**Министерство науки и высшего образования Российской Федерации**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Факультет безопасности информационных технологий**

**Дисциплина:**

«Информационная безопасность баз данных»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

«Защита базы данных»

**Выполнили:**

Бардышев Артём Антонович,

студент группы N3346

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

**Проверил:**

Салихов Максим Русланович,

преподаватель, ФБИТ

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(отметка о выполнении)

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

Содержание

[Введение 3](#_Toc210326334)

[1 Ход выполнения 4](#_Toc210326335)

[1.1 Мониторинг (логирование операций) 4](#_Toc210326336)

[1.2 Шифрование секретных данных 5](#_Toc210326337)

[1.3 Разграничение доступа 7](#_Toc210326338)

[Заключение 9](#_Toc210326339)

[Список использованных источников 10](#_Toc210326340)

Введение

Цель работы – получение навыков организации систем защиты баз данных: логирование операций, разграничение доступа и защита данных методами шифрования.

**Введение**

Современные системы управления базами данных предоставляют встроенные механизмы для обеспечения информационной безопасности. Среди них можно выделить:

* средства мониторинга (логирование действий пользователей и фиксация изменений);
* разграничение доступа с использованием ролей и привилегий;
* криптографическую защиту чувствительных данных.

В данной работе исследуются базовые механизмы PostgreSQL, которые позволяют реализовать перечисленные функции в учебной базе данных, разработанной ранее (Apple Store).

# Ход выполнения

## Мониторинг (логирование операций)

Создана таблица логов и функция-триггер, которая фиксирует изменения во всех таблицах БД.

-- Таблица логов

CREATE TABLE main\_log (

log\_item\_id SERIAL PRIMARY KEY,

operation\_type TEXT,

operation\_date TIMESTAMP,

user\_operator TEXT,

changed\_data JSONB

);

-- Триггерная функция

CREATE OR REPLACE FUNCTION logging() RETURNS TRIGGER AS $$

BEGIN

IF (TG\_OP = 'DELETE') THEN

INSERT INTO main\_log (operation\_type, operation\_date, user\_operator, changed\_data)

VALUES ('DELETE', now(), current\_user, row\_to\_json(OLD));

ELSIF (TG\_OP = 'UPDATE') THEN

INSERT INTO main\_log (operation\_type, operation\_date, user\_operator, changed\_data)

VALUES ('UPDATE', now(), current\_user, row\_to\_json(NEW));

ELSIF (TG\_OP = 'INSERT') THEN

INSERT INTO main\_log (operation\_type, operation\_date, user\_operator, changed\_data)

VALUES ('INSERT', now(), current\_user, row\_to\_json(NEW));

END IF;

RETURN NULL;

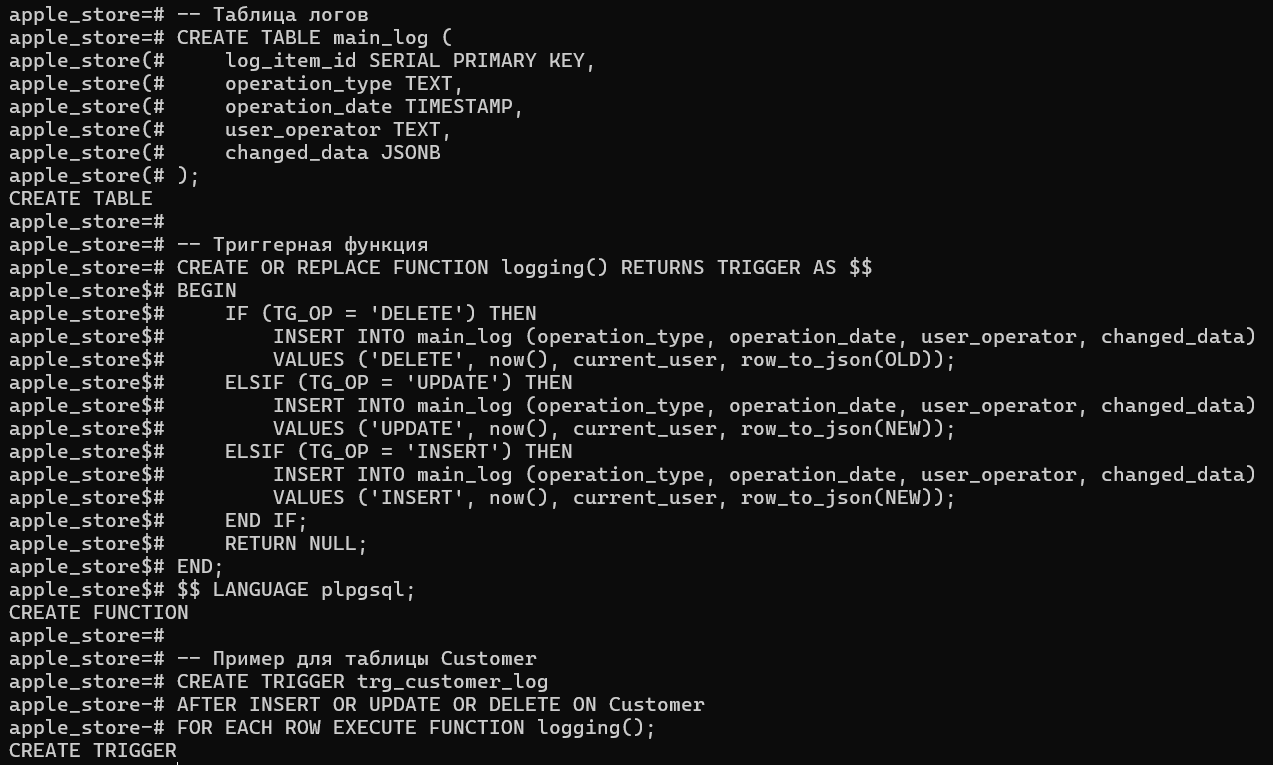
END;

