Министерство образования Новосибирской области ГБПОУ НСО «Новосибирский авиационный технический колледж имени Б.С.Галущака»

Практическая работа № 6 - триггеры и функции

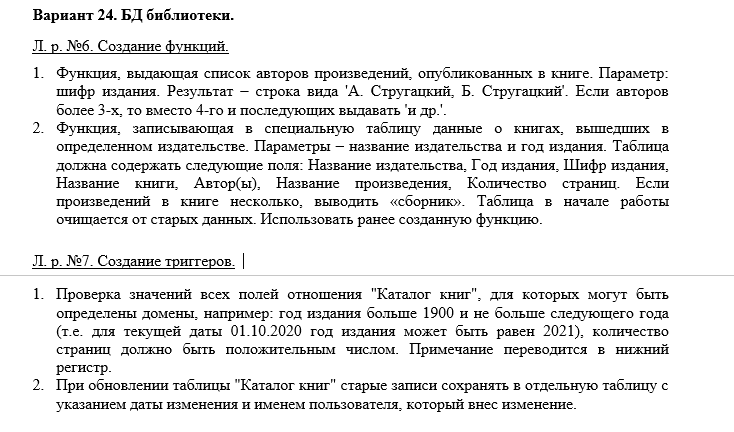
Вариант № 24

Работу выполнил:

студент группы ПР-21.102:

Ядыкин С.И.

Проверил: Еремеева Любовь Антоновна



Создание функций.

Функция 1: Получение списка авторов по шифру издания –

-- DROP FUNCTION public.get\_authors\_by\_book\_id(int4);

CREATE OR REPLACE FUNCTION public.get\_authors\_by\_book\_id(book\_id\_param integer)

RETURNS character varying

LANGUAGE plpgsql

AS $function$

DECLARE

authors\_list VARCHAR := '';

BEGIN

SELECT STRING\_AGG(author, ', ')

INTO authors\_list

FROM books

WHERE book\_id = book\_id\_param;

IF (SELECT COUNT(\*) FROM string\_to\_array(authors\_list, ', ')) > 3 THEN

authors\_list := (SELECT STRING\_AGG(author, ', ') || ' и др.' FROM books WHERE book\_id = book\_id\_param);

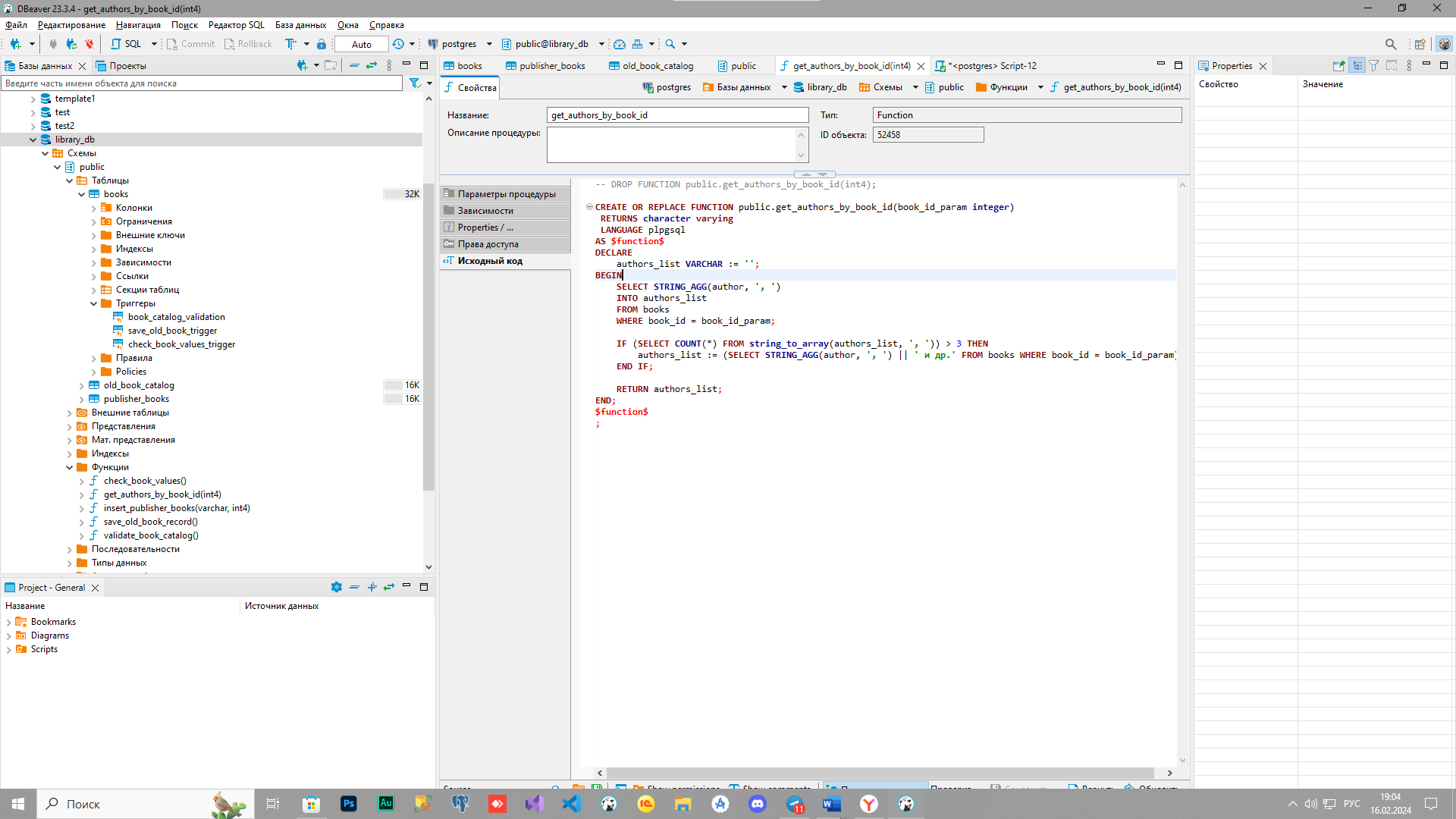
END IF;

