



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»**

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

Виконав: студент III курсу

ФПМ групи КВ-84

Ніколайчук Данило

Перевірив:

Київ – 2020

*Загальне завдання роботи полягає в такому:*

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 введення/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 підлеглий 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!**

Приклад генерації 100 псевдовипадкових чисел:

```
select trunc(random()*1000)::int
from generate_series(1,100)
```

	trunc integer	
1	368	
2	773	
3	29	
4	66	
5	497	
6	956	

Приклад генерації 5 псевдовипадкових рядків:

```
select chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int)
from generate_series(1,5)
```

	?column? text	
1	NE	
2	MQ	
3	RN	
4	DW	
5	DA	

Приклад генерації псевдовипадкової мітки часу з діапазону [доступний за посиланням](#).

Кількість даних для генерування має вводити користувач з клавіатури. Для тесту взяти 100 000 записів для однієї-двох таблиць.

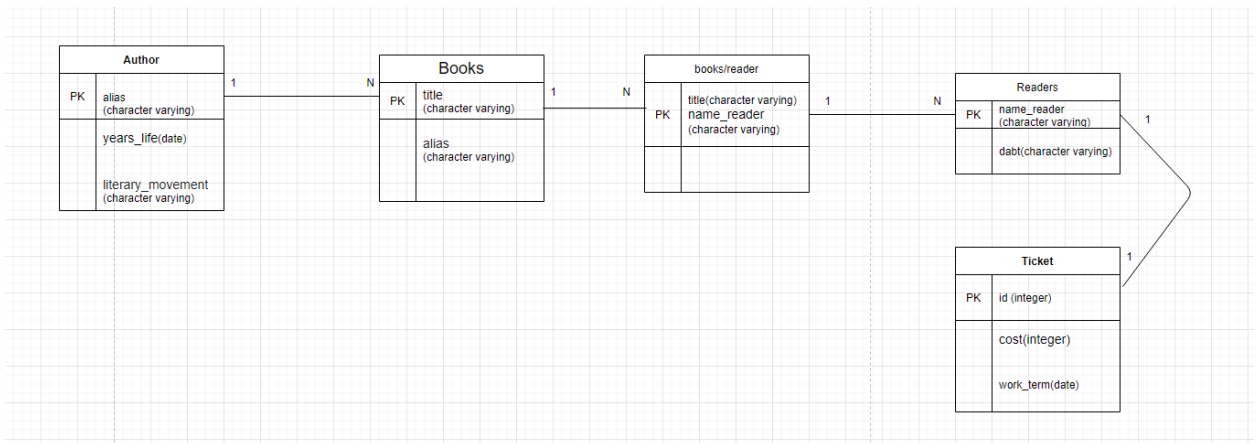
Особливу увагу слід звернути на відповідність даних вимогам зовнішніх ключів з метою уникнення помилок порушення обмежень цілісності foreign key).

- Для реалізації пошуку необхідно підготувати 3 запити, що включають дані з декількох таблиць і фільтрують рядки за 3-4 атрибутами цих таблиць. Забезпечити можливість введення 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](#). При цьому модель, подання та контролер мають бути реалізовані у окремих файлах. Для доступу до бази даних використовувати **лише мову SQL** без ORM).

Рекомендована бібліотека взаємодії з PostgreSQL Psycopg2: <http://initd.org/psycopg/docs/usage.html>)



### Відповідь на вимоги до пункту №1 деталізованого завдання:

Ілюстрації обробки виняткових ситуацій (помилки) при введенні/вилучення даних:

```
def add_if you want to update press 1
  If you want to add press 2
  count=If you want to delete press 3
  mass=[If you want to random press 4
  NULL_If you want to search press 5
  if(taEnter command : 3
    prEnter table name : FAKE_TABLENAME
    whThe table name is wrong ERROR
    Press any key to continue . . .
```

Ілюстрації валідації даних при уведенні користувачем:

```
print('If you want to update press 1\nIf you want to add press 2\nIf you want to delete press 3\nIf you want to random press 4\nIf you want to search press 5\nEnter command : 1\nEnter table name : books\nThere is a 1:N connection in Author and Books if the changes is in the "alias" column ,it will touch both tables : \nauthor table : ['alias', 'year_life', 'literary_movement']\nBooks table : ['title', 'alias']\nEnter column name : FAKE_COLUMNNAME\n42703\nERROR: ОШИБКА: столбец "fake_columnname" не существует\nLINE 1: SELECT FAKE_COLUMNNAME FROM books\n          ^\nPress any key to continue . . .
```

