

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

**Кафедра системного програмування та спеціалізованих комп’ютерних систем**

**Лабораторна робота №1**

з дисципліни **Бази даних і засоби управління**

*на тему: “Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL”*

Виконав:

студент ІII курсу

групи КВ-01

Рябенко Б. Ю.

Перевірив:

Київ – 2022

*Метою роботи* є здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

*Завдання* роботи полягає у наступному:

1. Розробити модель «сутність-зв’язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі».
2. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL.
3. Виконати нормалізацію схеми бази даних до третьої нормальної форми (3НФ).
4. Ознайомитись із інструментарієм PostgreSQL та pgAdmin 4 та внести декілька рядків даних у кожну з таблиць засобами pgAdmin 4.

*Вимоги до ER-моделі*

1. Сутності моделі предметної галузі мають містити зв’язки типу 1:N або N:M.
2. Кількість сутностей у моделі – 3-4. Кількість атрибутів у кожній сутності: від двох до п’яти.
3. Передбачити наявність зв’язку з атрибутом.
4. Для побудови ER-діаграм використовувати одну із нотацій: Чена, “Пташиної лапки (Crow’s foot)”, UML.

**Опис предметної області «Книжковий магазин»**

Обрана предметна область передбачає опис параметрів книжок. Книжка має декілька жанрів, а також знаходиться у книжковій серії. Книжкові серії, у свою чергу, написані автором.

**Опис сутностей предметної області**

Для побудови бази даних для обраної області було виділено сутності, зображені на рисунку 1:

1. Книга (book), з атрибутами id книги, назва, ціна, кількість, дата публікації, id книжкової серії. Зберігає інформацію про книгу, серію та автора.
2. Серія (series), з атрибутами id серії, назва, id автора. Зберігає інформацію про Серію та автора.
3. Автор (author), з атрибутами id автора, ім’я. Зберігає інформацію про автора.
4. Жанр (genre), з атрибутами id жанру, назва. Зберігає інформацію про жанр.

**Опис зв’язків між сутностями предметної області**

Сутність «Книга» має зв’язок 1:N по відношенню до серії, тому що в одній серії можуть бути декілька книжок, але одна книжка не може відноситися до декількох серій.

Сутність «Серія» має зв’язок 1:N по відношенню до автора, тому що автор може мати декілька серій, але серія не може мати декілька авторів.

Сутність «Жанр» має зв’язок M:N по відношенню до книги, тому що одна книжка може мати декілька жанрів, та один і той самий жанр може бути у декількох книжок.

**Концептуальна модель предметної області**

**“Книжковий магазин”**

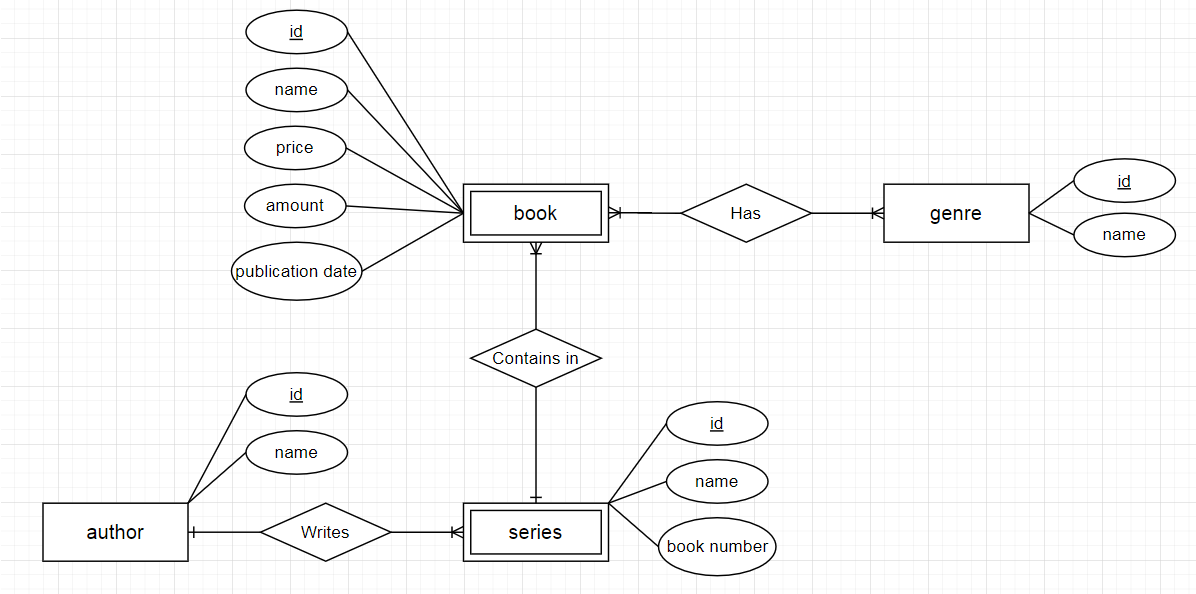
Концептуальна модель наведена на рисунку 1.

