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

Київ – 2022

Лабораторна робота № 2.

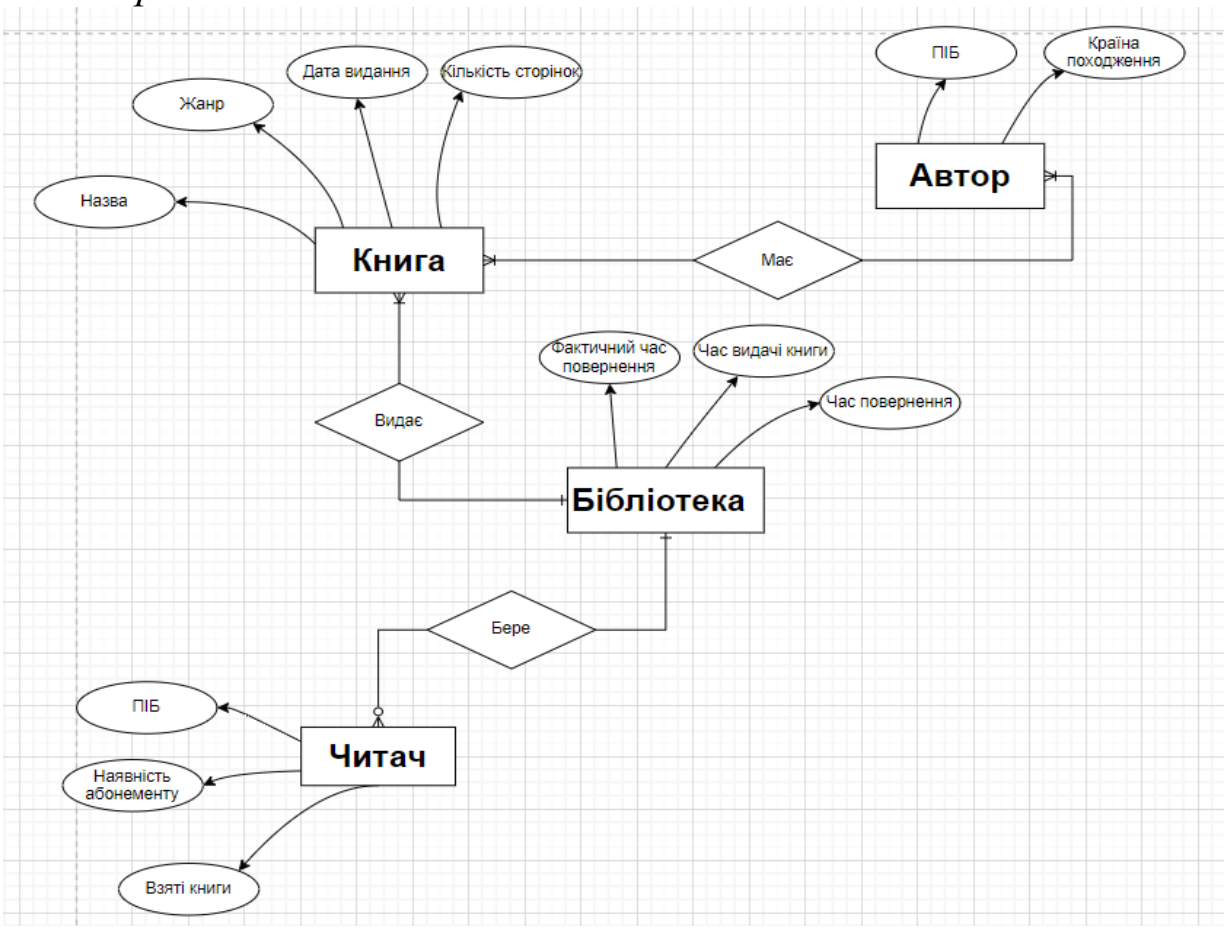
Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL

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

Загальне завдання роботи полягає у наступному:

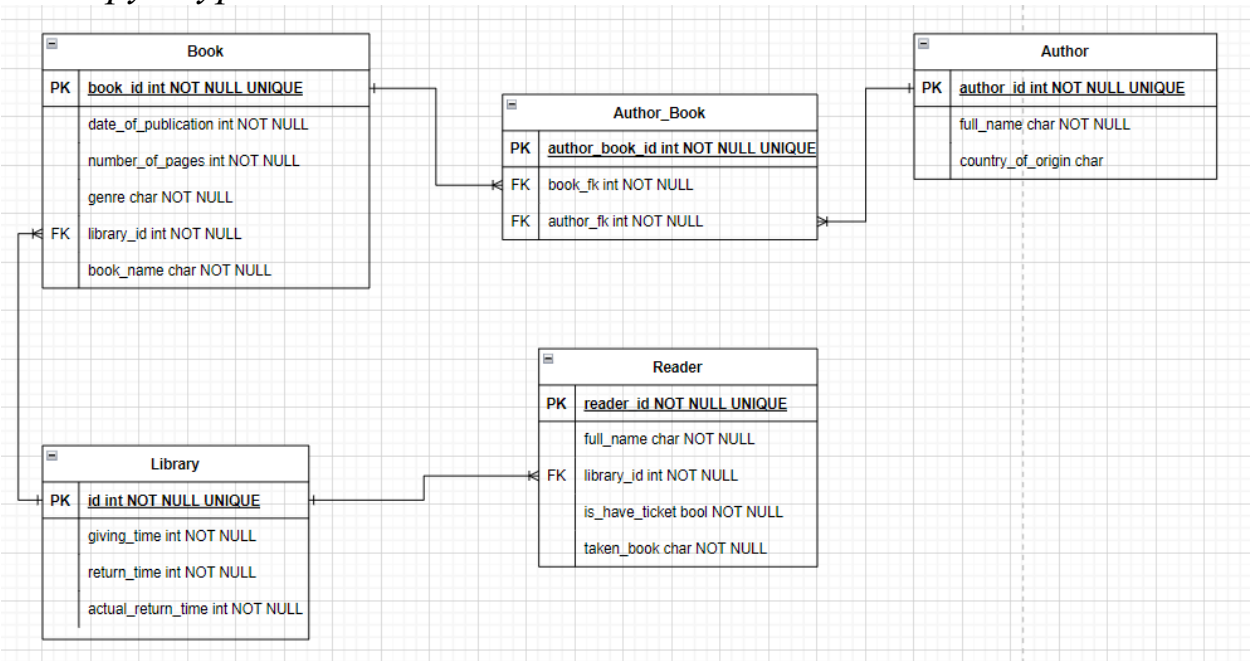
1. Реалізувати функції перегляду, внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. Забезпечити реалізацію пошуку за декількома атрибутами 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*
    - *1.1. Insert data*
        - *1.1.1. Choose table*
    - *1.2. Update data*
        - *1.2.1. Choose table*
    - *1.3. Delete data*
        - *1.3.1. Choose table*
    - *1.4. Exit*
2. *Create data*
    - *2.1. Choose table*
3. *Search*
    - *3.1. Query 1*
    - *3.2. Query 2*
    - *3.3. Query 3*

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;

