**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №5 «процедуры, функции, триггеры в PostgreSQL**»**

по дисциплине **«Проектирование и реализация баз данных»**

Автор: Едигарева Д.Р.

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

**Оглавление**

[Цель работы 3](#_Toc150736113)

[Практическое задание 3](#_Toc150736115)

[Вариант 12. БД «Прокат автомобилей» 3](#_Toc150736116)

[Выполнение 4](#_Toc150736117)

[Вывод 10](#_Toc150736118)

# Цель работы

# Овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

# Практическое задание

**Вариант 1 (max - 6 баллов)**

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
2. Создать триггер для логирования событий вставки, удаления, редактирования данных в базе данных PostgreSQL (согласно индивидуальному заданию, часть 5). Допустимо создать универсальный триггер или отдельные триггеры на логирование действий.

**Вариант 2 (max - 8 баллов)**

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).

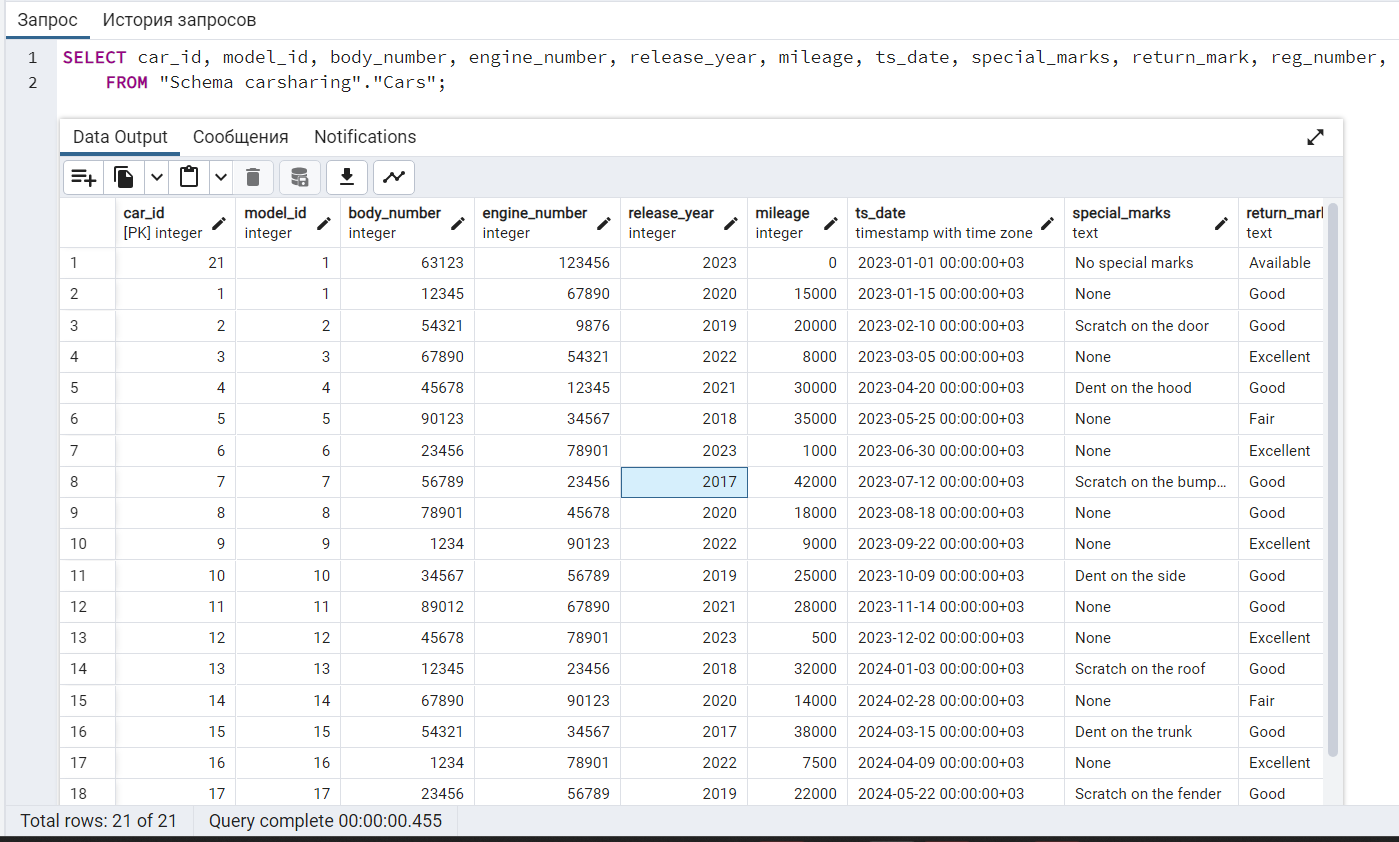
*2.1.* Модифицировать триггер (триггерную функцию) на проверку корректности входа и выхода сотрудника (см. Практическое задание 1 Лабораторного практикума (Приложение)) с максимальным учетом «узких» мест некорректных данных по входу и выходу).

