Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Факультет «Информационных технологий и анализа больших данных»

**ОТЧЕТ**

**по практической работе № 8**

**по теме «Триггеры»**

**по дисциплине Б.1.1.3.4 Системы управления базами данных**

**Вариант 17**

Выполнил: студент группы ИД23-1

Маслов Александр Николаевич

Проверил преподаватель: Цветков И.Е.

**Москва – 2025**

**Задание №1**

**Изображение выглядит как текст, Шрифт, снимок экрана, черно-белый

Контент, сгенерированный ИИ, может содержать ошибки.**

**Решение:**

**1.**

CREATE TABLE aircrafts\_tmp AS TABLE bookings.aircrafts\_data WITH NO DATA;

CREATE TABLE aircraft\_log (

aircraft\_code CHAR(3),

model JSONB,

range INTEGER,

log\_time TIMESTAMPTZ DEFAULT CURRENT\_TIMESTAMP

);

CREATE OR REPLACE FUNCTION trg\_fn\_log\_aircraft()

RETURNS TRIGGER AS $$

BEGIN

INSERT INTO aircraft\_log (aircraft\_code, model, range)

VALUES (NEW.aircraft\_code, NEW.model, NEW.range);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_log\_aircraft

AFTER INSERT ON aircrafts\_tmp

FOR EACH ROW

EXECUTE FUNCTION trg\_fn\_log\_aircraft();

**Проверка:**

INSERT INTO aircrafts\_tmp (aircraft\_code, model, range)

VALUES ('ABC', '{"en": "Test Plane"}', 5000);

SELECT \* FROM aircraft\_log;

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

**2.**

CREATE TABLE aircrafts\_copy AS TABLE bookings.aircrafts\_data;

CREATE TABLE flights\_copy AS TABLE bookings.flights;

CREATE TABLE seats\_copy AS TABLE bookings.seats;

CREATE OR REPLACE FUNCTION trg\_fn\_delete\_related()

RETURNS TRIGGER AS $$

BEGIN

DELETE FROM flights\_copy WHERE aircraft\_code = OLD.aircraft\_code;

DELETE FROM seats\_copy WHERE aircraft\_code = OLD.aircraft\_code;

RETURN OLD;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_on\_delete\_aircraft

BEFORE DELETE ON aircrafts\_copy

FOR EACH ROW

EXECUTE FUNCTION trg\_fn\_delete\_related();

**Проверка:**

INSERT INTO aircrafts\_copy (aircraft\_code, model, range)

VALUES ('XYZ', '{"en": "DeleteMe"}', 6000);

INSERT INTO flights\_copy (flight\_id, flight\_no, aircraft\_code, scheduled\_departure,

scheduled\_arrival, departure\_airport, arrival\_airport, status)

VALUES (9999, 'ZZ9999', 'XYZ', NOW(), NOW() + interval '2 hours', 'LED', 'DME', 'On Time');

INSERT INTO seats\_copy (aircraft\_code, seat\_no, fare\_conditions)

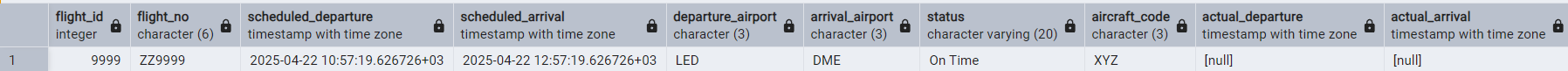
VALUES ('XYZ', '1A', 'Business');

SELECT \* FROM seats\_copy WHERE aircraft\_code = 'XYZ';

Изображение выглядит как текст, снимок экрана, Шрифт, линия

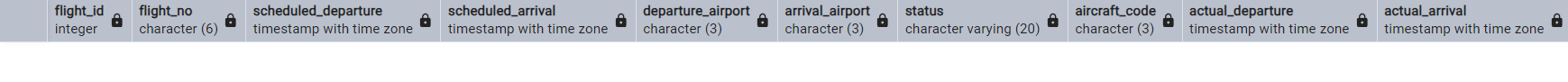
Контент, сгенерированный ИИ, может содержать ошибки.

