Климчука Нікіти Олеговича КП-91

**Лабораторна робота № 1.**

*Завдання* роботи полягає у наступному:

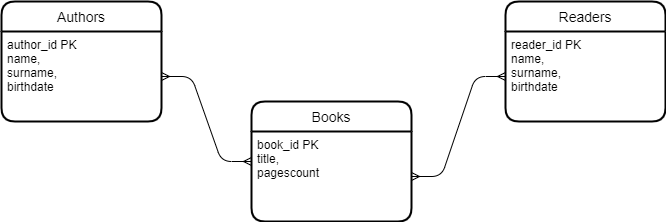
1. Розробити модель «сутність-зв’язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі».
2. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL.
3. Виконати нормалізацію схеми бази даних до третьої нормальної форми (3НФ).
4. Ознайомитись із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

**1.**

Сутність автор, автори можуть писати книги.

Сутність книга, у книги є автори та можуть бути читачі.

Сутність читач, читачі можуть читати книги.



ER модель.

Назва нотації: crow’s feet.

**2.**

Так як одна й та сама людина може читати книги і бути автором, для того щоб не дублювати інформацію щодо людини була створена окрема таблиця persons, в якій зберігається ця інформація.

Сутність автор була перетворена у таблицю authors, читач - у таблицю readers, книга - у таблицю books.

Зв’язок M:N між авторами та книгами зумовив появлення таблиці linksbooktoauthor. Аналогічно з читачами та книгами.

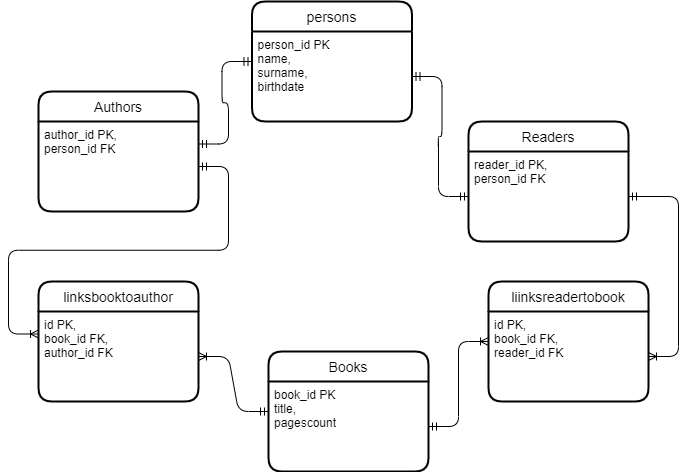


Схема бази даних

**3.**

Схема бд відповідає 1 нормальній формі, так як кожне поле містить лише одне значення.

2 нормальній формі, так як кожна таблиця має id, і інші поля таблиць ніяк від нього не залежать.

3 нормальній формі, так як всі дані, які можуть мати різні таблиці винесені в окремі таблиці.

**4.**

Скрипт створення таблиць:

create table persons

(

person\_id INT GENERATED ALWAYS AS IDENTITY,

name varchar(50) not null,

surname varchar(50) not null,

birthdate date,

primary key (person\_id)

);

create table authors

(

author\_id INT GENERATED ALWAYS AS IDENTITY,

person\_id int not null,

primary key (author\_id),

CONSTRAINT fk\_person

FOREIGN KEY(person\_id)

REFERENCES persons(person\_id)

);

create table readers

(

reader\_id INT GENERATED ALWAYS AS IDENTITY,

person\_id int not null,

primary key (reader\_id),

CONSTRAINT fk\_person

FOREIGN KEY(person\_id)

REFERENCES persons(person\_id)

);

create table books

(

book\_id INT GENERATED ALWAYS AS IDENTITY,

title varchar(50) not null,

PagesCount int,

primary key (book\_id)

);

create table linksbooktoauthor

(

id INT GENERATED ALWAYS AS IDENTITY,

book\_id int not null,

author\_id int not null,

primary key(id),

CONSTRAINT fk\_book

FOREIGN KEY(book\_id)

REFERENCES books(book\_id),

CONSTRAINT fk\_author

FOREIGN KEY(author\_id)

REFERENCES authors(author\_id)

);

create table linksreadertobook

(

id INT GENERATED ALWAYS AS IDENTITY,

book\_id int not null,

reader\_id int not null,

primary key(id),

CONSTRAINT fk\_book

FOREIGN KEY(book\_id)

