Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное
учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет инфокоммуникационных технологий

ЛАБОРАТОРНАЯ РАБОТА №3 «процедуры, функции, триггеры в PostgreSQL»

Выполнил:
студент : Аль-Мошки Исмаил Абдулвахаб
группа: K32401

Проверили:
Говорова Марина Михайловна

Санкт-Петербург
2023

Цель работы: овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД PostgreSQL, SQL Shell (psql).

Практическое задание:

Вариант 1

1. 2.Создать процедуры/функции согласно индивидуальному заданию и (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

1. Создать процедуры/функции согласно индивидуальному заданию и (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.

Указание. Работа выполняется в консоли SQL Shell (psql).

Вариант 12. БД «Прокат автомобилей»

Описание предметной области: Компания предоставляет прокат автомобилей. В пункт проката обращаются клиенты, данные о которых регистрируют в базе. Цена проката зависит от марки автомобиля, технических характеристик и года выпуска.

Для проката авто с клиентом заключается договор, в котором фиксируется период проката, вид страховки, стоимость страховки, залоговая стоимость. Залоговая стоимость возвращается полностью или частично клиенту, в зависимости от страховки, аварий и штрафов. Если залоговая стоимость уже возвращена клиенту, но на авто в компанию пришел штраф, то он оплачивается компанией, а не клиентом. При передаче авто клиенту составляется акт о передаче автомобиля клиенту. При возвращении автомобиля также составляется акт о передаче авто компании.

Если клиент не вернул автомобиль в срок и не оформил продление, ему назначается штраф за каждый час просрочки.

Постоянным клиентам предоставляются скидки.

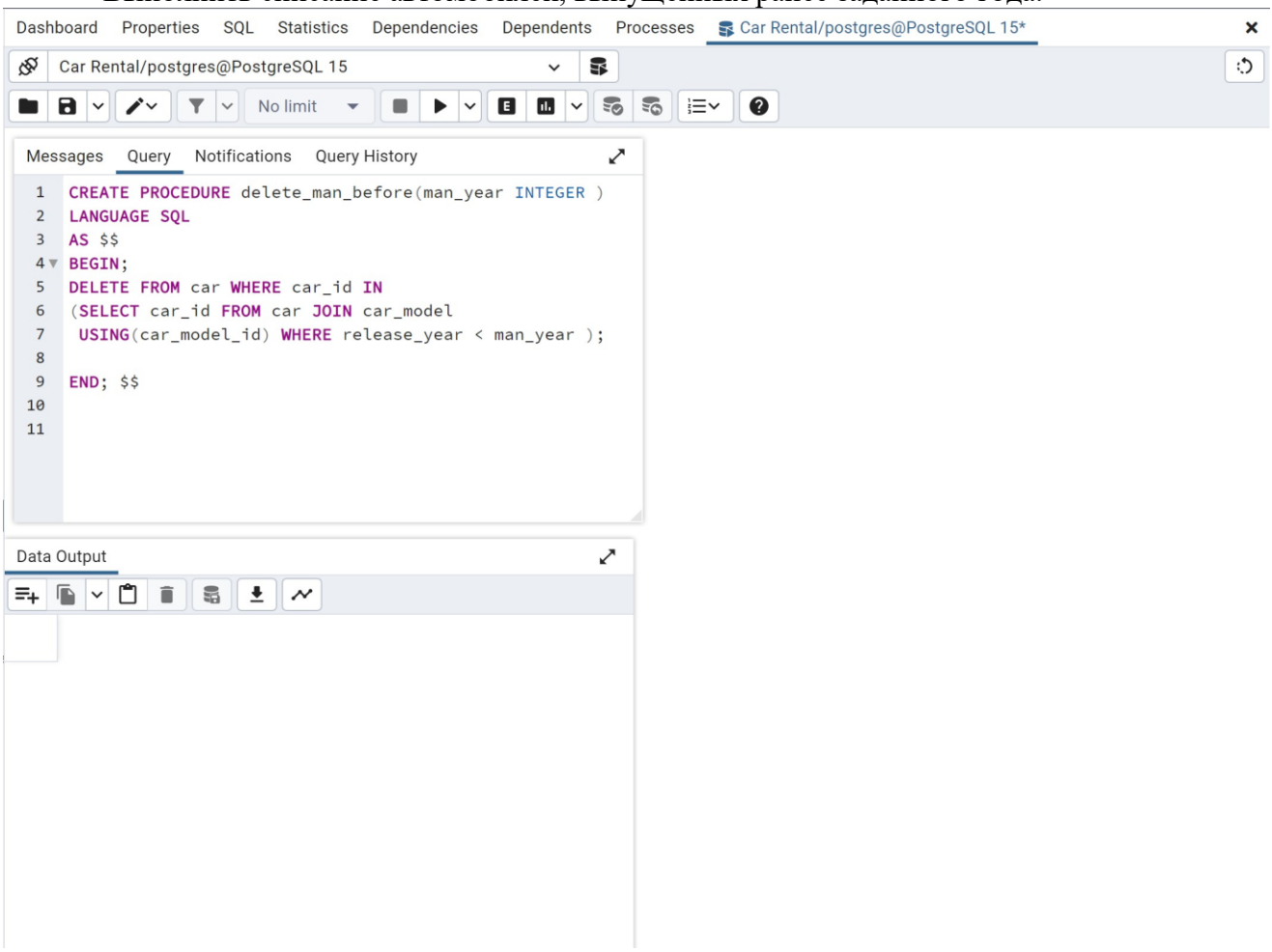
В системе необходимо хранить историю штрафов и аварий автомобилей.

