

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-02

Недоколенко Д. Р.

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.

*Зміст звіту*

1. Опис проблемного середовища;

2. Концептуальна модель предметної області;

3. Логічна модель (схема) даних БД;

4. Склад СУБД PostgreSQL;

5. Список обмежень цілісності в термінах СУБД PostgreSQL;

6. Фізична модель (схема) даних БД в pgAdmin III;

7. Приклад вмісту БД.

Посилання на github: <https://github.com/Zarrylon/db_kpi>

**Опис предметної області «Бібліотека»**

Обрана предметна область передбачає отримання читачем абонемента на книги. Читач може мати всього один абонемент, який, в свою чергу, може включати в себе багато книг. Книга має одного автора, проте автор може мати багато книг.

**Опис сутностей предметної області**

Для побудови бази даних для обраної області було виділено 4 сутності, зображені на Рисунку 1.

1. Книги (Books). Атрибути: id книг; id автора; назва; кількість сторінок; наявність. Призначення: збереження інформації про книги з перевіркою їх наявності;

2. Автор (Author). Атрибути: id автора; ім’я. Призначення: збереження інформації про авторів;

3. Абонемент (Subscription). Атрибути: id абонементу; id читача; id книг; дата початку; дата кінця. Призначення: збереження інформації про існуючі абонементи, їх початок та кінець, а також встановлює стан обраних книг як «ненаявні»;

4. Читач (Reader). Атрибути: id читача; ім’я, адреса, номер телефону. Призначення: збереження інформації про читача, який отримує абонемент.

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

Сутність «Книги» має зв’язок 1:N по відношенню до сутності «Автор», оскільки книга може мати тільки одного автора, проте той самий автор може написати більше, ніж одну книгу. Також ця сутність має зв’язок M:N по відношенню до «Абонемент». Це зумовлено тим, що одна й та сама книга в різний момент часу може входити в різні абонементи, а один абонемент може включати в себе різні книги.

Сутність «Абонемент» має зв’язок M:N по відношенню до сутності «Читачі», оскільки у одного й того самого читача в різний момент часу можуть бути різні абонементи, а абонемент можуть отримати багато читачів.

**Концептуальна модель предметної області «Бібліотека»**

Концептуальна модель наведена на Рисунку 1.

Слід також зазначити, що в даній ER-діаграмі може існувати ще один зв’язок M:N – між сутностями «Читач» та «Книги». Даний зв’язок буде зайвим, оскільки контроль над тим, які книги читач хоче взяти з собою здійснює сутність «Абонемент» через свої атрибути. Варто зазначити, що такий зв’язок мав би сенс лиш в тому випадку, коли нам потрібно зберігати інформацію про те, які книги читають в самій бібліотеці. Оскільки в справжніх бібліотеках такого не роблять, такий зв’язок не має сенсу.

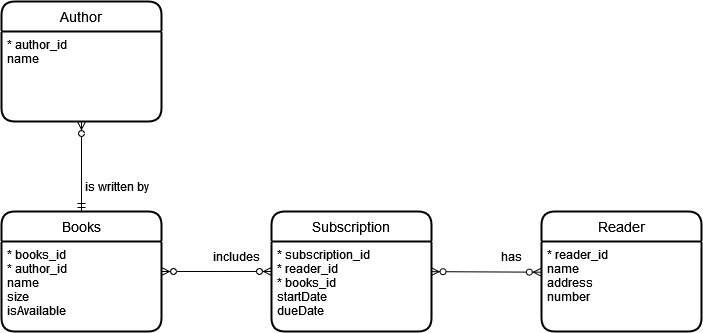


Рисунок 1 – ER-діаграма предметної галузі «Бібліотека», побудована за нотацією Crow’s foot (інструмент: draw.io)

**Перетворення концептуальної моделі у логічну схему бази даних**

Сутність «Автор» перетворено у таблицю «author».

Сутність «Книги» перетворено у таблицю «books».

