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 та засоби управління»

ТЕМА: «СТВОРЕННЯ ДОДАТКУ БАЗИ ДАНИХ, ОРІЄНТОВАНОГО НА ВЗАЄМОДІЮ З СУБД POSTGRESQL»

Виконав: студент II курсу

ФПМ групи КВ-04

Оніщук А.О.

Київ – 2022

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).

Для програмування була використана мова Python і PyCharm Community Edition. В програмі були використані бібліотеки:

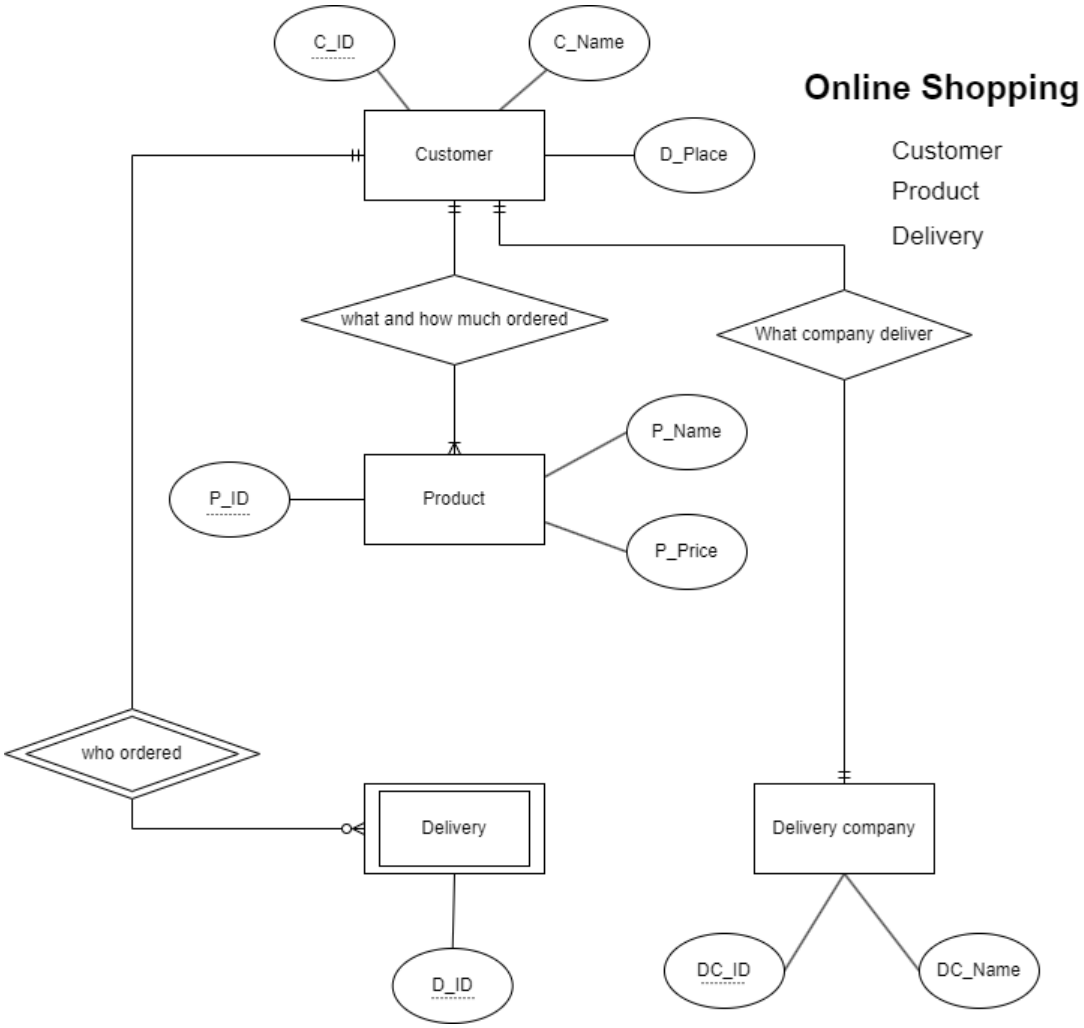
psycopg2 – для управління базою даних

