**Національний технічний університет України   
“Київський політехнічний інститут імені Ігоря Сікорського”**

Факультет прикладної математики   
Кафедра системного програмування і   
спеціалізованих комп’ютерних систем

**ЛАБОРАТОРНА РОБОТА №2**

з дисципліни   
“Бази даних і управління”

ТЕМА: **«Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL»**

Виконав студент групи КВ-03

Статечний Сергій

***Репозиторій:***

[***https://github.com/Code01KPI/Lab2***](https://github.com/Code01KPI/Lab2)

Київ – 2022

**Лабораторна робота № 2.**

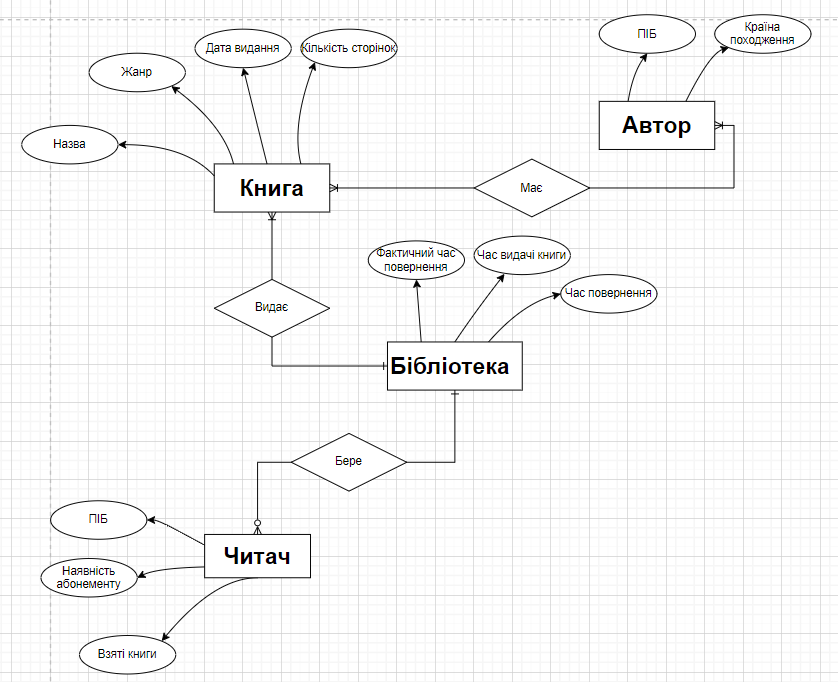
**Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL**

*Метою роботи* є здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.

*Загальне завдання* роботи полягає у наступному:

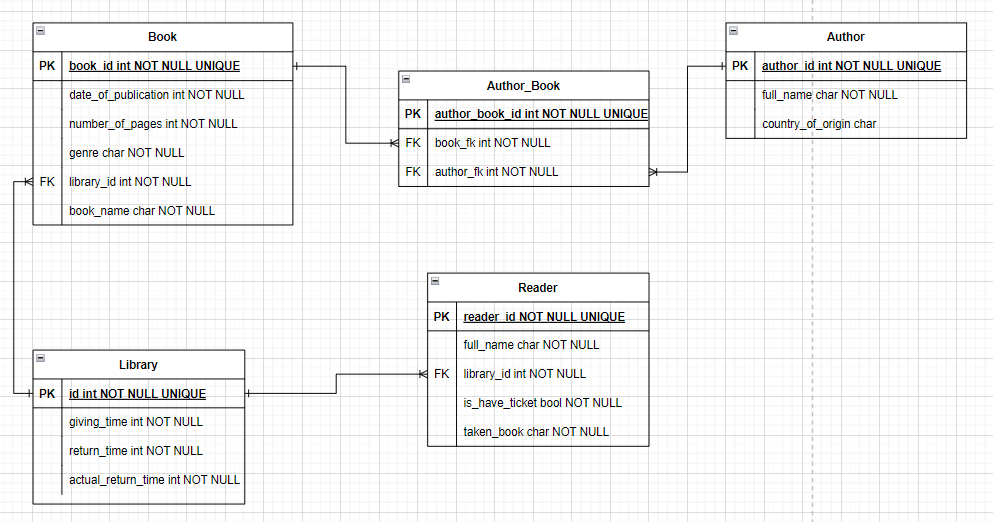
1. Реалізувати функції перегляду, внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

*ER діаграма*



*Нотація “Пташина лапка”*

*Структура бази даних*



ER модель – бібліотека(видача книг читачам)

*Опис сутностей*

1. Автор – сутність описує автора певної книги. Серед атрибутів має ПІБ(виступає в ролі id) та країну походження .
2. Книга – сутність описує певну книгу. Атрибутами є назва книги(виступає в ролі id), а також жанр, дату видання і кількість сторінок.
3. Бібліотека – сутність описує бібліотеку. Атрибутами є час видачі книги, очікуваний час повернення та фактичний час повернення книги.
4. Читач – сутність описує певного читача. Атрибутами є ПІБ(виступає в ролі id), наявність абонементу та перелік взятих книг.