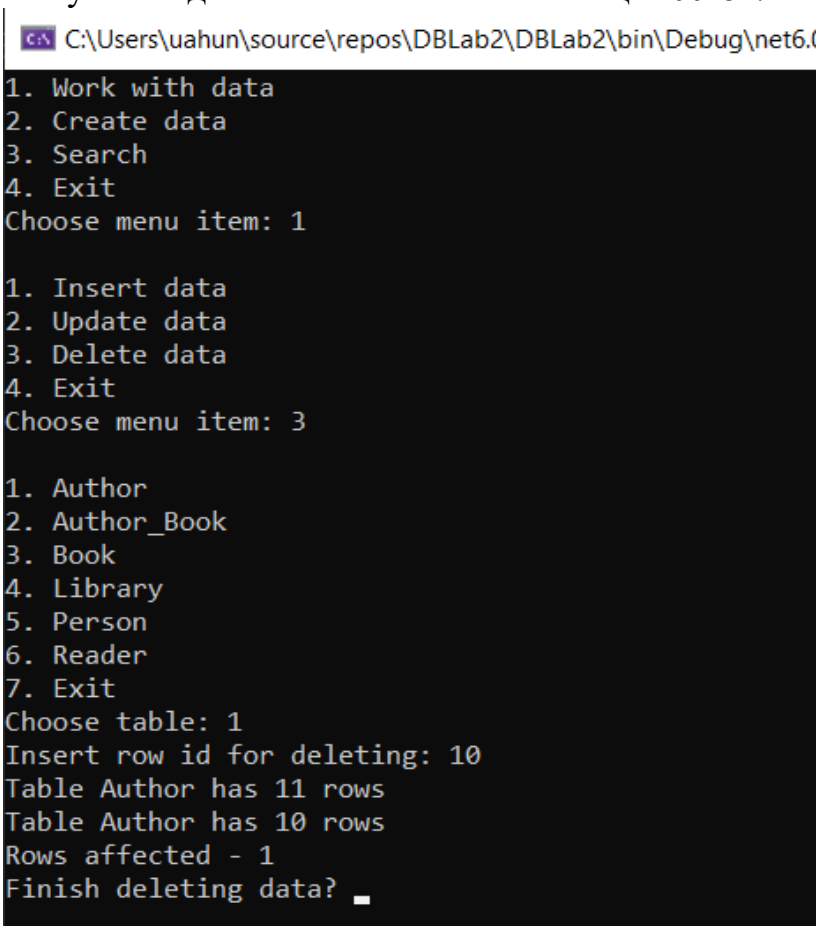
Рівень меню 3.3(3.1, 3.2, 3.3, 3.4) – вибір запиту пошуку;

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:

	author_id [PK] integer	full_name character varying	country_of_origin character varying
1	5	qwe	qwe
2	6	f6224280868bc4...	b21434467ef636...
3	7	0257a5b68f34fb...	344003ef610b83...
4	8	fe4c4b93540361...	5d95670fb13d58...
5	9	f882b76134b5d1...	d43dcb93d07e9...
6	10	d1e532c1900a09...	5c850d2cfc7889...
7	11	9f10e29d95e09b...	4cbf6c0a704951...

	author_id [PK] integer	full_name character varying	country_of_origin character varying
1	5	qwe	qwe
2	6	f6224280868bc4...	b21434467ef636...
3	7	0257a5b68f34fb...	344003ef610b83...
4	8	fe4c4b93540361...	5d95670fb13d58...
5	9	f882b76134b5d1...	d43dcb93d07e9...
6	11	9f10e29d95e09b...	4cbf6c0a704951...
7	12	537834cbf7552d...	9112e0dd07c235...

Дочірня таблиця *Author_Book* до та після вилучення:

	author_book_id [PK] integer	book_fk integer	author_fk integer
1	6	1	5
2	7	1	5
3	8	1	5
4	15	1	6
5	16	2	8
6	17	4	9
7	18	5	10

	author_book_id [PK] integer	book_fk integer	author_fk integer
1	6	1	5
2	7	1	5
3	8	1	5
4	15	1	6
5	16	2	8
6	17	4	9

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):

```
7. Exit
Choose table: 1
Insert row id for deleting: 100
Table don't have row with that id!
Insert row id for deleting:
```

Запис даних в дочірню таблицю *Reader*:

```
C:\Users\uahun\source\repos\DBLab2\DBLab2\bin\Debug\net6.0\DBLab2.exe
1. Work with data
2. Create data
3. Search
4. Exit
Choose menu item: 1

1. Insert data
2. Update data
3. Delete data
4. Exit
Choose menu item: 1

1. Author
2. Author_Book
3. Book
4. Library
5. Person
6. Reader
7. Exit
Choose table: 6
Insert *id*(Attention! PK should't be repeated): 8
Insert *library_id*(Attention! The id column from the Library table must have the same value!): 3
Insert *person_id*(Attention! The id column from the Person table must have the same value!): 2
Insert *taken_book*: Kobzar
Finished entering data? y
Table Reader has 5 rows
Table Reader has 6 rows
Rows affected - 1
```

