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:
Смирнов Тимур Олегович
Группа: К3243

Проверил:
Говорова Марина Михайловна

Санкт-Петербург
2021

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. 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.

Для базы данных «Автомастерская»:

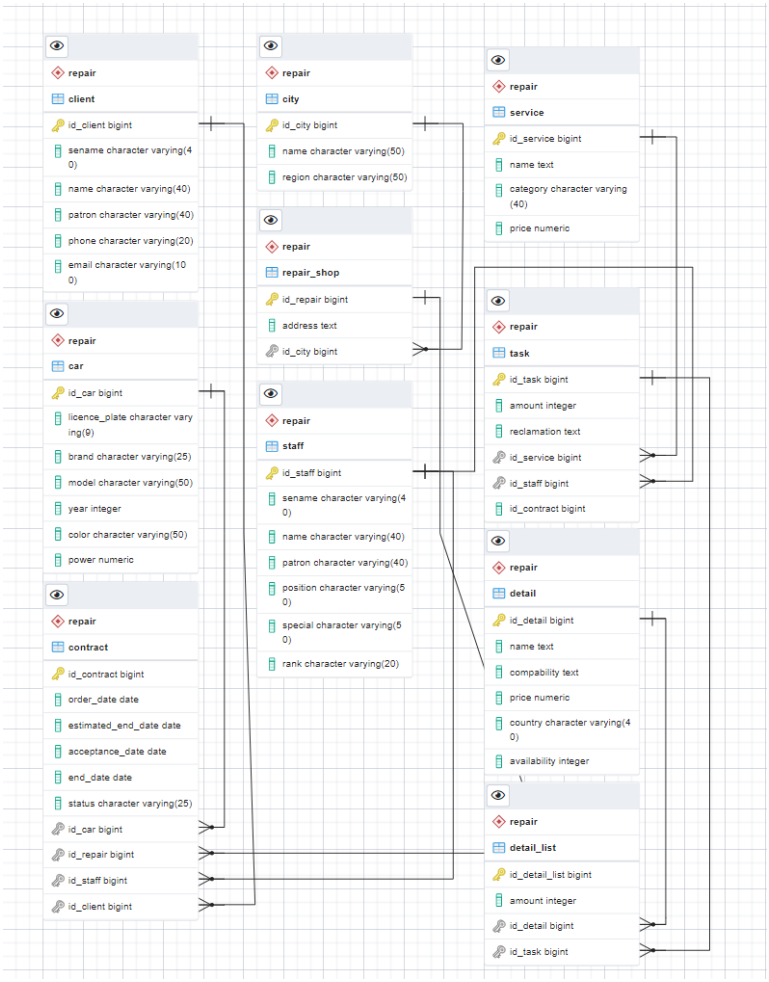

Рисунок 1 – ERD диаграмма БД

Выполнение:

Задание 1 – Создать хранимые процедуры:

- Повышения цены деталей для автомобиля “Ford” на 10 %.

	id_detail [PK] bigint	name text	compability text	price numeric	country character varying (40)	availability integer
1	1	dkde34df	bmw	2311.00	germany	2
2	2	sfgoo4	audi	13434.00	germany	134
3	3	adlkn43	kia	34134.00	korea	34
4	4	dkd3	ford	12454.00	italy	0
5	5	parasha	lada	10453.00	russia	3453
6	6	timur	ford	3944.00	russia	1

Рисунок 2 – Таблица до выполнения процедуры 1

Листинг 1 – Процедура 1

```
CREATE PROCEDURE repair.ford_price_raise()
LANGUAGE SQL
AS $$
    UPDATE repair.detail
    SET price = price * 1.1
    WHERE compability = 'ford'
$$;

CALL repair.ford_price_raise();
```

	id_detail [PK] bigint	name text	compability text	price numeric	country character varying (40)	availability integer
1	1	dkde34df	bmw	2311.00	germany	2
2	2	sfgoo4	audi	13434.00	germany	134
3	3	adlkn43	kia	34134.00	korea	34
4	4	dkd3	ford	13699.400	italy	0
5	5	parasha	lada	10453.00	russia	3453
6	6	timur	ford	4338.400	russia	1

Рисунок 3 – Таблица после выполнения процедуры 1

- Для повышения разряда тех мастеров, которые отремонтировали больше 3 автомобилей.

