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»

Обучающийся Милютин Никита Александрович
Факультет прикладной информатики
Группа K3239
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии 2023
Преподаватель Говорова Марина Михайловна

Санкт-Петербург

2025/2026

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 (min - 6 баллов, max - 10 баллов, доп. баллы - 3):

1. Создать 3 процедуры для индивидуальной БД согласно варианту (часть 4 ЛР 2). Допустимо использование IN/OUT параметров. Допустимо создать авторские процедуры. (3 балла)
2. Создать триггеры для индивидуальной БД согласно варианту:
   Вариант 2.1. 3 триггера - 3 балла (min). Допустимо использовать триггеры логирования из практического занятия по функциям и триггерам.
   Вариант 2.2. 7 оригинальных триггеров - 7 баллов (max).

Практическое задание (min - 6 баллов, max - 10 баллов, доп. баллы - 3):

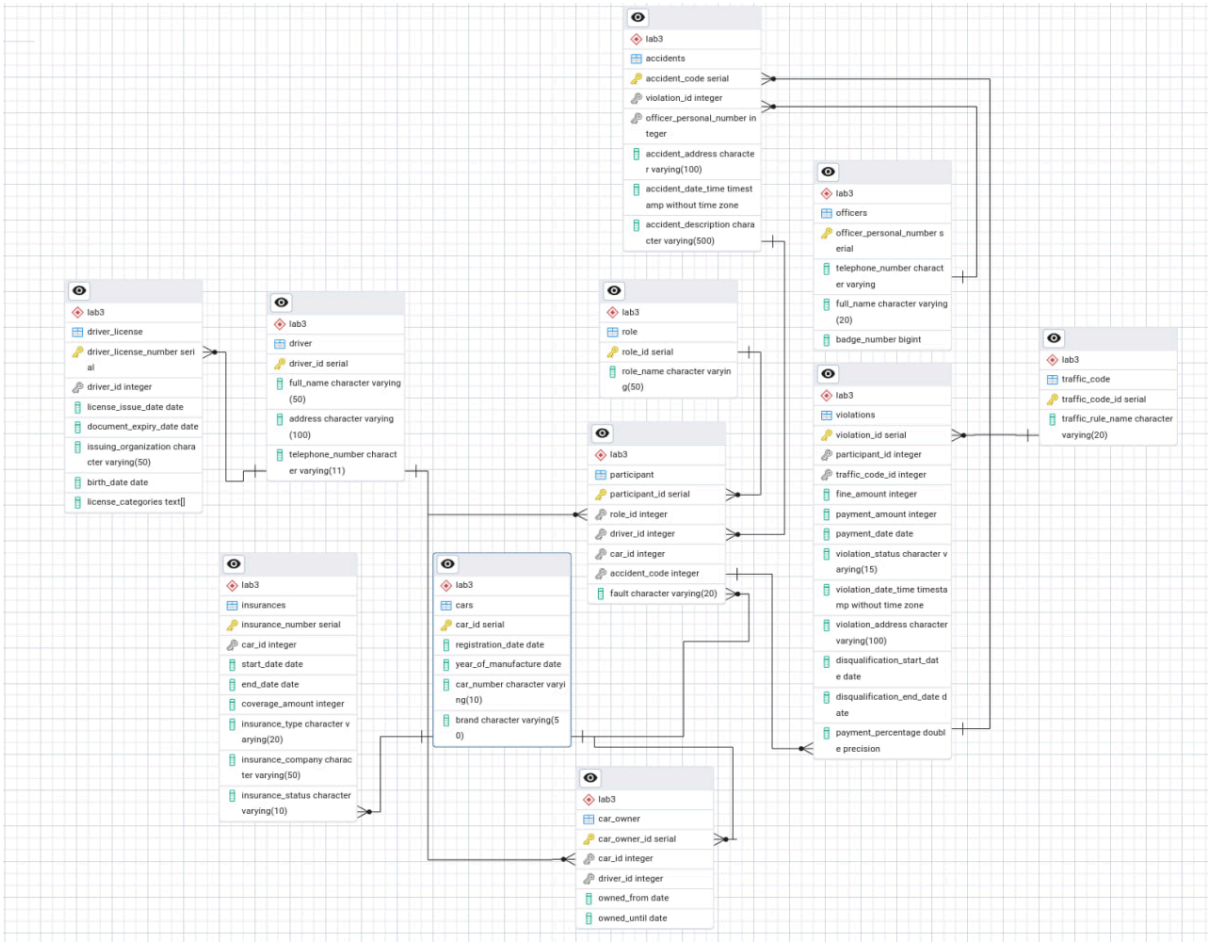
Создать 3 процедуры для индивидуальной БД согласно варианту (часть 4 ЛР 2). Допустимо использование IN/OUT параметров. Допустимо создать авторские процедуры. (3 балла)

Создать триггеры для индивидуальной БД согласно варианту:

Вариант 2.1. 3 триггера - 3 балла (min). Допустимо использовать триггеры логирования из практического занятия по функциям и триггерам.