REFERENCES books(book\_id),

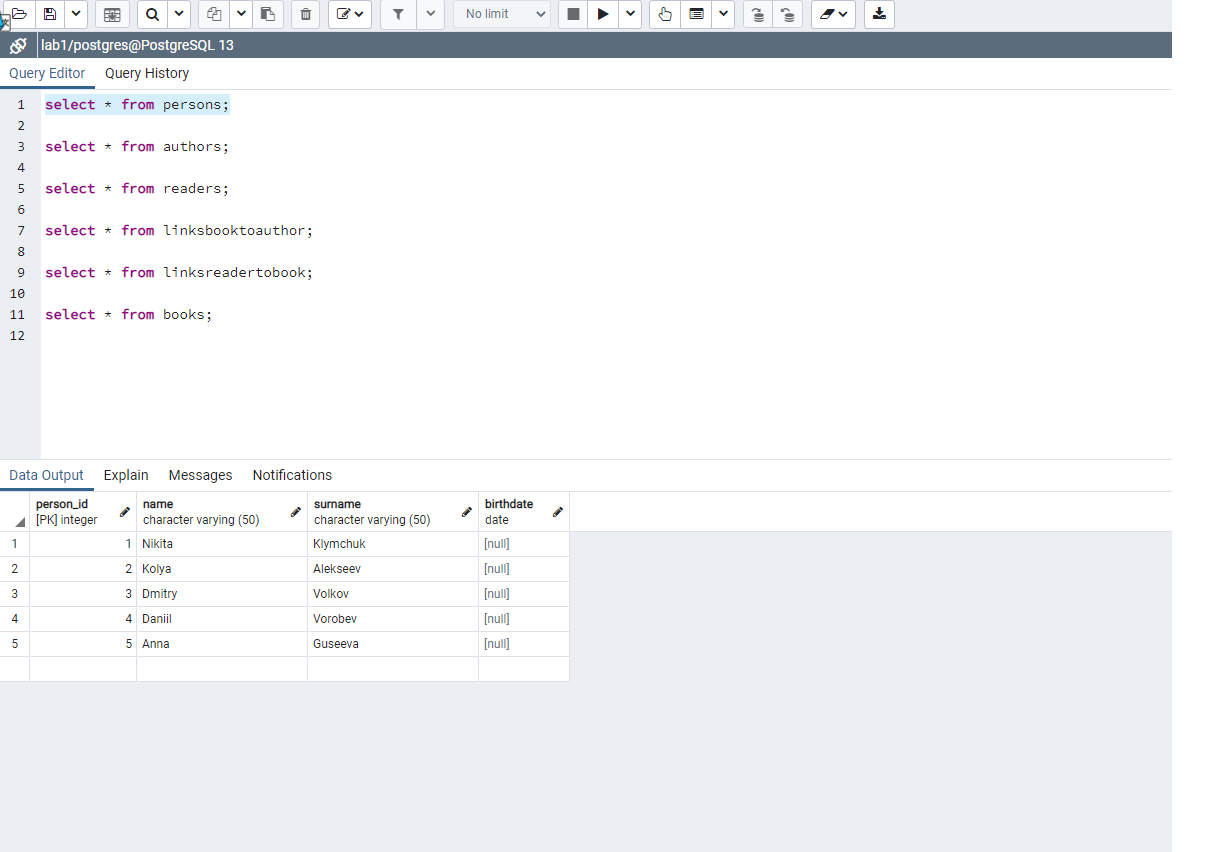
CONSTRAINT fk\_reader

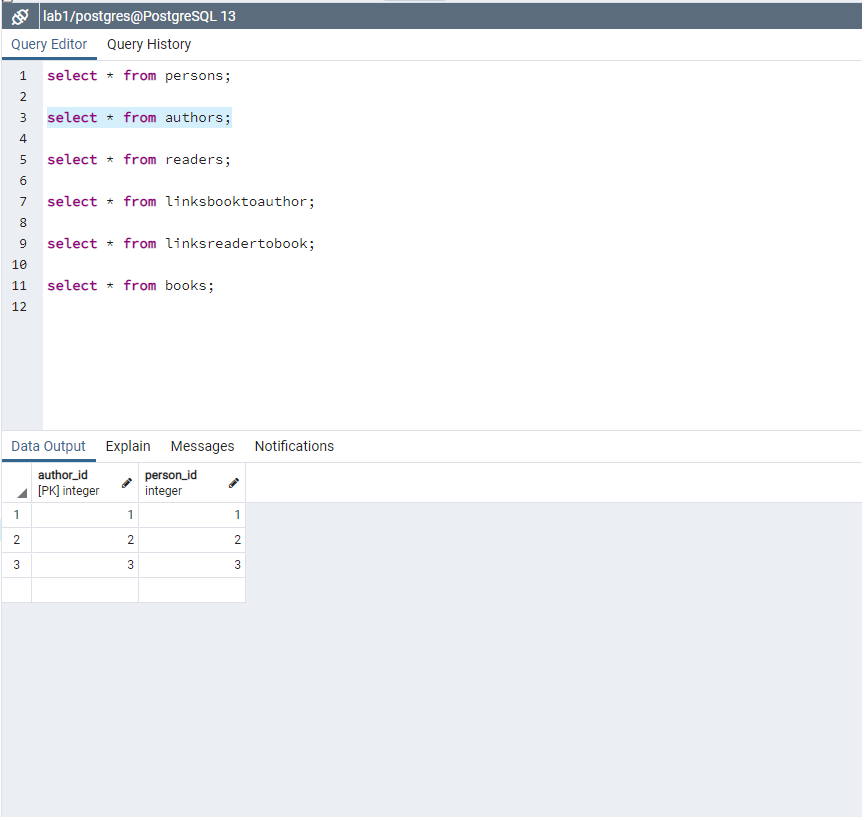