$$ LANGUAGE plpgsql;

-- Пример для таблицы Customer

CREATE TRIGGER trg\_customer\_log

AFTER INSERT OR UPDATE OR DELETE ON Customer

FOR EACH ROW EXECUTE FUNCTION logging();

## Шифрование секретных данных

Подключаем расширение:

CREATE EXTENSION IF NOT EXISTS pgcrypto;

Создаём таблицу:

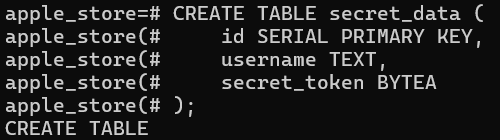
CREATE TABLE secret\_data (

id SERIAL PRIMARY KEY,

username TEXT,

secret\_token BYTEA

);



Генерация ключа (SHA-256 от пароля !stroNgpsw31234):

-- пример вставки зашифрованных данных

INSERT INTO secret\_data (username, secret\_token) VALUES

(

'admin\_user',

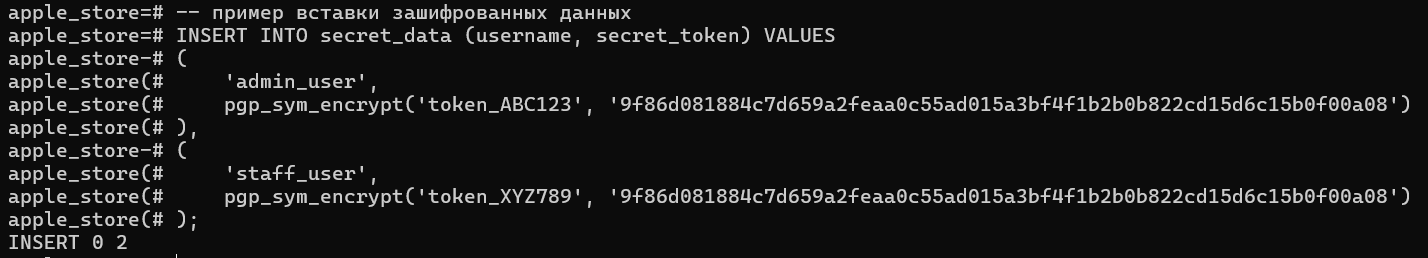
pgp\_sym\_encrypt('token\_ABC123', '9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08')

),

(

'staff\_user',

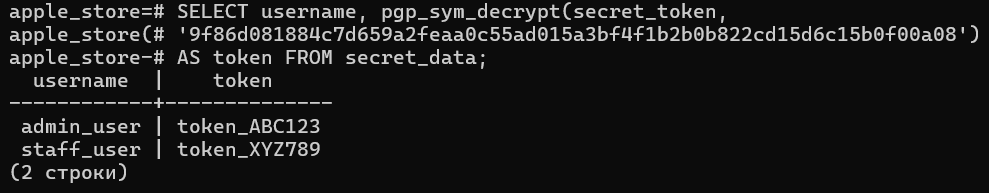
pgp\_sym\_encrypt('token\_XYZ789', '9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08')

);

Расшифровка доступна только при знании пароля:

SELECT username, pgp\_sym\_decrypt(secret\_token,

'9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08')

AS token FROM secret\_data;

## Разграничение доступа

Создаём три групповые роли: **student\_group\_role, staff\_group\_role, admin\_group\_role**.