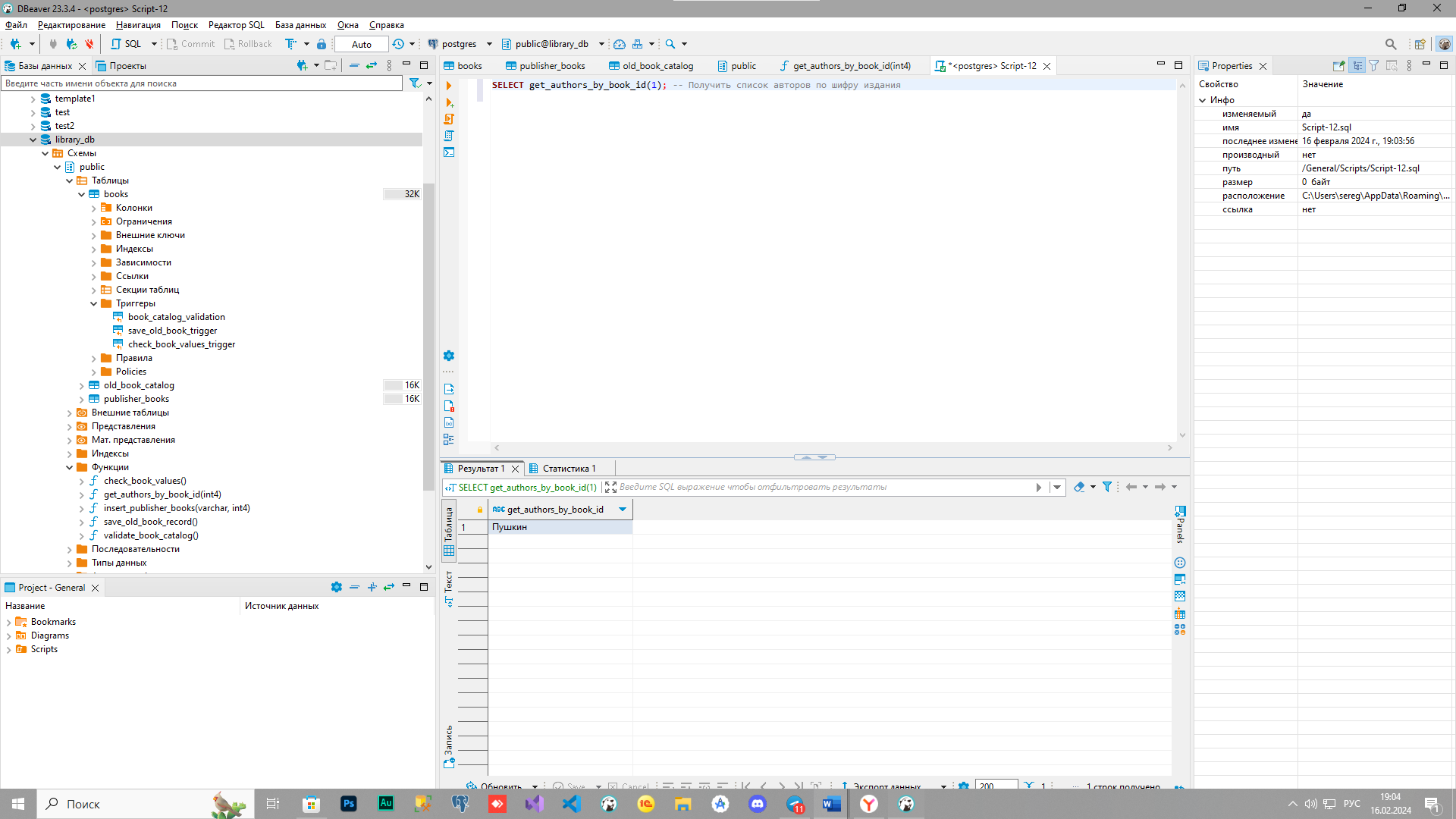
RETURN authors\_list;

END;

$function$

;





**Объяснение работы функции:** get\_authors\_by\_book\_id

Данная функция get\_authors\_by\_book\_id принимает идентификатор книги book\_id\_param в качестве параметра и возвращает строку, содержащую список авторов этой книги. Функция учитывает следующее:

Может быть несколько авторов у одной книги.

Если авторов больше 3, то вместо полного списка выводится "и др.".

authors\_list - строковая переменная для хранения списка авторов, инициализируется пустой строкой.

Выполняется запрос к таблице books (предполагается, что такая таблица существует), где выбираются все авторы (author) книги с идентификатором book\_id\_param.

Результат объединяется в строку с разделителями-запятыми (STRING\_AGG(author, ', ')) и записывается в переменную authors\_list.

Подсчитывается количество элементов в строке authors\_list, разделенной по запятым (SELECT COUNT(\*) FROM string\_to\_array(authors\_list, ', '))).

Если количество авторов больше 3:

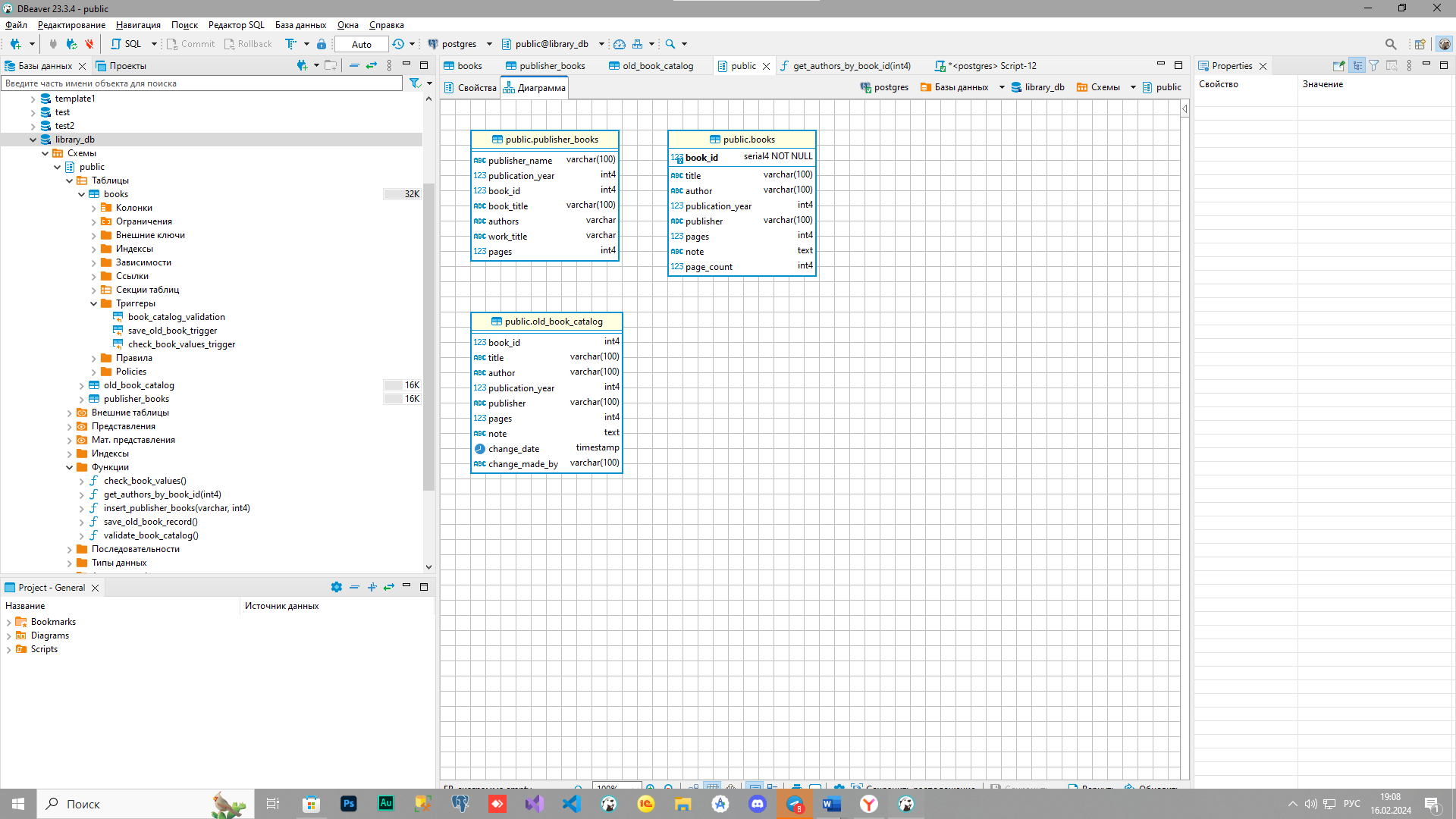
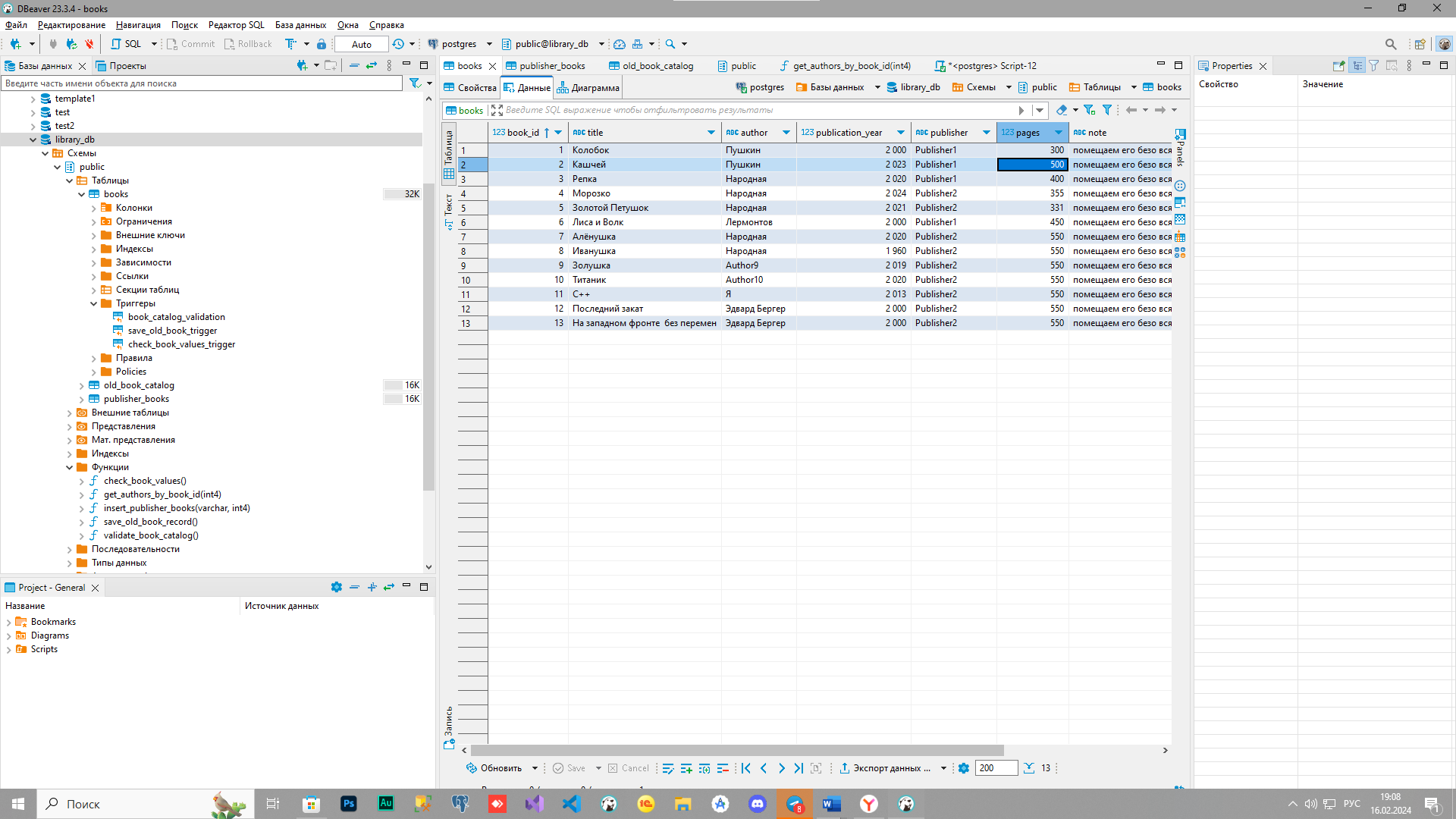
Выполняется повторный запрос к таблице books, аналогичный предыдущему, но к результату добавляется строка "и др." (STRING\_AGG(author, ', ') || ' и др.').

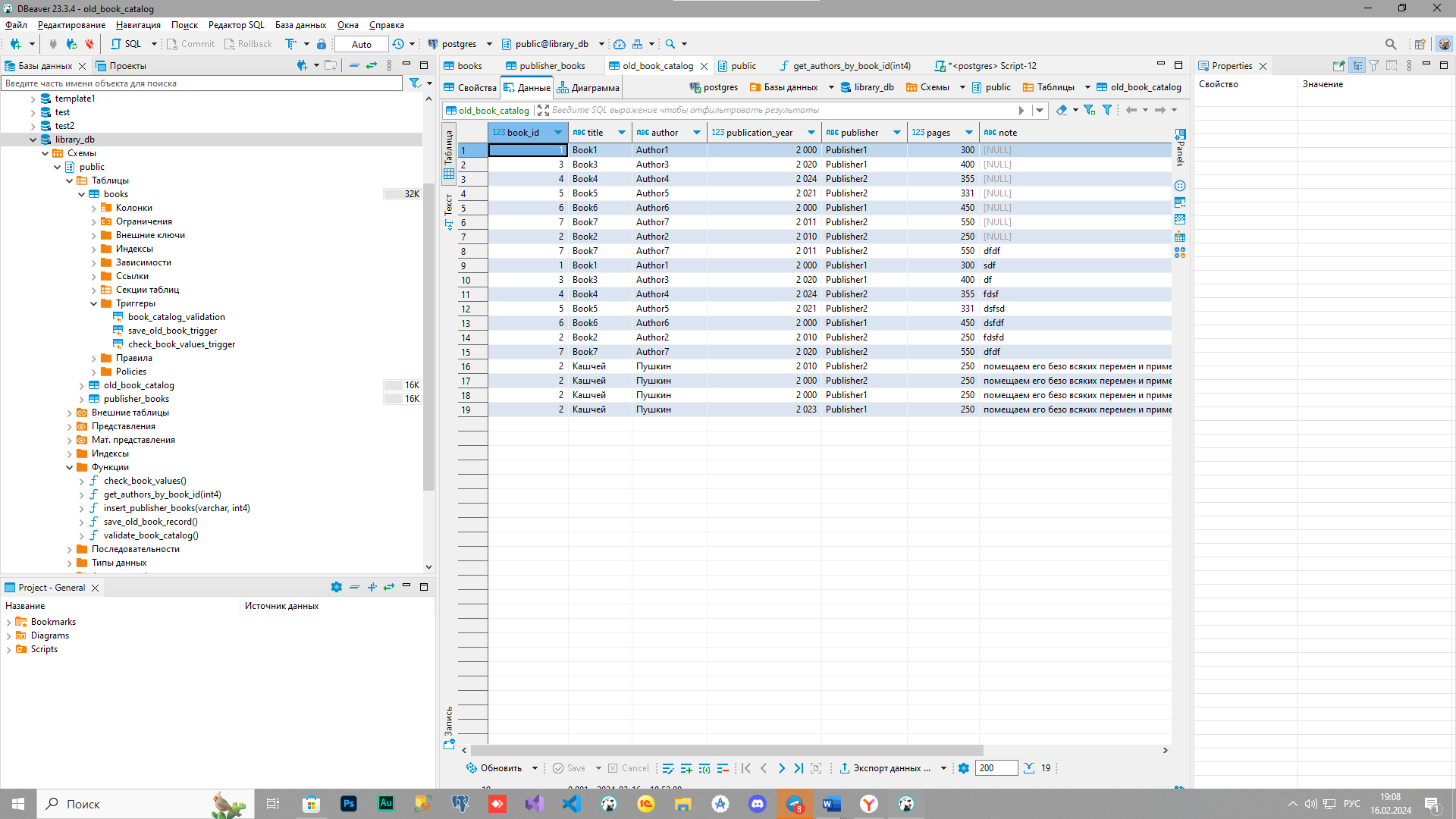
Обновляется значение переменной authors\_list новым результатом.

