# ТИТУЛЬНИК

# ЗАДАНИЕ

**СОДЕРЖАНИЕ**

[1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc135153979)

[1.1 Предметная область 5](#_Toc135153980)

[1.2 Основные бизнес-процессы 5](#_Toc135153981)

[2 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ДАННЫХ 6](#_Toc135153982)

[2.1 Концептуальное проектирование базы данных 6](#_Toc135153983)

[2.2 Логическое проектирование базы данных 8](#_Toc135153984)

[2.3 Выбор и описание СУБД 9](#_Toc135153985)

[2.4 Физическое проектирование базы данных 10](#_Toc135153986)

[3 ИСПОЛЬЗОВАНИЕ БАЗЫ ДАННЫХ 11](#_Toc135153987)

[3.1 Создание DataSet 16](#_Toc135153988)

[3.2 Добавление данных в базу 20](#_Toc135153989)

[3.3 Запросы к базе данных 28](#_Toc135153990)

[3.4 Тестирование функций и триггеров 33](#_Toc135153991)

# ВВЕДЕНИЕ

## Актуальность проблемы

Модернизация структуры базы данных становится неотъемлемой частью деятельности для всех современных компаний. Она обусловлена следующими факторами:

1. стремительно развивающиеся технологии выдвигают требования по обновлению и адаптации баз данных к новым возможностям и объемам информации;
2. изменения в бизнес-процессах подразумевают необходимость согласованной структуры баз данных для эффективного управления информационными потоками;
3. адаптация к изменяющемуся законодательству требует своевременной модернизации баз данных для соответствия обязательствам по хранению и обработке пользовательских данных.

Учет всех этих факторов позволяет компаниям поддерживать гибкость, эффективность и гарантировать защиту интересов клиентов, делая модернизацию баз данных важной составляющей успешной бизнес-стратегии.

## Цель работы:

Систематизация и закрепление полученных теоретических и практических умений по разработке баз данных с использованием СУБД PostgreSQL.

## Задачи работы:

1. изучить литературу по проектированию и разработке базы данных в СУБД PostgreSQL;
2. получить практический опыт разработки базы данных в СУБД PostgreSQL.

# 1 описание предметной области

## Предметная область

Вы являетесь руководителем библиотеки. Ваша библиотека решила зарабатывать деньги, выдавая напрокат некоторые книги, имеющиеся в небольшом количестве экземпляров. Вашей задачей является отслеживание финансовых показателей работы. У каждой книги, выдаваемой в прокат, есть название, автор, жанр. В зависимости от ценности книги вы определили для каждой из них залоговую стоимость (сумма, вносимая клиентом при взятии книги напрокат) и стоимость проката (сумма, которую клиент платит при возврате книги, получая назад залог). В библиотеку обращаются читатели. Все читатели регистрируются в картотеке, которая содержит стандартные анкетные данные (фамилия, имя, отчество, адрес, телефон). Каждый читатель может обращаться в библиотеку несколько раз. Все обращения читателей фиксируются, при этом по каждому факту выдачи книги запоминаются дата выдачи и ожидаемая дата возврата.

Несложный анализ показал, что стоимость проката книги должна зависеть не только от самой книги, но и от срока ее проката. Кроме того, необходимо добавить систему штрафов за вред, нанесенный книге, и систему скидок для некоторых категорий читателей.

## Основные бизнес-процессы

При изучении предметной области было выявлено 5 бизнес-процессов Подробнее с ними можно ознакомиться в «Таблице 1 – Список основных бизнес-процессов».

Таблица 1 - Список основных бизнес-процессов

|  |  |
| --- | --- |
| Номер бизнес-процесса | Наименование бизнес процесса |
| 1 | Добавление новой книги в прокат |
| 2 | Изъятие книги из проката |
| 3 | Регистрация пользователя в картотеке |
| 4 | Выдача книги в прокат клиенту |
| 5 | Получение книги после проката от клиента |

# 2 проектирование и разработка данных

## 2.1 Концептуальное проектирование базы данных

При анализе предметной области были выявлены следующие сущности: «книга», «прокат», «клиент».

Сущность «книга» содержит следующие атрибуты:

1. автор;
2. жанр книги (может быть несколько);
3. название;
4. размер депозита, вносимого клиентом при взятии в прокат;
5. стоимость аренды книги за день проката;
6. комментарий о состоянии книги;
7. статус доступности для проката.

Сущность «прокат» содержит следующие атрибуты:

1. длительность проката;
2. стоимость проката;
3. штраф за нанесенный книге ущерб;
4. книга, взятая в прокат;
5. клиент, воспользовавшийся услугой.

Сущность «клиент» содержит следующие атрибуты:

1. ФИО клиента;
2. категория клиента (школьник, студент, пенсионер, стандарт);
3. размер скидки в зависимости от категории;
4. контакты клиента;
5. адрес проживания.

Связи между сущностями отображены в «Рисунке 1 – концептуальная модель данных».

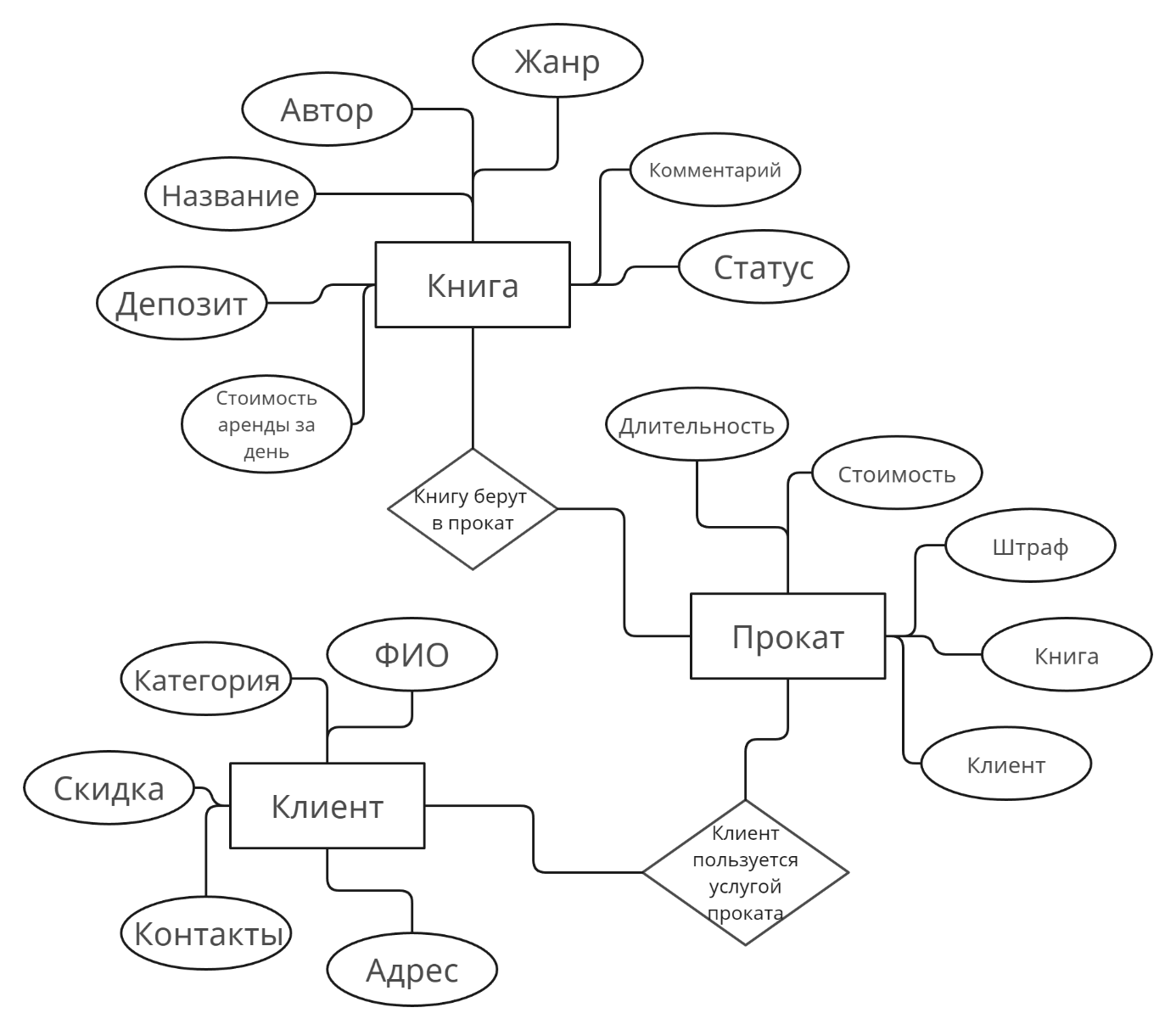


Рисунок 1 – концептуальная модель данных

Структуры сущностей приведены в последующих таблицах: «Таблица 2 – структура сущности книга», «Таблица 3 – структура сущности прокат», «Таблица 4 – структура сущности».

Таблица 2 – структура сущности книга

|  |  |  |  |
| --- | --- | --- | --- |
| Название параметра | Тип данных | Размер | Диапазон значений |
| Название | Текстовый | 100 |  |
| Автор | Текстовый | 100 |  |
| Жанр | Текстовый | 100 |  |
| Размер депозита | Числовой |  | От 0 |
| Стоимость аренды за день | Числовой |  | От 0 |
| Комментарий о состоянии книги | Текстовый |  |  |
| Статус доступности | Логический | 1 | TRUE – книга в наличии и доступна для проката.  FALSE – книга уже взята в прокат |

Таблица 3 – структура сущности прокат

|  |  |  |  |
| --- | --- | --- | --- |
| Название параметра | Тип данных | Размер | Диапазон значений |
| Длительность | Дата | 8 | ДД-ММ-ГГГГ |
| Стоимость | Числовой |  | От 0 |
| Штраф | Числовой |  | От 0 |
| Книга | Текстовый |  |  |
| Клиент | Текстовый |  |  |

Таблица 4 – структура сущности

|  |  |  |  |
| --- | --- | --- | --- |
| Название параметра | Тип данных | Размер | Диапазон значений |
| ФИО | Текстовый | 100 |  |
| Категория | Текстовый | 100 |  |
| Размер скидки | Числовой |  | От 0 до 1 |
| Контакты | Текстовый |  | Подходит под маску +Х (ХХХ) ХХХ-ХХ-ХХ |
| Адрес проживания | Текстовый |  |  |

## 2.2 Логическое проектирование базы данных

Для выявленных сущностей была создана логическая схема базы данных (Рисунок 2 – логическая схема базы данных).