Цены на прокат автомобилей могут меняться.

БД должна содержать следующий минимальный набор сведений: ФИО. Паспортные данные. Код должности. Наименование должности. Оклад. Обязанности. Код марки. Наименование. Технические характеристики. Описание. Код автомобиля. Регистрационный номер. Номер кузова. Номер двигателя. Год выпуска. Пробег. Цена автомобиля. Цена проката. Дата последнего ТО. Специальные отметки. Отметка о возврате. Код клиента. ФИО. Адрес. Телефон. Паспортные данные. Дата и время выдачи автомобиля. На сколько часов. Дата и время возврата автомобиля. Данные о нарушениях. Данные об авариях. Дата продления. Часов продления.

Создать хранимые процедуры:

- Выполнить списание автомобилей, выпущенных ранее заданного года.



Dashboard Properties SQL Statistics Dependencies Dependents Processes Car Rental/pos... Car Rental/postgres@PostgreSQL 15*

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

```
1  SELECT * FROM car JOIN car_model USING (car_model_id)
```

Data Output

	...l_marks ...er varying	in_rent boolean	car_code character varying (10)	car_price money	milage integer	model_name character varying (30)	release_year integer	rent_per_hour numeric	technical_characteristics character varying
2		false	run432	12,000.00 ر.س.	6000	TOYOTA	2015	800	[null]
3		false	rat654	11,500.00 ر.س.	6000	JEEP CHEROKI	2019	1000	[null]
4		false	run875	13,000.00 ر.س.	6000	Mutsubishi	2015	800	[null]
5		false	rat154	12,000.00 ر.س.	6000	Mazda car	2015	800	[null]
6		false	run134	11,000.00 ر.س.	6000	Mazda car	2015	800	[null]
7		false	rat243	16,000.00 ر.س.	6000	Hiundai	2019	1000	[null]
8		false	run123	12,000.00 ر.س.	7000	JEEP CHEROKI	2019	1000	[null]
9		false	can234	7,000.00 ر.س.	100000	matiz	2011	500	[null]
10		false	cvb234	7,000.00 ر.س.	90000	matiz	2011	500	[null]
11		false	cvv233	7,000.00 ر.س.	96500	matiz	2011	500	[null]

Dashboard Properties SQL Statistics Dependencies Dependents Processes Car Rental/pos... Car Rental/postgres@PostgreSQL 15*

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

```
1  CALL  delete_man_before(2012);
2
3  SELECT * FROM car JOIN car_model USING (car_model_id);
```

Data Output

	(10)	car_price money	milage integer	model_name character varying (30)	release_year integer	rent_per_hour numeric	technical_characteristics character varying	car_brand character varying (20)	collateral_sum numeric
1		11,000.00 ر.س.	6000	Mazda car	2015	800	[null]	Mazda	7000
2		12,000.00 ر.س.	6000	TOYOTA	2015	800	[null]	Mazda	7000
3		11,500.00 ر.س.	6000	JEEP CHEROKI	2019	1000	[null]	Jeep	10000
4		13,000.00 ر.س.	6000	Mutsubishi	2015	800	[null]	Mazda	7000
5		12,000.00 ر.س.	6000	Mazda car	2015	800	[null]	Mazda	7000
6		11,000.00 ر.س.	6000	Mazda car	2015	800	[null]	Mazda	7000
7		16,000.00 ر.س.	6000	Hiundai	2019	1000	[null]	Jeep	10000
8		12,000.00 ر.س.	7000	JEEP CHEROKI	2019	1000	[null]	Jeep	10000

