

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

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

**Лабораторна робота №2**

з дисципліни

**«**Бази даних і засоби управління**»**

Виконав студент ІII курсу

ФПМ групи КВ-83

Лазуткін О.

Перевірив: Павловський В.І.

Київ – 2020

**Ознайомлення з базовими операціями СУБД PostgreSQL**

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

*Загальне завдання* роботи полягає у наступному:

1. Реалізувати функції 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).

Деталізоване завдання:

1. Забезпечити можливість уведення/редагування/вилучення даних у таблицях бази даних з можливістю контролю відповідності типів даних атрибутів таблиць (рядків, чисел, дати/часу). Для контролю пропонується два варіанти: контроль при введенні (валідація даних) та перехоплення помилок (try..except) від сервера PostgreSQL при виконанні відповідної команди SQL. Особливу увагу варто звернути на дані таблиць, що мають зв’язок 1:N. При цьому з боку батьківської таблиці необхідно контролювати **вилучення** рядків за умови наявності даних у підлеглій таблиці. З точки зору підлеглої таблиці варто контролювати наявність відповідного рядка у батьківській таблиці при виконанні **внесення** нових даних. Унеможливити виведення програмою системних помилок на екрані шляхом їх перехоплення і адекватної обробки. Внесення даних виконується користувачем у консольному вікні програми.
2. Забезпечити можливість автоматичної генерації великої кількості даних у таблицях за допомогою вбудованих у PostgreSQL функцій роботи з псевдовипадковими числами. Дані мають бути згенерованими **не мовою програмування, а відповідним SQL-запитом**!
3. Для реалізації пошуку необхідно підготувати 3 запити, що включають дані з декількох таблиць і фільтрують рядки за 3-4 атрибутами цих таблиць. Забезпечити можливість уведення конкретних значень констант для фільтрації з клавіатури користувачем. Крім того, після виведення даних необхідно вивести час виконання запиту у мілісекундах. Перевірити швидкодію роботи запитів на попередньо згенерованих даних.
4. Програмний код організувати згідно шаблону Model-View-Controller(MVC).  Приклад організації коду згідно шаблону доступний [за даним посиланням](https://www.giacomodebidda.com/mvc-pattern-in-python-introduction-and-basicmodel/). При цьому модель, подання та контролер мають бути реалізовані у окремих файлах. Для доступу до бази даних використовувати **лише мову SQL** (без ORM).

Вимоги до оформлення звіту лабораторної роботи у електронному вигляді

Опис (файл README.md) лабораторної роботи у  **репозиторії** **GitHub** включає: назву лабораторної роботи, структуру бази даних з лабораторної роботи №1.

Репозиторій має містить файл звіту у форматі PDF та програмний код файлів мовою Python (або іншою).

Звіт у форматі PDF має містити: титульний аркуш, завдання, URL репозиторію з вихідним кодом та відповіді на вимоги до звітування щодо пунктів 1-4 деталізованого завдання:

Вимоги до пункту №1 деталізованого завдання:

* ілюстрації обробки виняткових ситуацій (помилок) при уведенні/вилучення даних;
* ілюстрації валідації даних при уведенні користувачем.

Вимоги до пункту №2 деталізованого завдання:

* копії екрану (ілюстрації) з фрагментами згенерованих даних таблиць;
* копії SQL-запитів, що ілюструють генерацію при визначених вхідних параметрах.

Вимоги до пункту №3 деталізованого завдання:

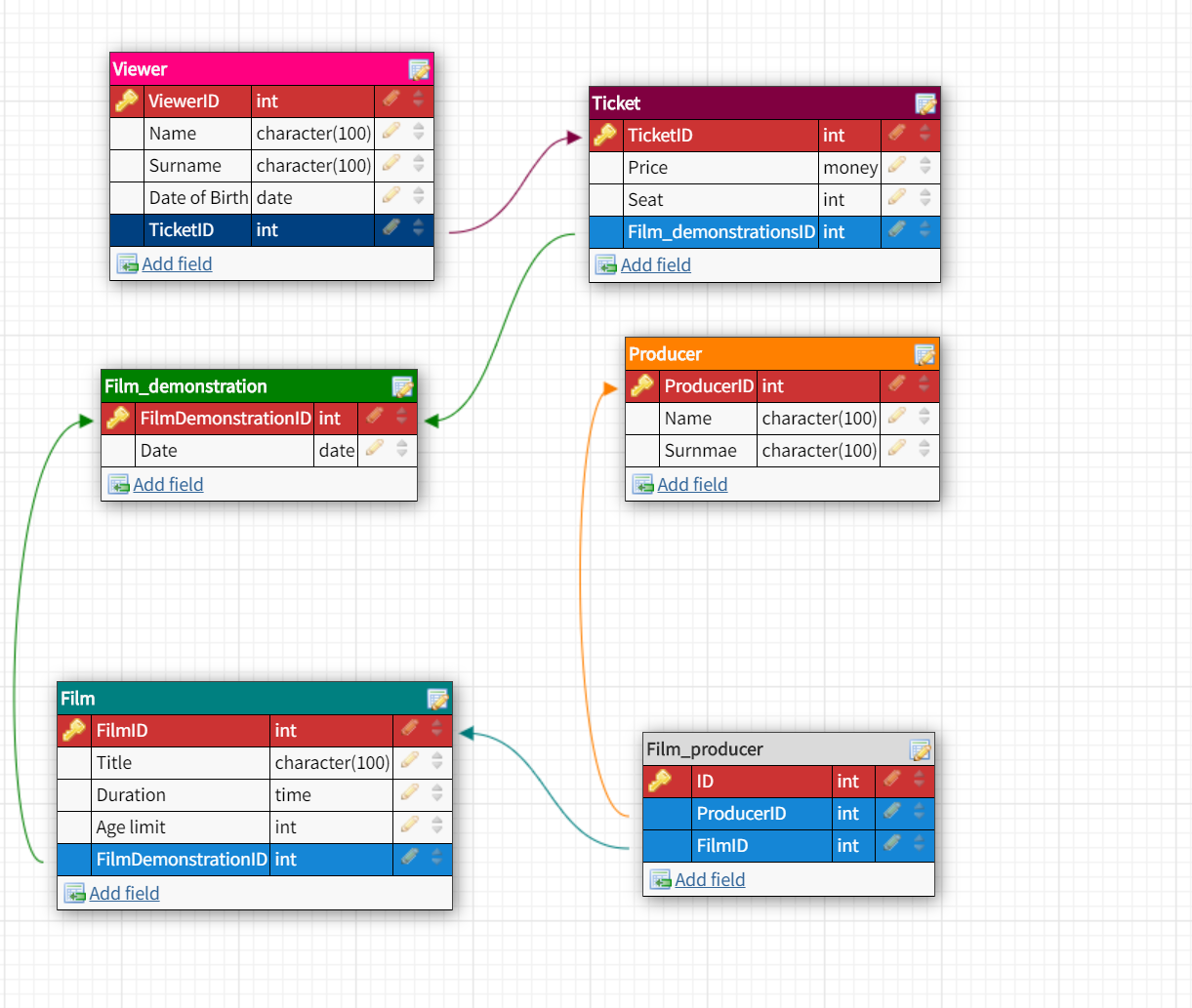
* ілюстрації уведення пошукового запиту та результатів виконання запитів;
* копії SQL-запитів, що ілюструють пошук з зазначеними початковими параметрами

Вимоги до пункту №4 деталізованого завдання:

ілюстрації програмного коду з репозиторію Git.

# Нормалізована модель даних БД «Доставка їжі»

На рисунку зображена нормалізована модель даних БД ‘Доставка їжі’, розроблена на першій лабораторній роботі.

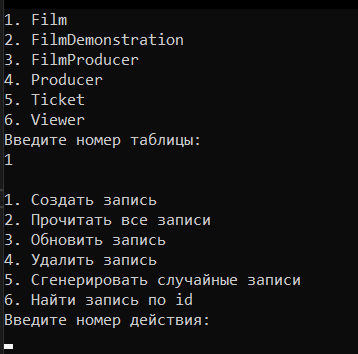


# Опис програми

Програма створена для управління базою даних за допомогою базових операцій СУБД PostgreSQL і мови програмування **c#**, та реалізовує функціональні вимоги, що наведені у завданні. Додаток використовує шаблон проектування MVC.

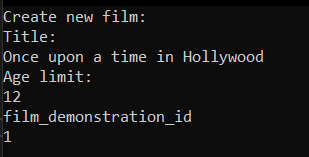
# Меню програми:

На зображенні показано головне меню програми з вибором таблиці, з якою хоче оперувати користувач. Після вибору таблиці, відображається меню вибору операції.

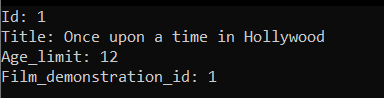


# Завдання 1

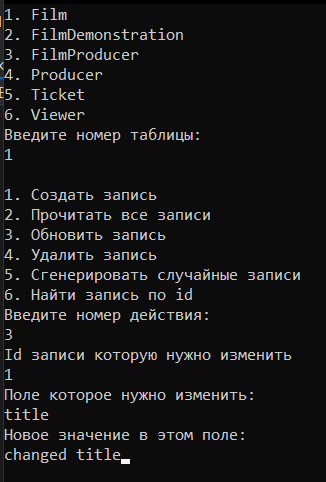
## Додавання даних до БД:



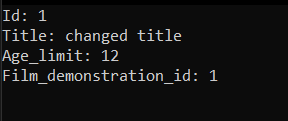
Результат:



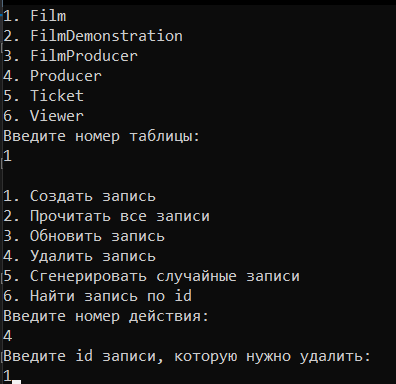
## Редагування даних



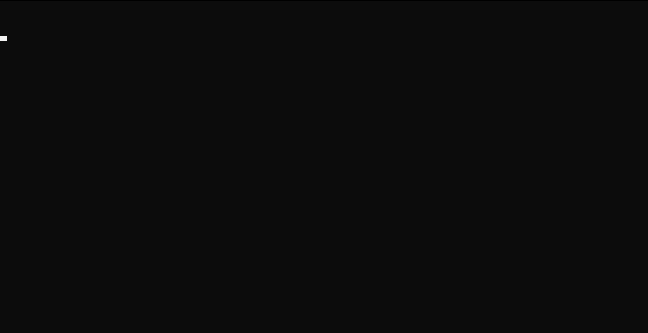
Результат:



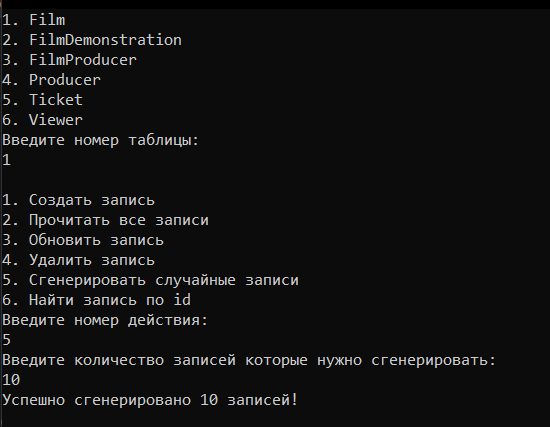
## Видалення даних:



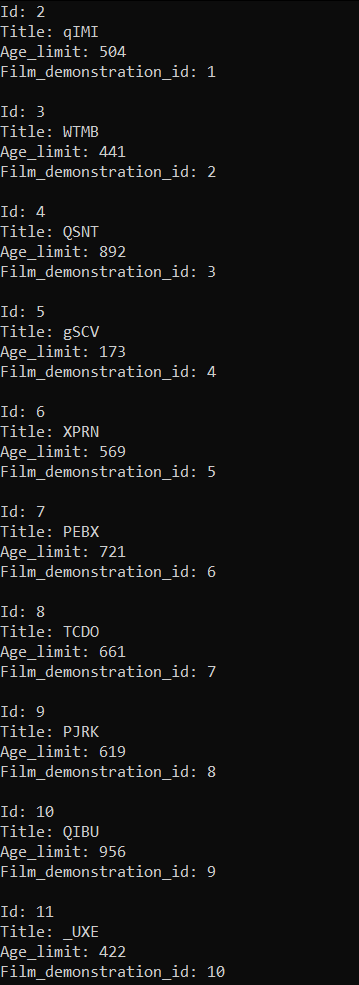
Результат (пустий екран, тобто записів немає):



# Завдання 2. Пакетне генерування даних:



Результат:



Запит SQL:

insert into

film(title, age\_limit, film\_demonstration\_id)

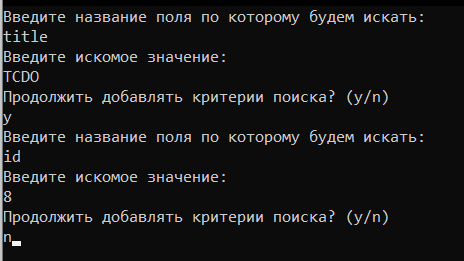
(select chr(trunc(65 + random() \* 50)::int) || chr(trunc(65 + random() \* 25)::int) || chr(trunc(65 + random() \* 25)::int) || chr(trunc(65 + random() \* 25)::int),

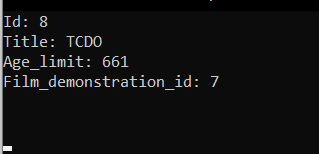
trunc(random()\*1000)::int,

film\_demonstration.id

from generate\_series(1, 10), film\_demonstration limit(10))

# Завдання 3. Пошук за двома-трьома атрибутами одночасно:





Запит SQL:

Select \* from film where title = ‘TCDO’ and id = 8