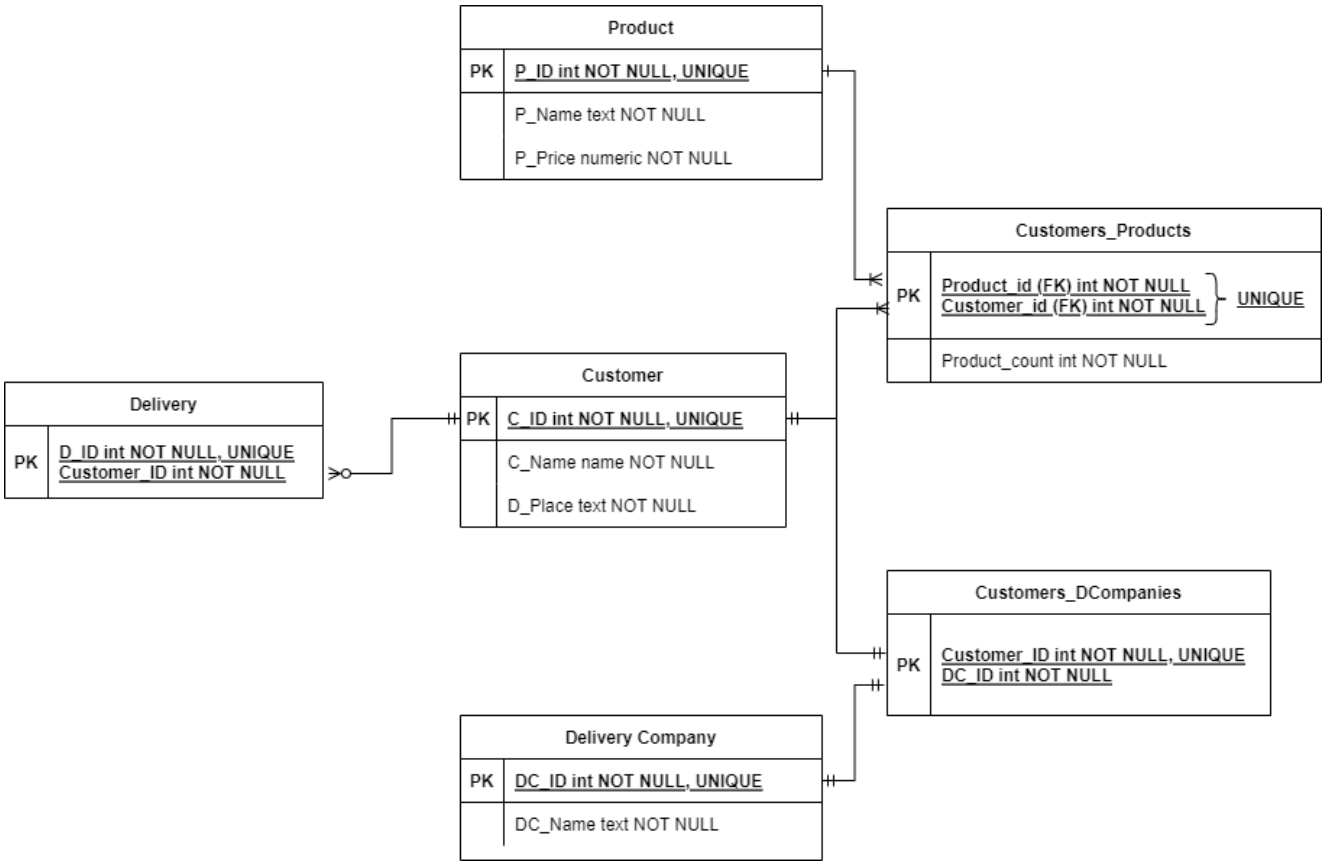
time – для заміру часу

Git-репозиторій: https://github.com/ViDjet-Git/Data-Base-University

Діаграма сутність-зв’язок та структура бази даних з лабораторної роботи №1

Варіант завдання: Для завдання була обрана галузь “Інтернет-магазин (Покупець, продукція, доставка, компанія доставки)”. Покупець обирає, що замовити, через яку компанію виконати доставку, і далі звертається до служби доставки, щоб вона виконала його замовлення.