Total rows: 8 of 8 | Query complete 00:00:00.252 | Ln 3, Col 55

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

```
CREATE PROCEDURE create_contract (con_id INT, cus_id INT,
                                  car1_id INT, ins_id INT ,
                                  pickup_date DATE , expected_return_date DATE
                                 ,rent_hours INT
                                 )
language plpgsql
as $$
BEGIN
INSERT INTO contract VALUES(con_id,cus_id,car1_id,ins_id
                                                ,pickup_date,expected_return_date,
                                                null,false,rent_hours,(
                                                        (rent_hours*
                                                                (SELECT rent_per_hour FROM car
JOIN car_model
                                                                 USING(car_model_id)        WHERE
car_id = car1_id))-
                                                        (
                                                                 rent_hours*(SELECT rent_per_hour
FROM car JOIN
                                                                        car_model
USING(car_model_id) WHERE
                                                                        car_id = car1_id)*
                                                                (SELECT        discount_percentage
FROM
                                                                         customer JOIN customer_type
                                                                        USING(customer_type_id)
                                                                        WHERE customer_id = cus_id)
                                                        )
                                                )
                                                ,
                                                        0 , 0 , false

                                                        );
END; $$
```

Dashboard Properties SQL Statistics Dependencies Dependents Processes Car Rental/pos... Car Rental/postgres@PostgreSQL 15*

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

```
CREATE PROCEDURE create_contract (con_id INT, cus_id INT,
                car1_id INT, ins_id INT ,
                pickup_date DATE , expected_return_date DATE
               ,rent_hours INT
               )
language plpgsql
as $$
BEGIN
INSERT INTO contract VALUES(con_id,cus_id,car1_id,ins_id
                           ,pickup_date,expected_return_date,
                           null,false,rent_hours,(
                               (rent_hours*
                                (SELECT rent_per_hour FROM car JOIN car_model
                                 USING(car_model_id) WHERE car_id = car1_id))-
                               (
                                   rent_hours*(SELECT rent_per_hour FROM car JOIN
                                    car_model USING(car_model_id) WHERE
                                    car_id = car1_id)*
                                   (SELECT discount_percentage FROM
                                     customer JOIN customer_type
                                    USING(customer_type_id)
                                    WHERE customer_id = cus_id)
                               )
                           )
                           ,
                            0 , 0 , false

                               );
END; $$

```

Dashboard Properties SQL Statistics Dependencies Dependents Processes Car Rental/pos... Car Rental/postgres@PostgreSQL 15*