*2.2.* Создать авторский триггер по варианту индивидуального задания.

# Выполнение

# Процедуры/функции согласно индивидуальному заданию (часть 4).

* Выполнить списание автомобилей, выпущенных ранее заданного года.



CREATE OR REPLACE PROCEDURE remove\_cars(

IN Year INT

)

LANGUAGE plpgsql

AS $$

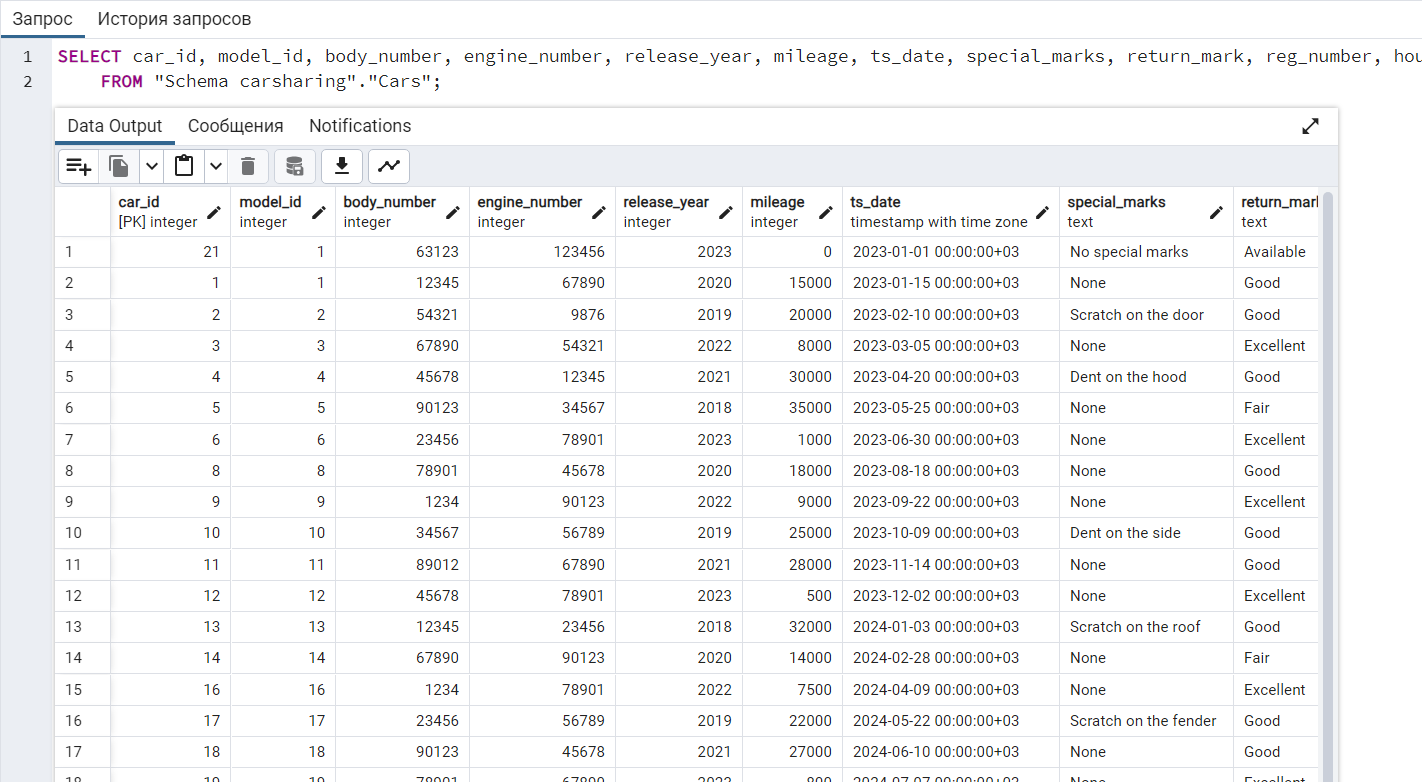
BEGIN

DELETE FROM "Cars" WHERE "release\_year" < Year;

END;

$$;

CALL remove\_cars(2018);



* Выдачи автомобиля и расчета стоимости с учетом скидки постоянным клиентам.

CREATE OR REPLACE FUNCTION Rental\_car\_with\_discount(

ClientID INT,

CarID INT,

StaffID INT,

ActID INT,

RentalDate TIMESTAMP WITH TIME ZONE,