Рисунок 1 - ER-діаграма, побудована за нотацією Чена (інструмент: draw.io)

**Перетворення концептуальної моделі у схему баз даних**

1. Якщо зв'язок типу 1:N і клас приналежності сутності на стороні N є обов'язковим, то необхідно побудувати таблицю для кожної сутності. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Первинний ключ сутності на стороні 1 додається як атрибут в таблицю для сутності на стороні N. Ми можемо спостерігати даний зв’язок між сутностями author і series, book і series.

2. Якщо зв’язок типу N:M, то необхідно побудувати три таблиці - по одній для кожної сутності і одну для зв'язку. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Таблиця для зв'язку серед своїх атрибутів повинна мати ключі обох сутностей. Даний зв’язок можна спостерігати між сутностями book і genre.

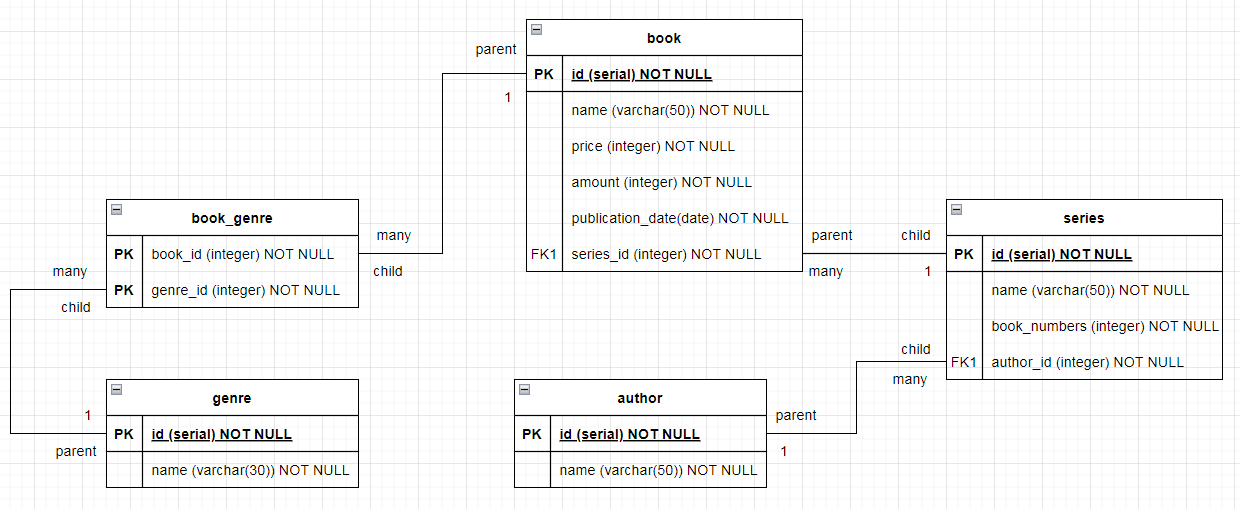
Логічну схему бази даних наведено на рисунку 2.

Рисунок 2 - Схема бази даних (інструмент: draw.io)

**Опис об’єктів бази даних у вигляді таблиці**

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| book – *містить інформацію про всі книги* | id – унікальний ідентифікатор книги  name –назва книги  price – ціна  amount – кількість книжок  publication\_date – дата публікації  series\_id – зовнішній ключ ceрії | serial  character varying  integer  integer  date  integer |
| series – *містить інформацію про серію книг* | id – унікальний ідентифікатор серії  name –назва серії  book\_numbers – кількість книжок у серії  author\_id - зовнішній ключ автора | serial  character varying  integer  integer |
| author – *містить інформацію про автора* | id – унікальний ідентифікатор автора  name –ім’я автора | serial  character varying |
| genre – *містить інформацію про жанр* | id – унікальний ідентифікатор жанру  name –назва жанру | serial  character varying |
| book\_genre –*містить інформацію про жанр книг* | book\_id – зовнішній ключ книги  genre\_id – зовнішній ключ жанру | integer  integer |