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

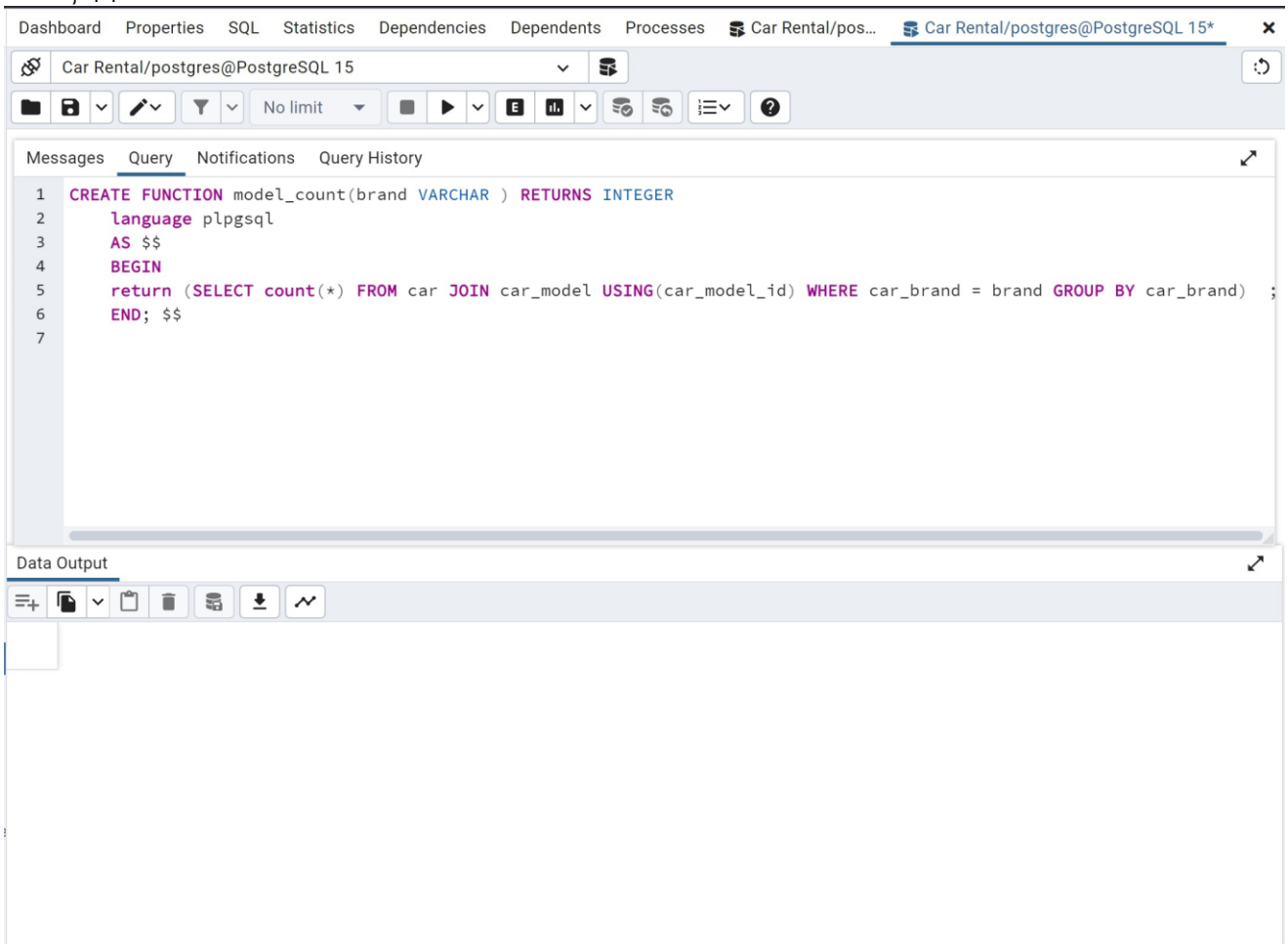
```
CALL create_contract (3,3,3,2,CURRENT_DATE, '2023-06-01',48);
SELECT * FROM contract;
```

Data Output

	ar_pickup_date ate	car_expected_return_date date	car_actual_return_date date	is_extended boolean	rent_hours integer	rent_cost money	extension_hours integer	hours_late integer	is_return boolean
1	023-05-15	2023-05-17	[null]	false	70	3,800.00 ر.س.	[null]	[null]	false
2	023-05-15	2023-05-22	[null]	false	40	4,800.00 ر.س.	[null]	[null]	false
3	023-05-15	2023-05-30	[null]	false	40	4,800.00 ر.س.	[null]	30	false
4	023-05-25	2023-06-01	[null]	false	24	18,960.00 ر.س.	0	0	false
5	023-05-25	2023-06-01	[null]	false	24	18,960.00 ر.س.	0	0	false
6	023-05-25	2023-06-01	[null]	false	24	19,200.00 ر.س.	0	0	false
7	023-05-25	2023-06-01	[null]	false	48	38,400.00 ر.س.	0	0	false

- Для вычисления количества автомобилей заданной марки.

```
CREATE FUNCTION model_count(brand VARCHAR ) RETURNS INTEGER
language plpgsql
AS $$
BEGIN
return (SELECT count(*) FROM car JOIN car_model USING(car_model_id) WHERE car_brand = brand GROUP BY car_brand) ;
END; $$
```



Dashboard Properties SQL Statistics Dependencies Dependents Processes Car Rental/pos... Car Rental/postgres@PostgreSQL 15*

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

```
1  SELECT car_brand,count(*) FROM car JOIN car_model USING(car_model_id) GROUP BY car_brand;
2
```

Data Output

	car_brand character varying (20)	count bigint
1	Mazda	5
2	Jeep	3

Dashboard Properties SQL Statistics Dependencies Dependents Processes Car Rental/pos... Car Rental/postgres@PostgreSQL 15*