ReturnDate TIMESTAMP WITH TIME ZONE

)

RETURNS DOUBLE PRECISION

LANGUAGE plpgsql

AS $$

DECLARE

Discount INT;

TotalCost DOUBLE PRECISION;

BEGIN

SELECT COALESCE("Clients"."discount", 0)

INTO Discount

FROM "Schema carsharing"."Clients"

WHERE client\_id = ClientID;

SELECT COALESCE(car\_rental\_price, 0) \* EXTRACT(HOUR FROM ReturnDate - RentalDate) \* (1 - Discount / 100.0)

INTO TotalCost

FROM "Schema carsharing"."Rental rate";

INSERT INTO "Schema carsharing"."Contract" (car\_id, client\_id, staff\_id, act\_id, start\_date\_and\_time, end\_date\_and\_time)

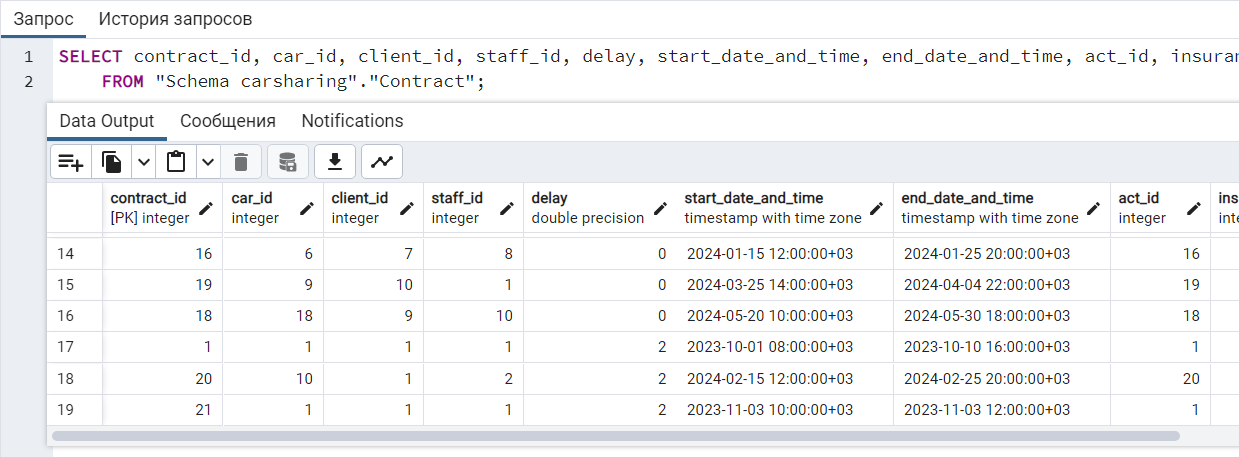
VALUES (CarID, ClientID, StaffID, ActID, RentalDate, ReturnDate);

