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 Дроздов Матвей Вячеславович
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
2024/2025

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

Практическое задание:

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).

Дополнительные баллы - 3:

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).

Выполнение:

1) Создать хранимые процедуры:

- Для проверки наличия экземпляров заданной книги в библиотеке (процедура должна возвращать количество экземпляров книги).

```
library=# CREATE OR REPLACE PROCEDURE check_copies_by_title(book_name TEXT)
library-# LANGUAGE plpgsql
library-# AS $$
library$# DECLARE
library$#     copy_count INTEGER;
library$# BEGIN
library$#     SELECT COUNT(*)
library$#     INTO copy_count
library$#     FROM copies_of_books cb
library$#     JOIN books b ON cb.book_id = b.book_id
library$#     WHERE b.name_publication = book_name
library$#       AND cb.inventory_number NOT IN (
library$#           SELECT inventory_number FROM disposal_documents
library$#       );
library$#
library$#     RAISE NOTICE 'Количество экземпляров книги "%": %', book_name, cop
y_count;
library$# END;
library$# $$;
CREATE PROCEDURE
```

- Для ввода в базу данных новой книги.

```
library=# CREATE OR REPLACE PROCEDURE add_new_book(
library(#     _year_of_issue DATE,
library(#     _compiler_publication TEXT,
library(#     _place_publication TEXT,
library(#     _language_translation TEXT,
library(#     _name_publisher TEXT,
library(#     _volume_number INTEGER,
library(#     _translator TEXT,
library(#     _type_publication TEXT,
library(#     _region_knowledge TEXT,
library(#     _name_publication TEXT,
library(#     _author_publication TEXT
library(# )
library-# LANGUAGE plpgsql
library-# AS $$
library$# BEGIN
library$#     INSERT INTO books (
library$#         year_of_issue, compiler_publication, place_publication,
library$#         language_translation, name_publisher, volume_number,
library$#         translator, type_publication, region_knowledge,
library$#         name_publication, author_publication
library$#     )
library$#     VALUES (
library$#         _year_of_issue, _compiler_publication, _place_publication,
library$#         _language_translation, _name_publisher, _volume_number,
library$#         _translator, _type_publication, _region_knowledge,
library$#         _name_publication, _author_publication
library$#     );
library$# END;
library$# $$;
CREATE PROCEDURE
library=# CALL add_new_book(
library(#     '2025-05-28',
library(#     'Петров В.В.',
library(#     'Москва',
library(#     'русский',
library(#     'Просвещение',
library(#     1,
library(#     NULL,
library(#     'монография',
library(#     'философия',
library(#     'Смысл жизни',
library(#     'Сидоров А.А.'
library(# );
CALL
```

- Для ввода нового читателя (необходимо проверить наличие читателя в картотеке, чтобы не назначить ему номер вторично).

```
library=# CREATE OR REPLACE PROCEDURE add_new_reader(
library(#     _library_card INT,
library(#     _full_name TEXT,
library(#     _passport_data TEXT,
library(#     _address TEXT,
library(#     _email TEXT,
library(#     _telephone TEXT,
library(#     _education TEXT
library(# )
library-# LANGUAGE plpgsql
library-# AS $$
library$# DECLARE
library$#     exists_count INT;
library$# BEGIN
library$#     SELECT COUNT(*) INTO exists_count
library$#     FROM reader
library$#     WHERE library_card = _library_card
library$#        OR passport_data = _passport_data;
library$#
library$#     IF exists_count > 0 THEN
library$#         RAISE NOTICE 'Читатель уже существует';
library$#     ELSE
library$#         INSERT INTO reader (
library$#             library_card, full_name, passport_data,
library$#             address, email, telephone_number, education
library$#         )
library$#         VALUES (
library$#             _library_card, _full_name, _passport_data,
library$#             _address, _email, _telephone, _education
library$#         );
library$#         RAISE NOTICE 'Читатель успешно добавлен.';
library$#     END IF;
library$# END;
library$# $$;
CREATE PROCEDURE
library=# CALL add_new_reader(
library(#     4010,
library(#     'Анна Сергеевна Морозова',
library(#     '5500 112233',
library(#     'г. Пермь, ул. Свободы, д. 3',
library(#     'morozova.a@mail.ru',
library(#     '+7-905-111-22-33',
library(#     'Высшее'
library(# );
NOTICE:  Читатель успешно добавлен.
CALL
```

-

```
library=# CALL add_new_reader(
library(#     4010,
library(#     'Анна Сергеевна Морозова',
library(#     '5500 112233',
library(#     'г. Пермь, ул. Свободы, д. 3',
library(#     'morozova.a@mail.ru',
library(#     '+7-905-111-22-33',
library(#     'Высшее'
library(# );
NOTICE:  Читатель уже существует
CALL
```

2) Создание триггеров

- Автоматическая установка fine = 0 при своевременной сдаче (если return_date - date_of_issue <= 10)

```
library=# CREATE OR REPLACE FUNCTION auto_clear_fine()
library-# RETURNS TRIGGER AS $$
library$# BEGIN
library$#     IF NEW.return_date IS NOT NULL AND NEW.return_date <= NEW.date_of_issue + INTERVAL '10 days' THEN
library$#         UPDATE book_distribution
library$#         SET fine = 0
library$#         WHERE id_distrubution = NEW.id_distrubution;
library$#     END IF;
library$#     RETURN NEW;
library$# END;
library$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
library=#
library=# CREATE TRIGGER trg_auto_clear_fine
library-# AFTER UPDATE ON book_distribution
library-# FOR EACH ROW
library-# EXECUTE FUNCTION auto_clear_fine();
CREATE TRIGGER
library=#
```