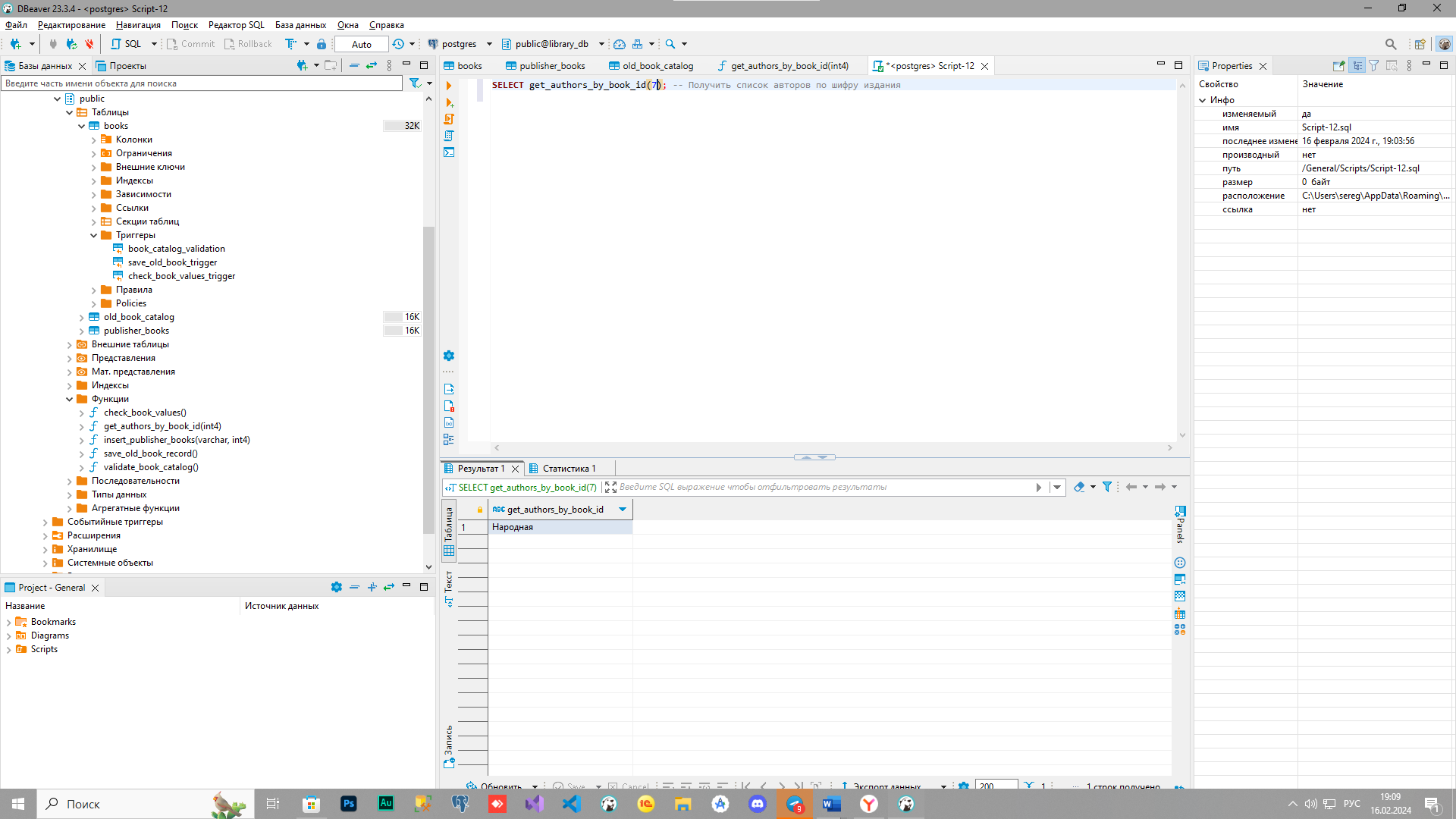
Возврат результата:

Функция возвращает значение переменной authors\_list, которая содержит список авторов книги в требуемом формате.

**Как выглядит сама база:**

**  
 **



Принцип работы 1 Функции:   


Выводит авторов по id

**Функция 2, записывающая в специальную таблицу данные о книгах, вышедших в определенном издательстве. Параметры – название издательства и год издания. Таблица должна содержать следующие поля: Название издательства, Год издания, Шифр издания, Название книги, Автор(ы), Название произведения, Количество страниц. Если произведений в книге несколько, выводить «сборник». Таблица в начале работы очищается от старых данных. Использовать ранее созданную функцию.**

-- DROP FUNCTION public.insert\_publisher\_books(varchar, int4);

**CREATE** **OR** **REPLACE** **FUNCTION** public.insert\_publisher\_books(publisher\_name\_param **character** **varying**, publication\_year\_param **integer**)

**RETURNS** **void**

**LANGUAGE** plpgsql

**AS** **$function$**

**BEGIN**

**DELETE** **FROM** publisher\_books;

**INSERT** **INTO** publisher\_books (publisher\_name, publication\_year, book\_id, book\_title, authors, work\_title, pages)

**SELECT** publisher, publication\_year, book\_id, title, author,

**CASE** **WHEN** **COUNT**(\*) > 1 **THEN** **'сборник'** **ELSE** title **END**,

pages

**FROM** books

**WHERE** publisher = publisher\_name\_param **AND** publication\_year = publication\_year\_param

**GROUP** **BY** publisher, publication\_year, book\_id, title, author, pages;

**END**;

**$function$**

;

Данная функция insert\_publisher\_books принимает два параметра:

publisher\_name\_param: название издательства

publication\_year\_param: год издания

Функция очищает таблицу publisher\_books и затем заполняет ее данными о книгах, опубликованных данным издательством в указанный год.

Функция начинает с очистки таблицы publisher\_books, удаляя все существующие записи с помощью команды DELETE FROM publisher\_books.

Выполняется запрос к таблице books (предполагается, что такая таблица существует), где выбираются все книги, соответствующие заданным параметрам:

publisher: совпадает с publisher\_name\_param

publication\_year: равен publication\_year\_param

Для каждой книги выбираются поля:

book\_id: идентификатор книги

title: название книги

author: автор(ы) книги

pages: количество страниц книги

Формирование списка авторов и название произведения:

Используется функция CASE WHEN COUNT(\*) > 1 THEN 'сборник' ELSE title END для формирования имени произведения:

Если книга содержит несколько произведений (т.е. у нее несколько записей в таблице books), то присваивается значение "сборник".

В противном случае используется само название книги (title).

Автор(ы) объединяются в одну строку с разделителем по запятым.

Выполняется команда INSERT INTO для добавления отобранных данных в таблицу publisher\_books.