Меню користувача

```
1. Add Element To Table
2. Update Element In Table
3. Delete Element From Table
4. Show Tables Names
5. Show Table
Enter number:
```

1 пункт – додає один елемент до таблиці

2 пункт – обновляє елемент з таблиці

3 пункт – видаляє елемент з таблиці

4 пункт – виводить назви таблиць бази даних

5 пункт – виводить на екран таблицю з її елементами

Пункт 1. Реалізація функцій

Операція виведення таблиці

```
Enter table name: Delivery Company
[INFO] Select all from Delivery Company
(1, 'CIAEC')
(2, 'DAXQG')
(3, 'YLHLK')
(4, 'IKXBU')
(5, 'RBQGX')
```

Операція додавання елементу

```
Enter table name: Customers_Products
Enter Customer ID: 5
Enter Product ID: 2
Enter Count of Product: 5
[INFO] product id=2 (Count: 5) was added to customer id=5
[INFO] Select all from Customers_Products
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
We have just added (5, 2, 5) to our Customers_Products table!
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Редагування елементу

```
3. Delete Element From Table
4. Show Tables Names
5. Show Table
Enter number: 2
Enter table name: Customer
Enter Customer ID: 1
Enter NEW Customer Name: Ivan
Enter NEW Delivery Place: Lviv
[INFO] Select customer by id=1
[INFO] updated customer id=1
[INFO] Select customer by id=1
---   ---   ---   ---   ---   ---   ---   ---   ---   ---   --
Change item from "[(1, 'Oleg', 'Kyiv')]" to "[(1, 'Ivan', 'Lviv')]" in table Customer
---   ---   ---   ---   ---   ---   ---   ---   ---   ---   --
```

Видалення елементу

```
2. Update Element In Table
3. Delete Element From Table
4. Show Tables Names
5. Show Table
Enter number: 3
Enter table name: Customer
Enter Customer ID: 2
[INFO] Select customers_products by customer id=2
[INFO] Select customers_dcompanies by customer id=2
[INFO] Select delivery by customer id=2
[INFO] deleted customer id=2
----------------------------------------------------------------
We have just removed id=2 from our Customer table
----------------------------------------------------------------
```

Помилка при невірному використанні операцій додавання видалення та оновлення

```
1. Add Element To Table
2. Update Element In Table
3. Delete Element From Table
4. Show Tables Names
5. Show Table
Enter number: 3
Enter table name: Customer
Enter Customer ID: 1
[INFO] Select customers_products by customer id=1
****************************************************************
[ERROR] Input incorrect
****************************************************************
```

```
1. Add Element To Table
2. Update Element In Table
3. Delete Element From Table
4. Show Tables Names
5. Show Table
Enter number: 2
Enter table name: Customer
Enter Customer ID: 5
[INFO] Select customer by id=5
****************************************************************
[ERROR] Input incorrect
****************************************************************
```

```
1. Add Element To Table
2. Update Element In Table
3. Delete Element From Table
4. Show Tables Names
5. Show Table
Enter number: 1
Enter table name: Customers_Products
Enter Customer ID: 7
Enter Product ID: 1
Enter Count of Product: 2
[INFO] Select customer by id=7
************************************************************
[ERROR] Input incorrect
************************************************************
```

Пункт 2. Генерація випадкових даних

Додавання 100 000 випадкових елементів

```
[INFO] Delivery company ('QFFLLODKST',) was added
[INFO] Delivery company ('IWABPOPBAE',) was added
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
We have just added 100000 random items to our Delivery Company table!
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Вибірка цих елементів(в таблиці до додавання ще було 5 елементів)

```
(100000, 'UNQJLGSVKS')
(100001, 'HNTVFQVWIV')
(100002, 'CGPSVAPDRN')
(100003, 'BTXWHWCSPF')
(100004, 'QFFLLODKST')
(100005, 'IWABPOPBAE')
```

Генерування 5 випадкових елементів

```
[INFO] Delivery company ('CIAEC',) was added
[INFO] Delivery company ('DAXQG',) was added
[INFO] Delivery company ('YLHLK',) was added
[INFO] Delivery company ('IKXBU',) was added
[INFO] Delivery company ('RBQGX',) was added
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
We have just added 5 random items to our Delivery Company table!
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

SQL запити для генерування даних (в нашому випадку два рядки довжиною два символи)

```
SELECT chr(trunc(65 + random()*25)::int) || chr(trunc(65 + random()*25)::int) from generate_series(1,10)
SELECT chr(trunc(65 + random()*25)::int) || chr(trunc(65 + random()*25)::int) from generate_series(1,10)
[INFO] customer ('HV',) (City: ('IE',)) was added
[INFO] customer ('QW',) (City: ('GE',)) was added
```

Пункт 3. Пошукові запити

Пошуковий запит 1 (За мінімальною ціною товару, за першою літерою імені покупця і за першою літерою продукту)