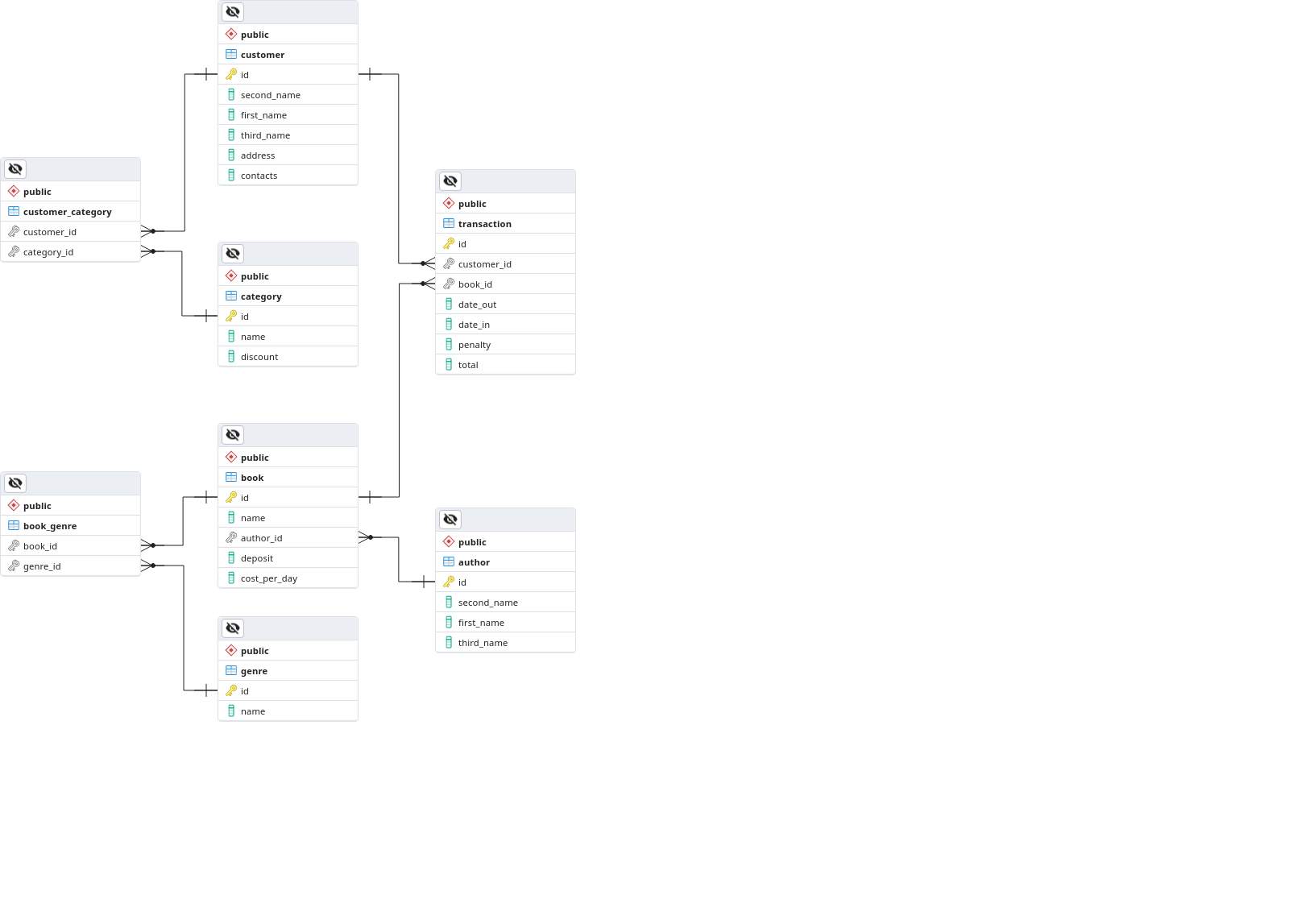


Рисунок 2 – логическая модель данных

## 2.3 Выбор и описание СУБД

Для реализации курсовой работы по базам данных была выбрана СУБД PostgreSQL.

PostgreSQL является СУБД с открытым исходным кодом, что позволяет как можно глубже изучить внутреннее устройство СУБД. PostgreSQL также обладает широким набором функций, обеспечивающих гибкость при разработке баз данных, поддерживает различные типы данных. Надежность и стабильность PostgreSQL также являются важными преимуществами. СУБД активно поддерживается и разрабатывается сообществом разработчиков, что гарантирует безопасность и исправление ошибок. Встроенные механизмы резервного копирования, восстановления и системы журналирования обеспечивают сохранность данных и доступность базы данных. Кроме того, PostgreSQL обладает высокой производительностью и масштабируемостью. Он способен эффективно обрабатывать большие объемы данных и поддерживает параллельную обработку запросов, оптимизацию запросов и кэширование. Наконец, PostgreSQL имеет широкое сообщество разработчиков и пользователей, а также обширную документацию. Это облегчает изучение и работу с PostgreSQL, а также предоставляет возможность получить поддержку и советы от опытных пользователей. Все эти причины делают PostgreSQL отличным выбором для реализации курсовой работы.

## 2.4 Физическое проектирование базы данных

Используя функционал PostgreSQL была создана физическая схема базы данных (Рисунок 3 – физическая модель данных).

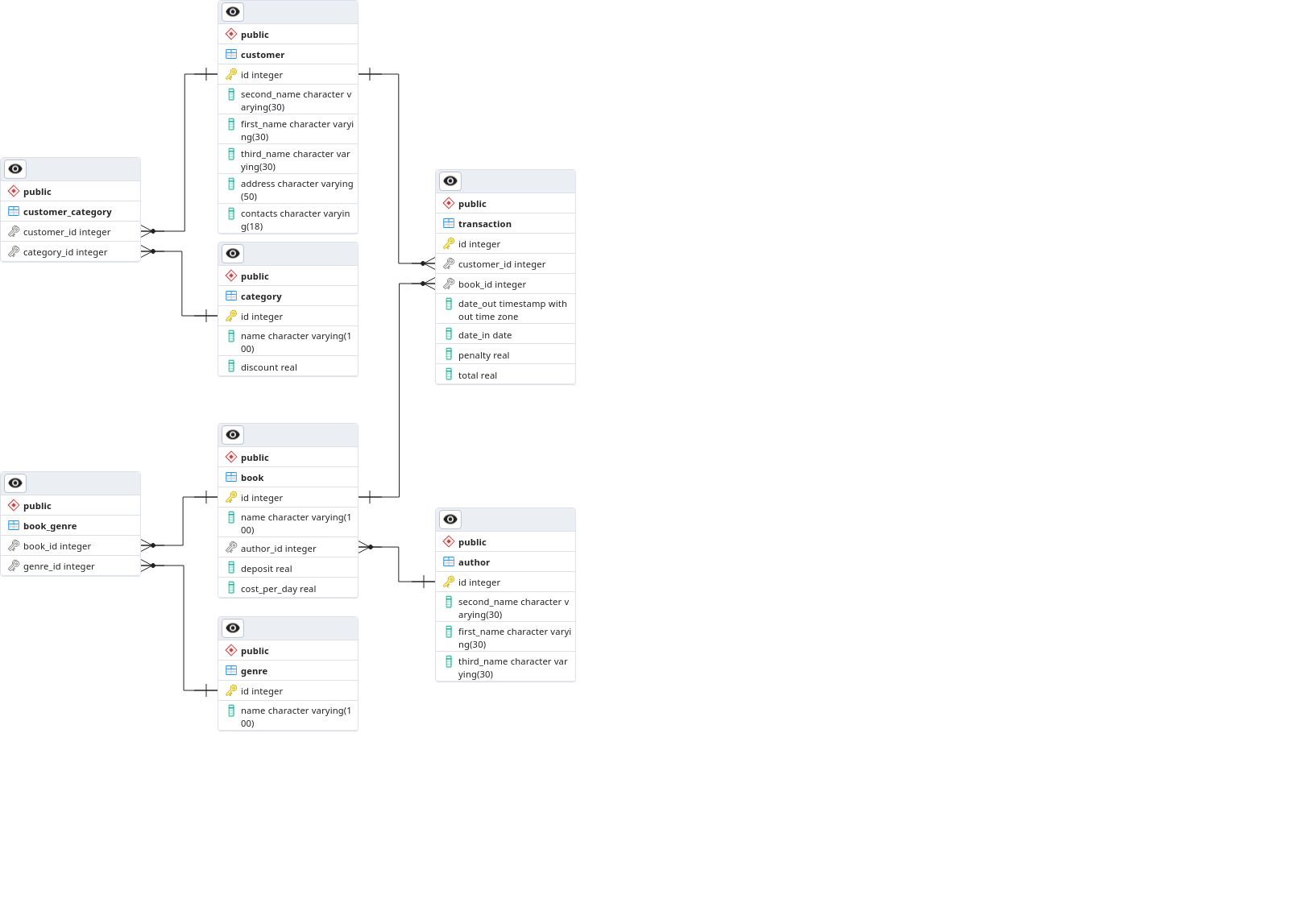


Рисунок 3 – физическая модель данных

Далее модель была реализована при помощи PostgreSQL. Для создания таблиц и связей между ними были использованы SQL-запросы, которые указаны в «Листинге 1 – инициализация базы данных».