Таблиця *Reader* після додавання даних:

	id [PK] integer	library_id integer	person_id integer	taken_book character varying
1	1	2	1	Kobzar
2	2	1	1	Zakhar Berkut
3	5	5	2	Painted fox
4	6	1	2	adf86d5f7e48b4...
5	7	1	2	a4be5c9d8fb765...
6	8	3	2	Kobzar

Приклад валідації даних при їх записі:

```
Choose table: 6
Insert *id*(Attention! PK should't be repeated): 9
Insert *library_id*(Attention! The id column from the Library table must have the same value!): 1000
It's incorrect value of *library_id*!(There is no required data in the parent table!)
Insert *id*(Attention! PK should't be repeated):
```

Завдання 2

Генерація псевдовипадкових даних таблиці *Library*:

```
1. Work with data
2. Create data
3. Search
4. Exit
Choose menu item: 2

1. Author
2. Author_Book
3. Book
4. Library
5. Person
6. Reader
7. Exit
Choose table: 4
Enter amount of generation data: 10000
Table Library has 35 rows
Table Library has 10035 rows
Finish create data?
```

Результат:

	id [PK] integer	giving_time date	return_time date	actual_return_time date
3186	3190	2020-11-13	2022-11-04	2021-02-26
3187	3191	2020-03-15	2022-10-15	2021-07-04
3188	3192	2020-03-30	2022-06-19	2021-08-16
3189	3193	2020-09-12	2022-10-26	2021-07-24
3190	3194	2020-11-28	2022-03-10	2021-06-25
3191	3195	2020-05-28	2022-08-01	2021-12-05
3192	3196	2020-11-25	2022-01-12	2021-11-17
3193	3197	2020-02-05	2022-05-18	2021-01-02
3194	3198	2020-03-22	2022-05-09	2021-07-01