**Функціональні залежності для кожної таблиці**

BOOK:

id → name, price, amount, publication\_date, series\_id

id → name (назва книги залежить від коду книги)

id → price (ціна залежить від коду книги)

id → amount (кількість залежить від коду книги)

id → publication\_date (дата залежить від коду книги)

id → series\_id (код серії залежить від коду книжки)

SERIES:

id → name, book\_numbers, author\_id

id → name (назва серії залежить від коду серії)

id → book\_numbers (кількість книг залежить від коду серії)

id → author\_id (код автора залежить від коду серії)

AUTHOR:

id → name (ім’я автора залежить від коду автора)

GENRE:

id → name (назва жанру залежить від коду жанру)

Схема відповідає 1НФ, тому що:

1. В таблиці немає дубльованих рядків.

2. В кожній комірці зберігається атомарне (скалярне) значення.

3. В кожному стовпці зберігаються дані одного типу.

Схема відповідає 2НФ, тому що:

1. Вона відповідає 1НФ.

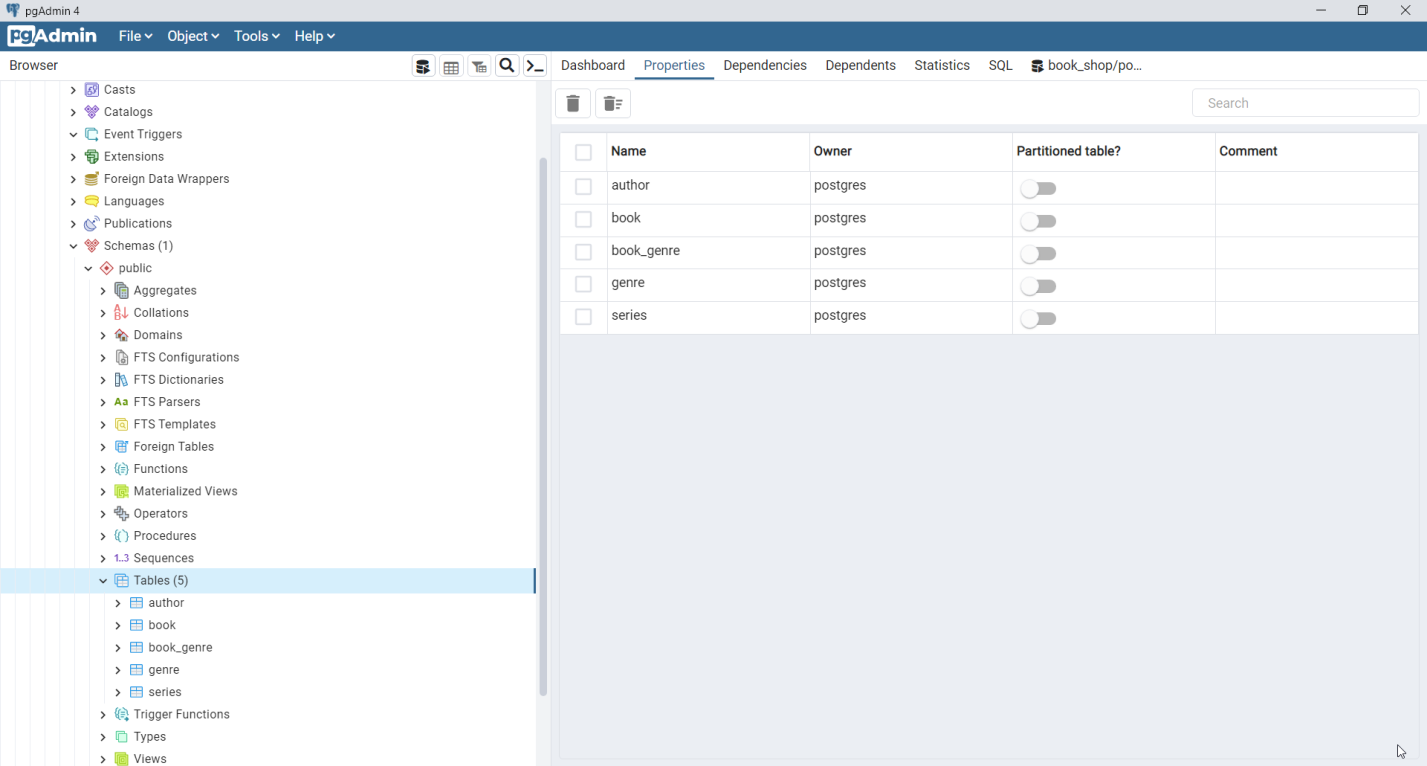
2. Має первинний ключ, а всі не ключові стовпці таблиці залежать від первинного ключа.