Вставляются следующие поля:

publisher\_name: издательство (из параметра функции)

publication\_year: год издания (из параметра функции)

book\_id: идентификатор книги

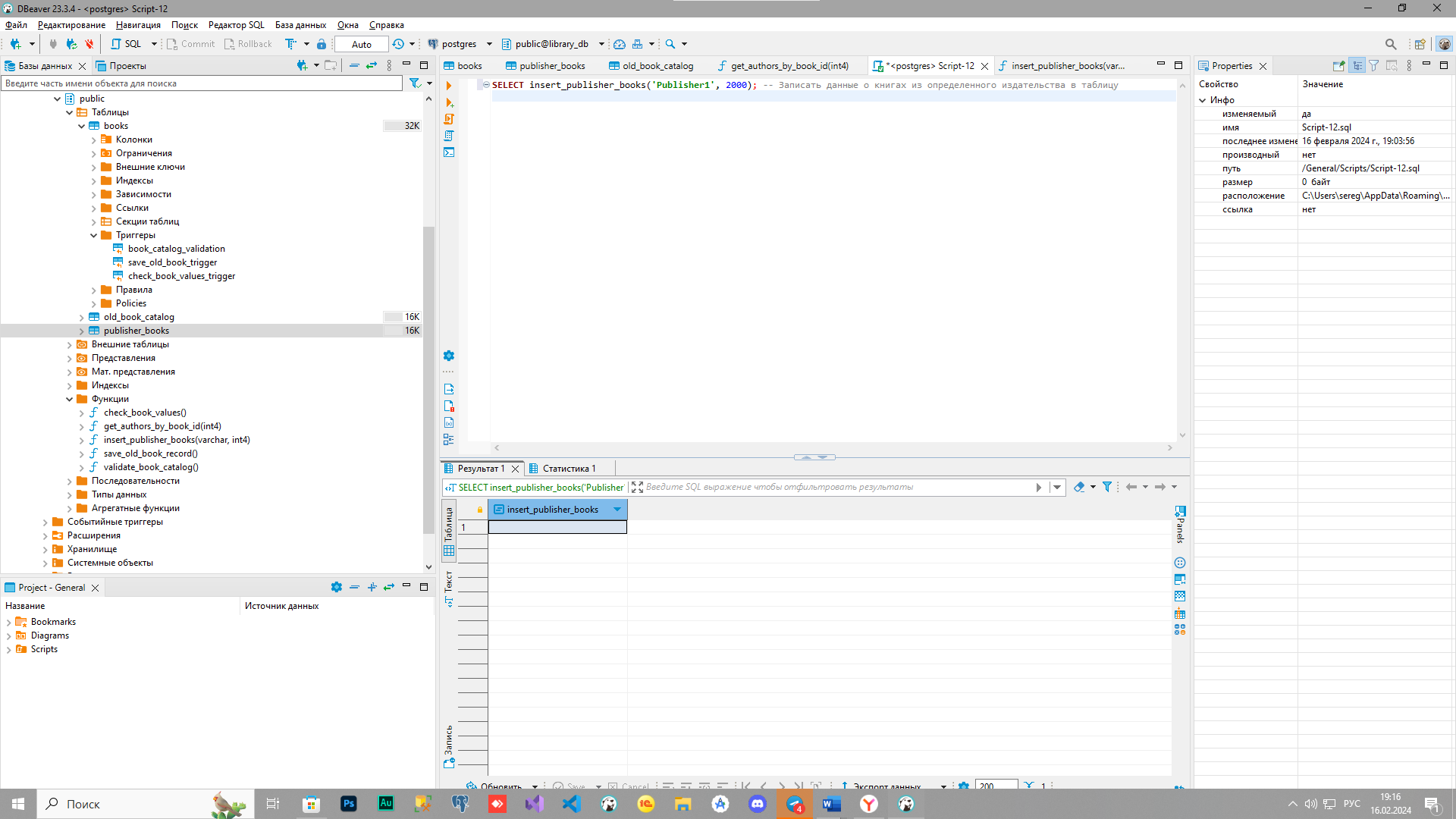
book\_title: название книги

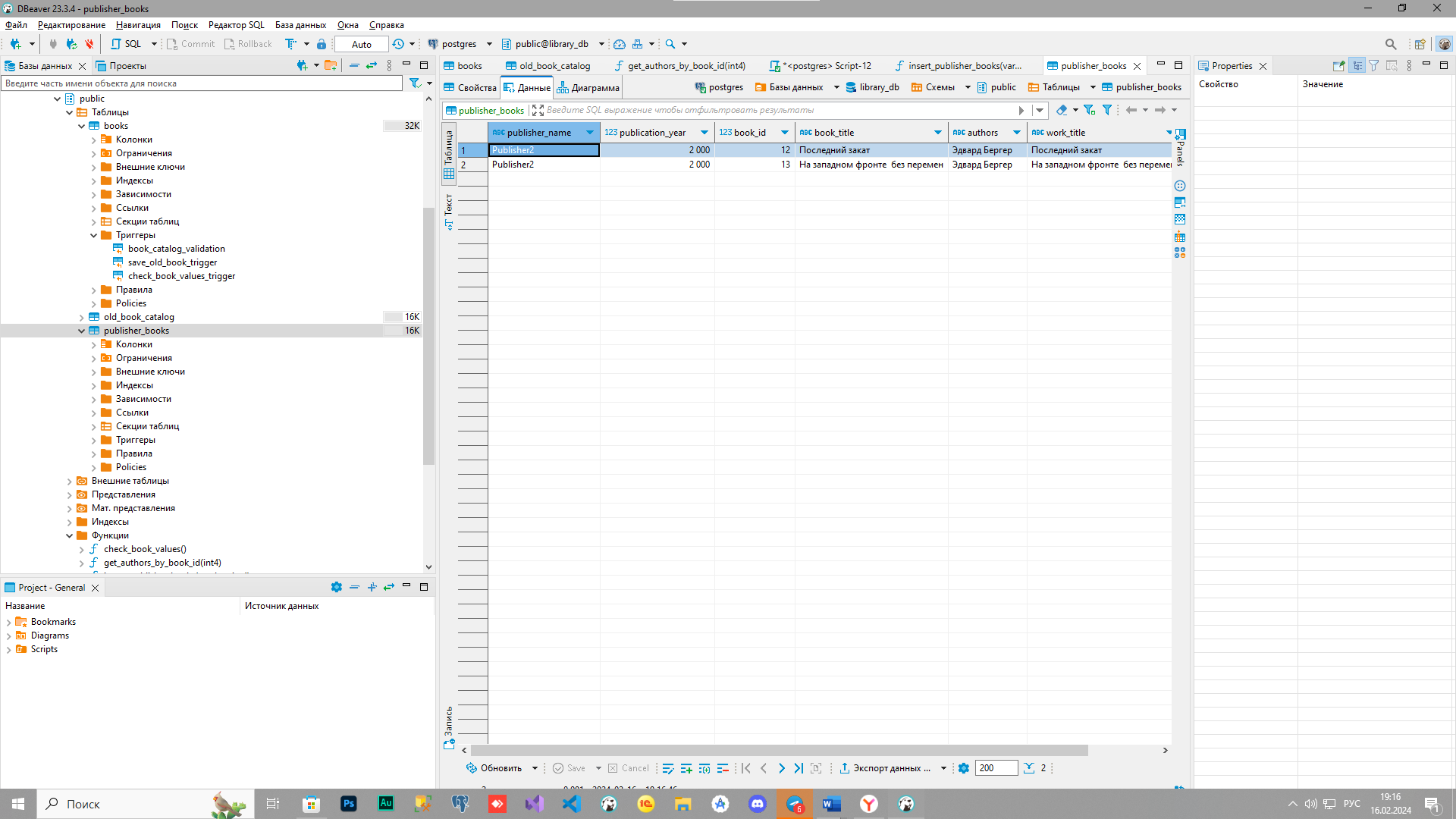
authors: список авторов книги

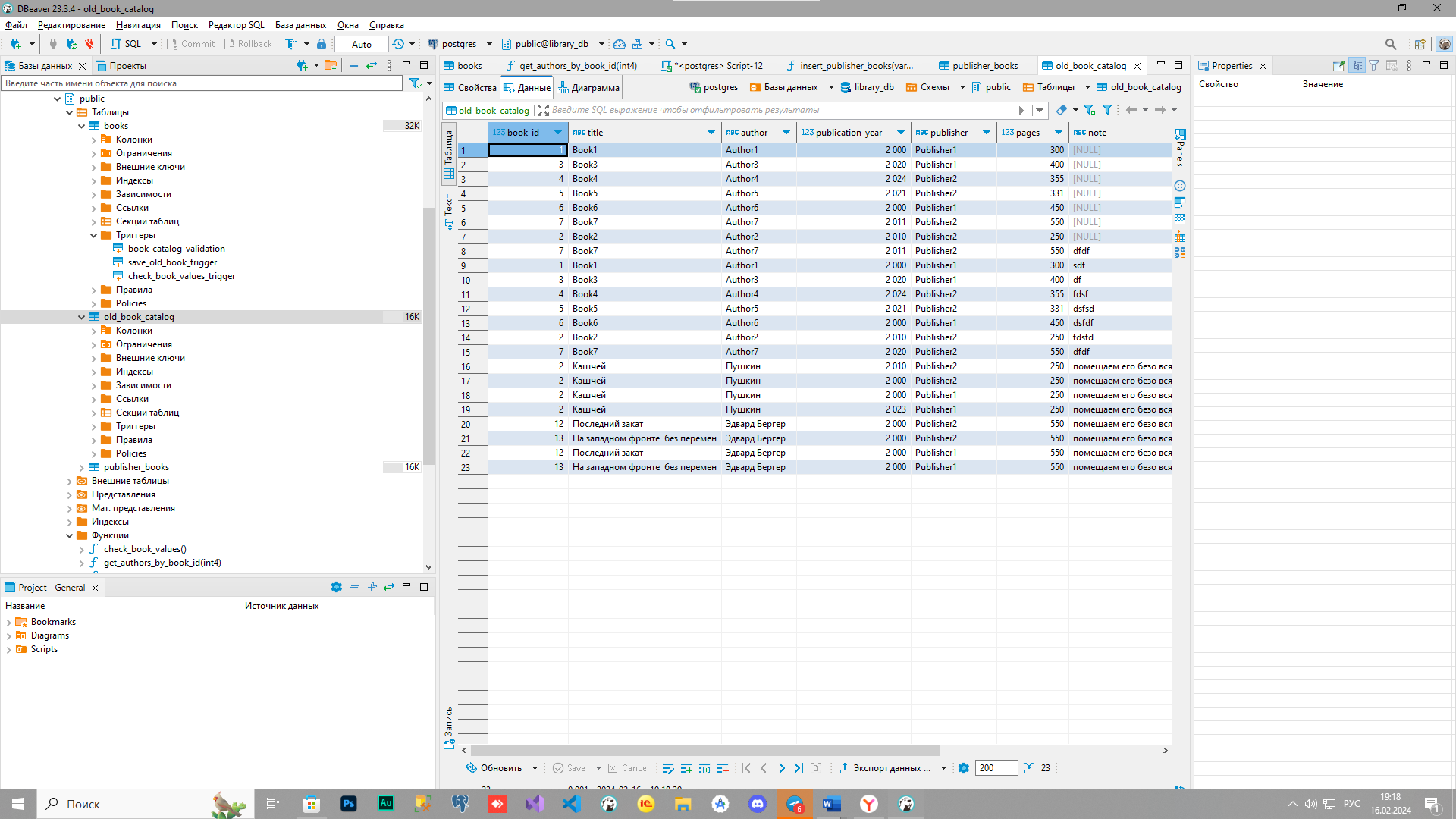
work\_title: название произведения (сборник или название книги)

pages: количество страниц книги

Запрос в пункте 2 использует GROUP BY, чтобы избежать дублирования записей в publisher\_books для книг с несколькими произведениями. Группировка осуществляется по издательству, году издания, идентификатору книги, названию книги, автору и количеству страниц.





**Как можно заметить, данная функция помогает обновить данные в таблице.   
**

**А в этой таблице идёт фиксация.**

**Л. р. №7. Создание триггеров.  
create** **trigger** book\_catalog\_validation **before**

**insert**

**or**

**update**

**on**

public.books **for** **each** **row** **execute** **function** validate\_book\_catalog()

Этот триггер book\_catalog\_validation будет запускать функцию validate\_book\_catalog() каждый раз перед вставкой или обновлением записи в таблице public.books. Это позволит проверить данные, которые собираются быть добавлены или изменены в таблице, и выполнить какие-то действия по валидации перед тем, как изменения будут фактически применены. Это помогает обеспечить целостность данных и соблюдение заданных правил в каталоге книг.

**create** **trigger** save\_old\_book\_trigger **after**

**update**

**on**

public.books **for** **each** **row** **execute** **function** save\_old\_book\_record()

Этот триггер `save\_old\_book\_trigger` будет запускаться после обновления записи в таблице `public.books` и будет вызывать функцию `save\_old\_book\_record()` для сохранения старой версии обновленной записи книги. Это позволит сохранить историю изменений и иметь доступ к предыдущей информации о книге после каждого обновления.   
  
