МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

**ОТЧЕТ**  
по лабораторному практикуму по дисциплине

«Распределённые БД»

Лабораторная работа № 5

Обучающийся группы 6131-020402D В.Д. Гижевская

Обучающийся группы 6131-020402D Д.А. Кремнёв

Обучающийся группы 6131-020402D М.Г. Алкеев

Проверил М.В. Додонов

Самара 2023

ЗАдание

Тема работы: создание и использование хранимых процедур.

Цель работы: знакомство с возможностями СУБД по созданию хранимых процедур.

Подготовка к работе:

1. Изучите теоретические основы, команды создания хранимых процедур, функций и пакетов, рассмотренные в лекционном курсе, методических указаниях или других источниках.

2. Ответьте на вопросы к лабораторной работе.

3. Подготовьте отчет к оформлению хода выполнения работы.

Порядок выполнения работы:

1. Создайте хранимые процедуры для добавления и изменения записей в таблицах, проверяющих существование редактируемой записи и определяющих значение суррогатного первичного ключа по умолчанию для новых записей.

2. Внесите такие изменения в хранимые процедуры добавления и изменения записей в таблицах, которые не позволят добавить или изменить записи с дублирующими названиями.

3. Создайте хранимые процедуры удаления записей из таблиц, удаляющих связанные записи из дочерних таблиц.

4. Внесите такие изменения в хранимые процедуры удаления, добавления и изменения записей в таблицах, которые позволят выполнить операции только для заданных пользователей.

5. Проверьте работоспособность всех созданных хранимых процедур.

6. Оформите все созданные хранимые процедуры для каждой таблицы в виде пакетов.

7. Проверьте работоспособность пакетов.

8. Оформите отчет о выполнении лабораторной работы.

вариант

Предметная область: Книжный сайт.

Основные предметно-значимые сущности: Серии книг, Авторы, Книги, Главы.

Основные предметно-значимые атрибуты сущностей:

* серии книг – название серии книг;
* авторы – ФИО автора;
* книги – название книги, описание книги, ключевые слова;
* главы – название главы, время чтения, количество страниц.

Основные требования к функциям системы:

* выбрать все книги по определённому автору;
* выбрать все главы определённой книги;
* вывести рейтинг книг по времени чтения;
* вывести все книги определённой серии;
* вывести все книги по определённому ключевому слову;
* подсчитать количество страниц нужной книги;
* подсчитать количество страниц нужной серии;
* подсчитать время чтения нужной книги;
* подсчитать время чтения выбранной серии.

1. Создайте хранимые процедуры для добавления и изменения записей в таблицах, проверяющих существование редактируемой записи и определяющих значение суррогатного первичного ключа по умолчанию для новых записей

**Создание хранимой процедуры для добавления записи для таблицы authors:**

CREATE OR REPLACE PROCEDURE add\_author(name varchar)

LANGUAGE 'plpgsql'

AS

$BODY$

BEGIN

IF EXISTS(SELECT FROM authors WHERE author\_name = name) THEN

RAISE EXCEPTION 'Record with name % already exists', name;

ELSE

INSERT INTO authors (id\_author, author\_name) VALUES (

(SELECT COALESCE(MAX(id\_author), 0) + 1 FROM authors),

name

);

END IF;

END;

$BODY$;

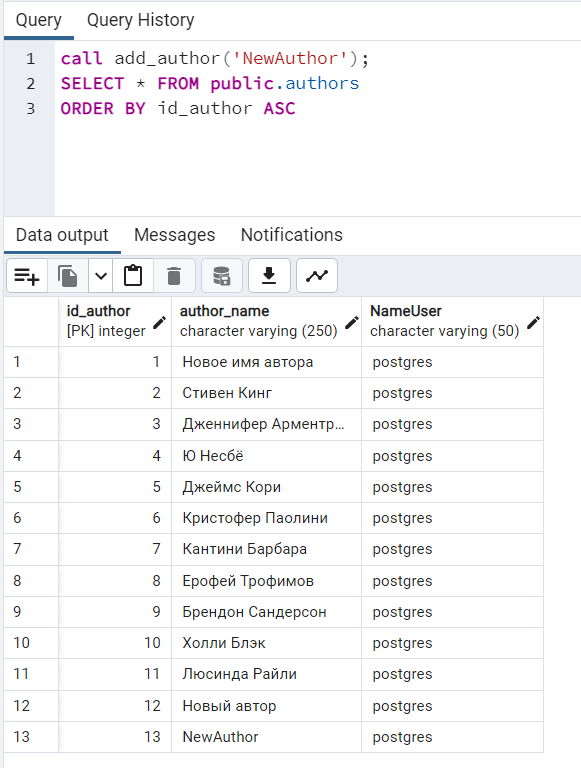


Рисунок 1 – Проверка работоспособности процедуры по добавлению записи при вводе уникального значения