*Схема меню розробленої програми*

1. *Work with data*
   1. *Insert data*
      1. *Choose table*
   2. *Update data*
      1. *Choose table*
   3. *Delete data*
      1. *Choose table*
   4. *Exit*
2. *Create data*
   1. *Choose table*
3. *Search*
   1. *Query 1*
   2. *Query 2*
   3. *Query 3*
   4. *Exit*
4. *Exit*

Рівень меню 1(1, 2, 3, 4) – вибір завдання лабораторної, яке буде виконувати розроблена програма;

Рівень меню 1.1(1.1, 1.2, 1.3, 1.4) – уведення/редагування/видалення даних з таблиці користувачем; 1.1.1, 1.2.1, 1.3.1 – вибір таблиці для маніпуляцій з даними;

Рівень меню 2.1(2.1) – вибір таблиці для автоматичного введення рандомізованиз даних;

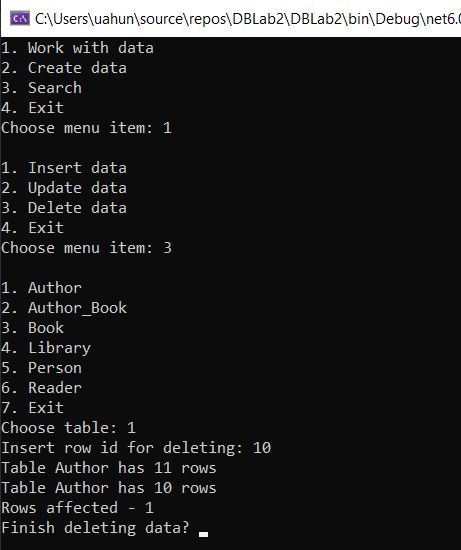
Рівень меню 3.3(3.1, 3.2, 3.3, 3.4) – вибір запиту пошуку;

*Мова програмування та використані бібліотеки*

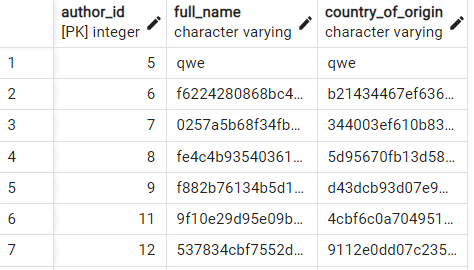
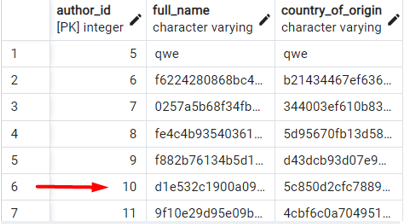
Програма реалізована на мові C#, з використанням NuGet-пакету Npgsql.

**Завдання 1**

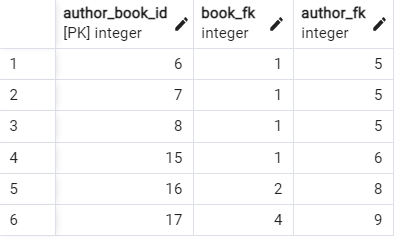
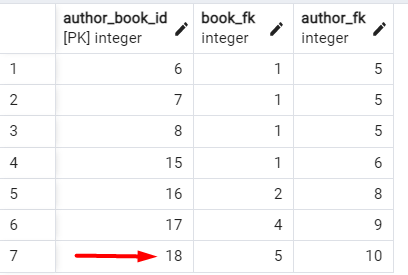
Вилучення даних з батьківської таблиці *Author*:



Батьківська таблиця *Author* до та після вилучення:

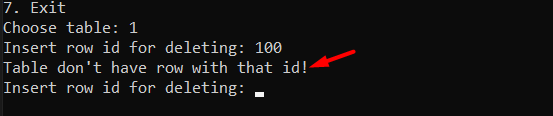


Дочірня таблиця *Author\_Book* до та після вилучення:

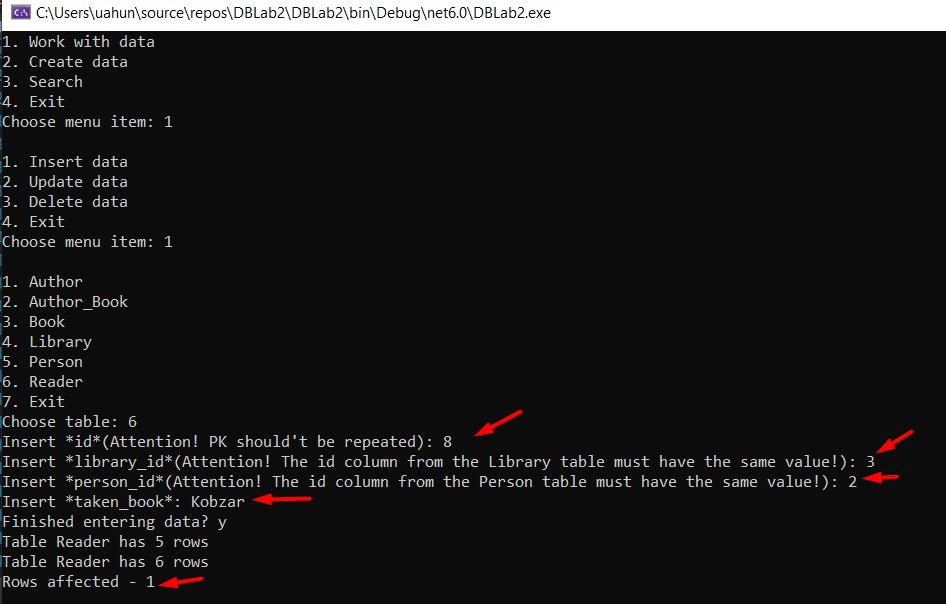


При вилучені даних з батьківської таблиці програма автоматично видаляє всі пов’язані дані з дочірніх таблиць.

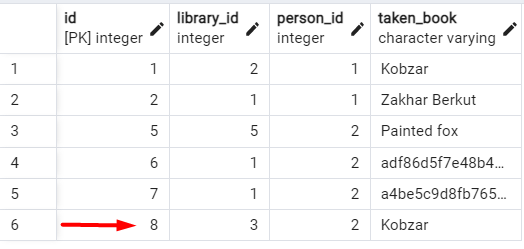
Приклад валідації даних(спроба видалення рядка якого не існує):



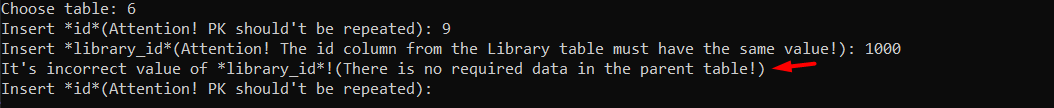
Запис даних в дочірню таблицю *Reader*:



Таблиця *Reader* після додавання даних:

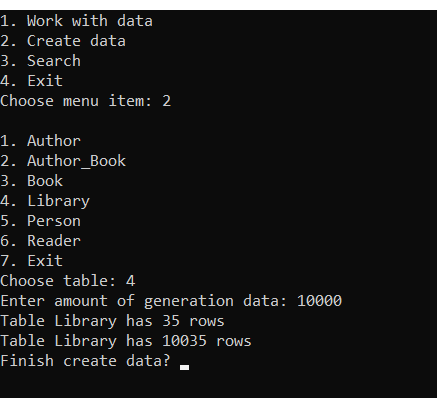


Приклад валідації даних при їх записі:

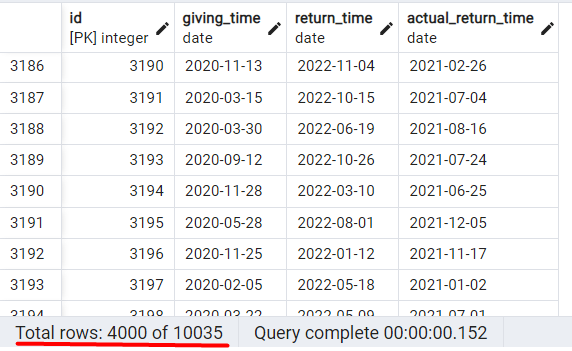


**Завдання 2**

Генерація псевдовипадкових даних таблиці *Library*:



Результат:



Копія SQL-запиту:

INSERT INTO Library(id, giving\_time, return\_time, actual\_return\_time)

SELECT generate\_series(1, 100),

timestamp '2020-01-01 00:00:00' + random() \* (timestamp '2021-01-01 00:00:00' - timestamp '2020-01-02 00:00:00'),

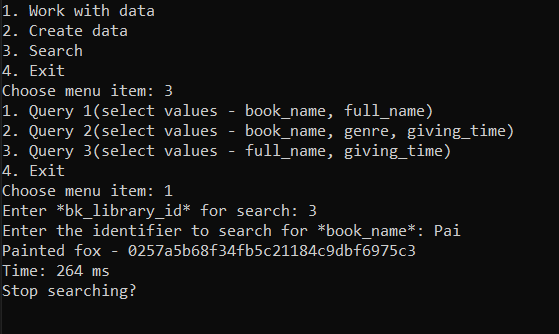
timestamp '2022-01-01 00:00:00' + random() \* (timestamp '2023-01-01 00:00:00' - timestamp '2022-01-02 00:00:00'),

timestamp '2021-01-01 00:00:00' + random() \* (timestamp '2022-01-01 00:00:00' - timestamp '2021-01-02 00:00:00');

В generate\_series() передається кількість даних яка буде згенерована.