Сутність «Читач» перетворено у таблицю «reader».

Зв’язок N:M між сутностями «Книги» та «Читач» зумовив появу сутності «Абонемент», яку перетворено в таблицю «subscription» із зовнішніми ключами books\_id та reader\_id.

**Логічна модель (схема) БД «Бібліотека»**

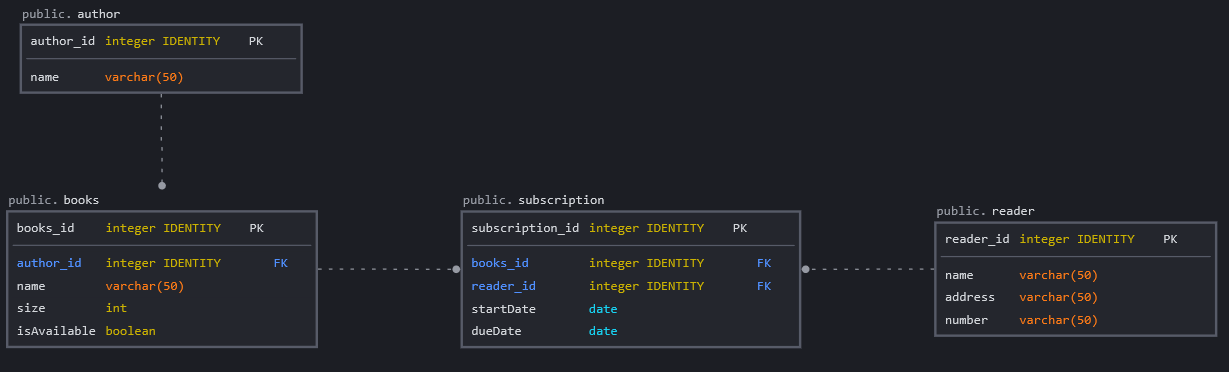
Логічну модель (схему бази даних) наведено на Рисунку 2.

Рисунок 2. – Схема бази даних (інстурмент: sqldb.com)

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

Опис сутностей та їх атрибутів наведено в Таблиця 1.

Таблиця 1 – Опис структури бази даних «Бібліотека».

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| **author** – містить дані про автора | **author\_id** – унікальний ідентифікатор автора. Не допускає NULL  **name** – ім’я автора. Не допускає NULL. | **integer** (числовий)  **character varying** (рядок) |
| **books** – містить дані про всі книги. | **books\_id** – унікальний ідентифікатор книги. Не допускає NULL  **author\_id** – унікальний ідентифікатор автора.  **name** – назва книги. Не допускає NULL.  **size** – кількість сторінок в книзі. Не допускає NULL.  **isAvailable** – ідентифікатор наявності книги. | **integer** (числовий)  **integer** (числовий)  **character varying** (рядок)  **integer** (числовий)  **boolean** (логічний) |
| **reader** – містить дані про читача. | **reader\_id** – унікальний ідентифікатор читача. Не допускає NULL  **name** – ПІБ читача. Не допускає NULL.  **address** – адреса читача. Не допускає NULL.  **number** – номер телефону читача. Не допускає NULL. | **integer** (числовий)  **character varying** (рядок)  **character varying** (рядок)  **character varying** (рядок) |
| **subscription** – містить дані про абонемент. | **subscription\_id** – унікальний ідентифікатор абонементу. Не допускає NULL.  **books\_id** – унікальний ідентифікатор книги.  **reader\_id** – унікальний ідентифікатор читача.  **startDate** – дата початку дії абонементу. Не допускає NULL.  **dueDate** – дата закінчення дії абонементу. Не допускає NULL. | **integer** (числовий)  **integer** (числовий)  **integer** (числовий)  **date** (час)  **date** (час) |

**Функціональні залежності для кожної таблиці**

Author (author\_id, name):

PK: author\_id

author\_id → name

Books (books\_id, author\_id, name, size, isAvailable):