Total rows: 4000 of 10035 | Query complete 00:00:00.152

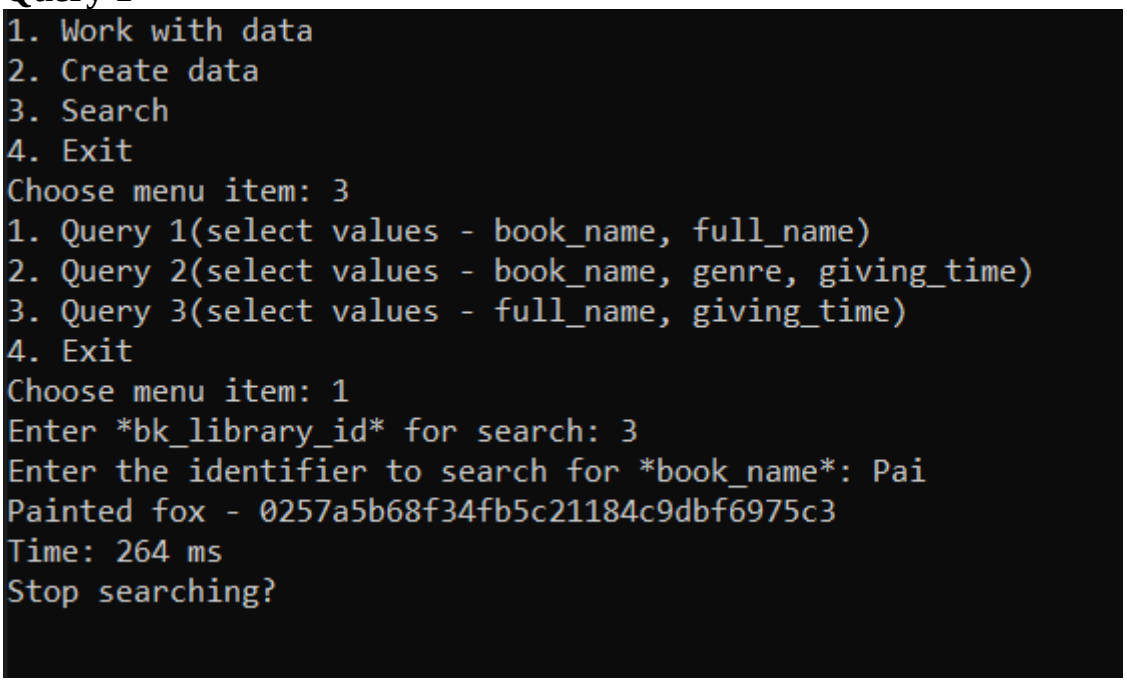
Копія SQL-запиту:

```
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

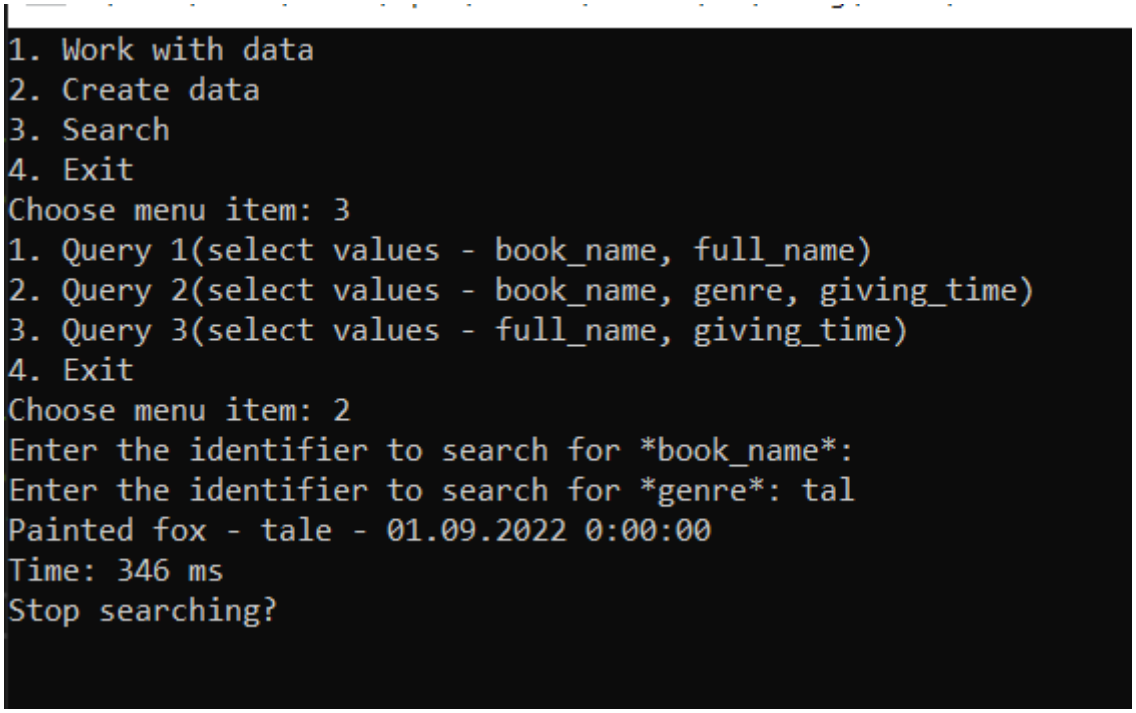


Копія SQL-запиту 1:

```
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

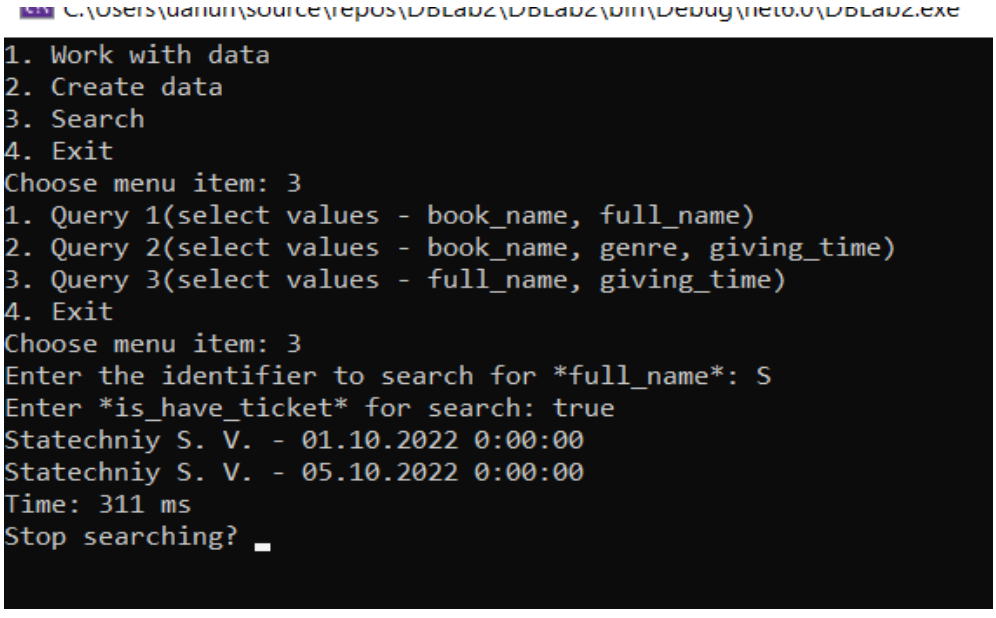


Копія SQL-запиту 2:

```
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

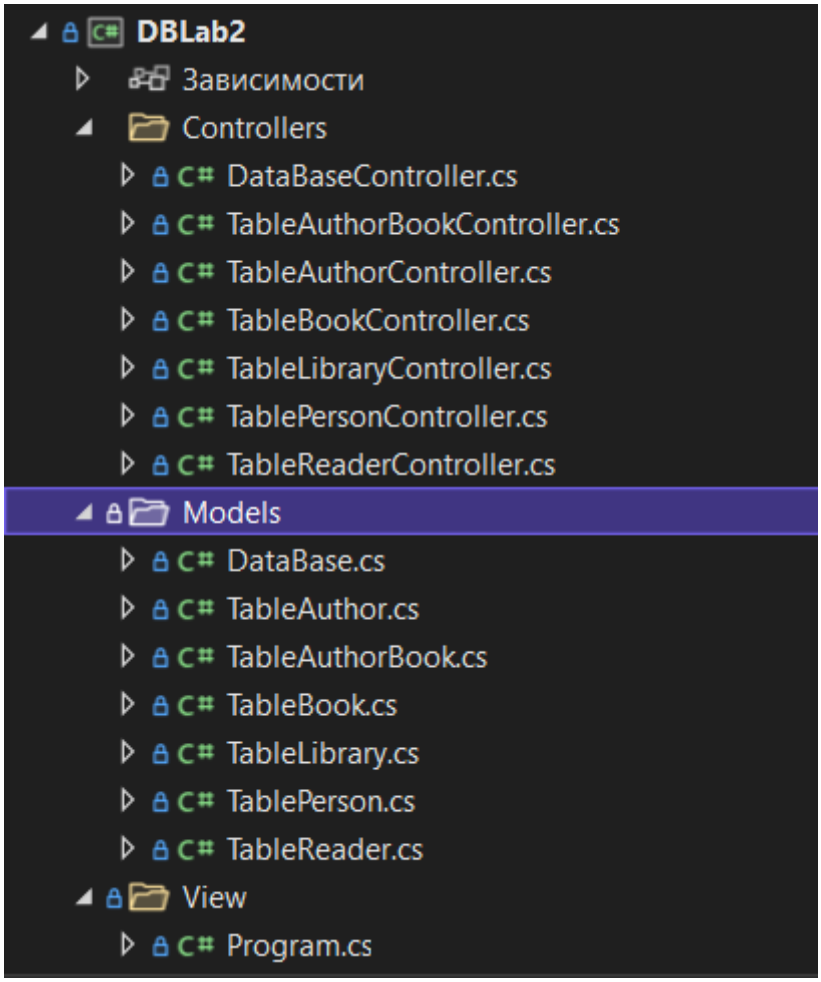


Копія SQL-запиту 3:

```
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*.