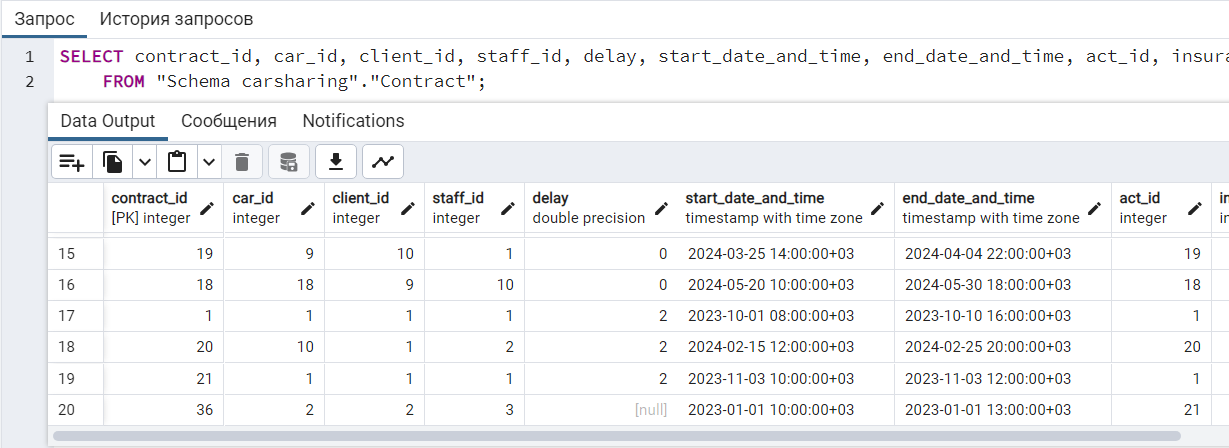
RETURN TotalCost;

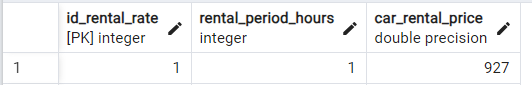
END;

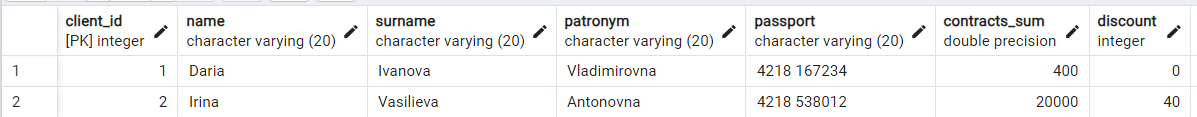
$$;

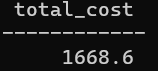
SELECT Rental\_car\_with\_discount(2, 2, 3, 21, '2023-01-01 10:00:00 MSK'::TIMESTAMP WITH TIME ZONE, '2023-01-01 13:00:00'::TIMESTAMP WITH TIME ZONE) AS total\_cost;











* Для вычисления количества автомобилей заданной марки.