PK: books\_id

books\_id → author\_id

books\_id → name

books\_id → size

books\_id → isAvailable

books\_id → author\_id, name, size, isAvailable

Reader (reader\_id, name, address, number):

PK: reader\_id

reader\_id → name

reader\_id → address

reader\_id → number

reader\_id → name, address, number

Subscription (subscription\_id, books\_id, reader\_id, startDate, dueDate):

PK: subscription\_id

subscription\_id → books\_id

subscription\_id → reader\_id

subscription\_id → startDate

subscription\_id → dueDate

subscription\_id → books\_id, reader\_id, startDate, dueDate

**Відповідність схеми бази даних до третьої нормальної форми**

Схема відповідає 1НФ, тому що всі комірки (атрибути) таблиці є атомарними, тобто кожна комірка містить лише одне значення, а кожний рядок є унікальним.

Схема відповідає 2НФ, тому що вона відповідає 1НФ і всі неключові атрибути функціонально залежать від цілого ключа, а не його частини.

Схема відповідає 3НФ, тому що вона відповідає 2НФ і всі неключові атрибути таблиці функціонально залежать від первинного ключа.

**Фізична модель БД «Бібліотека» у pgAdmin3**

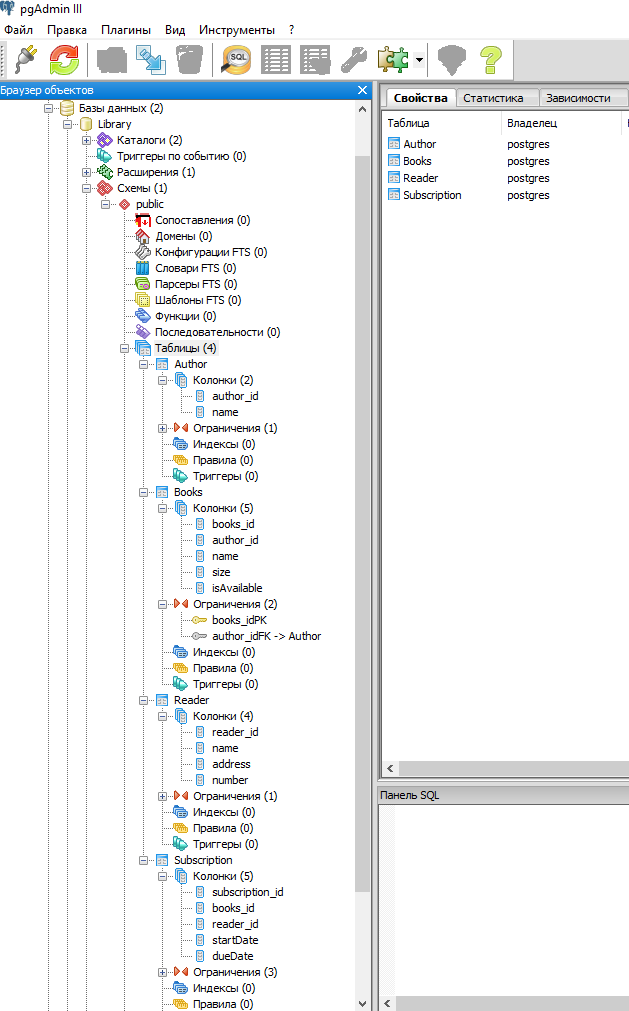


Рисунок 3. – Схема бази даних (інструмент: pgAdmin3)

**Фотографії вмісту таблиць**

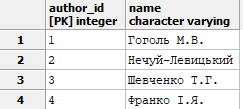


Рисунок 4.1. – Дані таблиці Author

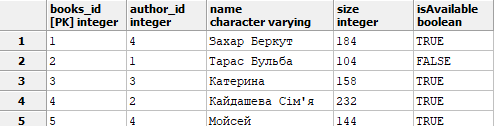


Рисунок 4.2. – Дані таблиці Books

.