Листинг 1 – инициализация базы данных

-------------------------------------------------------------

-- Table author ---------------------------------------------

DROP TABLE IF EXISTS author CASCADE;

CREATE TABLE IF NOT EXISTS author

(

id serial PRIMARY KEY,

second\_name varchar (30) NOT NULL,

first\_name varchar (30) NOT NULL,

third\_name varchar (30) DEFAULT (NULL)

);

-------------------------------------------------------------

-- Table book -----------------------------------------------

DROP TABLE IF EXISTS book CASCADE;

CREATE TABLE IF NOT EXISTS book

(

id serial PRIMARY KEY,

name varchar (100) NOT NULL,

author\_id integer REFERENCES author (id) NOT NULL,

deposit real NOT NULL,

cost\_per\_day real NOT NULL,

comment text DEFAULT ('Новая книга'),

available boolean DEFAULT (TRUE)

);

-------------------------------------------------------------

-- Table genre ----------------------------------------------

DROP TABLE IF EXISTS genre CASCADE;

CREATE TABLE IF NOT EXISTS genre

(

id serial PRIMARY KEY,

name varchar(100) NOT NULL

);

-------------------------------------------------------------

-- Table book\_genre -----------------------------------------

DROP TABLE IF EXISTS book\_genre CASCADE;

CREATE TABLE IF NOT EXISTS book\_genre

(

book\_id serial REFERENCES book (id) NOT NULL,

genre\_id integer REFERENCES genre (id) NOT NULL

);

-------------------------------------------------------------

-- Table customer -------------------------------------------

DROP TABLE IF EXISTS customer CASCADE;

CREATE TABLE IF NOT EXISTS customer

(

id serial PRIMARY KEY,

second\_name varchar(30) NOT NULL,

first\_name varchar(30) NOT NULL,

third\_name varchar(30) DEFAULT (''),

address varchar(50) NOT NULL,

contacts varchar(18) NOT NULL

);

-------------------------------------------------------------

-- Table category -------------------------------------------

DROP TABLE IF EXISTS category CASCADE;

CREATE TABLE IF NOT EXISTS category

(

id serial PRIMARY KEY,

name varchar(100) NOT NULL,

discount real NOT NULL

);

-------------------------------------------------------------

-- Table customer\_category ----------------------------------

DROP TABLE IF EXISTS customer\_category CASCADE;

CREATE TABLE IF NOT EXISTS customer\_category

(

customer\_id integer REFERENCES customer (id) NOT NULL,

category\_id integer REFERENCES category (id) NOT NULL

);

-------------------------------------------------------------

-- Table transaction ----------------------------------------

DROP TABLE IF EXISTS transaction CASCADE;

CREATE TABLE IF NOT EXISTS transaction

(

id serial PRIMARY KEY,

customer\_id integer REFERENCES customer (id) NOT NULL,

book\_id integer REFERENCES book (id) NOT NULL,

date\_out date DEFAULT (CURRENT\_DATE),

date\_in date DEFAULT (NULL),

penalty real DEFAULT (0),

total real DEFAULT (NULL)

);

-------------------------------------------------------------

# 3 использование базы данных

## 3.1 Создание DataSet

Для дальнейшей работы с базой данных был разработан Dataset, указанный в «Таблице 5 – dataset таблицы author», «Таблице 6 – dataset таблицы book», «Таблице 7 – dataset таблицы book\_genre», «Таблице 8 – dataset таблицы category», «Таблице 9 – dataset таблицы customer\_category», «Таблице 10 – dataset таблицы customer», «Таблице 11 – dataset таблицы genre», «Таблице 12 – dataset таблицы transaction».

Таблица 5 – dataset таблицы author

|  |  |  |  |
| --- | --- | --- | --- |
| id  (идентификатор автора в базе данных | second\_name (фамилия) | first\_name  (имя) | third\_name  (отчество) |
| 1 | Толстой | Лев | Николаевич |
| 2 | Пушкин | Александр | Сергеевич |
| 3 | Достоевский | Фёдор | Михайлович |
| 4 | Чехов | Антон | Павлович |
| 5 | Гоголь | Николай | Васильевич |

Таблица 6 – dataset таблицы book

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id  (иденти-  фикатор книги в базе данных) | name  (название книги) | author\_id  (иденти-  фикатор автора в базе данных) | deposit  (размер депозита) | cost\_per\_day  (стоимость за день аренды книги) | comment  (комментарий о состоянии книги) |
| 1 | Война и мир | 1 | 1000 | 10 | Отличное состояние |
| 2 | Анна Каренина | 1 | 500 | 20 | Хорошее состояние |
| 3 | Детство | 1 | 1500 | 30 | Новая книга |
| 4 | Юность | 1 | 800 | 40 | Слегка изношенная |
| 5 | Отрочество | 1 | 1200 | 50 | Идеальное состояние |
| 6 | Медный всадник | 2 | 1000 | 10 | Идеальное состояние |
| 7 | Капитанская дочка | 2 | 800 | 20 | Отличное состояние |
| 8 | Дубровский | 2 | 1500 | 30 | Хорошее состояние |
| 9 | Сказка о царе Салтане | 2 | 1200 | 40 | Новая книга |
| 10 | Сказка о рыбаке и рыбке | 2 | 1000 | 50 | Слегка изношенная |
| 11 | Преступление и наказание | 3 | 1000 | 10 | Отличное состояние |
| 12 | Братья Карамазовы | 3 | 500 | 20 | Хорошее состояние |
| 13 | Бесы | 3 | 1500 | 30 | Новая книга |
| 14 | Бедные люди | 3 | 800 | 40 | Слегка изношенная |
| 15 | Двойник | 3 | 1200 | 50 | Идеальное состояние |
| 16 | Рассказы | 4 | 1000 | 10 | Отличное состояние |
| 17 | Человек в футляре | 4 | 800 | 20 | Хорошее состояние |
| 18 | Хамелеон | 4 | 1500 | 30 | Новая книга |
| 19 | Дама с собачкой | 4 | 1200 | 40 | Слегка изношенная |
| 20 | Палата №6 | 4 | 1000 | 50 | Идеальное состояние |
| 21 | Мёртвые души | 5 | 1000 | 10 | Отличное состояние |
| 22 | Тарас Бульба | 5 | 500 | 20 | Хорошее состояние |
| 23 | Ночь перед Рождеством | 5 | 1500 | 30 | Новая книга |
| 24 | Вий | 5 | 800 | 40 | Слегка изношенная |
| 25 | Нос | 5 | 1200 | 50 | Идеальное состояние |

Таблица 7 – dataset таблицы book\_genre

|  |  |
| --- | --- |
| book\_id (идентификатор книги в базе данных) | genre\_id  (идентификатор жанра в базе данных) |
| 1 | 7 |
| 1 | 10 |
| 2 | 10 |
| 2 | 2 |
| 3 | 10 |
| 3 | 7 |
| 4 | 10 |
| 4 | 7 |
| 5 | 10 |
| 5 | 7 |
| 6 | 6 |
| 7 | 10 |
| 8 | 10 |
| 9 | 12 |
| 10 | 12 |
| 11 | 10 |
| 12 | 10 |
| 12 | 14 |
| 13 | 10 |
| 13 | 14 |
| 14 | 9 |
| 15 | 9 |
| 15 | 13 |
| 16 | 9 |
| 17 | 9 |
| 18 | 9 |
| 19 | 9 |
| 20 | 9 |
| 21 | 6 |
| 21 | 11 |
| 22 | 9 |
| 23 | 9 |
| 23 | 5 |
| 24 | 9 |
| 24 | 5 |
| 25 | 9 |
| 25 | 11 |
|  |  |

Таблица 8 - dataset таблицы category

|  |  |  |
| --- | --- | --- |
| id  (идентификатор категории в базе данных) | name  (название категории) | discount  (размер скидки в %) |
| 1 | Школьник | 15 |
| 2 | Студент | 10 |
| 3 | Пенсионер | 30 |
| 4 | Стандарт | 0 |

Таблица 9 – dataset таблицы customer\_category

|  |  |
| --- | --- |
| customer\_id  (идентификатор клиента в базе данных) | category\_id  (идентификатор категории в базе данных) |
| 1 | 3 |
| 2 | 4 |
| 3 | 4 |
| 4 | 4 |
| 5 | 4 |
| 6 | 3 |
| 7 | 2 |
| 8 | 2 |
| 9 | 1 |
| 10 | 4 |

Таблица 10 – dataset таблицы customer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id  (иденти-фикатор клиента в базе данных) | second\_name  (фамилия) | first\_name  (имя) | third\_name  (отчество) | address  (адрес проживания) | contacts  (контакты клиента) |
| 1 | Иванов | Иван | Иванович | ул. Ленина, 10 | +7 (912) 345-67-89 |
| 2 | Петров | Петр | Петрович | ул. Пушкина, 5 | +7 (923) 456-78-90 |
| 3 | Сидорова | Анна | Викторовна | пр. Гагарина, 20 | +7 (934) 567-89-01 |
| 4 | Смирнов | Алексей | Игоревич | ул. Московская, 15 | +7 (945) 678-90-12 |
| 5 | Козлова | Мария | Сергеевна | пр. Ленинский, 30 | +7 (956) 789-01-23 |
| 6 | Николаев | Дмитрий | Анатольевич | ул. Советская, 25 | +7 (967) 890-12-34 |
| 7 | Васильева | Екатерина | Александровна | пр. Победы, 12 | +7 (978) 901-23-45 |
| 8 | Морозов | Игорь | Владимирович | ул. Зеленая, 8 | +7 (989) 012-34-56 |
| 9 | Федоров | Артем | Олегович | ул. Пролетарская, 18 | +7 (990) 123-45-67 |
| 10 | Кузнецова | Анастасия | Павловна | пр. Свердлова, 7 | +7 (901) 234-56-78 |

Таблица 11 – dataset таблицы genre

|  |  |
| --- | --- |
| id  (идентификатор жанра в базе данных) | name  (название жанра) |
| 1 | Детектив |
| 2 | Драма |
| 3 | Комедия |
| 4 | Лирика |
| 5 | Мистика |
| 6 | Поэма |
| 7 | Проза |
| 8 | Пьеса |
| 9 | Рассказ |
| 10 | Роман |
| 11 | Сатира |
| 12 | Сказка |
| 13 | Фантастика |
| 14 | Философия |
| 15 | Фэнетези |