CREATE OR REPLACE FUNCTION Count\_cars\_by\_brand(need\_brand VARCHAR)

RETURNS INT

LANGUAGE plpgsql

AS $$

DECLARE

car\_count INT;

BEGIN

SELECT COUNT(\*)

INTO car\_count

FROM "Schema carsharing"."Cars"

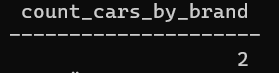
WHERE model\_id in (SELECT model\_id FROM "Schema carsharing"."Model" WHERE brand = need\_brand);

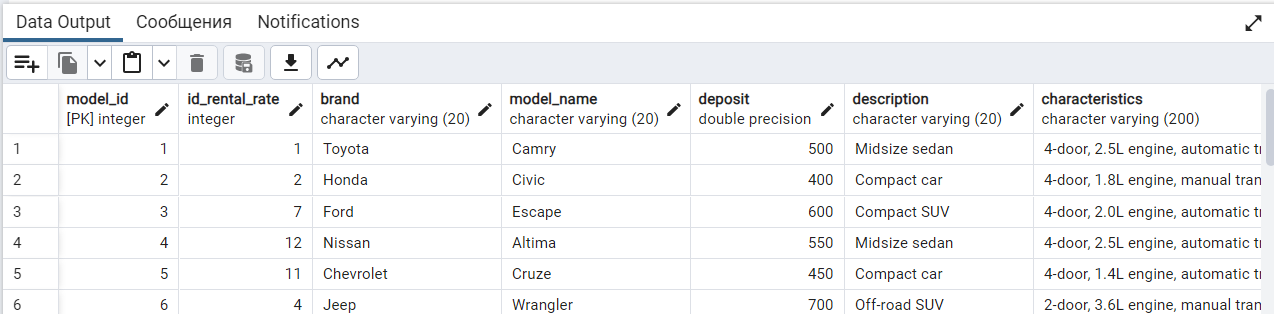
RETURN car\_count;

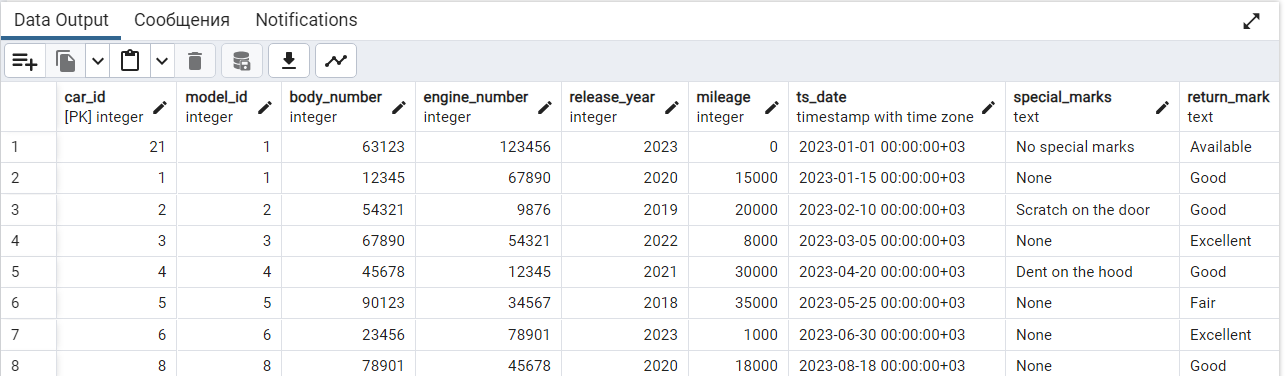
END;

$$;

SELECT Count\_cars\_by\_brand('Toyota');







**Авторский триггер по варианту индивидуального задания.**

CREATE OR REPLACE FUNCTION check\_contract\_extension()

RETURNS TRIGGER AS $$

BEGIN

IF NEW.start\_date\_and\_time <> (SELECT end\_date\_and\_time FROM "Schema carsharing"."Contract" WHERE contract\_id = NEW.contract\_id) THEN

RAISE EXCEPTION 'The beginning of the contract extension must be equal to the end of the contract itself.';

END IF;

RETURN NEW;