Вимоги до пункту №2 деталізованого завдання:  
Меню генерації:

```
():\nt("If you want to If you want to update press 1\nIf you want to add press 2\nIf you want to delete press 3\nIf you want to random press 4\nIf you want to search press 5\nEnter command : 4\nEnter table name : books\nThere is a 1:N connection in Author and Books if the changes is in the "alias" column ,it will touch both tables : \nauthor table : ['alias', 'year_life', 'literary_movement']\nBooks table : ['title', 'alias']\nENTER the N value : 2
```

Копії екрану з фрагментами згенерованих даних таблиць:

9977	焔	燭
9978	燭	燭
9979	燭	燭
9980	燭	燭
9981	燭	燭
9982	燭	燭
9983	燭	燭
9984	燭	燭
9985	燭	燭
9986	燭	燭
9987	燭	燭
9988	燭	燭
9989	燭	燭
9990	燭	燭
9991	燭	燭
9992	燭	燭
9993	燭	燭
9994	燭	燭
9995	燭	燭
9996	燭	燭
9997	燭	燭
9998	燭	燭
9999	燭	燭
10000	燭	燭

Копії SQLзапитів, що ілюструють генерацію при визначених  
вхідних параметрах:

```
print('If you want to update press 1\nIf you want to add press 2\nIf you want to delete press 3\nIf you want to random press 4\nIf you want to search press 5\nEnter command : 4\nEnter table name : books\nThere is a 1:N connection in Author and Books if the changes is in the "alias" column ,it will touch both tables : \nauthor table : ['alias', 'year_life', 'literary_movement']\nBooks table : ['title', 'alias']\nENTER the N value : 4\nprint(WITH test AS(INSERT INTO books SELECT chr(trunc(65+random()*500)::int), chr(trunc(65 + random()*500)::int) FROM generate_series(1,4) RETURNING alias)\nINSERT INTO author SELECT alias FROM test\nPress any key to continue . . .
```

### Вимоги до пункту №3 деталізованого завдання:

Ілюстрації введення пошукового запиту та результатів виконання запитів:

```
comm If you want to update press 1
table If you want to add press 2
comm If you want to delete press 3
table If you want to random press 4
comm If you want to search press 5
error Enter command : 5
() Input quantity of attributes to search by >>> 2
Errc Input name of the attribute number 1 to search by >>> id
rcode Input name of the attribute number 2 to search by >>> work_term
OR: ['id', 'work_term']
s')
comm col_names_str: SELECT table_name FROM INFORMATION_SCHEMA.COLUMNS WHERE information_schema.columns.column_name LIKE 'id'
le) INTERSECT ALL SELECT table_name FROM information_schema.columns WHERE information_schema.columns.column_name LIKE 'work_
comm term'
comm ['integer', 'date']
tab Enter left limit DATA for work_term 2013-01-21
comm Enter right limit DATA for work_term 2021-01-01
table Enter left limit for id 10
comm Enter right limit for id 300
table ((13, 200, datetime.date(2020, 12, 12)), (27, 806, datetime.date(2015, 8, 1)))
comm Time:0.007994890213012695 seconds
ou
()
```

Копії SQL-запитів, що ілюструють генерацію при визначених запитів,  
що ілюструють пошук з зазначеними початковими параметрами

```
If you want to update press 1
If you want to add press 2
If you want to delete press 3
If you want to random press 4
If you want to search press 5
Enter command : 5
Input quantity of attributes to search by >>> 1
Input name of the attribute number 1 to search by >>> alias
['alias']
col_names_str: SELECT table_name FROM INFORMATION_SCHEMA.COLUMNS WHERE information_schema.columns.column_name LIKE 'alias'
['character varying', 'character varying']
Input string for alias to search by >>> Lesia Ukrainka
[('Lesia Ukrainka',), ('Lesia Ukrainka',)]
Time:0.003996610641479492 seconds
SELECT alias FROM author WHERE alias LIKE 'Lesia Ukrainka' UNION ALL SELECT alias FROM books WHERE alias LIKE 'Lesia Ukrainka'
```

Вимоги до пункту №4 деталізованого завдання:

Ілюстрації програмного коду з репозиторію Git:

master

KPI\_DB\_labs / DB\_lab2 /

Go to file

Add file

s-person27 Create README.md

fcc9188 now History

..

README.md	Create README.md	now
SCHEMA.png	Add files via upload	6 minutes ago
control_func.py	Add files via upload	6 minutes ago
model.py	Add files via upload	6 minutes ago
view.py	Add files via upload	6 minutes ago

README.md

Лабораторна робота № 2 Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL Структура бази даних з лабораторної роботи №1 У файлі SCHEMA.PNG