Таблица 12 – dataset таблицы customer

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| id  (иденти-фикатор транзакции в базе данных | customer\_id  (иденти-фикатор клиента в базе данных | book\_id  (иденти-фикатор книги в базе данных | date\_out (дата выдачи книги в прокат) | date\_in  (дата возвращения книги из проката) | penalty  (штраф) | total  (итоговая стоимость проката) |
| 1 | 1 | 3 | 2023-01-10 | 2023-02-01 | 0 | 461 |
| 2 | 5 | 3 | 2023-02-15 | 2023-03-02 | 0 | 450 |
| 3 | 9 | 3 | 2023-04-01 | 2023-04-07 | 0 | 153 |
| 4 | 2 | 6 | 2023-01-11 | 2023-02-08 | 100 | 280 |
| 5 | 2 | 8 | 2023-01-16 | 2023-02-08 | 0 | 690 |
| 6 | 3 | 9 | 2023-01-18 | 2023-03-14 | 0 | 2280 |
| 7 | 3 | 25 | 2023-02-21 | 2023-02-26 | 200 | 450 |
| 8 | 4 | 1 | 2023-04-02 | 2023-05-08 | 0 | 280 |
| 9 | 10 | 1 | 2023-01-11 | 2023-02-08 | 0 | 280 |
| 10 | 6 | 20 | 2023-03-16 | 2023-03-24 | 0 | 280 |
| 11 | 7 | 21 | 2023-03-03 | 2023-03-23 | 0 | 180 |
| 12 | 5 | 7 | 2023-04-15 | 2023-05-01 | 0 | 320 |
| 13 | 6 | 14 | 2023-02-03 | 2023-02-22 | 50 | 582 |
| 14 | 7 | 13 | 2023-03-30 | 2023-04-01 | 600 | 672 |

## 3.2 Добавление данных в базу

Разработанный dataset был добавлен в таблицу при помощи SQL-запросов, указанных в «Листинге 2 – внесение данных в систему»

Листинг 2 – внесение данных в систему

-------------------------------------------------------------

-- Dataset author -------------------------------------------

INSERT INTO author (second\_name, first\_name, third\_name)

VALUES

('Толстой', 'Лев', 'Николаевич'),

('Пушкин', 'Александр', 'Сергеевич'),

('Достоевский', 'Фёдор', 'Михайлович'),

('Чехов', 'Антон', 'Павлович'),

('Гоголь', 'Николай', 'Васильевич');

-------------------------------------------------------------

-- Dataset book ---------------------------------------------

INSERT INTO book (name, author\_id, deposit, cost\_per\_day, comment)

VALUES

-- Автор 1: Толстой

('Война и мир', 1, 1000, 10, 'Отличное состояние'),

('Анна Каренина', 1, 500, 20, 'Хорошее состояние'),

('Детство', 1, 1500, 30, 'Новая книга'),

('Юность', 1, 800, 40, 'Слегка изношенная'),

('Отрочество', 1, 1200, 50, 'Идеальное состояние'),

-- Автор 2: Пушкин

('Медный всадник', 2, 1000, 10, 'Идеальное состояние'),

('Капитанская дочка', 2, 800, 20, 'Отличное состояние'),

('Дубровский', 2, 1500, 30, 'Хорошее состояние'),

('Сказка о царе Салтане', 2, 1200, 40, 'Новая книга'),

('Сказка о рыбаке и рыбке', 2, 1000, 50, 'Слегка изношенная'),

-- Автор 3: Достоевский

('Преступление и наказание', 3, 1000, 10, 'Отличное состояние'),

('Братья Карамазовы', 3, 500, 20, 'Хорошее состояние'),

('Бесы', 3, 1500, 30, 'Новая книга'),

('Бедные люди', 3, 800, 40, 'Слегка изношенная'),

('Двойник', 3, 1200, 50, 'Идеальное состояние'),

-- Автор 4: Чехов

('Рассказы', 4, 1000, 10, 'Отличное состояние'),

('Человек в футляре', 4, 800, 20, 'Хорошее состояние'),

('Хамелеон', 4, 1500, 30, 'Новая книга'),

('Дама с собачкой', 4, 1200, 40, 'Слегка изношенная'),

('Палата №6', 4, 1000, 50, 'Идеальное состояние'),

-- Автор 5: Гоголь

('Мёртвые души', 5, 1000, 10, 'Отличное состояние'),

('Тарас Бульба', 5, 500, 20, 'Хорошее состояние'),

('Ночь перед Рождеством', 5, 1500, 30, 'Новая книга'),

('Вий', 5, 800, 40, 'Слегка изношенная'),

('Нос', 5, 1200, 50, 'Идеальное состояние');

-------------------------------------------------------------

-- Dataset genre --------------------------------------------

INSERT INTO genre (name)

VALUES

('Детектив'),

('Драма'),

('Комедия'),

('Лирика'),

('Мистика'),

('Поэма'),

('Проза'),

('Пьеса'),

('Рассказ'),

('Роман'),

('Сатира'),

('Сказка'),

('Фантастика'),

('Философия'),

('Фэнтези');

-------------------------------------------------------------

-- Dataset book\_genre ---------------------------------------

INSERT INTO book\_genre (book\_id, genre\_id) VALUES

-- Автор 1: Толстой

-- Книга 1: Война и мир

(1, 10), -- Роман

(1, 7), -- Проза

-- Книга 2: Анна Каренина

(2, 10), -- Роман

(2, 2), -- Драма

-- Книга 3: Детство

(3, 10), -- Роман

(3, 7), -- Проза

-- Книга 4: Юность

(4, 10), -- Роман

(4, 7), -- Проза

-- Книга 5: Отрочество

(5, 10), -- Роман

(5, 7), -- Проза

-- Автор 2: Пушкин

-- Книга 6: Мёртвые души

(6, 6), -- Поэма

-- Книга 7: Капитанская дочка

(7, 10), -- Роман

-- Книга 8: Дубровский

(8, 10), -- Роман

-- Книга 9: Сказка о царе Салтане

(9, 12), -- Сказка

-- Книга 10: Сказка о рыбаке и рыбке

(10, 12), -- Сказка

-- Автор 3: Достоевский

-- Книга 11: Преступление и наказание

(11, 10), -- Роман

-- Книга 12: Братья Карамазовы

(12, 10), -- Роман

(12, 14), -- Философия

-- Книга 13: Бесы

(13, 10), -- Роман

(13, 14), -- Философия

-- Книга 14: Бедные люди

(14, 9), -- Рассказ

-- Книга 15: Двойник

(15, 9), -- Рассказ

(15, 13), -- Фантастика

-- Автор 4: Чехов

-- Книга 16: Рассказы

(16, 9), -- Рассказ

-- Книга 17: Человек в футляре

(17, 9), -- Рассказ

-- Книга 18: Хамелеон

(18, 9), -- Рассказ

-- Книга 19: Дама с собачкой

(19, 9), -- Рассказ

-- Книга 20: Палата №6

(20, 9), -- Рассказ

-- Автор 5: Гоголь

-- Книга 21: Мёртвые души

(21, 6), -- Поэма

(21, 11), -- Сатира

-- Книга 22: Тарас Бульба

(22, 9), -- Рассказ

-- Книга 23: Ночь перед Рождеством

(23, 9), -- Рассказ

(23, 5), -- Мистика

-- Книга 24: Вий

(24, 9), -- Рассказ

(24, 5), -- Мистика

-- Книга 25: Нос

(25, 9), -- Рассказ

(25, 11); -- Сатира

-------------------------------------------------------------

-- Dataset category -----------------------------------------

INSERT INTO category (name, discount) VALUES

('Школьник', 15),

('Студент', 10),

('Пенсионер', 30),

('Стандарт', 0);

-------------------------------------------------------------

-- Dataset customer -----------------------------------------

INSERT INTO customer (second\_name, first\_name, third\_name, address, contacts) VALUES

('Иванов', 'Иван', 'Иванович', 'ул. Ленина, 10', '+7 (912) 345-67-89'),

('Петров', 'Петр', 'Петрович', 'ул. Пушкина, 5', '+7 (923) 456-78-90'),

('Сидорова', 'Анна', 'Викторовна', 'пр. Гагарина, 20', '+7 (934) 567-89-01'),

('Смирнов', 'Алексей', 'Игоревич', 'ул. Московская, 15', '+7 (945) 678-90-12'),

('Козлова', 'Мария', 'Сергеевна', 'пр. Ленинский, 30', '+7 (956) 789-01-23'),

('Николаев', 'Дмитрий', 'Анатольевич', 'ул. Советская, 25', '+7 (967) 890-12-34'),

('Васильева', 'Екатерина', 'Александровна', 'пр. Победы, 12', '+7 (978) 901-23-45'),

('Морозов', 'Игорь', 'Владимирович', 'ул. Зеленая, 8', '+7 (989) 012-34-56'),

('Федоров', 'Артем', 'Олегович', 'ул. Пролетарская, 18', '+7 (990) 123-45-67'),

('Кузнецова', 'Анастасия', 'Павловна', 'пр. Свердлова, 7', '+7 (901) 234-56-78');

-------------------------------------------------------------

-- Dataset customer\_category --------------------------------

INSERT INTO customer\_category

VALUES

(1, 3),

(2, 4),

(3, 4),

(4, 4),

(5, 4),

(6, 3),

(7, 2),

(8, 2),

(9, 1),

(10, 4);

-------------------------------------------------------------

-- Dataset transaction --------------------------------------

INSERT INTO TRANSACTION

(customer\_id, book\_id, date\_out, date\_in,penalty,total)

VALUES

-- Книги забирали и вернули из проката

(1, 3, '2023-01-10', '2023-02-01', 0, 461),

(5, 3, '2023-02-15', '2023-03-02', 0, 450),

(9, 3, '2023-04-01', '2023-04-07', 0, 153),

(2, 6, '2023-01-11', '2023-02-08', 100, 280),

(2, 8, '2023-01-16', '2023-02-08', 0, 690),

(3, 9, '2023-01-18', '2023-03-14', 0, 2280),

(3, 25, '2023-02-21', '2023-02-26', 200, 450),

(4, 1, '2023-04-02', '2023-05-08', 300, 360),

(10, 1, '2023-01-11', '2023-02-08', 0, 280),

(6, 20, '2023-03-16', '2023-03-24', 0, 280),

(7, 21, '2023-03-03', '2023-03-23', 0, 180),

(5, 7, '2023-04-15', '2023-05-01', 0, 320),

(6, 14, '2023-02-03', '2023-02-22', 50, 582),

(7, 13, '2023-03-30', '2023-04-01', 600, 672);

INSERT INTO TRANSACTION

(customer\_id, book\_id, date\_out)