**Завдання 3**

**Query 1**



Копія SQL-запиту 1:

SELECT B.book\_name, A.full\_name

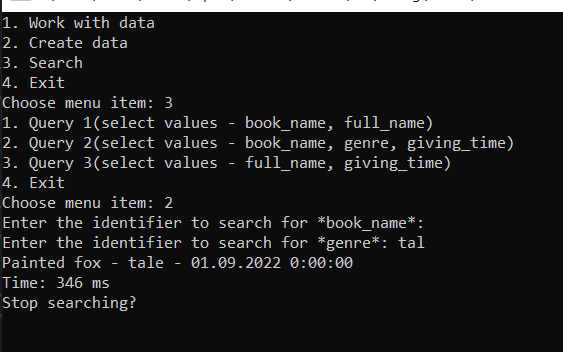
FROM Book as B, Author as A, Author\_Book as AB

WHERE AB.book\_fk = B.book\_id AND AB.author\_fk = A.author\_id AND B.bk\_library\_id = {param1} AND B.book\_name LIKE '{param2}%';

param1 – значення, яке вводить користувач для фільтрації по стовпцю *bk\_library\_id*;

param2 – шаблон рядка, для пошуку;

**Query 2**



Копія SQL-запиту 2:

SELECT B.book\_name, B.genre, L.giving\_time

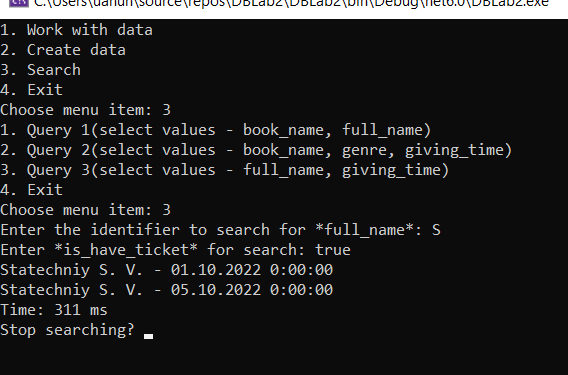
FROM Book as B, Library as L

WHERE book\_name LIKE '{param1}%' AND bk\_library\_id = id AND genre LIKE '{param2}%';

param1 – шаблон, який вводить користувач для фільтрації потрібного рядка по стовпцю *book\_name*;

param2 – шаблон, який вводить користувач для фільтрації потрібного рядка по стовпцю *genre*;

**Query 3**



Копія SQL-запиту 3:

SELECT P.full\_name, L.giving\_time

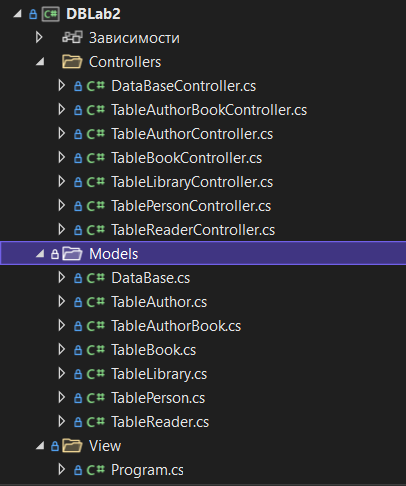
FROM public.\"Person\" as P, public.\"Library\" as L, public.\"Reader\" as R

WHERE full\_name LIKE '{param1}%' AND R.library\_id = L.id AND R.person\_id = P.person\_id AND is\_have\_ticket = {param2};

param1 – шаблон, який вводить користувач для фільтрації потрібного рядка по стовпцю *book\_name*;

param2 –значення для пошуку по стовпцю *is\_have\_ticket*;

**Завдання 4**



Модуль *Models* – містить моделі всіх таблиць бази даних, а також клас самої БД, клас має метод для підключення до заданої БД, який приймає рядок підключення в якості параметра. Також в класі є property *NpgsqlDataSource? DataSource* { get; set; }з допомогою якого можливе створення та реалізація запитів.

Класи таблиць мають всі поля конкретної таблиці, при такому підході зручно зберігати/додавати дані в задану таблицю.

Вся логіка знаходиться в модулі *Controllers.* Який має для кожної таблиці свій клас з логікою, що наслідується від головного класу *DataBaseController*.