-- Студенты: могут только читать часть представлений

CREATE ROLE student\_group\_role NOLOGIN;

GRANT SELECT ON compact\_shedule, consultations TO student\_group\_role;

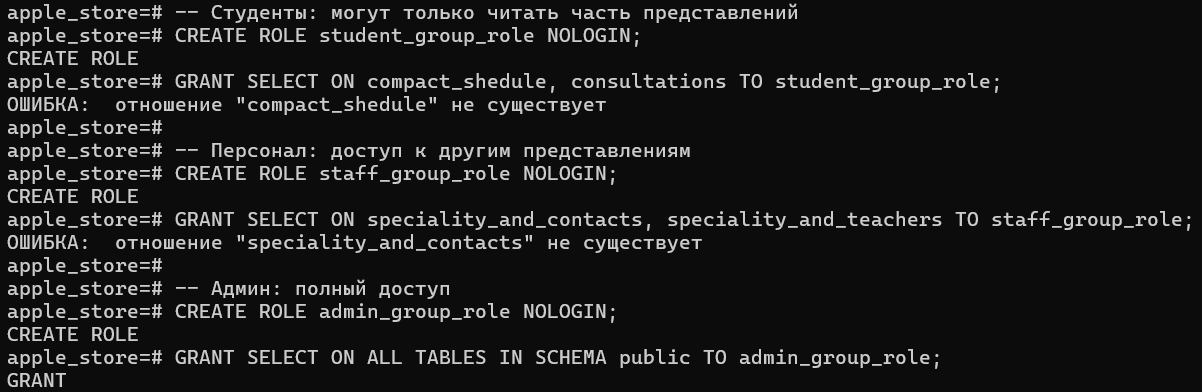
-- Персонал: доступ к другим представлениям

CREATE ROLE staff\_group\_role NOLOGIN;

GRANT SELECT ON speciality\_and\_contacts, speciality\_and\_teachers TO staff\_group\_role;

-- Админ: полный доступ

CREATE ROLE admin\_group\_role NOLOGIN;

GRANT SELECT ON ALL TABLES IN SCHEMA public TO admin\_group\_role;

Создаём индивидуальных пользователей:

CREATE ROLE ivan\_student LOGIN PASSWORD 'studpass';

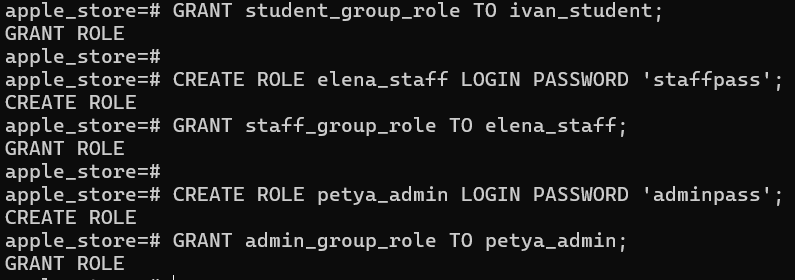
GRANT student\_group\_role TO ivan\_student;

CREATE ROLE elena\_staff LOGIN PASSWORD 'staffpass';

GRANT staff\_group\_role TO elena\_staff;

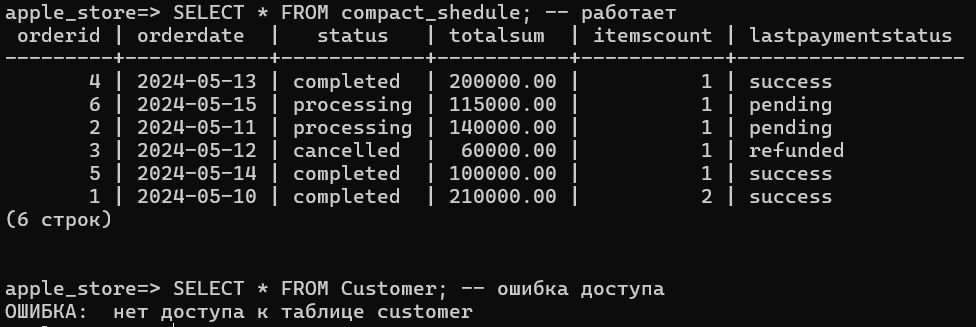
CREATE ROLE petya\_admin LOGIN PASSWORD 'adminpass';

GRANT admin\_group\_role TO petya\_admin;

Проверка:

SET ROLE ivan\_student;

SELECT \* FROM compact\_shedule; -- работает

SELECT \* FROM Customer; -- ошибка доступа

При создании некоторых ролей и зависимостей возникли некоторые трудности, внимательно разобравшись и войдя в профиль суперюзера root, было пересоздано все что не удалось и скриншот выше – итоговый правильный результат.

Заключение

В ходе работы разработана система защиты БД средствами PostgreSQL.

* Реализовано логирование операций с данными с помощью триггеров.
* Создана таблица секретных данных с симметричным шифрованием на основе pgcrypto.
* Настроено разграничение доступа на основе ролей пользователей, реализован принцип минимальных привилегий.

Таким образом, выполнены базовые задачи по обеспечению безопасности базы данных: контроль действий, защита чувствительных данных и управление правами доступа.

Список использованных источников

1. Новиков Б. А., Горшкова Е. А., Графеева Н. Г. **Основы технологий баз данных.** – 2-е изд. – М.: ДМК Пресс, 2020. – 582 с.
2. Хомоненко А. Д. (ред.). **Базы данных.** – 6-е изд., доп. – СПб.: КОРОНА-Век, 2009. – 736 с.
3. Документация PostgreSQL: <https://www.postgresql.org/docs/>
4. Модуль pgcrypto: <https://www.postgresql.org/docs/current/pgcrypto.html>