VALUES

-- Книги еще не вернули из проката

(8, 11, '2023-04-08'),

(9, 23, '2023-05-11'),

(10, 2, '2023-02-06');

UPDATE book SET available = FALSE

WHERE id = 11 or id = 23 or id = 2;

-------------------------------------------------------------

## 3.3 Запросы к базе данных

Для работы с базой данных были разработаны следующие запросы:

1. Вывод всех названий книг. Код запроса указан в «Листинге 3 – запрос 1»;

Листинг 3 – запрос 1

SELECT name FROM book;

1. Вывод размера скидки для каждого клиента. Код запроса указан в «Листинге 4 – запрос 2»;

Листинг 4 – запрос 2

SELECT second\_name, first\_name, third\_name, discount

FROM customer, customer\_category, category

WHERE customer\_id = customer.id AND

category\_id = category.id;

1. Вывод контактов клиента, названия книги для каждой транзакции в базе данных. Код запроса указан в «Листинге 5 – запрос 3»;

Листинг 5 – запрос 3

SELECT contacts, book.name, date\_out, date\_in, penalty, total

FROM transaction, customer, book

WHERE customer\_id = customer.id AND

book\_id = book.id;

1. Вывод суммы всех транзакций. Код запроса указан в «Листинге 6 – запрос 4»;

Листинг 6 – запрос 4

SELECT SUM(total) FROM transaction

WHERE total IS NOT NULL;

1. Вывод суммы транзакций каждого клиента. Код запроса указан в «Листинге 7 – запрос 5»;

Листинг 7 – запрос 5

SELECT second\_name, first\_name, third\_name, SUM(total)

FROM transaction, customer

WHERE total IS NOT NULL AND

customer\_id = customer.id

GROUP BY second\_name, first\_name, third\_name;

1. Вывод суммы всех транзакций без учета скидки. Код запроса указан в «Листинге 8 – запрос 6»;

Листинг 8 – запрос 6

SELECT id, customer\_id, book\_id, date\_out, date\_in,

penalty, total, SUM(interval\_counter(date\_out, date\_in) \*

(select cost\_per\_day from book where book\_id = book.id))

OVER (PARTITION BY id) AS "total without discount"

FROM

transaction;

1. Вывод средней длительности проката книги. Код запроса указан в «Листинге 9 – запрос 7»;

Листинг 9 – запрос 7

SELECT AVG(interval\_counter(date\_out, date\_in))

FROM transaction WHERE date\_in IS NOT NULL;

1. Вывод количества клиентов для каждой категории. Код запроса указан в «Листинге 10 – запрос 8»;

Листинг 10 – запрос 8

SELECT name, COUNT(\*)

FROM category, customer\_category

WHERE category.id = category\_id

GROUP BY name, category\_id;

1. Вывод сводки о клиентах. Если у клиента более одного проката, то выводится «Частный клиент», иначе – «Обычный клиент». Код запроса указан в «Листинге 11 – запрос 9»;

Листинг 11 – запрос 9

SELECT second\_name, first\_name, third\_name,

CASE

WHEN COUNT(\*) > 1 THEN 'Частый клиент'

ELSE 'Обычный клиент'

END AS "information"

FROM customer, transaction

WHERE customer\_id = customer.id AND

date\_in IS NOT NULL

GROUP BY second\_name, first\_name, third\_name;

1. Вывод всех книг и их авторов. Код запроса указан в «Листинге 12 – запрос 10»;

Листинг 12 – запрос 10

SELECT second\_name, first\_name, third\_name, name

FROM author

JOIN (SELECT author\_id, name FROM book) b

ON author.id = b.author\_id;

1. Вывод последних 5 транзакций. Код запроса указан в «Листинге 13 – запрос 11»;

Листинг 13 – запрос 11

SELECT \* FROM transaction

ORDER BY date\_out DESC

LIMIT 5;

1. Вывод ФИО каждого пенсионера. Код запроса указан в «Листинге 14 – запрос 12»;

Листинг 14 – запрос 12

SELECT second\_name || ' ' || first\_name || ' ' || third\_name AS "ФИО"

FROM customer, category, customer\_category

WHERE customer\_id = customer.id AND

category.id = category\_id AND

category.name = 'Пенсионер';

1. Вывод информации для каждого жанра: сколько книг такого жанра, какие авторы у книг этого жанра. Код запроса указан в «Листинге 15 – запрос 13»;

Листинг 15 – запрос 13

SELECT g.name AS genre\_name,

COUNT(\*) AS book\_count,

STRING\_AGG(DISTINCT a.first\_name || ' ' || a.second\_name, ', ') AS authors

FROM transaction t

JOIN customer c ON t.customer\_id = c.id

JOIN book b ON t.book\_id = b.id

JOIN book\_genre bg ON b.id = bg.book\_id

JOIN genre g ON bg.genre\_id = g.id

JOIN author a ON b.author\_id = a.id

WHERE c.id = customer\_id

GROUP BY g.name

ORDER BY book\_count DESC;

1. Вывод любимого жанра клиента. Код запроса указан в «Листинге 16 – запрос 14»;

Листинг 16 – запрос 14

SELECT g.name AS favorite\_genre

FROM genre g

JOIN book\_genre bg ON g.id = bg.genre\_id

JOIN book b ON bg.book\_id = b.id

JOIN transaction t ON b.id = t.book\_id

WHERE t.customer\_id = 1

GROUP BY g.name

ORDER BY COUNT(\*) DESC

LIMIT 1;

1. Вывод книг, которые ни разу не брали в прокат. Код запроса указан в «Листинге 17 – запрос 15»;

Листинг 17 – запрос 15

SELECT name

FROM book

WHERE

id NOT IN (SELECT book\_id FROM transaction)

1. Вывод информации для каждой книги: сколько раз ее брали в прокат. Код запроса указан в «Листинге 18 – запрос 16»;

Листинг 18 – запрос 16

SELECT b.name AS book\_name, COUNT(\*) AS rent\_count

FROM book b

JOIN transaction t ON b.id = t.book\_id

GROUP BY b.name

ORDER BY rent\_count DESC;

1. Вывод суммы транзакций для каждой книги. Код запроса указан в «Листинге 19 – запрос 17»;

Листинг 19 – запрос 17

SELECT b.name AS book\_name, SUM(t.total) AS total\_income

FROM book b

JOIN transaction t ON b.id = t.book\_id

WHERE t.date\_in IS NOT NULL

GROUP BY b.name

ORDER BY total\_income DESC;

1. Вывод списка клиентов, кто сейчас арендует книгу. Код запроса указан в «Листинге 20 – запрос 18»;

Листинг 20 – запрос 18

SELECT second\_name, first\_name, third\_name, contacts

FROM customer

JOIN transaction t ON customer.id = t.customer\_id

WHERE date\_in IS NULL;

1. Вывод всех доступных для аренды книг. Код запроса указан в «Листинге 21 – запрос 19»;

Листинг 21 – запрос 19

SELECT name

FROM book

WHERE available = TRUE;

1. Вывод количества книг для каждого автора. Код запроса указан в «Листинге 22 – запрос 20»;

Листинг 22 – запрос 20

SELECT second\_name, first\_name, third\_name, COUNT(b.id) AS book\_count

FROM author

JOIN book b ON author.id = b.author\_id

GROUP BY author.id, first\_name, second\_name;

## 3.4 Тестирование функций и триггеров

Для работы с базой данных были разработаны следующие функции и триггеры:

1. функция open\_transaction (customer\_contacts character varying (18), book\_name character varying(100)), которая используется для создания новой записи о выдаче книги в прокат. Функция автоматически выставляет время открытия транзакции на текущую дату. Код функции и ее инициализация указаны в «Листинге 23 – функция open\_transaction»;

Листинг 23 – функция open\_transaction

CREATE OR REPLACE PROCEDURE open\_transaction

(

customer\_contacts character varying(18),

book\_name character varying(100)

)

AS $$

BEGIN

INSERT INTO transaction

(customer\_id, book\_id)

VALUES

((SELECT id FROM customer WHERE contacts = contacts\_formatter(customer\_contacts)),

(SELECT id FROM book WHERE name = book\_name));

END;

$$ LANGUAGE PLPGSQL;

1. функция close\_transaction(customer\_contacts character varying(18), book\_name character varying(100)) и close\_transaction (customer\_contacts character varying(18), book\_name character varying(100), customer\_penalty real, new\_comment text)), которая используется для закрытия транзакции в ситуации, когда клиент вернул книгу из проката. Функция автоматически подсчитывает дату закрытия транзакции и заносит эти данные в базу данных. Использована перегрузка функции, чтобы предусмотреть варианты возвращения книги в изначальном состоянии, или при порче книги. Код функции и ее инициализация указаны в «Листинге 24 – функция close\_transaction»;

Листинг 24 – функция close\_transaction

-- Если нет штрафа за состояние книги

CREATE OR REPLACE PROCEDURE close\_transaction

(

customer\_contacts character varying(18),

book\_name character varying(100)

)

AS $$

BEGIN

UPDATE transaction SET date\_in = CURRENT\_DATE

WHERE date\_in IS NULL AND

customer\_id = (SELECT id FROM customer

WHERE contacts = contacts\_formatter(customer\_contacts)) AND

book\_id = (SELECT id FROM book WHERE name = book\_name);

END;

$$ LANGUAGE plpgsql;

-- если есть штраф за состояние книги

CREATE OR REPLACE PROCEDURE close\_transaction

(

customer\_contacts character varying(18),

book\_name character varying(100),

customer\_penalty real,

new\_comment text

)

AS $$

BEGIN

UPDATE transaction SET date\_in = CURRENT\_DATE, penalty = customer\_penalty

WHERE date\_in IS NULL AND

customer\_id = (SELECT id FROM customer

WHERE contacts = contacts\_formatter(customer\_contacts)) AND

book\_id = (SELECT id FROM book WHERE name = book\_name);

UPDATE book SET comment = new\_comment WHERE name = book\_name;

END;

$$ LANGUAGE plpgsql;

1. функция contacts\_formatter (contacts character varying(18)), которая используется другими функциями и триггерами для приведения вводимого номера телефона к необходимому формату для хранения в базе данных. Функция преобразует ХХХХХХХХХХХ в +Х (ХХХ) ХХХ-ХХ-ХХ. Код функции и ее инициализация указаны в «Листинге 25 – функция contacts\_formatter»;