**Создание хранимой процедуры для изменения записи для таблицы authors:**

CREATE OR REPLACE PROCEDURE update\_author(id integer, name varchar)

LANGUAGE 'plpgsql'

AS

$BODY$

BEGIN

IF EXISTS(SELECT FROM authors WHERE author\_name = name) THEN

RAISE EXCEPTION 'Record with name % already exists', name;

ELSE

UPDATE authors SET author\_name = name WHERE id\_author = id;

END IF;

END;

$BODY$;

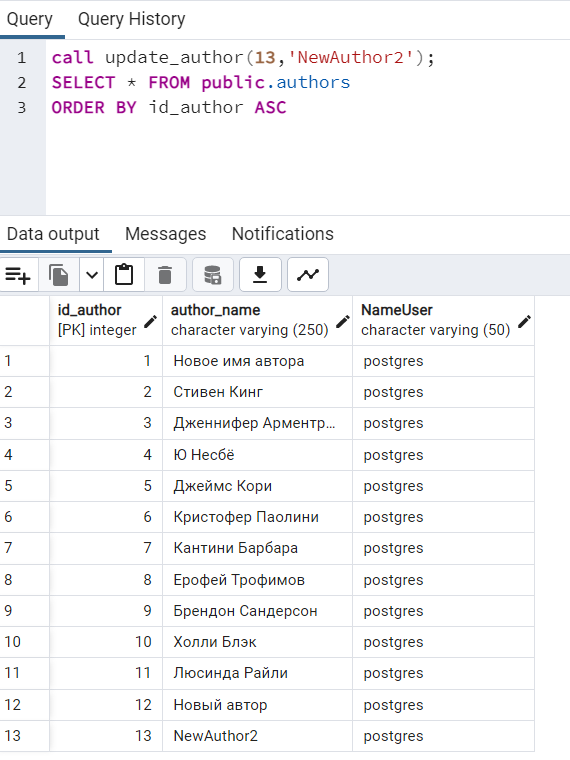


Рисунок 2 – Проверка работоспособности процедуры по обновлению записи при вводе уникального значения

1. Создайте хранимые процедуры удаления записей из таблиц, удаляющих связанные записи из дочерних таблиц

**Создание хранимой процедуры удаления записей из таблицы authors, удаляющей связанные записи дочерних таблиц:**

CREATE OR REPLACE PROCEDURE delete\_author(id\_to\_delete integer)

LANGUAGE 'plpgsql'

AS

$BODY$

BEGIN

DELETE FROM capters

WHERE id\_book

IN (SELECT id\_book FROM book WHERE id\_book\_series IN

(SELECT id\_book\_series FROM book\_series WHERE id\_author = id\_to\_delete));

DELETE FROM books

WHERE id\_book\_series IN

(SELECT id\_book\_series FROM book\_series WHERE id\_author = id\_to\_delete);

DELETE FROM book\_series

WHERE id\_author = id\_to\_delete;

DELETE FROM author

WHERE id\_author = id\_to\_delete;

END;

$BODY$;

