

Лабораторна робота №2

З дисципліни «Бази даних»

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

Виконав студент групи КП-93

Мельник Іван Олександрович

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 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!**

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).

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 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). При цьому модель, подання та контролер мають бути реалізовані у окремих файлах. Для доступу до бази даних використовувати **лише мову SQL** (без ORM).

Посилання на репозиторій: <https://github.com/Melnikiva/databases>

Демонстрація роботи програми

Оброблення помилок





```
Select operation:
1. Add new purchase
2. Update product price
3. Delete purchase
4. Generate N purchases
5. Search for product
6. Exit

1
Input product code:
123
Invalid input
```

```
4. Generate N purchases
5. Search for product
6. Exit

2
Input product code:
12348
Input new price:
qwerty
Not a float
```

Генерація даних

	 id [PK] bigint 	product_code bigint 	date timestamp without time zone 
1	149254	12379	2020-06-17 19:05:50.836867
2	149255	12490	2020-10-20 13:18:09.901348
3	149256	12361	2020-01-21 18:16:30.541862
4	149257	12410	2020-05-08 04:11:55.945044
5	149258	12375	2020-07-15 05:42:03.707531
6	149259	12471	2020-05-04 17:50:41.062399
7	149260	12460	2020-09-04 07:06:25.600131
8	149261	12465	2020-06-18 22:37:04.796995
9	149262	12442	2020-10-07 09:47:12.559251
10	149263	12398	2020-09-19 03:10:24.425462
11	149264	12369	2020-10-31 07:17:35.546419
12	149265	12469	2020-01-20 23:15:05.482422
13	149266	12352	2020-03-08 21:04:00.896942
14	149267	12444	2020-03-04 17:02:25.218335
15	149268	12455	2020-07-12 15:40:12.612516

```
self.cursor.execute("INSERT INTO sales (product_code, date) "
                    "VALUES ({}, NOW() - (random() * "
                    "(NOW() + '1 year' - NOW())))".format(products[randint(0, len(products) - 1)))
```

Пошук даних

```
Select operation:
1. Add new purchase
2. Update product price
3. Delete purchase
4. Generate N purchases
5. Search for product
6. Exit

5
Input string to search
be
Code: 12351 | Description: beef | Price: 16.36 | Storage: 66
Code: 12368 | Description: canned beer | Price: 26.10 | Storage: 57
Code: 12370 | Description: coffee | Price: 19.48 | Storage: 32
Code: 12374 | Description: curd cheese | Price: 17.25 | Storage: 11
Code: 12384 | Description: processed cheese | Price: 9.88 | Storage: 24
Code: 12393 | Description: bottled beer | Price: 8.91 | Storage: 66
Code: 12402 | Description: hard cheese | Price: 12.72 | Storage: 16
Code: 12403 | Description: cream cheese | Price: 17.86 | Storage: 79
Code: 12419 | Description: sliced cheese | Price: 13.25 | Storage: 22
Code: 12422 | Description: specialty cheese | Price: 8.98 | Storage: 63
Code: 12431 | Description: house keeping products | Price: 12.13 | Storage: 41
Code: 12432 | Description: spread cheese | Price: 4.01 | Storage: 65
Code: 12440 | Description: artif. sweetener | Price: 26.74 | Storage: 14
Code: 12464 | Description: sweet spreads | Price: 3.46 | Storage: 77
Code: 12468 | Description: instant coffee | Price: 14.01 | Storage: 13
Code: 12471 | Description: soft cheese | Price: 22.70 | Storage: 40
Code: 12501 | Description: flower (seeds) | Price: 26.10 | Storage: 25
Time: 4.997014999389648 ms
```

```
self.cursor.execute("SELECT code, description, price, quantity FROM products CROSS JOIN storage "
                    "WHERE code = product_code AND description LIKE '%{}%'".format(search_string))
```

Ілюстрація коду з репозиторію git

За посиланням: <https://github.com/Melnikiva/databases>

Висновки

Ми навчилися створювати додатки, за допомогою яких можна користуватися розробленою раніше базою даних PostgreSQL.