Листинг 25 – функция contacts\_formatter

CREATE OR REPLACE FUNCTION contacts\_formatter(contacts character varying(18))

RETURNS character varying(18)

AS $$

DECLARE

formated\_contacts character varying(18);

BEGIN

formated\_contacts = CONCAT(

'+', substr (contacts, 1, 1),

' (', substr (contacts, 2, 3), ') ',

substr (contacts, 5, 3), '-',

substr (contacts, 8, 2), '-',

substr( contacts, 10, 2));

RETURN formated\_contacts;

END;

$$ LANGUAGE plpgsql;

1. функция interval\_counter (date\_o date, date\_i date), которая используется другими функциями и триггерами для подсчета длительности проката книги в днях. Код функции и ее инициализация указаны в «Листинге 26 – функция interval\_counter»;

Листинг 26 – функция interval\_counter

CREATE OR REPLACE FUNCTION interval\_counter(date\_o date, date\_i date)

RETURNS integer

AS $$

DECLARE

days integer;

BEGIN

days = (EXTRACT(EPOCH FROM AGE(date\_i, date\_o))/60/60/24)::integer;

RETURN days;

END

$$ LANGUAGE plpgsql;

1. триггер auto\_category, который автоматически создает запись о категории нового клиента при внесении новой записи в таблицу customer. Код триггера и его инициализация указаны в «Листинге 27 – триггер auto\_category»;

Листинг 27 – триггер auto\_category

CREATE OR REPLACE FUNCTION auto\_category()

RETURNS TRIGGER

AS $$

BEGIN

INSERT INTO customer\_category

VALUES (NEW.id, 4);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER auto\_category

AFTER INSERT ON customer

FOR EACH ROW EXECUTE FUNCTION auto\_category();

1. триггер auto\_contacts\_formatter, который автоматически применяет функцию contacts\_formatter(contacts character varying(18)) к значению атрибута contacts в новой записи в таблице customer. Код триггера и его инициализация указаны в «Листинге 28 – триггер auto\_contacts\_formatter»;

Листинг 28 – триггер auto\_contacts\_formatter

CREATE OR REPLACE FUNCTION auto\_contacts\_formater()

RETURNS TRIGGER

AS $$

BEGIN

NEW.contacts = contacts\_formatter(NEW.contacts);

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER auto\_contacts\_formater

BEFORE INSERT ON customer

FOR EACH ROW

EXECUTE FUNCTION auto\_contacts\_formater();

1. триггер auto\_unique\_contacts\_checker, который проверяет при создании новой записи в таблицу customer уникальность данных, внесенных в атрибут contacts. Если такой номер телефона уже зарегистрирован, то в регистрации клиенту будет отказано. Код триггера и его инициализация указаны в «Листинге 29 – триггер auto\_unique\_contacts\_checker»;

Листинг 29 – триггер auto\_unique\_contacts\_checker

CREATE OR REPLACE FUNCTION unique\_contacts\_checker()

RETURNS TRIGGER

AS $$

BEGIN

IF NEW.contacts IN (SELECT contacts FROM customer) THEN

RAISE EXCEPTION 'Уже есть клиент с таким номером телефона!';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER auto\_unique\_contacts\_checker

BEFORE INSERT OR UPDATE ON customer

FOR EACH ROW

EXECUTE FUNCTION unique\_contacts\_checker();

1. триггер book\_status\_changer, который при создании или обновлении записи в таблице transaction меняет значения атрибута available на противоположный, чтобы обеспечить согласованность данных в базе данных автоматически. Код триггера и его инициализация указаны в «Листинге 30 – триггер book\_status\_changer»;

Листинг 30 – триггер book\_status\_changer

CREATE OR REPLACE FUNCTION book\_status\_changer()

RETURNS TRIGGER

AS $$

BEGIN

IF NEW.date\_in IS NOT NULL THEN

UPDATE book SET available = TRUE WHERE id = NEW.book\_id;

ELSE

UPDATE book SET available = FALSE WHERE id = NEW.book\_id;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER book\_status\_changer

BEFORE INSERT OR UPDATE ON transaction

FOR EACH ROW

EXECUTE FUNCTION book\_status\_changer();

1. триггер total\_counter, который при обновлении записи в таблице transaction автоматически пересчитывает стоимость транзакции, записывает ее в базу данных и выводит сумму в сообщении для пользователя базы данных. Код триггера и его инициализация указаны в «Листинге 31 – триггер total\_counter»;

Листинг 31 – триггер total\_counter

CREATE OR REPLACE FUNCTION total\_counter()

RETURNS TRIGGER

AS $$

DECLARE

discount\_size real;

BEGIN

discount\_size := 1 - (SELECT(discount/100)

FROM customer, customer\_category, category

WHERE customer.id = NEW.customer\_id AND

customer.id = customer\_category.customer\_id AND

category.id = customer\_category.category\_id);

IF NEW.date\_in IS NOT NULL THEN

NEW.total = interval\_counter(NEW.date\_out, NEW.date\_in) \*(SELECT cost\_per\_day from book where id = NEW.book\_id);

NEW.total = FLOOR(NEW.total\*discount\_size) + NEW.penalty;

RAISE NOTICE 'Итоговая стоимость составит: % рублей', NEW.total;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER total\_counter

BEFORE UPDATE ON transaction

FOR EACH ROW

EXECUTE FUNCTION total\_counter();

Далее было проведено тестирование функций на новых наборах данных, был описан ожидаемый результат и фактический результат и сделан вывод о работоспособности функции.

1. функция функция open\_transaction (customer\_contacts character varying (18), book\_name character varying(100)).

Входные данные: «79123456789», «Тарас Бульба»

Ожидаемый результат: появление в таблице transaction новой записи с текущей датой (на момент проведения тестов 29.05.2023) в поле date\_out, значения 22 в поле book\_id, значения 1 в поле customer\_id.

Фактический результат представлен на «Рисунке 4 – тестирование функции open\_transaction».

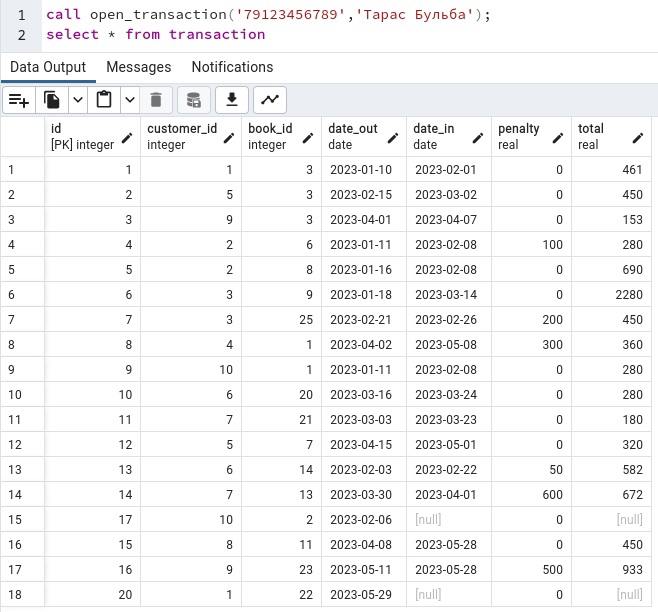


Рисунок 4 – тестирование функции open\_transaction

Вывод о работе функции: функция работает исправно;

1. функция close\_transaction(customer\_contacts character varying(18), book\_name character varying(100)) и close\_transaction (customer\_contacts character varying(18), book\_name character varying(100), customer\_penalty real, new\_comment text)).

Входные данные: воспользуемся уже открытыми транзакциями. Данные для возврата без штрафа: «79012345678», «Анна Каренина».

Ожидаемый результат: обновление записи с id = 15, должно появиться значение в столбце date\_in = 29.05.2023 – текущая дата на момент тестирования.

Результат тестирования приведен на «Рисунке 5 – тестирование close\_transaction 1».

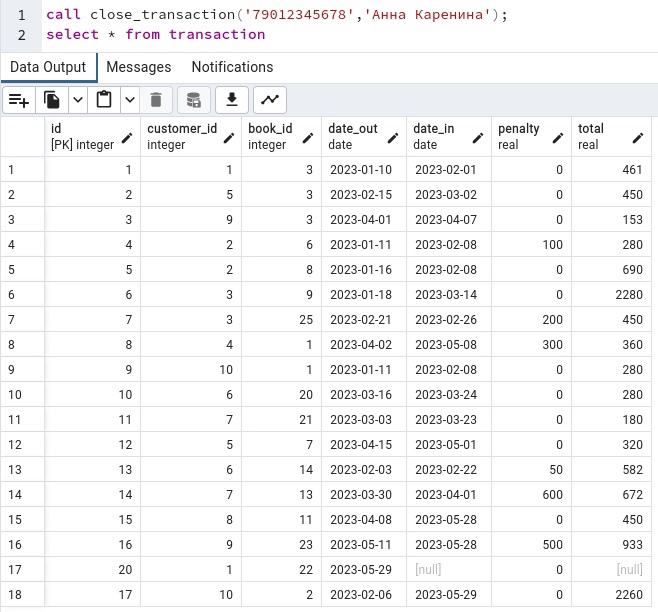


Рисунок 5 – тестирование close\_transaction 1

Вывод о работе функции: функция работает исправно.

Данные для возврата книги со штрафом: «79123456789», «Тарас Бульба», 400, «Порвана обложка».

Ожидаемый результат: date\_in = 2023-05-29, penalty = 400 в таблице transaction, comment = «Порвана обложка» в таблице book.

Фактический результат приведен в «Рисунке 6 – тестирование close\_transaction 2», «Рисунке 7 – тестирование close\_transaction 3».