Схема відповідає 3НФ, тому що:

1. Вона відповідає 2НФ.

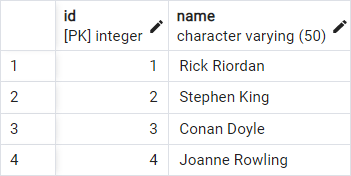
2. Всі не ключові атрибути таблиці залежать винятково від усього первинного ключа, а не його частини. Тобто кожен неключовий атрибут нетранзитивно (без посередника) залежить від первинного ключа

**Фізична модель БД «Книжковий магазин» у pgAdmin3**

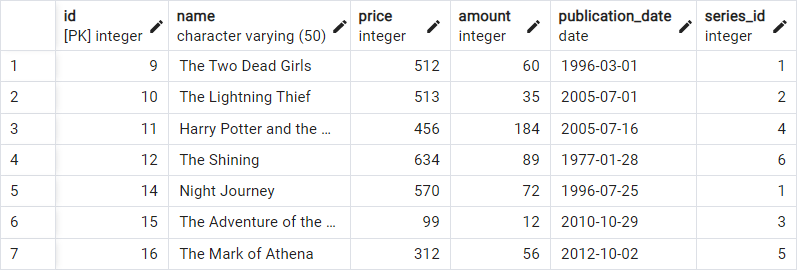


**Фотографії вмісту таблиць**

Таблиця “author”



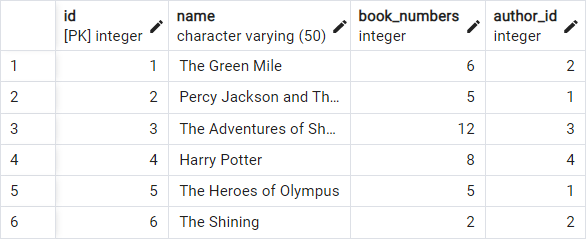
Таблиця “book”



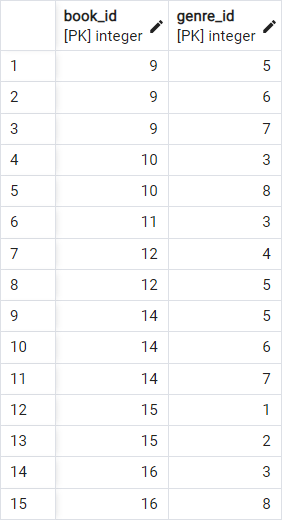
Таблиця “genre”



Таблиця “series”



Таблиця “book\_genre”



**Sql-текст опису БД** **«Книжковий магазин»**

-- Database: book\_shop

-- DROP DATABASE IF EXISTS book\_shop;

**CREATE** **DATABASE** book\_shop

**WITH**

**OWNER** = postgres

**ENCODING** = 'UTF8'

LC\_COLLATE = 'Russian\_Ukraine.1251'

LC\_CTYPE = 'Russian\_Ukraine.1251'

TABLESPACE = pg\_default

**CONNECTION** **LIMIT** = -**1**

IS\_TEMPLATE = **False**;

-- Table: public.author

-- DROP TABLE IF EXISTS public.author;

**CREATE** **TABLE** IF **NOT** **EXISTS** **public**.author

(

id integer **NOT** **NULL** **GENERATED** ALWAYS **AS** **IDENTITY** ( **INCREMENT** **1** **START** **1** **MINVALUE** **1** **MAXVALUE** **2147483647** **CACHE** **1** ),

name character varying(**50**) **COLLATE** pg\_catalog."default" **NOT** **NULL**,

**CONSTRAINT** author\_pkey **PRIMARY** **KEY** (id),

**CONSTRAINT** author\_name\_key **UNIQUE** (name)

)

TABLESPACE pg\_default;