Вариант 2.2. 7 оригинальных триггеров - 7 баллов (max).

Схема


lab3
accidents
accident_code serial
violation_id integer
officer_personal_number integer
accident_address character varying(100)
accident_date_time timestamp without time zone
accident_description character varying(500)
lab3
officers
officer_personal_number serial
telephone_number character varying
full_name character varying(20)
badge_number bigint
lab3
driver_license
driver_license_number serial
driver_id integer
license_issue_date date
document_expiry_date date
issuing_organization character varying(50)
birth_date date
license_categories text[]
lab3
driver
driver_id serial
full_name character varying(50)
address character varying(100)
telephone_number character varying(11)
lab3
role
role_id serial
role_name character varying(50)
lab3
violations
violation_id serial
participant_id integer
traffic_code_id integer
fine_amount integer
payment_amount integer
payment_date date
violation_status character varying(15)
violation_date_time timestamp without time zone
violation_address character varying(100)
disqualification_start_date date
disqualification_end_date date
payment_percentage double precision
lab3
traffic_code
traffic_code_id serial
traffic_rule_name character varying(20)
lab3
participant
participant_id serial
role_id integer
driver_id integer
car_id integer
accident_code integer
fault character varying(20)
lab3
insurances
insurance_number serial
car_id integer
start_date date
end_date date
coverage_amount integer
insurance_type character varying(20)
insurance_company character varying(50)
insurance_status character varying(10)
lab3
cars
car_id serial
registration_date date
year_of_manufacture date
car_number character varying(10)
brand character varying(50)
lab3
car_owner
car_owner_id serial
car_id integer
driver_id integer
owned_from date
owned_until date

Выполнение
Процедуры и функции

Создадим по заданию №4 из 2 лабы процедуры и функции которые от нас требуют.

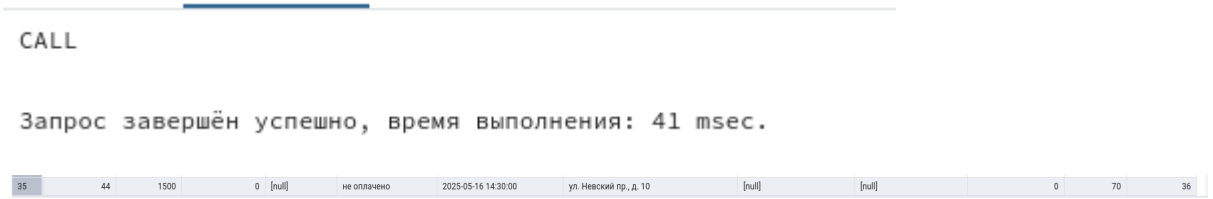
Добавить данные о новом штрафе водителя

```
CREATE OR REPLACE PROCEDURE lab3.add_violation_by_license(
    license_num INTEGER,
    rule_name TEXT,
    fine INTEGER,
    address TEXT,
    v_date TIMESTAMP
)
LANGUAGE plpgsql
AS $$
DECLARE
    pid INT;
    tcid INT;
BEGIN
    SELECT p.participant_id INTO pid
    FROM lab3.driver_license dl
    JOIN lab3.driver d ON dl.driver_id = d.driver_id
    JOIN lab3.participant p ON p.driver_id = d.driver_id
    WHERE dl.driver_license_number = license_num
    LIMIT 1;

    IF pid IS NULL THEN
        RAISE EXCEPTION 'Не найден участник с лицензией %', license_num;
    END IF;

    SELECT traffic_code_id INTO tcid
    FROM lab3.traffic_code
    WHERE traffic_rule_name = rule_name
    LIMIT 1;

    IF tcid IS NULL THEN
            RAISE EXCEPTION 'Не найдено нарушение с названием %',
rule_name;
    END IF;
```

```
    INSERT INTO lab3.violations (
        participant_id, traffic_code_id, fine_amount, payment_amount,
        payment_date, violation_status, violation_date_time,
        violation_address, disqualification_start_date,
        disqualification_end_date, payment_percentage
    ) VALUES (
        pid, tcid, fine, 0, NULL, 'не оплачено',
        v_date, address, NULL, NULL, 0.0
    );
END;
$$;
```

Запрос:

```
CALL lab3.add_violation_by_license(
    12345,
    'Превышение скорости',
    1500,
    'ул. Невский пр., д. 10',
    TIMESTAMP '2025-05-16 14:30:00'
);
```