```
Enter min price: 100
Enter First Letter of Customer Name: G
Enter First Letter of Product Name: M

SEARCH_1:
C_Name, P_Name, P_Price, Product_count:
[('GFWVX', 'MKAMF', Decimal('883'), 2)]
```

Пошуковий запит 2 (За максимальним ID компанії доставки, за останньою літерою імені покупця і за частиною назви продукту)

```
Enter max DC_ID: 1000
Enter Last Letter of Customer Name: X
Enter Part of Product Name: E

SEARCH_2:
C_Name, D_Place DC_Name:
[('GFWVX', 'CCEAK', 'IKXBU')]
```

Пошуковий запит 3 (В діапазоні ID доставок, за останньою літерою імені покупця і за частиною назви продукту)

```
Enter min D_ID: 1
Enter max D_ID: 1000
Enter Last Letter of Customer Name: X
Enter Part of Product Name: E

SEARCH_3:
C_Name, D_ID, D_Place
[('GFWVX', 1, 'CCEAK')]
Search 3 Finished in 2.4521 milliseconds
```

Генерування SQL запитів

```
sql = 'SELECT "C_Name", "P_Name", "P_Price", "Product_count"' \
        'FROM "Customers_Products" ' \
        'JOIN "Customer" ' \
        'ON "Customer"."C_ID" = "Customers_Products"."Customer_id" ' \
        'JOIN "Product" ' \
        'ON "Product"."P_ID" = "Customers_Products"."Product_id" ' \
        'WHERE "P_Price" > {} ' \
        'AND "C_Name" LIKE \'{}%\' ' \
        'AND "P_Name" LIKE \'{}%\' '.format(min_price, first_letter_c_name, first_letter_p_name)
```

```
sql = 'SELECT "C_Name", "D_Place", "DC_Name" ' \
        'FROM "Customers_DCompanies" ' \
        'JOIN "Customer" ' \
        'ON "Customer"."C_ID" = "Customers_DCompanies"."Customer_ID" ' \
        'JOIN "Delivery Company" ' \
        'ON "Delivery Company"."DC_ID" = "Customers_DCompanies"."DC_ID" ' \
        'WHERE "Customers_DCompanies"."DC_ID" < {} ' \
        'AND "C_Name" LIKE \'%{}\' ' \
        'AND "D_Place" LIKE \'%{}%\''.format(max_d_id, last_c_name, inside_d_place)
```

```
sql = 'SELECT "C_Name", "D_ID", "D_Place" ' \
        'FROM "Delivery" ' \
        'JOIN "Customer" ' \
        'ON "Customer"."C_ID" = "Delivery"."Customer_ID" ' \
        'WHERE "D_ID" >= {} AND "D_ID" <= {} ' \
        'AND "C_Name" LIKE \'%{}\' ' \
        'AND "D_Place" LIKE \'%{}%\''.format(min_id, max_id, l_c_name, i_d_place)
```

Пункт 4. Шаблон програмного коду

model.py з описом функцій

```
import backend as b
import time


class ModelBasic(object):

    #додавання покупця
    def add_new_customer(self, connection, customerName, deliveryPlace):
        b.add_customer(connection, customerName, deliveryPlace)

    #редагування покупця
    def update_customer(self, connection, _id, customerName, deliveryPlace):
        b.update_customer(connection, _id, customerName, deliveryPlace)

    #видалення покупця
    def delete_customer(self, connection, _id):
        b.delete_customer(connection, _id)
```

```
    #додання продукту
    def add_new_product(self, connection, productName, productPrice):
        b.add_product(connection, productName, productPrice)

    #редагування продукту
    def update_product(self, connection, _id, productName, productPrice):
        b.update_product(connection, _id, productName, productPrice)

    #видалення продукту
    def delete_product(self, connection, _id):
        b.delete_product(connection, _id)

    #додавання компанії
    def add_new_delivery_company(self, connection, companyName):
        b.add_delivery_company(connection, companyName)

    #редагування компанії
    def update_delivery_company(self, connection, _id, companyName):
        b.update_delivery_company(connection, _id, companyName)

    #видалення компанії
    def delete_delivery_company(self, connection, _id):
        b.delete_delivery_company(connection, _id)

    #доданя елементу в таблицю Покупець/Продукти
    def add_product_to_customer(self, connection, customerID, productID, count):
        b.add_product_to_customer(connection, customerID, productID, count)