FOREIGN KEY(reader\_id)

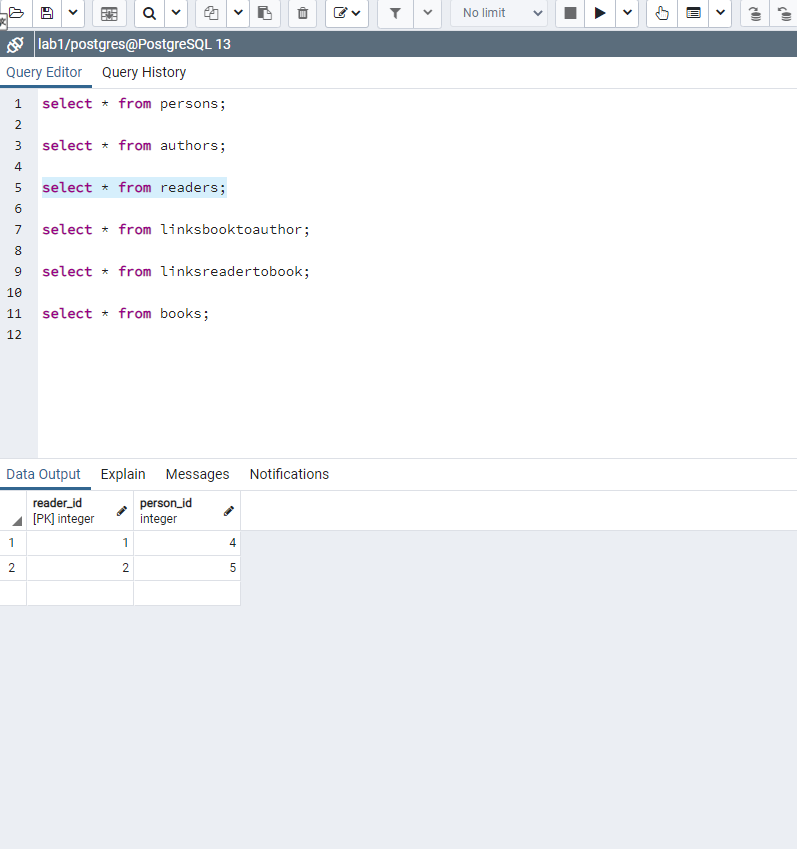
REFERENCES readers(reader\_id)