	id_staff [PK] bigint	sename character varying (40)	name character varying (40)	patron character varying (40)	position character varying (50)	special character varying (50)	rank integer
1	1	Григорян	Александр		помогите	Чистка	4
2	2	Московская	Мария	Ивановна	мне	Починка	3
3	3	Григорян	Арина	Кришкина	я	Менеджмент	2
4	4	Рускоид	Даниил		устал	Починка	5
5	5	Петросян	Анна		это	Хз	3
6	6	Иванов	Иван	Иванович	заполнять	я устал	1

Рисунок 4 – Таблица до выполнения процедуры 2

Листинг 2 – Процедура 2

```
CREATE PROCEDURE repair.raise_staff_rank()
LANGUAGE SQL
AS $$
    UPDATE repair.staff
    SET rank = rank - 1
    WHERE
        rank >= 1 AND
        id_staff IN (
            SELECT DISTINCT id_staff FROM repair.task
            GROUP BY id_staff
            HAVING COUNT(*) > 3
        )
$$;

CALL repair.raise_staff_rank();
```

	id_staff [PK] bigint	sename character varying (40)	name character varying (40)	patron character varying (40)	position character varying (50)	special character varying (50)	rank integer
1	1	Григорян	Александр		помогите	Чистка	4
2	2	Московская	Мария	Ивановна	мне	Починка	3
3	3	Григорян	Арина	Кришкина	я	Менеджмент	1
4	4	Рускоид	Даниил		устал	Починка	5
5	5	Петросян	Анна		это	Хз	3
6	6	Иванов	Иван	Иванович	заполнять	я устал	1

Рисунок 5 – Таблица после выполнения процедуры 2

- Сколько автомобилей отремонтировал каждый механик за истекший квартал.

Листинг 3 – Фунцкия

```
CREATE FUNCTION repair.show_staff_cars() RETURNS TABLE(id bigint, "Фамилия" character
varying, "Кол-во машин" int)
AS
$$
SELECT id_staff, sename, COUNT(*) from repair.task NATURAL JOIN repair.staff
WHERE id_contract IN (
                        SELECT id_contract FROM repair.contract
                        WHERE end_date > current_date - interval '3 month'
                    )
GROUP BY id_staff, sename
$$
LANGUAGE SQL;

select * from repair.show_staff_cars();
```

	id bigint	Фамилия character varying	Кол-во машин integer
1	5	Петросян	1
2	3	Григорян	7

Рисунок 6 – Результат функции

Задание 2 – Создать тригер для логирования событий вставки, удаления, редактирования.

Была создана таблица logs.

Листинг 4 – Триггер

```
CREATE OR REPLACE FUNCTION repair.add_to_log() RETURNS TRIGGER AS $$
DECLARE
    mstr varchar(30);
    astr varchar(100);
    retstr varchar(254);
BEGIN
    IF TG_OP = 'INSERT' THEN
        astr = NEW;
        mstr := 'Add new data ';
        retstr := mstr||astr;
        INSERT INTO repair.logs(text, added, table_name) values (retstr, NOW(),
TG_TABLE_NAME);
        RETURN NEW;
```

```
    ELSIF TG_OP = 'UPDATE' THEN
        astr = NEW;
        mstr := 'Update data ';
        retstr := mstr||astr;
        INSERT INTO repair.logs(text, added, table_name) values (retstr, NOW(),
TG_TABLE_NAME);
        RETURN NEW;
    ELSIF TG_OP = 'DELETE' THEN
        astr = OLD;
        mstr := 'Remove data ';
        retstr := mstr || astr;
        INSERT INTO repair.logs(text, added, table_name) values (retstr, NOW(),
TG_TABLE_NAME);
        RETURN OLD;
    END IF;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER tr_car AFTER INSERT OR UPDATE OR DELETE
ON repair.car FOR EACH ROW EXECUTE PROCEDURE repair.add_to_log();


INSERT INTO repair.car(licence_plate, brand, model, year, color, power)
VALUES ('k345df12', 'kia', 'optima', 2019, 'black', 3.14);

UPDATE repair.car SET color = 'red' WHERE licence_plate = 'k345df12';

DELETE FROM repair.car WHERE licence_plate = 'k345df12';

SELECT * FROM repair.logs
```

	text text	added date	table_name character varying
1	Add new data (15,k345df12,kia,optima,2019,black,3.14)	2022-05-06	car
2	Update data (15,k345df12,kia,optima,2019,red,3.14)	2022-05-06	car
3	Remove data (15,k345df12,kia,optima,2019,red,3.14)	2022-05-06	car

Рисунок 7 – Результат вывода событий с тригером

Вывод:

В ходе работы были изучены принципы работы и процесс создания процедур и функций, а также триггеры.