    #видалення елементу з таблиці Покупець/Продукти
    def delete_product_from_customer(self, connection, customer_id, product_id):
        b.delete_product_from_customer(connection, customer_id, product_id)

    #доданя елементу в таблицю Покупець/Компанія доставки
    def add_delivery_company_to_customer(self, connection, customerID, companyID):
        b.add_delivery_company_to_customer(connection, customerID, companyID)

    #редагування елементу в таблиці Покупець/Компанія доставки
    def update_delivery_company_to_customer_by_customer_id(self, connection,
customerID, companyID):
        b.update_delivery_company_to_customer_by_customer_id(connection,
customerID, companyID)

    #видалення елементу з таблиці Покупець/Компанія доставки
    def delete_delivery_company_from_customer(self, connection, _id):
        b.delete_delivery_company_from_customer(connection, _id)

    #доданя елементу в таблицю Доставка
    def add_delivery(self, connection, customerID):
        b.add_delivery(connection, customerID)

    #видалення елементу з таблиці Доставка
    def delete_delivery(self, connection, _id):
        b.delete_delivery(connection, _id)

    #вибірка всіх елементів таблиці
    def select_all(self, connection, tableName):
        return b.select_all(connection, tableName)

    #вибірка елемента покупця за його id
    def select_customer_by_id(self, connection, _id):
        return b.select_customer_by_id(connection, _id)

    # вибірка елемента продукту за його id
    def select_product_by_id(self, connection, _id):
        return b.select_product_by_id(connection, _id)

    # вибірка елемента компанії доставки за його id
```

```python
    def select_delivery_company_by_id(self, connection, _id):
        return b.select_delivery_company_by_id(connection, _id)

    # вибірка елемента Покупець/Продукт за id покупця
    def select_product_to_customer_by_customer_id(self, connection, _id):
        return b.select_product_to_customer_by_customer_id(connection, _id)

    # вибірка елемента Покупець/Продукт за id продукту
    def select_product_to_customer_by_product_id(self, connection, _id):
        return b.select_product_to_customer_by_product_id(connection, _id)

    # вибірка елемента Покупець/Компанія Доставки за id покупця
    def select_company_to_customer_by_customer_id(self, connection, _id):
        return b.select_company_to_customer_by_customer_id(connection, _id)

    # вибірка елементів Покупець/Компанія Доставки за id компанії доставки
    def select_company_to_customer_by_company_id(self, connection, _id):
        return b.select_company_to_customer_by_company_id(connection, _id)

    # вибірка елемента Доставки за id
    def select_delivery_by_id(self, connection, _id):
        return b.select_delivery_by_id(connection, _id)

    # вибірка елемента Доставки за id покупця
    def select_delivery_by_customer_id(self, connection, _id):
        return b.select_delivery_by_customer_id(connection, _id)

    # видалення всієї таблиці
    def delete_all_from_table(self, connection, tableName):
        b.delete_all_from_table(connection, tableName)

    # генерування чисел
    def auto_gen_int(self, connection, max_val, rows_number):
        return b.auto_gen_int(connection, max_val, rows_number)

    # генерування рядків
    def auto_gen_char(self, connection, str_len, rows_number):
        return b.auto_gen_char(connection, str_len, rows_number)

    # пошук 1 і замір часу
    def search_1(self, connection, min_price, f_c_name, f_p_name):
        start_time = time.perf_counter()
        b.search1(connection, min_price, f_c_name, f_p_name)
        end_time = time.perf_counter()
        run_time = (end_time - start_time) * 1000
        print(f"Search 1 Finished in {run_time:.4f} milliseconds")

    # пошук 2 і замір часу
    def search_2(self, connection, max_id, l_c_name, i_d_place):
        start_time = time.perf_counter()
        b.search2(connection, max_id, l_c_name, i_d_place)
        end_time = time.perf_counter()
        run_time = (end_time - start_time) * 1000
        print(f"Search 1 Finished in {run_time:.4f} milliseconds")

    # пошук 3 і замір часу
    def search_3(self, connection, min_id, max_id, l_c_name, i_d_place):
        start_time = time.perf_counter()
        b.search3(connection, min_id, max_id, l_c_name, i_d_place)
        end_time = time.perf_counter()
        run_time = (end_time - start_time) * 1000
        print(f"Search 3 Finished in {run_time:.4f} milliseconds")
```