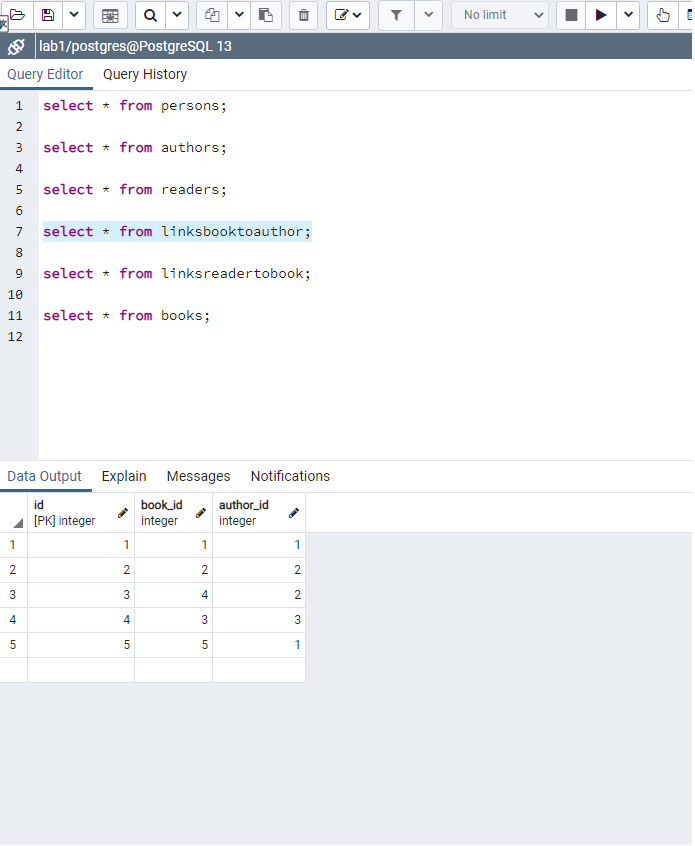
);

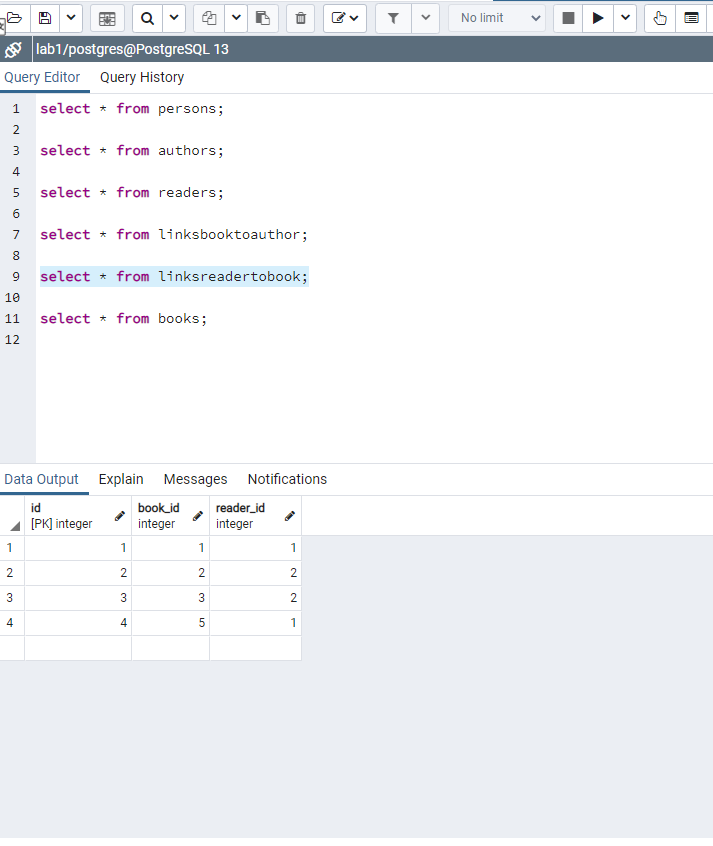
Скріншоти екрану з даними в таблицях:

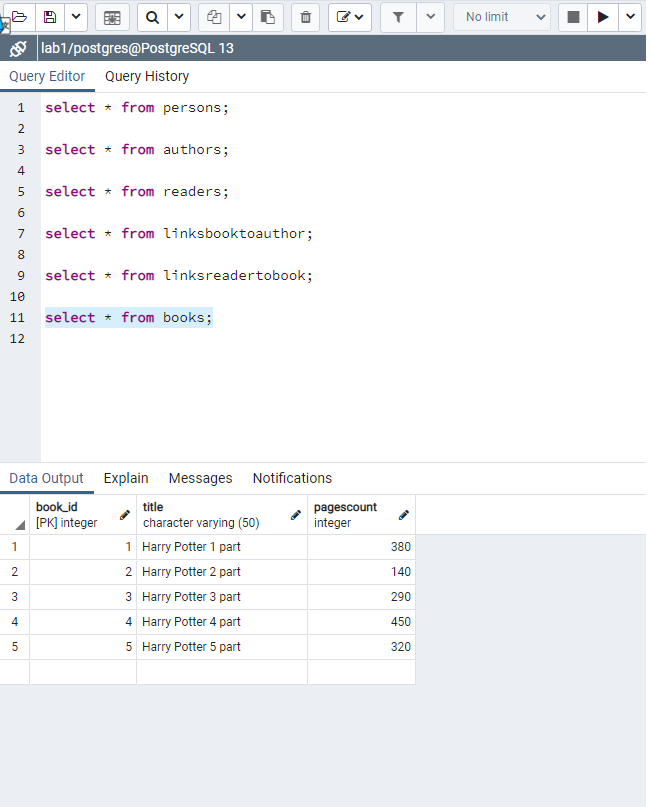


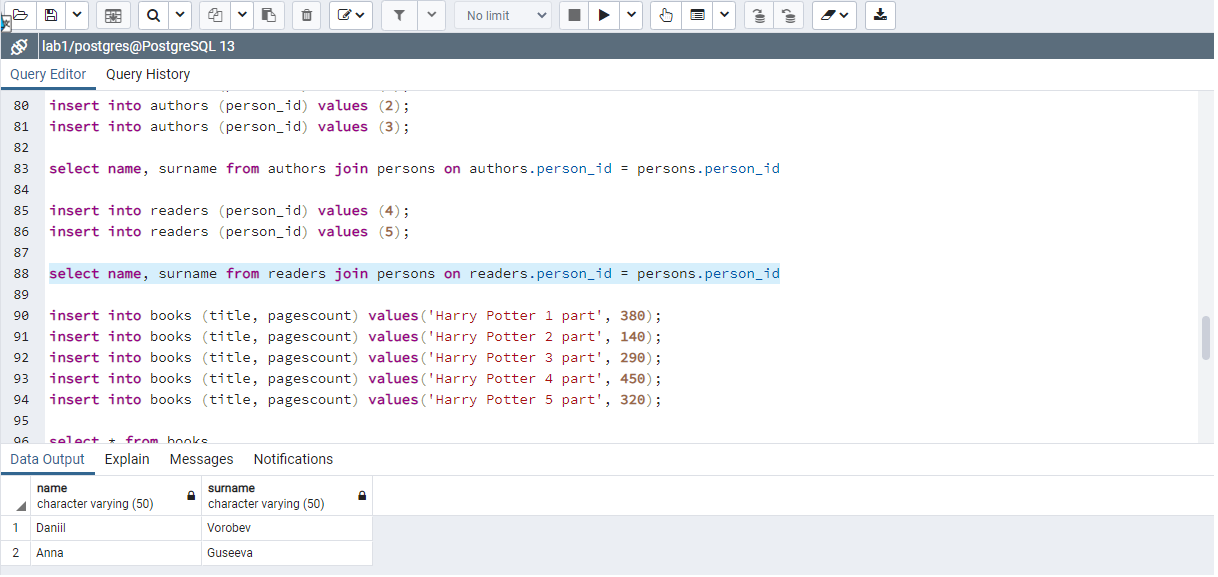


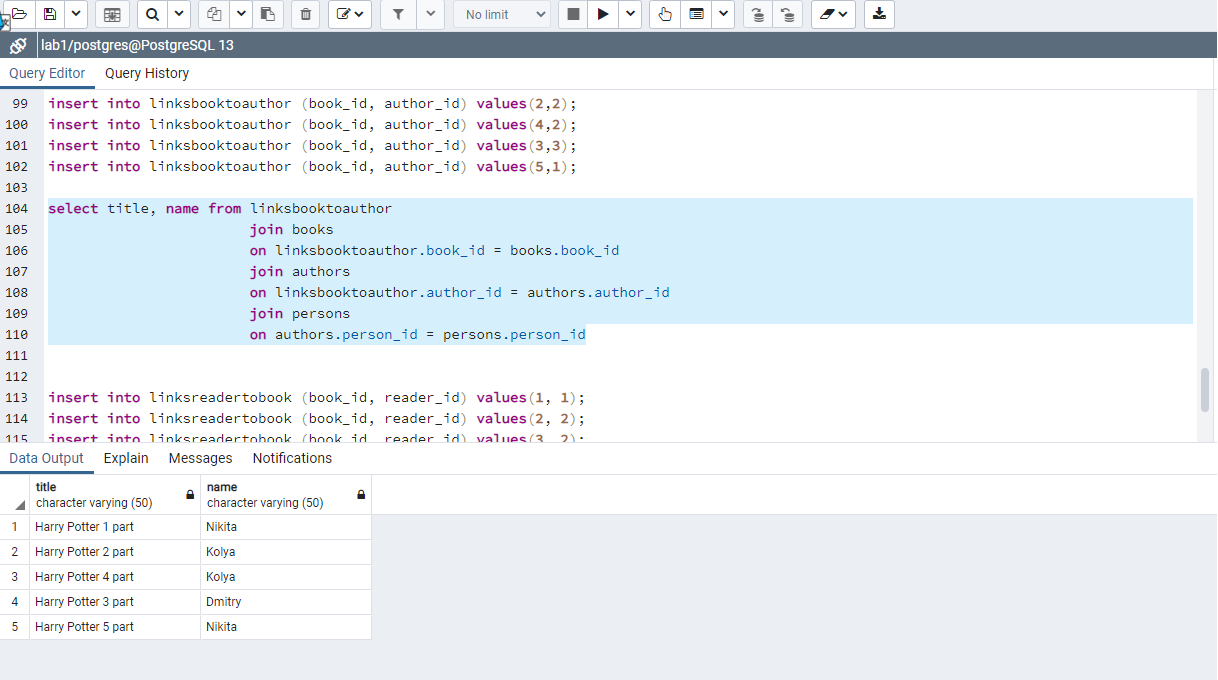


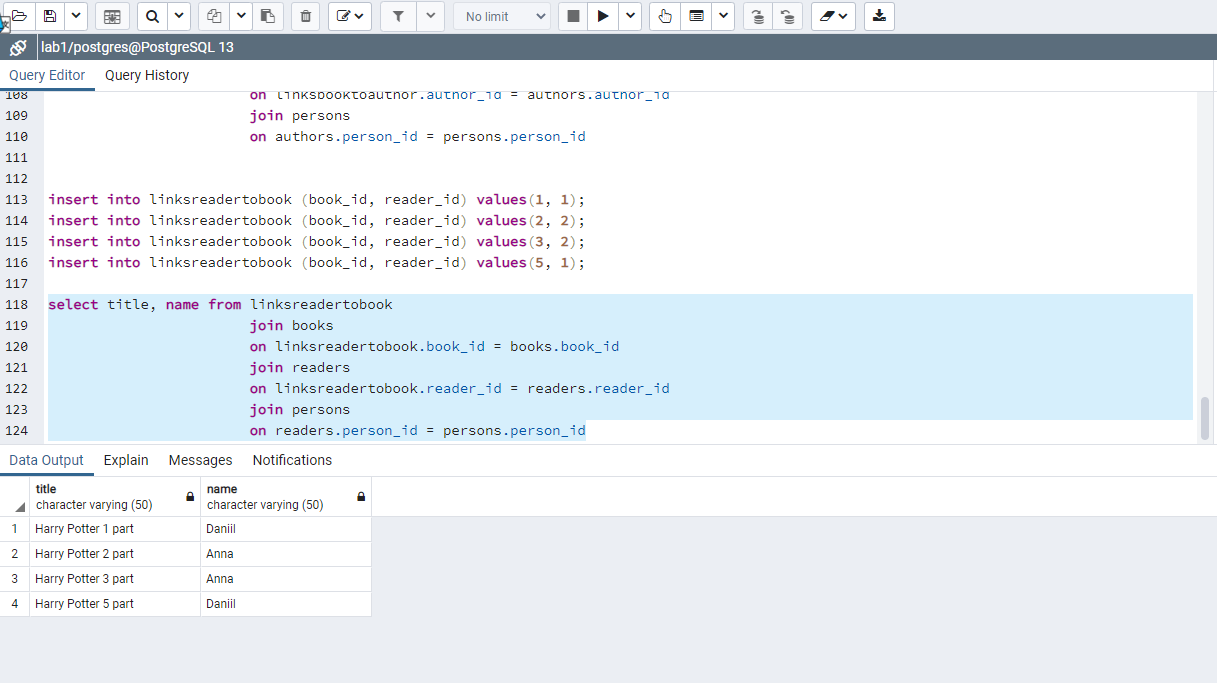












Відповіді на запитання:

1. **Сформулювати призначення діаграм типу «сутність-зв’язок».**

Схематично змоделювати сутності бази даних та зв’язки між ними.

1. **Назвати основні об’єкти схеми PostgreSQL.**

Таблиці, функції, тригери, процедури, представлення, типи.

1. **Навести приклади різних типів зв’язків у базах даних (1:1, 1:N, N:M).**

N:M - наприклад книга і автор (у одного автора може бути багато книг, а у книги - багато авторів).

1:N - наприклад у соціальній мережі користувач може ставити багато лайків.

1:1 - наприклад у нас є користувач, у нього є багато полів, ми можемо розбити інформацію про нього у декілька окремих таблиць і між цими таблицями буде зв’язок 1:1.