SELECT \* FROM flights\_copy WHERE aircraft\_code = 'XYZ';



DELETE FROM aircrafts\_copy WHERE aircraft\_code = 'XYZ';

SELECT \* FROM flights\_copy WHERE aircraft\_code = 'XYZ';



SELECT \* FROM seats\_copy WHERE aircraft\_code = 'XYZ';

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Контент, сгенерированный ИИ, может содержать ошибки.

**3.**

CREATE OR REPLACE FUNCTION trg\_fn\_update\_total\_amount()

RETURNS TRIGGER AS $$

DECLARE

booking\_ref CHAR(6);

new\_total NUMERIC;

BEGIN

SELECT t.book\_ref INTO booking\_ref

FROM bookings.tickets t

WHERE t.ticket\_no = COALESCE(NEW.ticket\_no, OLD.ticket\_no);

SELECT SUM(tf.amount)

INTO new\_total

FROM bookings.ticket\_flights tf

JOIN bookings.tickets t ON tf.ticket\_no = t.ticket\_no

WHERE t.book\_ref = booking\_ref;

UPDATE bookings.bookings

SET total\_amount = COALESCE(new\_total, 0)

WHERE book\_ref = booking\_ref;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_total\_amount\_insert

AFTER INSERT ON bookings.ticket\_flights

FOR EACH ROW

EXECUTE FUNCTION trg\_fn\_update\_total\_amount();

CREATE TRIGGER trg\_total\_amount\_update

AFTER UPDATE ON bookings.ticket\_flights

FOR EACH ROW

EXECUTE FUNCTION trg\_fn\_update\_total\_amount();

CREATE TRIGGER trg\_total\_amount\_delete

AFTER DELETE ON bookings.ticket\_flights

FOR EACH ROW

EXECUTE FUNCTION trg\_fn\_update\_total\_amount();

**Проверка:**

INSERT INTO bookings.bookings (book\_ref, book\_date, total\_amount)

VALUES ('BR001', NOW(), 0);

INSERT INTO bookings.tickets (ticket\_no, book\_ref, passenger\_id, passenger\_name, contact\_data)

VALUES ('TICK001', 'BR001', 'P001', 'John Doe', '{"phone": "+1234567890"}');

INSERT INTO bookings.ticket\_flights (ticket\_no, flight\_id, fare\_conditions, amount)

VALUES ('TICK001', 1, 'Economy', 200.00);

SELECT book\_ref, total\_amount FROM bookings.bookings WHERE book\_ref = 'BR001';

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.

UPDATE bookings.ticket\_flights

SET amount = 300

WHERE ticket\_no = 'TICK001';

SELECT book\_ref, total\_amount FROM bookings.bookings WHERE book\_ref = 'BR001';

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Контент, сгенерированный ИИ, может содержать ошибки.

DELETE FROM bookings.ticket\_flights WHERE ticket\_no = 'TICK001';

SELECT book\_ref, total\_amount FROM bookings.bookings WHERE book\_ref = 'BR001';

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Контент, сгенерированный ИИ, может содержать ошибки.

**Задание №2**

**Изображение выглядит как текст, снимок экрана, документ, чек

Контент, сгенерированный ИИ, может содержать ошибки.**

**Решение:**

**1.1**

CREATE OR REPLACE FUNCTION trg\_fn\_prevent\_delete\_store1()

RETURNS TRIGGER AS $$

BEGIN

IF OLD.Номер = 1 THEN

RAISE EXCEPTION 'Удаление магазина №1 запрещено!';

END IF;

RETURN OLD;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_no\_delete\_store1

BEFORE DELETE ON "Магазины"

FOR EACH ROW

EXECUTE FUNCTION trg\_fn\_prevent\_delete\_store1();

**Проверка:**

DELETE FROM "Магазины" WHERE "Номер" = 1;