**ALTER** **TABLE** IF **EXISTS** **public**.author

**OWNER** **to** postgres;

-- Table: public.book

-- DROP TABLE IF EXISTS public.book;

**CREATE** **TABLE** IF **NOT** **EXISTS** **public**.book

(

id integer **NOT** **NULL** **GENERATED** ALWAYS **AS** **IDENTITY** ( **INCREMENT** **1** **START** **1** **MINVALUE** **1** **MAXVALUE** **2147483647** **CACHE** **1** ),

name character varying(**50**) **COLLATE** pg\_catalog."default" **NOT** **NULL**,

price integer **NOT** **NULL**,

amount integer **NOT** **NULL**,

publication\_date date **NOT** **NULL**,

series\_id integer **NOT** **NULL**,

**CONSTRAINT** book\_pkey **PRIMARY** **KEY** (id),

**CONSTRAINT** book\_seriesid\_fkey **FOREIGN** **KEY** (series\_id)

**REFERENCES** **public**.series (id) **MATCH** **SIMPLE**

**ON** **UPDATE** **NO** ACTION

**ON** **DELETE** **CASCADE**

)

TABLESPACE pg\_default;

**ALTER** **TABLE** IF **EXISTS** **public**.book

**OWNER** **to** postgres;

-- Table: public.book\_genre

-- DROP TABLE IF EXISTS public.book\_genre;

**CREATE** **TABLE** IF **NOT** **EXISTS** **public**.book\_genre

(

book\_id integer **NOT** **NULL**,

genre\_id integer **NOT** **NULL**,

**CONSTRAINT** book\_genre\_pkey **PRIMARY** **KEY** (book\_id, genre\_id),

**CONSTRAINT** bookgenre\_bookid\_fkey **FOREIGN** **KEY** (book\_id)

**REFERENCES** **public**.book (id) **MATCH** **SIMPLE**

**ON** **UPDATE** **NO** ACTION

**ON** **DELETE** **CASCADE**,

**CONSTRAINT** bookgenre\_genreid\_fkey **FOREIGN** **KEY** (genre\_id)

**REFERENCES** **public**.genre (id) **MATCH** **SIMPLE**

**ON** **UPDATE** **NO** ACTION

**ON** **DELETE** **CASCADE**

)

TABLESPACE pg\_default;

**ALTER** **TABLE** IF **EXISTS** **public**.book\_genre

**OWNER** **to** postgres;

-- Table: public.genre

-- DROP TABLE IF EXISTS public.genre;

**CREATE** **TABLE** IF **NOT** **EXISTS** **public**.genre

(

id integer **NOT** **NULL** **GENERATED** ALWAYS **AS** **IDENTITY** ( **INCREMENT** **1** **START** **1** **MINVALUE** **1** **MAXVALUE** **2147483647** **CACHE** **1** ),

name character varying(**30**) **COLLATE** pg\_catalog."default" **NOT** **NULL**,

**CONSTRAINT** genre\_pkey **PRIMARY** **KEY** (id),

**CONSTRAINT** genre\_name\_key **UNIQUE** (name)

)

TABLESPACE pg\_default;

**ALTER** **TABLE** IF **EXISTS** **public**.genre

**OWNER** **to** postgres;

-- Table: public.series

-- DROP TABLE IF EXISTS public.series;

**CREATE** **TABLE** IF **NOT** **EXISTS** **public**.series

(

id integer **NOT** **NULL** **GENERATED** ALWAYS **AS** **IDENTITY** ( **INCREMENT** **1** **START** **1** **MINVALUE** **1** **MAXVALUE** **2147483647** **CACHE** **1** ),

name character varying(**50**) **COLLATE** pg\_catalog."default" **NOT** **NULL**,

book\_numbers integer **NOT** **NULL**,

author\_id integer **NOT** **NULL**,

**CONSTRAINT** series\_pkey **PRIMARY** **KEY** (id),

**CONSTRAINT** series\_name\_key **UNIQUE** (name),

**CONSTRAINT** series\_authorid\_fkey **FOREIGN** **KEY** (author\_id)

**REFERENCES** **public**.author (id) **MATCH** **SIMPLE**

**ON** **UPDATE** **NO** ACTION

**ON** **DELETE** **CASCADE**

)

TABLESPACE pg\_default;

**ALTER** **TABLE** IF **EXISTS** **public**.series

**OWNER** **to** postgres;