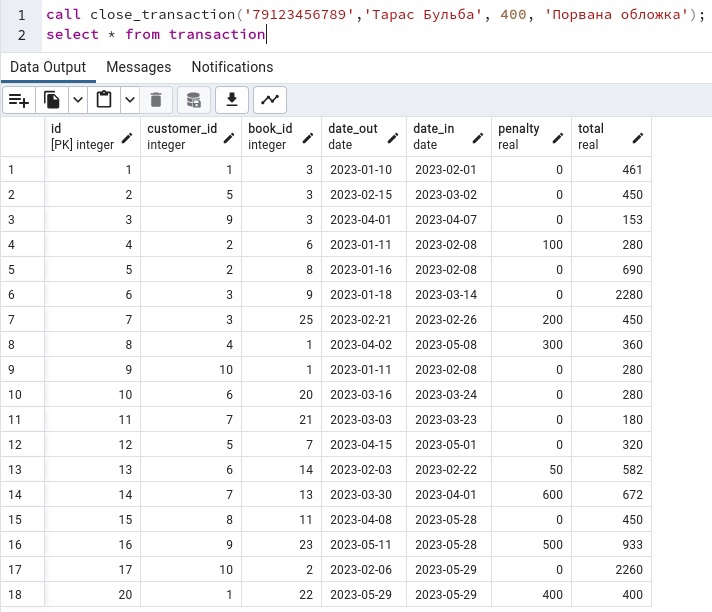


Рисунок 6 – тестирование close\_transaction 2

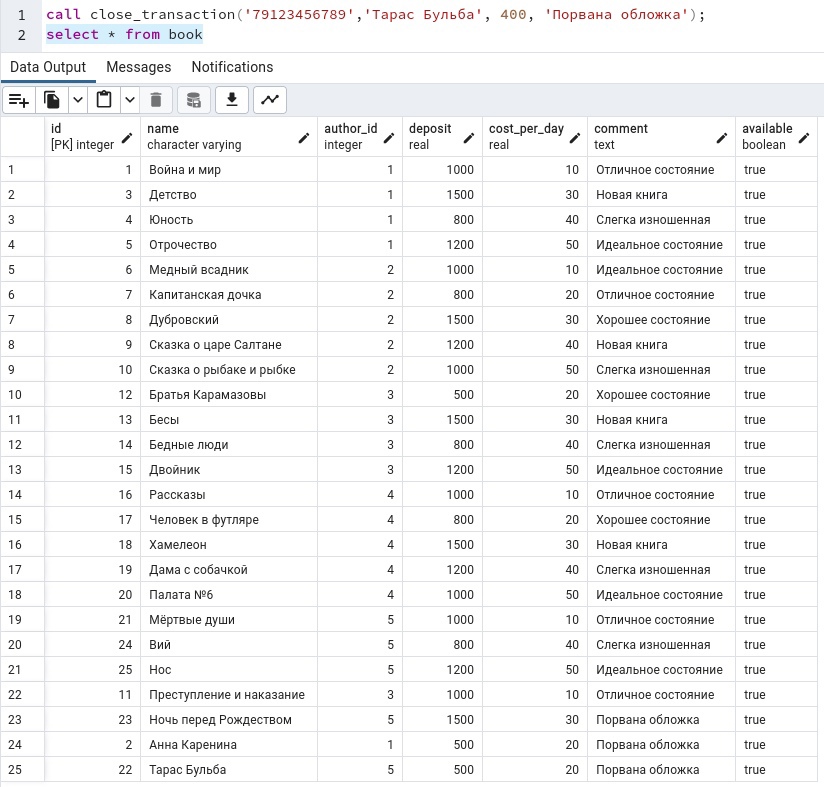


Рисунок 7 – тестирование close\_transaction 3

Вывод о работе функции: функция работает исправно;

1. Функция contacts\_formatter (contacts character varying(18)).

Входные данные: «71230001122»

Ожидаемый результат: выходные данные: +7 (123) 000-11-22

Фактический результат приведен в «Рисунке 8 – тестирование contacts\_formatter»

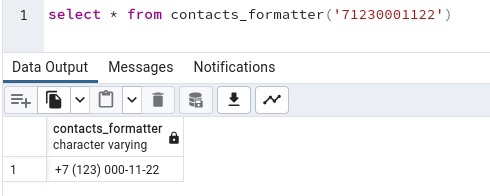


Рисунок 8 – тестирование contacts\_formatter

Вывод о работе функции: функция работает исправно;

1. функция interval\_counter (date\_o date, date\_i date).

Входные данные: «2023-01-01», «2023-01-03»

Ожидаемый результат: 2

Фактический результат приведен в «Рисунке 9 – тестирование interval\_counter 1»

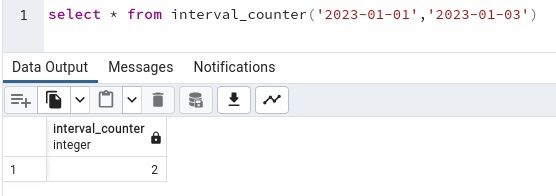


Рисунок 9 – тестирование interval\_counter 1

Входные данные: «2023-01-01», «2023-02-03»

Ожидаемый результат: 32

Фактический результат приведен в «Рисунке 10 – тестирование interval\_counter 2»

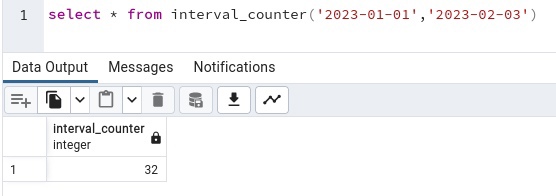


Рисунок 10 – тестирование interval\_counter 2

Вывод о работе функции: функция работает исправно;

1. триггер auto\_category.

Для проверки работы триггера необходимо создать нового пользователя.

Входные данные: создание нового пользователя с полями «Ушаков», «Матвей», «Викторович», «ул. Прянишникова, 2А», «79990001234».

Ожидаемый результат: создание новой записи в таблице customer\_category под id = 11.

Фактический результат приведен в «Рисунке 11 – тестирование auto\_category».

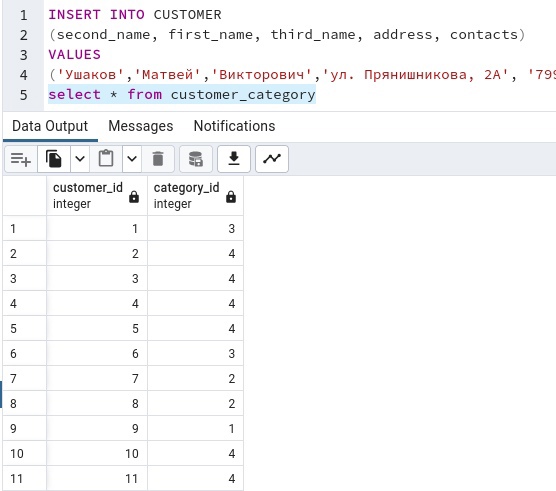


Рисунок 11 – тестирование auto\_category

Вывод о работе триггера: триггер работает корректно;

1. триггер auto\_contacts\_formatter.

При проверке предыдущей функции был создан новый пользователь, при добавлении уже сработала функция. Проверим запись в таблице customer.

Ожидаемый результат: +7 (999) 000-12-34 в поле contacts у нового клиента с фамилией Ушаков.

Фактический результат приведен в «Рисунке 12 – тестирование auto\_contacts\_formatter».

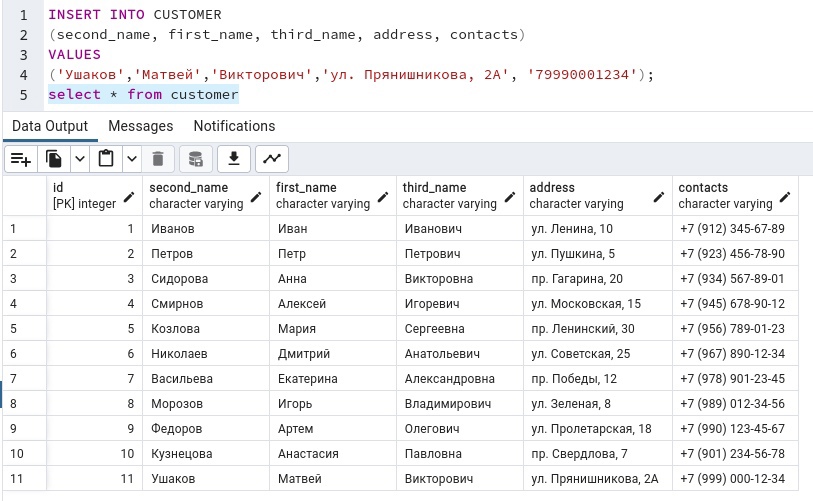


Рисунок 12 – тестирование auto\_contacts\_formatter

Вывод о работе триггера: триггер успешно выполнил свою работу;

1. триггер auto\_unique\_contacts\_checker.

Входные данные: создание нового пользователя с полями «Ушаков», «Матвей», «Викторович», «ул. Прянишникова, 2А», «79990001234», повторяющие поля существующего пользователя.

Ожидаемый результат: ошибка создания нового пользователя.

Фактический результат приведен в таблице 13 – тестирование auto\_unique\_contacts\_checker.

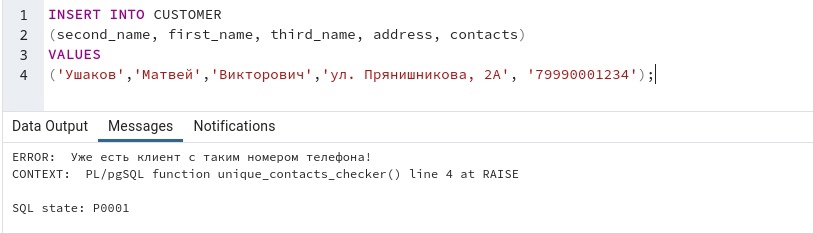


Рисунок 13 – тестирование auto\_unique\_contacts\_checker

Вывод о работе триггера: триггер работает;

1. триггер book\_status\_changer.

Входные данные: создание новой транзакции с книгой «Мёртвые души».

Ожидаемый результат: смена значения в поле available на FALSE у записи книги «Мёртвые души».