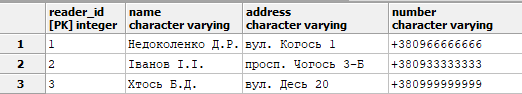


Рисунок 4.3. – Дані таблиці Reader

.

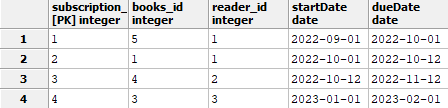


Рисунок 4.4. – Дані таблиці Subscription

**SQL-текст опису БД «Бібліотека»**

1. Library (Database):

-- Database: "Library"

-- DROP DATABASE "Library";

**CREATE** **DATABASE** "Library"

**WITH** OWNER = postgres

ENCODING = 'UTF8'

**TABLESPACE** = pg\_default

LC\_COLLATE = 'Russian\_Russia.1251'

LC\_CTYPE = 'Russian\_Russia.1251'

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

1. Author (Table):

-- Table: "Author"

-- DROP TABLE "Author";

**CREATE** **TABLE** "Author"

(

author\_id serial **NOT** **NULL**,

name **character** **varying** **NOT** **NULL**,

**CONSTRAINT** "author\_idPK" **PRIMARY** **KEY** (author\_id)

)

**WITH** (

OIDS=**FALSE**

);

**ALTER** **TABLE** "Author"

OWNER **TO** postgres;

1. Books (Table):

-- Table: "Books"

-- DROP TABLE "Books";

**CREATE** **TABLE** "Books"

(

books\_id serial **NOT** **NULL**,

author\_id serial **NOT** **NULL**,

name **character** **varying** **NOT** **NULL**,

**size** **integer** **NOT** **NULL**,

"isAvailable" **boolean** **NOT** **NULL**,

**CONSTRAINT** "books\_idPK" **PRIMARY** **KEY** (books\_id),

**CONSTRAINT** "author\_idFK" **FOREIGN** **KEY** (author\_id)

**REFERENCES** "Author" (author\_id) **MATCH** **FULL**

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

)

**WITH** (

OIDS=**FALSE**

);

**ALTER** **TABLE** "Books"

OWNER **TO** postgres;

1. Reader (Table):

-- Table: "Reader"

-- DROP TABLE "Reader";

**CREATE** **TABLE** "Reader"

(

reader\_id serial **NOT** **NULL**,

name **character** **varying** **NOT** **NULL**,

address **character** **varying** **NOT** **NULL**,

"number" **character** **varying** **NOT** **NULL**,

**CONSTRAINT** "reader\_idPK" **PRIMARY** **KEY** (reader\_id)

)

**WITH** (

OIDS=**FALSE**

);

**ALTER** **TABLE** "Reader"

OWNER **TO** postgres;

1. Subscription (Table):

-- Table: "Subscription"

-- DROP TABLE "Subscription";

**CREATE** **TABLE** "Subscription"

(

subscription\_id serial **NOT** **NULL**,

books\_id serial **NOT** **NULL**,

reader\_id serial **NOT** **NULL**,

"startDate" **date** **NOT** **NULL**,

"dueDate" **date** **NOT** **NULL**,

**CONSTRAINT** "subscription\_idPK" **PRIMARY** **KEY** (subscription\_id),

**CONSTRAINT** "books\_idFK" **FOREIGN** **KEY** (books\_id)

**REFERENCES** "Books" (books\_id) **MATCH** **FULL**

**ON** **UPDATE** **NO** **ACTION** **ON** **DELETE** **NO** **ACTION**,

**CONSTRAINT** "reader\_idFK" **FOREIGN** **KEY** (reader\_id)

**REFERENCES** "Reader" (reader\_id) **MATCH** **FULL**

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

)

**WITH** (

OIDS=**FALSE**

);

**ALTER** **TABLE** "Subscription"

OWNER **TO** postgres;