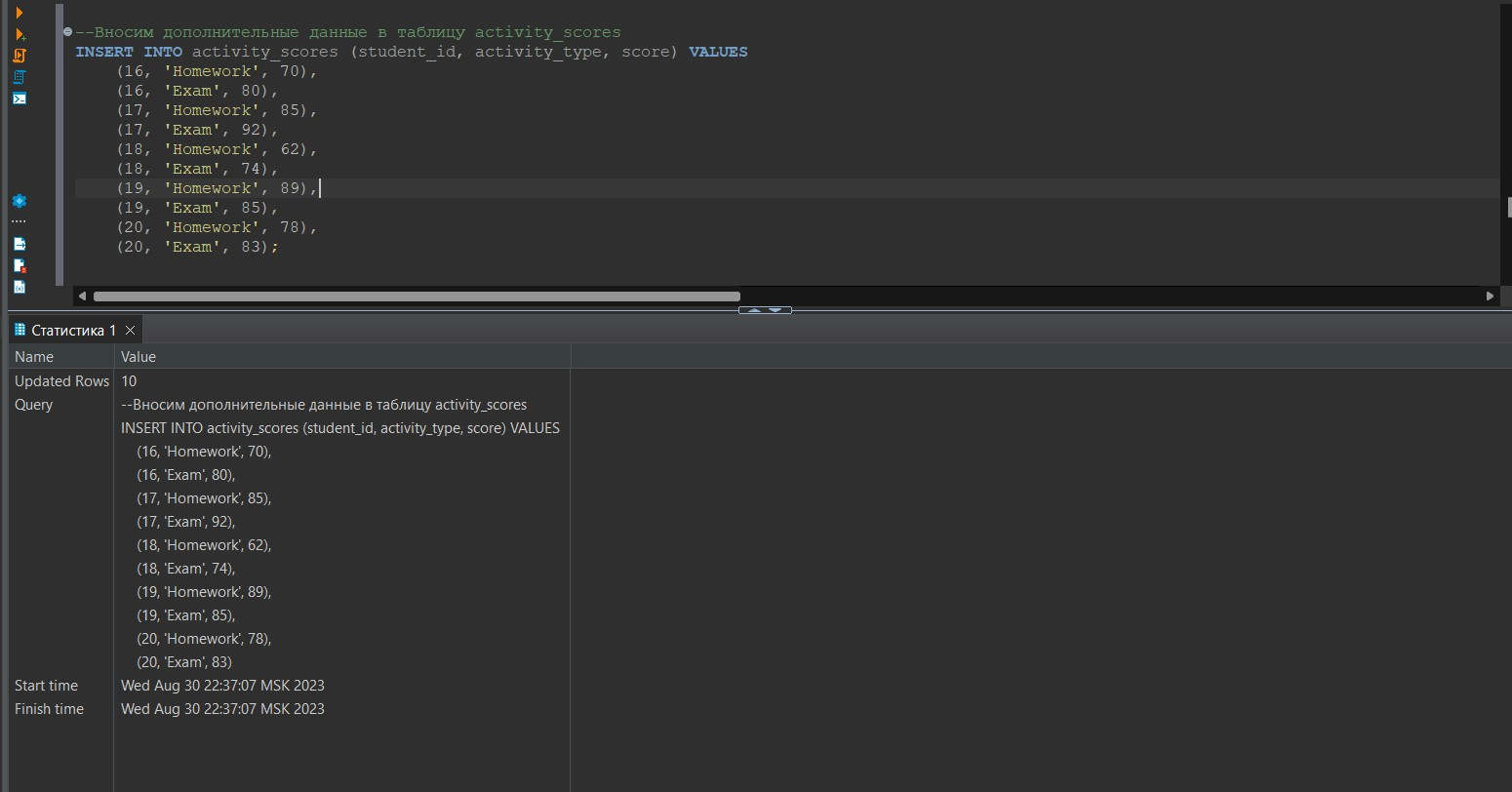
(18, 'Exam', 74),

(19, 'Homework', 89),

(19, 'Exam', 85),

(20, 'Homework', 78),

(20, 'Exam', 83);



--Проверяем, что общий балл каждого студента автоматически обновлен в таблице students.

**SELECT** \* **FROM** students;