****

**1.2**

CREATE OR REPLACE FUNCTION forbid\_price\_decrease()

RETURNS trigger AS $$

BEGIN

IF NEW."Цена" < OLD."Цена" THEN

RAISE EXCEPTION 'Снижение цены запрещено! Была: %, стало: %', OLD."Цена", NEW."Цена";

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_no\_price\_decrease

BEFORE UPDATE ON "Техника"

FOR EACH ROW

EXECUTE FUNCTION forbid\_price\_decrease();

**Проверка:**

UPDATE "Техника"

SET "Цена" = 5000

WHERE "Код" = 2;



**2.1**

CREATE OR REPLACE VIEW "Наличие\_техники\_в" AS

SELECT

nt."Техника" AS "Код\_техники",

t."Название" AS "Название\_техники",

nt."Магазин" AS "Номер\_магазина",

m."ФИО директора" AS "ФИО\_директора",

nt."Количество"

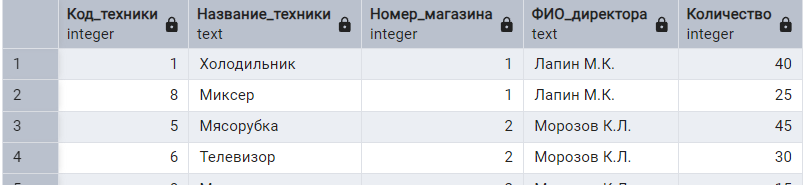
FROM "НаличиеТехники" nt

JOIN "Магазины" m ON nt."Магазин" = m."Номер"

JOIN "Техника" t ON nt."Техника" = t."Код";

**Проверка:**

SELECT \* FROM "Наличие\_техники\_в"



**2.2**

CREATE OR REPLACE FUNCTION insert\_into\_наличие\_техники\_в()

RETURNS trigger AS $$

DECLARE

tech\_exists BOOLEAN;

shop\_exists BOOLEAN;

duplicate\_exists BOOLEAN;

BEGIN

SELECT EXISTS (

SELECT 1 FROM "Техника" WHERE "Код" = NEW."Код\_техники"

) INTO tech\_exists;

SELECT EXISTS (

SELECT 1 FROM "Магазины" WHERE "Номер" = NEW."Номер\_магазина"

) INTO shop\_exists;

IF NOT tech\_exists THEN

RAISE EXCEPTION 'Ошибка: техника с кодом % не существует', NEW."Код\_техники";

ELSIF NOT shop\_exists THEN

RAISE EXCEPTION 'Ошибка: магазин с номером % не существует', NEW."Номер\_магазина";

END IF;

SELECT EXISTS (

SELECT 1 FROM "НаличиеТехники"

WHERE "Техника" = NEW."Код\_техники"

AND "Магазин" = NEW."Номер\_магазина"

) INTO duplicate\_exists;

IF duplicate\_exists THEN

RAISE EXCEPTION 'Ошибка: техника с кодом % уже учтена в магазине %',

NEW."Код\_техники", NEW."Номер\_магазина";

END IF;

INSERT INTO "НаличиеТехники"("Магазин", "Техника", "Количество")

VALUES (NEW."Номер\_магазина", NEW."Код\_техники", NEW."Количество");

RETURN NULL;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg\_insert\_наличие\_техники\_в

INSTEAD OF INSERT ON "Наличие\_техники\_в"

FOR EACH ROW

EXECUTE FUNCTION insert\_into\_наличие\_техники\_в();

**Проверка:**

INSERT INTO "Наличие\_техники\_в"

("Код\_техники", "Номер\_магазина", "Количество")

VALUES (1, 2, 15);



INSERT INTO "Наличие\_техники\_в"

("Код\_техники", "Номер\_магазина", "Количество")

VALUES (999, 3, 15);



INSERT INTO "Наличие\_техники\_в"

("Код\_техники", "Номер\_магазина", "Количество")

VALUES (1, 999, 15);