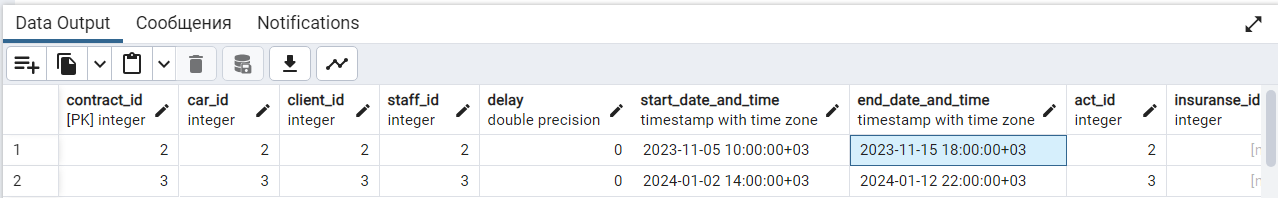
END;

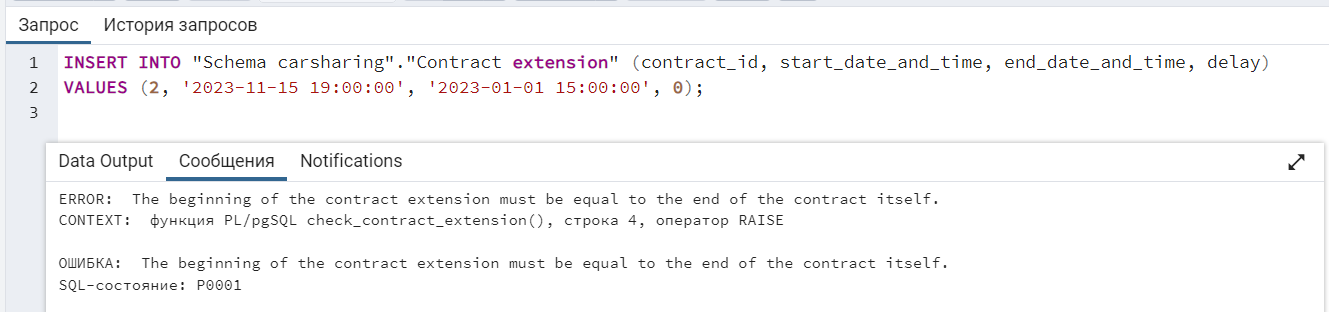
$$ LANGUAGE plpgsql;

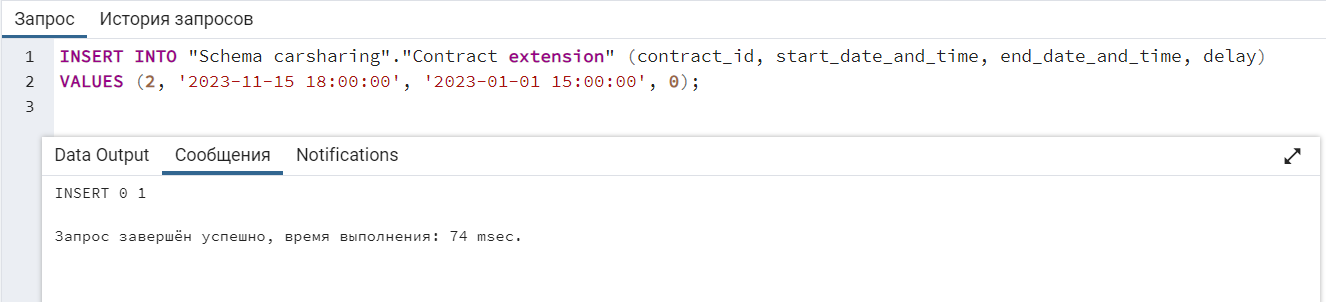
CREATE TRIGGER trg\_check\_contract\_extension

BEFORE INSERT OR UPDATE ON "Schema carsharing"."Contract extension"

FOR EACH ROW EXECUTE FUNCTION check\_contract\_extension();







# Вывод

# Мы овладели практическими навыками создания и использования процедур, функций и триггеров в базе данных PostgreSQL, научились писать и тестировать их в консоли SQL Shell.