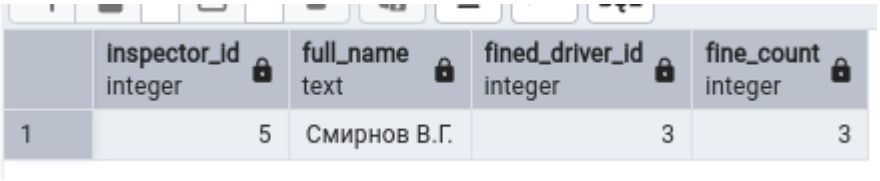
Вывести данные инспектора, оштрафовавшего одного и того же водителя более одного раза

```
CREATE OR REPLACE FUNCTION lab3.inspectors_multiple_fines()
RETURNS TABLE (
    inspector_id INT,
    full_name TEXT,
    fined_driver_id INT,
    fine_count INT
)
LANGUAGE sql
AS $$
SELECT
    a.officer_personal_number AS inspector_id,
```

```
    o.full_name,
    p.driver_id AS fined_driver_id,
    COUNT(*) AS fine_count
FROM lab3.accidents a
JOIN lab3.violations v ON a.violation_id = v.violation_id
JOIN lab3.participant p ON v.participant_id = p.participant_id
JOIN lab3.officers o ON a.officer_personal_number = o.officer_personal_number
GROUP BY a.officer_personal_number, o.full_name, p.driver_id
HAVING COUNT(*) > 1;
$$;
```

Запрос:

```
SELECT * FROM lab3.inspectors_multiple_fines();
```



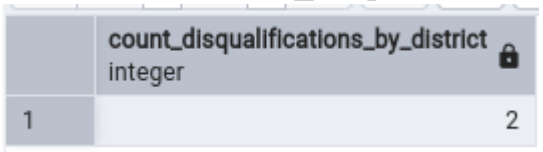
	inspector_id integer	full_name text	fined_driver_id integer	fine_count integer
1	5	Смирнов В.Г.	3	3

Вывести количество нарушений, повлекших лишение прав в заданном, как параметр районе

```
CREATE OR REPLACE FUNCTION lab3.count_disqualifications_by_district(district TEXT)
RETURNS INTEGER
LANGUAGE sql
AS $$
    SELECT COUNT(*)
    FROM lab3.violations
    WHERE disqualification_start_date IS NOT NULL
      AND disqualification_end_date IS NOT NULL
      AND violation_address ILIKE '%' || district || '%';
$$;
```

Запрос:

```
SELECT lab3.count_disqualifications_by_district('Москва');
```



	count_disqualifications_by_district integer
1	2

Триггеры

Нужно придумать 3 триггера к моей базе данных.

Автоматическое выставление статуса "оплачено", если payment_amount = fine_amount

Функция триггер

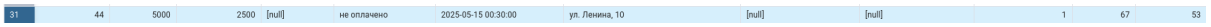
```
CREATE OR REPLACE FUNCTION lab3.auto_update_payment_status()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.payment_amount = NEW.fine_amount THEN
        NEW.violation_status := 'оплачено';
        NEW.payment_date := now();
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

Создание триггера

```
CREATE TRIGGER trg_auto_payment_status
BEFORE INSERT OR UPDATE ON lab3.violations
FOR EACH ROW
EXECUTE FUNCTION lab3.auto_update_payment_status();
```

Работает:
UPDATE lab3.violations SET payment_amount = 5000 WHERE violation_id = 67;

Было:



31	44	5000	2500	[null]	не оплачено	2025-05-15 00:30:00	ул. Ленина, 10	[null]	[null]	1	67	53

Стало:

43	44	5000	5000	2025-05-22	оплачено	2025-05-15 00:30:00	ул. Ленина, 10	[null]	[null]	1	67	53

Логирование добавленных нарушений (violations_log)

Создадим табличку для логирования

```
CREATE TABLE lab3.violations_log (
    log_id SERIAL PRIMARY KEY,
    violation_id INT,
    action_time TIMESTAMP DEFAULT now()
);
```

Функция триггер

```
CREATE OR REPLACE FUNCTION lab3.log_violation_insert()
RETURNS TRIGGER AS $$
BEGIN
    INSERT INTO lab3.violations_log (violation_id)
    VALUES (NEW.violation_id);
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

Создание триггера

```
CREATE TRIGGER trg_log_violation_insert
AFTER INSERT ON lab3.violations
FOR EACH ROW
EXECUTE FUNCTION lab3.log_violation_insert();
```

Работает

	log_id [PK] integer	violation_id integer	action_time timestamp without time zone
1	1	74	2025-05-22 13:16:10.57332
2	2	75	2025-05-22 13:16:10.57332
3	3	76	2025-05-22 13:16:10.57332

Автоматическая установка даты оплаты

Функция триггер

```
CREATE OR REPLACE FUNCTION lab3.auto_set_payment_date()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.payment_amount > 0 THEN
        NEW.payment_date := CURRENT_TIMESTAMP;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

Создаем триггер

```
CREATE TRIGGER trg_auto_set_payment_date
BEFORE INSERT OR UPDATE ON lab3.violations
FOR EACH ROW
EXECUTE FUNCTION lab3.auto_set_payment_date();
```

Работает:

До:

После:

Вывод

В этой лабораторной работе стало ясно, что такое триггеры, процедуры и функции. Научились их создавать и применять.