**Проверка работы процедуры:**

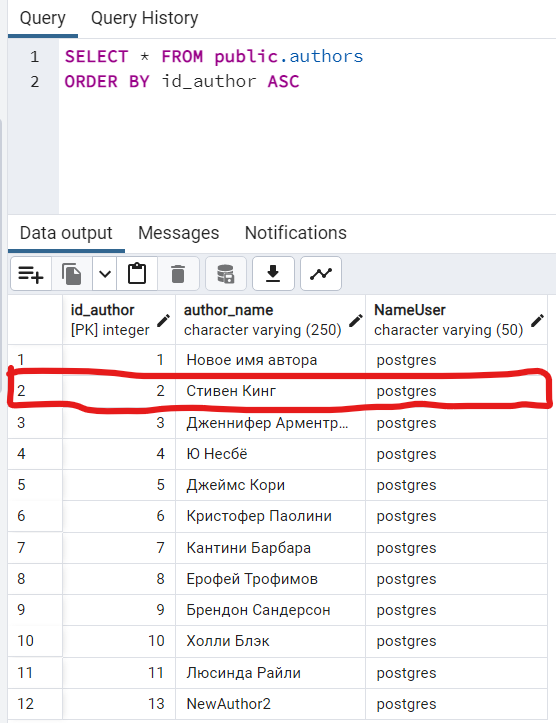
****

Рисунок 3 – Состояние таблицы authors до применения процедуры

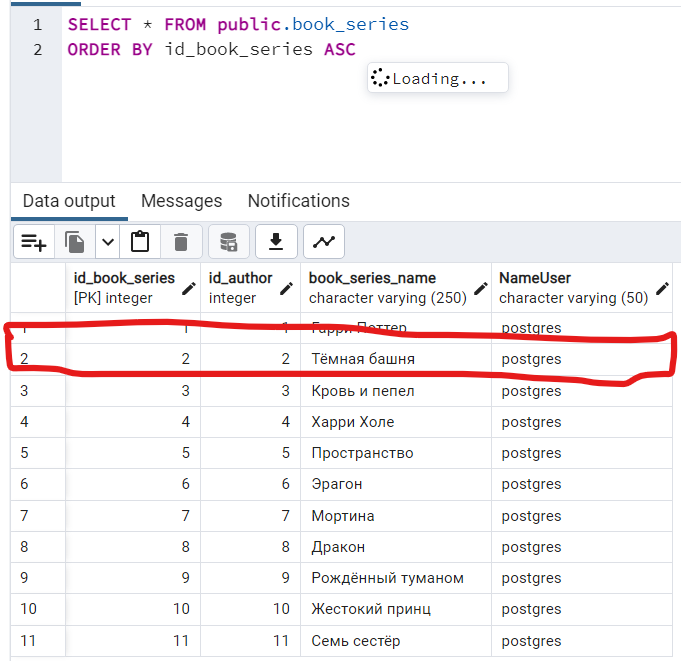


Рисунок 4 – Состояние таблицы book\_series до применения процедуры

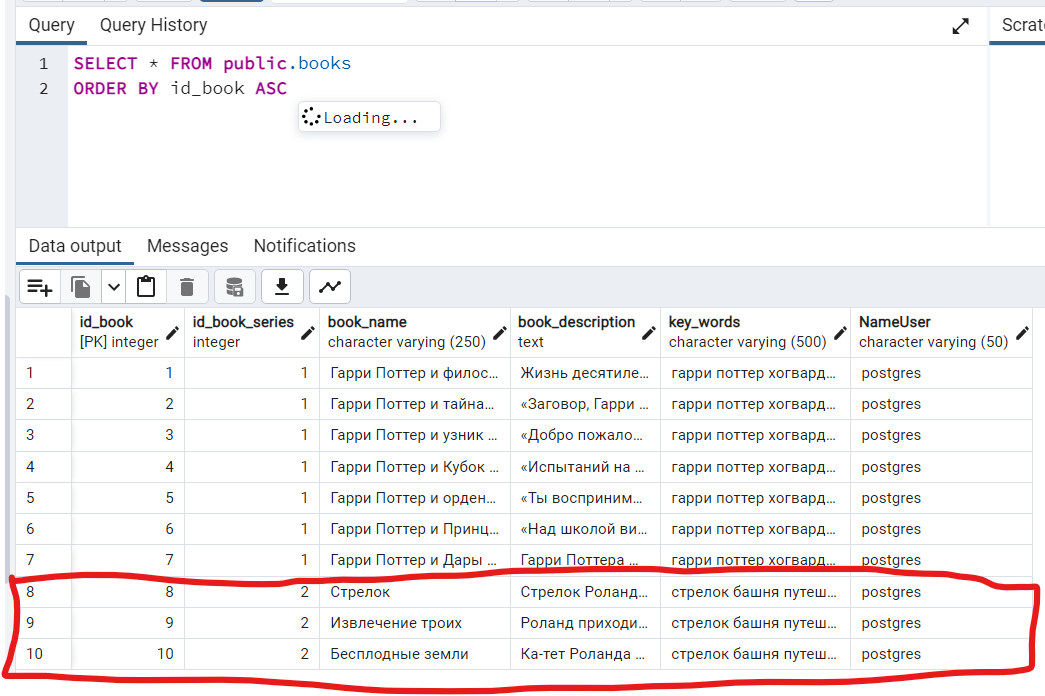


Рисунок 5 – Состояние таблицы books до применения процедуры

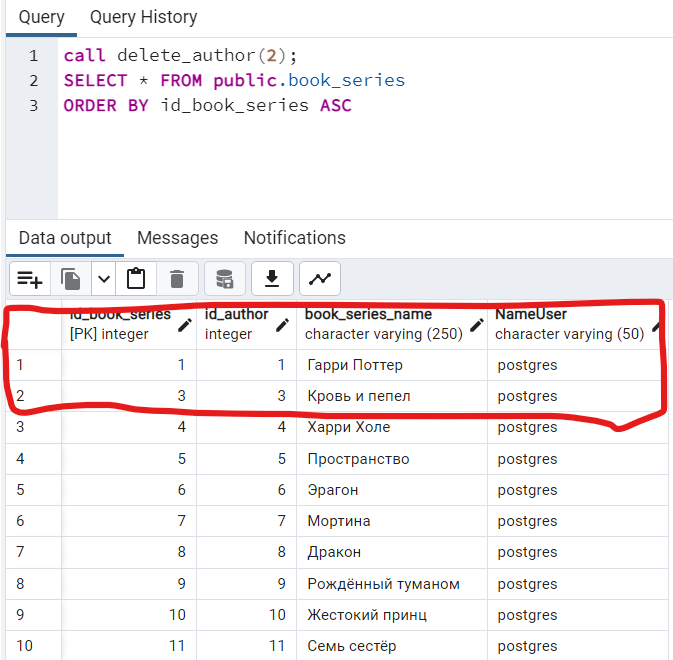


Рисунок 6 – Проверка работоспособности процедуры

1. Внесите такие изменения в хранимые процедуры удаления, добавления и изменения записей в таблицах, которые позволят выполнить операции только для заданных пользователей

**Создание хранимой процедуры удаления записей из таблицы authors, только для пользователя postgres:**

CREATE OR REPLACE PROCEDURE delete\_author(id\_to\_delete integer)

LANGUAGE 'plpgsql'

AS

$BODY$

DECLARE userName character varying;

BEGIN

userName = 'postgres';

IF USER IS DISTINCT FROM userName THEN

RAISE EXCEPTION 'Access Denied';

END IF;

DELETE FROM capters

WHERE id\_book

IN (SELECT id\_book FROM book WHERE id\_book\_series IN

(SELECT id\_book\_series FROM book\_series WHERE id\_author = id\_to\_delete));

DELETE FROM books

WHERE id\_book\_series IN

(SELECT id\_book\_series FROM book\_series WHERE id\_author = id\_to\_delete);

DELETE FROM book\_series

WHERE id\_author = id\_to\_delete;

DELETE FROM author

WHERE id\_author = id\_to\_delete;

END;

$BODY$;