Car Rental/postgres@PostgreSQL 15

No limit

Messages Query Notifications Query History

```
1  SELECT model_count('Jeep');
2
```

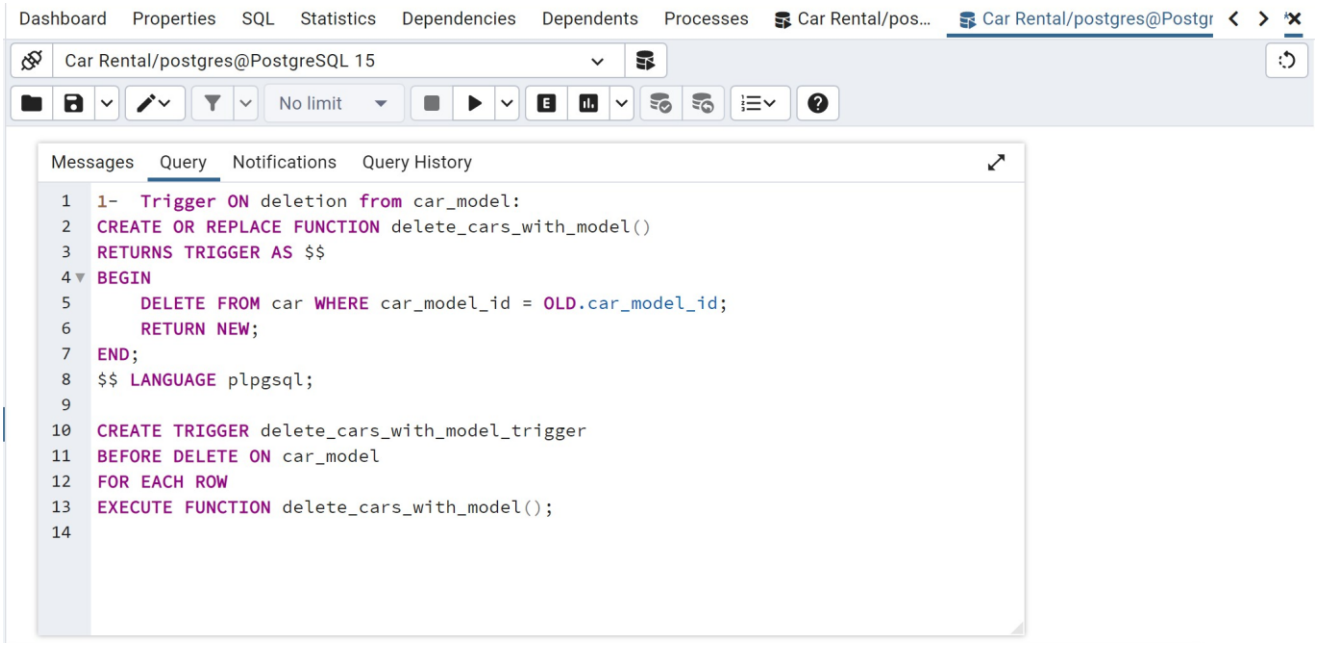
Data Output

	model_count integer
1	3

Создать необходимые триггеры:

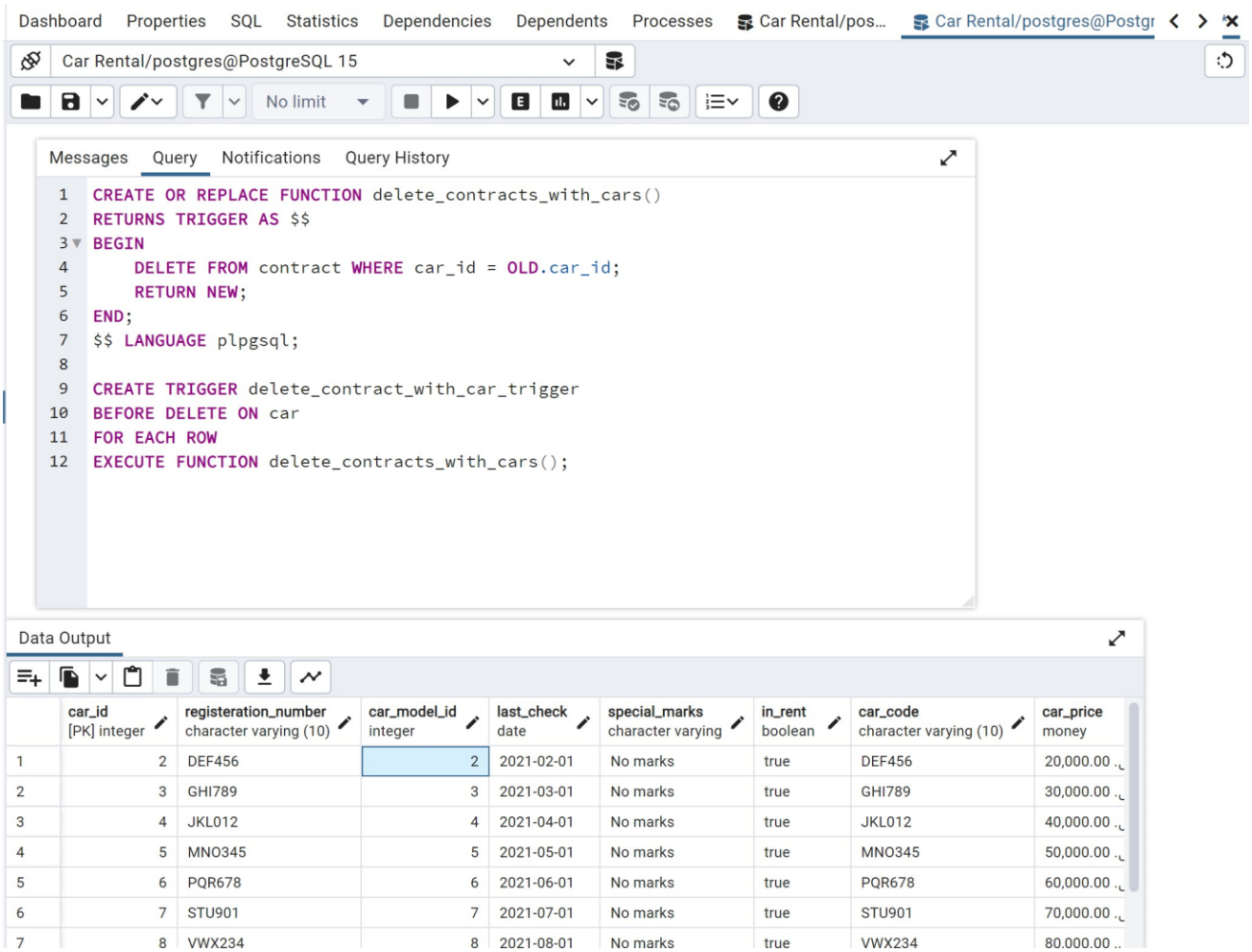
1- Trigger ON deletion from car_model:

```
CREATE OR REPLACE FUNCTION delete_cars_with_model()
RETURNS TRIGGER AS $$
BEGIN
   DELETE FROM car WHERE car_model_id = OLD.car_model_id;
   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER delete_cars_with_model_trigger
BEFORE DELETE ON car_model
FOR EACH ROW
EXECUTE FUNCTION delete_cars_with_model();
```



2-Trigger ON deletion from car:

```
CREATE OR REPLACE FUNCTION delete_contracts_with_cars()
RETURNS TRIGGER AS $$
BEGIN
   DELETE FROM contract WHERE car_id = OLD.car_id;
   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER delete_contract_with_car_trigger
BEFORE DELETE ON car
FOR EACH ROW
EXECUTE FUNCTION delete_contracts_with_cars();
```



	car_id [PK] integer	registeration_number character varying (10)	car_model_id integer	last_check date	special_marks character varying	in_rent boolean	car_code character varying (10)	car_price money
1	2	DEF456	2	2021-02-01	No marks	true	DEF456	20,000.00
2	3	GHI789	3	2021-03-01	No marks	true	GHI789	30,000.00
3	4	JKL012	4	2021-04-01	No marks	true	JKL012	40,000.00
4	5	MNO345	5	2021-05-01	No marks	true	MNO345	50,000.00
5	6	PQR678	6	2021-06-01	No marks	true	PQR678	60,000.00
6	7	STU901	7	2021-07-01	No marks	true	STU901	70,000.00
7	8	VWX234	8	2021-08-01	No marks	true	VWX234	80,000.00

Вывод:
Триггеры помогают поддерживать целостность данных, а функции и процедуры помогают автоматизировать процесс запроса данных.