Фактический результат приведен в «Рисунке 14 – тестирование book\_status\_changer 1»

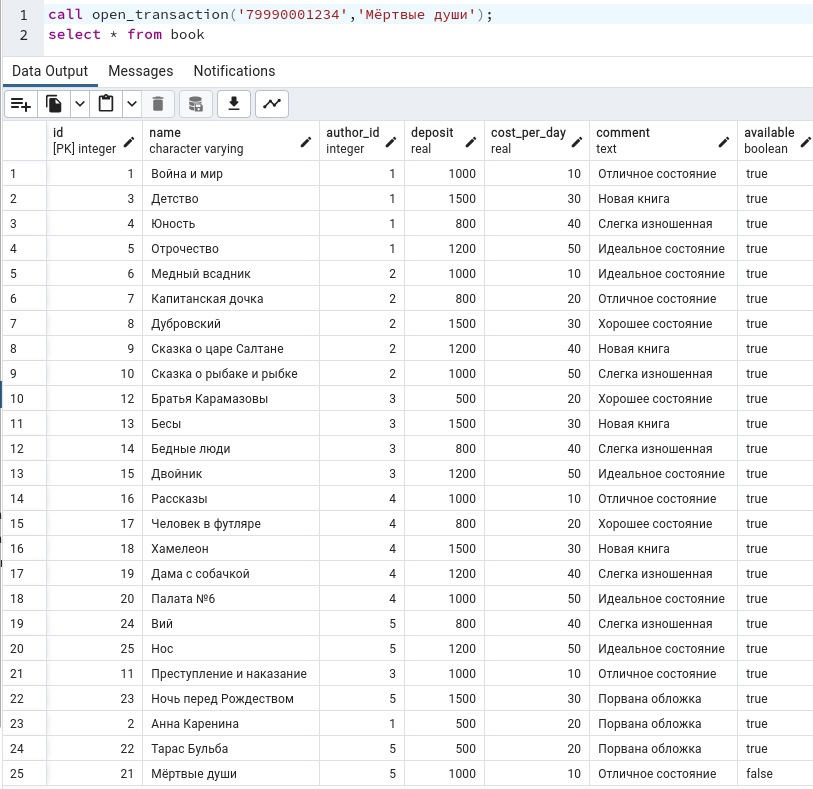


Рисунок 14 – тестирование book\_status\_changer 1

Входные данные: закрытие транзакции с книгой «Мёртвые души».

Ожидаемый результат: смена значения в поле available на FALSE у записи книги «Мёртвые души».

Фактический результат приведен в «Рисунке 14 – тестирование book\_status\_changer 2»

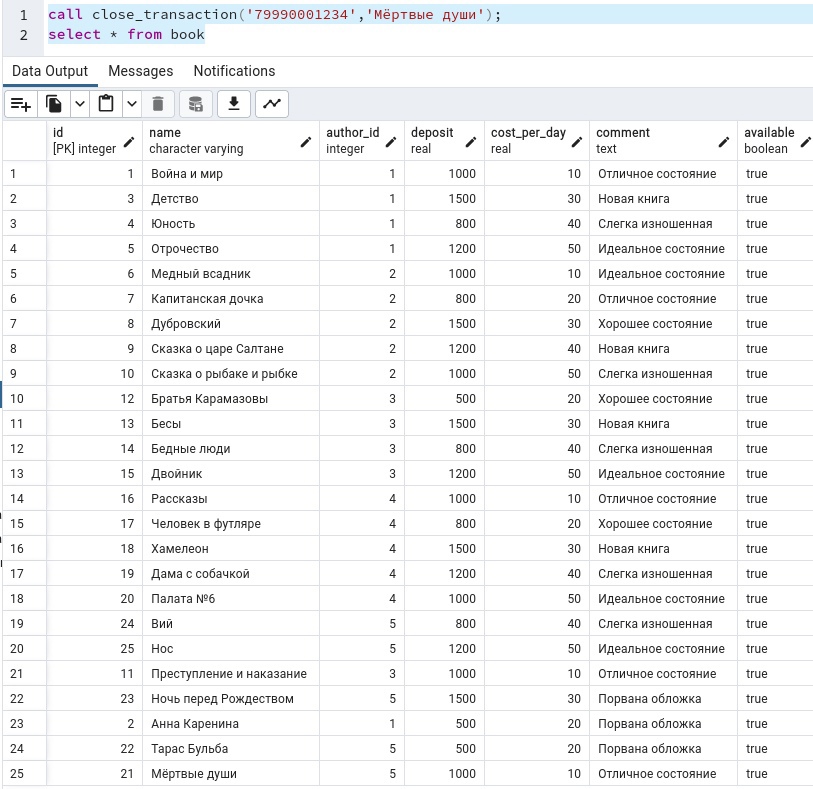


Рисунок 15 – тестирование book\_status\_changer 2

Вывод о работе триггера: триггер работает корректно;

1. триггер total\_counter.

Для проверки работы триггера создадим новую транзакцию, при закрытии транзакции должно появиться сообщение со стоимостью проката.

Входные данные: вызов функции open\_transaction(‘79990001234’, ’Мёртвые души’), close\_transaction(‘79990001234’, ’Мёртвые души’, 1000, ‘Тестирование’).

Ожидаемый результат: вывод сообщения «Итоговая стоимость составит: 1000 рублей».

Фактический результат приведен в «Рисунке 16 – тестирование total\_counter».

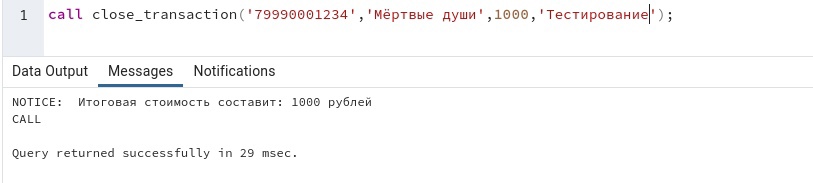


Рисунок 16 – тестирование total\_counter

Вывод о работе триггера: триггер работает корректно.

# заключение

В ходе данной курсовой работы была создана база данных с использованием системы управления базами данных PostgreSQL. Этот проект позволил мне приобрести ценные навыки в проектировании и разработке баз данных, а также в освоении SQL-запросов и работе с конкретной СУБД.

В процессе работы я изучил основные принципы моделирования данных, определил структуру базы данных, создал таблицы, определил связи между ними и заполнил их данными. Я также изучил различные возможности PostgreSQL, такие как создание и использование функций, триггеров и процедур.

Создание этой базы данных помогло мне лучше понять важность эффективной организации и хранения данных для успешного функционирования информационной системы. Я осознал значимость грамотного проектирования базы данных для обеспечения эффективности и масштабируемости системы, а также для обеспечения целостности и безопасности данных.

В процессе работы с PostgreSQL я также столкнулся с некоторыми трудностями и проблемами, но благодаря глубокому изучению документации, я смог успешно их преодолеть и получить ценный опыт.

В целом, данная курсовая работа позволила мне углубить свои знания и навыки в области баз данных и PostgreSQL в частности. Я приобрел опыт проектирования, разработки и оптимизации баз данных, что будет полезным для моей дальнейшей карьеры в области информационных технологий. PostgreSQL является мощной и гибкой СУБД, и я уверен, что полученные мною знания и навыки будут применяться мной в будущих проектах для создания надежных и эффективных баз данных.

# список использованных источников

1. ГОСТ 19.201-78. Режим доступа: https://www.swrit.ru/doc/espd/19.201-78.pdf (дата обращения: 18.04.2023).

2. Курс в ЛМС Московского политеха «Разработка технических текстов и документации». Режим доступа: https://online.mospolytech.ru/course/view.php?id=1689 (дата обращения: 18.04.2023).

3. Официальная документация разработчика PostgreSQL. Режим доступа: <https://www.postgresql.org/> (дата обращения: 29.05.2023).

4. Статья «Изучаем PostgreSQL. Часть 1. Знакомимся с архитектурой». Режим доступа: <https://habr.com/ru/companies/otus/articles/706346/> (дата обращения: 29.05.2023).

5. Информационный портал «Habr». Режим доступа: <https://habr.com/ru/all/> (дата обращения: 29.05.2023).

6 ГОСТ 7.32-2017. Режим доступа: <https://files.stroyinf.ru/Data2/1/4293742/4293742537.pdf> (дата обращения: 29.05.2023).

7. «Оптимизация запросов PostgreSQL» — Г. Домбровская, Б. Новиков, А. Бейликова. Режим доступа: <https://dmkpress.com/files/PDF/978-5-97060-963-7.pdf> (дата обращения 29.05.2023).

8. «Основы технологий баз данных: учебное пособие» — Б. Новиков, Е. Горшкова, Н. Графеева. Режим доступа: <https://postgrespro.ru/education/books/dbtech> (дата обращения: 29.05.2023).

9. «PostgreSQL изнутри» — Е. Рогов. Режим доступа: <https://postgrespro.ru/education/books/internals> (дата обращения 29.05.2023).

10. «Postgres: первое знакомство» — П. Лузанов, Е. Рогов, И. Лёвшин. Режим доступа: <https://postgrespro.ru/education/books/introbook> (дата обращения 29.05.2023).

11. «PostgreSQL. Основы языка SQL» — Е. Моргунов. Режим доступа: <https://postgrespro.ru/education/books/sqlprimer> (дата обращения 29.05.2023).

12. Руководство по подготовке курсовых работ (проектов) и выпускных квалификационных работ. Режим доступа: <http://www.skf-mtusi.ru/files/vkr/MU_DP_KP_MTUSI.pdf> (дата обращения 29.05.2023)

13. Правила оформления курсовых и дипломных работ. Режим доступа: <https://studfile.net/preview/3009968/page:5/> (дата обращения 29.05.2023).

14. Правила оформления отчета к лабораторным и курсовым работам. Режим доступа: <https://publications.hse.ru/mirror/pubs/share/direct/227003831> (дата обращения 29.05.2023).

15. Статья «Моделирование данных: обзор». Режим доступа <https://habr.com/ru/articles/556790/> (дата обращения 29.05.2023).

16. Нотация Питера Чена. Режим доступа: <https://studme.org/77222/informatika/notatsiya_pitera_chena> (дата обращения 29.05.2023).

17. Презентация «Модель «сущность-связь». Режим доступа: <https://foreva.susu.ru/courses/db/lecture3.pdf> (дата обращения 29.05.2023)

18. Триггерные процедуры в PostgreSQL. Режим доступа: <https://postgrespro.ru/docs/postgresql/9.6/plpgsql-trigger> (дата обращения 29.05.2023)

19. Форум по PosgreSQL Wiki.postgresql. Режим доступа: <https://wiki.postgresql.org/wiki/Main_Page/ru> (дата обращения 29.05.2023).