**Создание хранимой процедуры для изменения записи для таблицы authors, только для пользователя postgres:**

CREATE OR REPLACE PROCEDURE update\_author(id integer, name varchar)

LANGUAGE 'plpgsql'

AS

$BODY$

DECLARE userName character varying;

BEGIN

userName = 'postgres';

IF USER IS DISTINCT FROM userName THEN

RAISE EXCEPTION 'Access Denied';

END IF;

IF EXISTS(SELECT FROM authors WHERE author\_name = name) THEN

RAISE EXCEPTION 'Record with name % already exists', name;

ELSE

UPDATE authors SET author\_name = name WHERE id\_author = id;

END IF;

END;

$BODY$;

**Создание хранимой процедуры для добавления записи для таблицы authors:**

CREATE OR REPLACE PROCEDURE add\_author(name varchar)

LANGUAGE 'plpgsql'

AS

$BODY$

DECLARE userName character varying;

BEGIN

userName = 'postgres';

IF USER IS DISTINCT FROM userName THEN

RAISE EXCEPTION 'Access Denied';

END IF;

IF EXISTS(SELECT FROM authors WHERE author\_name = name) THEN

RAISE EXCEPTION 'Record with name % already exists', name;

ELSE

INSERT INTO authors (id\_author, author\_name) VALUES (

(SELECT COALESCE(MAX(id\_author), 0) + 1 FROM authors),

name

);

END IF;

END;

$BODY$;

**Проверка работы процедуры:**

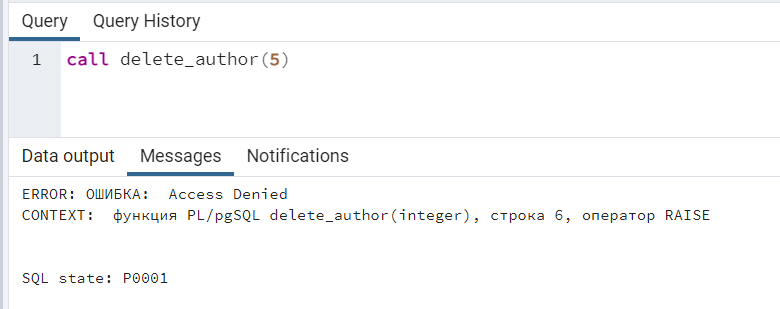
****

Рисунок 7 – Проверка процедуры под другим пользователем

Данная лабораторная работа выполняется с использованием СУБД PostgreSQL, в которой нет функциональности пакетов, которые присутствуют в функциональности СУБД Oracle.