**create** **trigger** check\_book\_values\_trigger **before**

**insert**

**or**

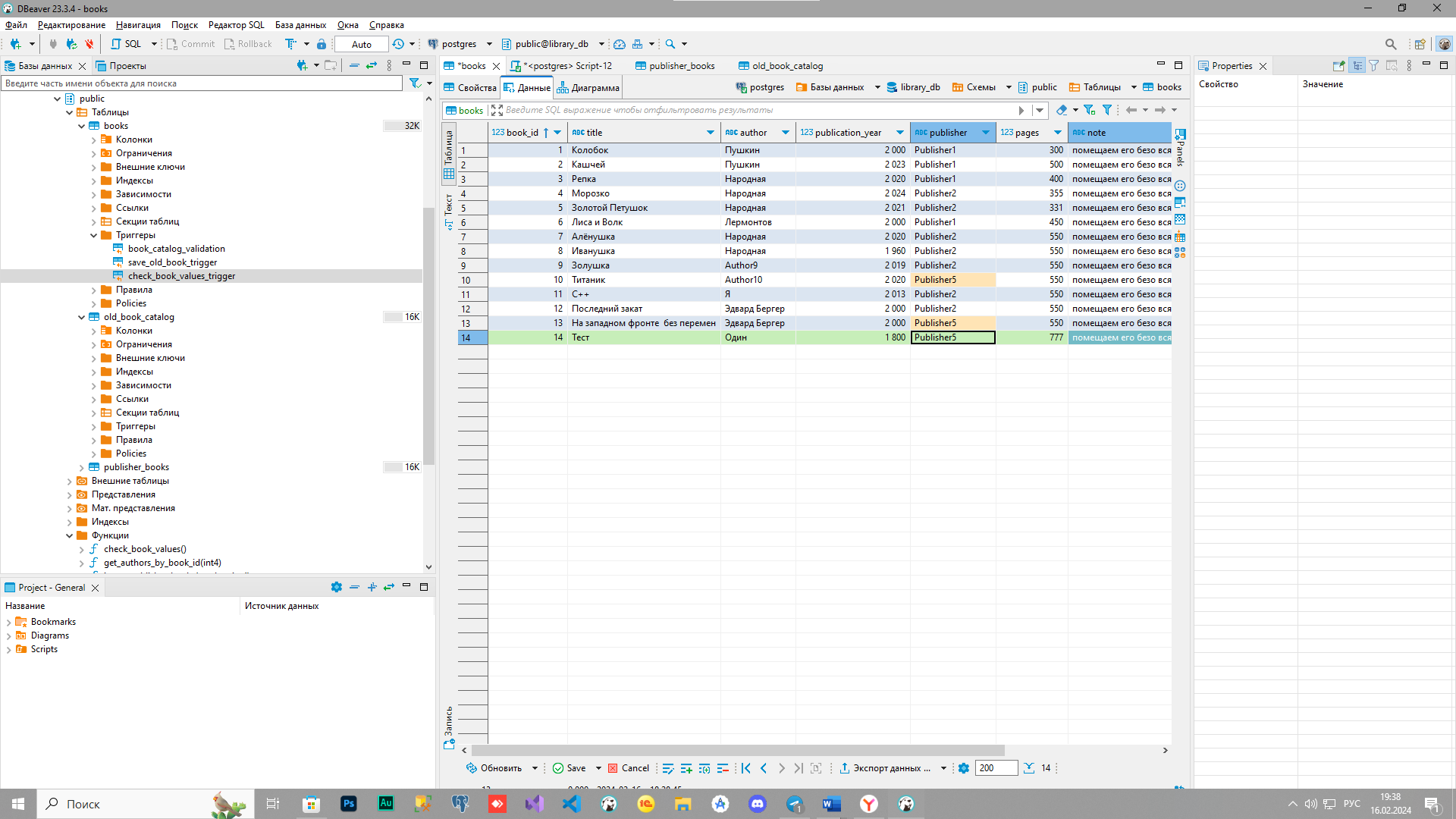
**update**

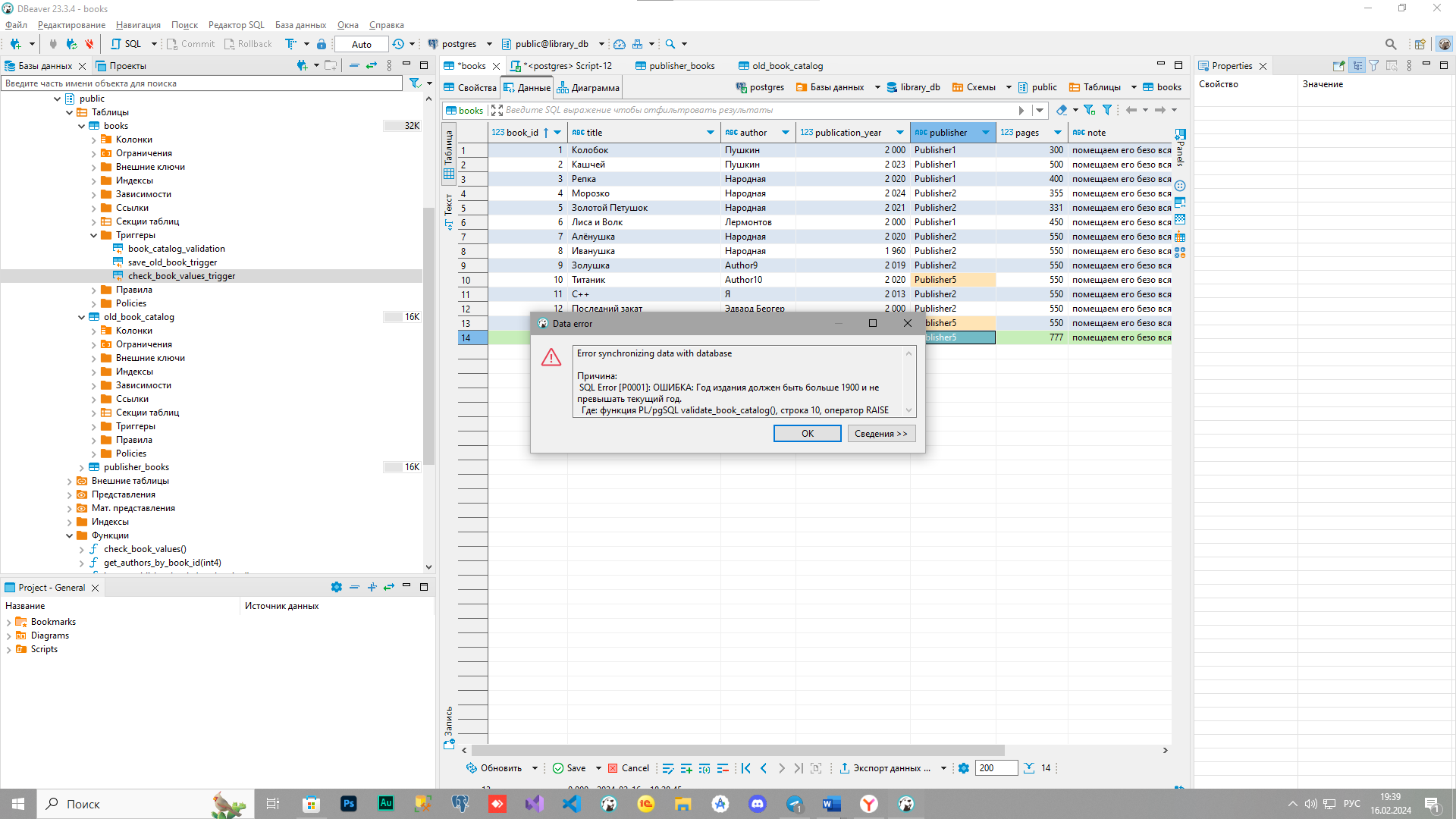
**on**

public.books **for** **each** **row** **execute** **function** check\_book\_values()