- Контроль допустимого значения штрафа (не больше 1000)

```
library=# CREATE OR REPLACE FUNCTION check_fine_limit()
library-# RETURNS TRIGGER AS $$
library$# BEGIN
library$#     IF NEW.fine > 1000 THEN
library$#         RAISE EXCEPTION 'Штраф не может превышать 1000 руб.';
library$#     END IF;
library$#     RETURN NEW;
library$# END;
library$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
library=#
library=# CREATE TRIGGER trg_check_fine_limit
library-# BEFORE INSERT OR UPDATE ON book_distribution
library-# FOR EACH ROW
library-# EXECUTE FUNCTION check_fine_limit();
CREATE TRIGGER
```

- Запрет выдачи книг читателям с неполными паспортными данными

```
library=# CREATE OR REPLACE FUNCTION prevent_issue_if_no_passport()
library-# RETURNS TRIGGER AS $$
library$# DECLARE
library$#     passport TEXT;
library$# BEGIN
library$#     SELECT passport_data INTO passport
library$#     FROM reader
library$#     WHERE library_card = NEW.library_card;
library$#
library$#     IF passport IS NULL OR TRIM(passport) = '' THEN
library$#         RAISE EXCEPTION 'Читатель не может получить книгу без паспортных данных.';
library$#     END IF;
library$#
library$#     RETURN NEW;
library$# END;
library$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
library=#
library=# CREATE TRIGGER trg_prevent_issue_if_no_passport
library-# BEFORE INSERT ON book_distribution
library-# FOR EACH ROW
library-# EXECUTE FUNCTION prevent_issue_if_no_passport();
CREATE TRIGGER
```

- Предотвращение вставки книги без названия

```
library=# CREATE OR REPLACE FUNCTION prevent_book_without_name()
library-# RETURNS TRIGGER AS $$
library$# BEGIN
library$#     IF NEW.name_publication IS NULL OR TRIM(NEW.name_publication) = '' THEN
library$#         RAISE EXCEPTION 'Название книги не может быть пустым.';
library$#     END IF;
library$#     RETURN NEW;
library$# END;
library$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
library=#
library=# CREATE TRIGGER trg_prevent_book_without_name
library-# BEFORE INSERT ON books
library-# FOR EACH ROW
library-# EXECUTE FUNCTION prevent_book_without_name();
CREATE TRIGGER
```

- Запрет удаления книги, если на неё есть экземпляры в copies_of_books

```
library=# CREATE OR REPLACE FUNCTION prevent_book_delete_if_copies_exist()
library-# RETURNS TRIGGER AS $$
library$# BEGIN
library$#     IF EXISTS (
library$#         SELECT 1 FROM copies_of_books
library$#         WHERE book_id = OLD.book_id
library$#     ) THEN
library$#         RAISE EXCEPTION 'Нельзя удалить книгу: на неё существуют экземпляры.';
library$#     END IF;
library$#     RETURN OLD;
library$# END;
library$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
library=#
library=# CREATE TRIGGER trg_prevent_book_delete
library-# BEFORE DELETE ON books
library-# FOR EACH ROW
library-# EXECUTE FUNCTION prevent_book_delete_if_copies_exist();
CREATE TRIGGER
```

- Читатель не может взять более 3 книг одновременно

```
library=# CREATE OR REPLACE FUNCTION limit_books_on_hand()
library-# RETURNS TRIGGER AS $$
library$# DECLARE
library$#     active_books_count INT;
library$# BEGIN
library$#     SELECT COUNT(*) INTO active_books_count
library$#     FROM book_distribution
library$#     WHERE library_card = NEW.library_card
library$#       AND return_date IS NULL;
library$#
library$#     IF active_books_count > 3 THEN
library$#         RAISE EXCEPTION 'Читатель уже держит 3 книги. Больше нельзя.';
library$#     END IF;
library$#
library$#     RETURN NEW;
library$# END;
library$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
library=#
library=# CREATE TRIGGER trg_limit_books_on_hand
library-# BEFORE INSERT ON book_distribution
library-# FOR EACH ROW
library-# EXECUTE FUNCTION limit_books_on_hand();
CREATE TRIGGER
```

- Автоматическая установка значения type_publication = 'прочее', если оно не указано

```
library=# CREATE OR REPLACE FUNCTION default_type_publication()
library-# RETURNS TRIGGER AS $$
library$# BEGIN
library$#     IF NEW.type_publication IS NULL OR TRIM(NEW.type_publication) = '' THEN
library$#         NEW.type_publication := 'прочее';
library$#     END IF;
library$#     RETURN NEW;
library$# END;
library$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
library=#
library=# CREATE TRIGGER trg_default_type_publication
library-# BEFORE INSERT ON books
library-# FOR EACH ROW
library-# EXECUTE FUNCTION default_type_publication();
CREATE TRIGGER
```

Вывод: овладел практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.