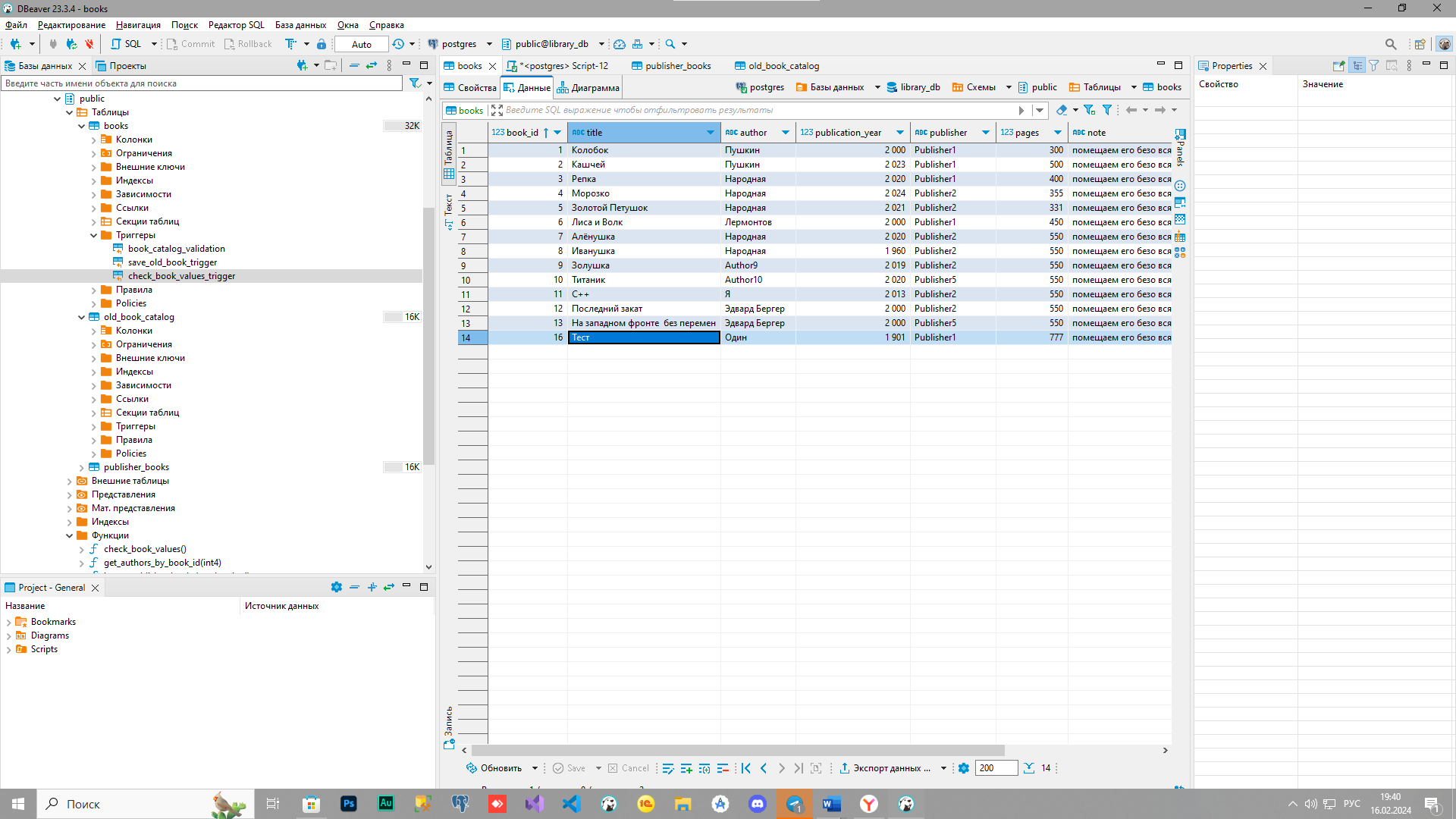
Этот триггер `check\_book\_values\_trigger` будет запускаться перед вставкой новой записи или обновлением существующей записи в таблице `public.books`. Триггер будет вызывать функцию `check\_book\_values()`, которая будет проверять значения, которые собираются быть вставлены или обновлены в таблице книг. Это позволит осуществлять дополнительные проверки перед выполнением операций вставки или обновления, что может помочь обеспечить целостность данных или применить дополнительные логику обработки в зависимости от значений.

**Демонстрация работы триггера:**

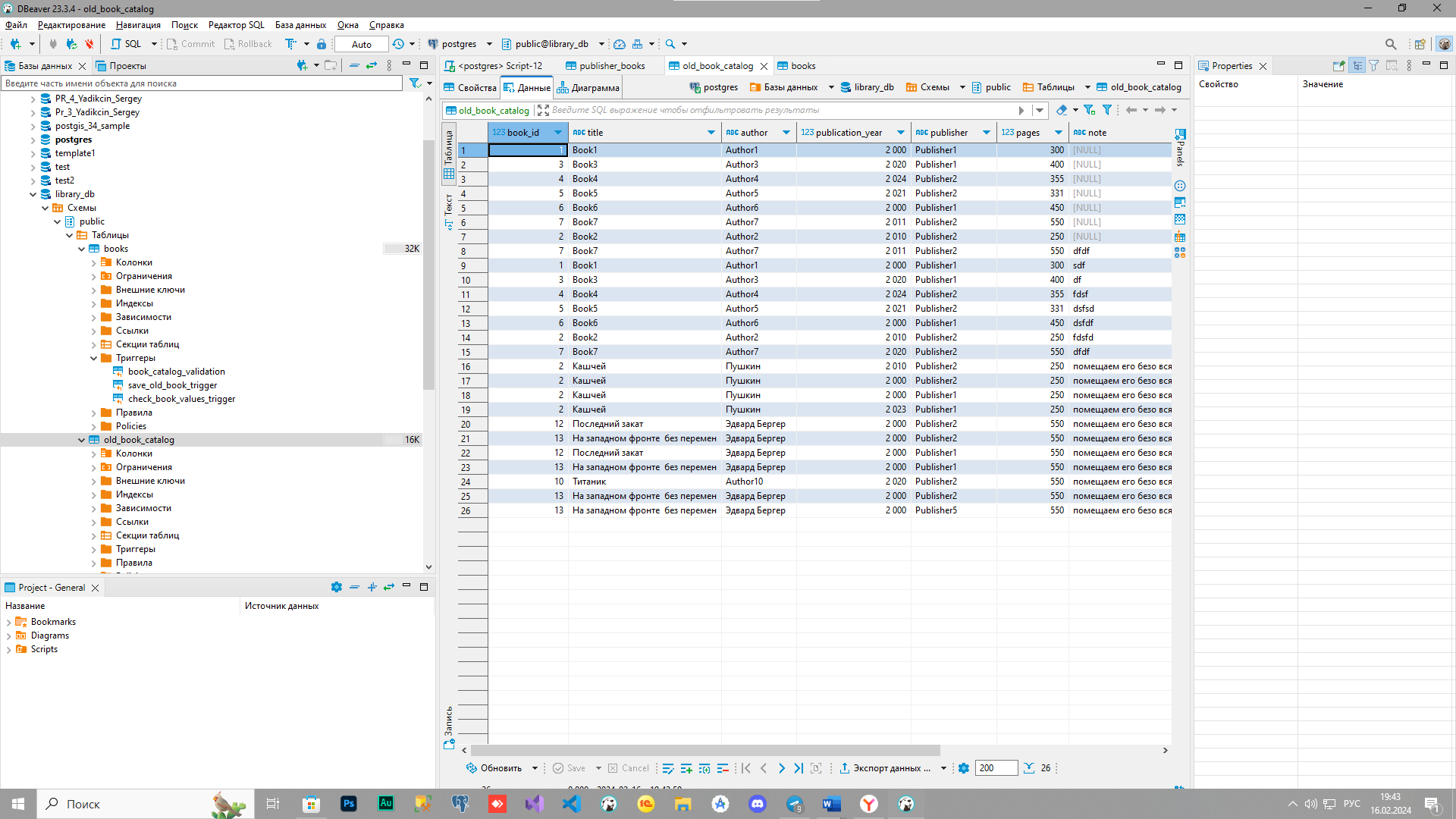
**Допустим, я хочу создать новую запись :   
**

Но при сохранении записи выдаст следующую ошибку:   


Т.к год у нас 1800.



Но поставим 1901 год и данные успешно будут сохранены.



Данный триггер позволяет провести проверку.