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»

Виконала: студентка III курсу

ФПМ групи КВ-84

Величко М. М.

Перевірив:

Київ – 2020

Завдання:

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).

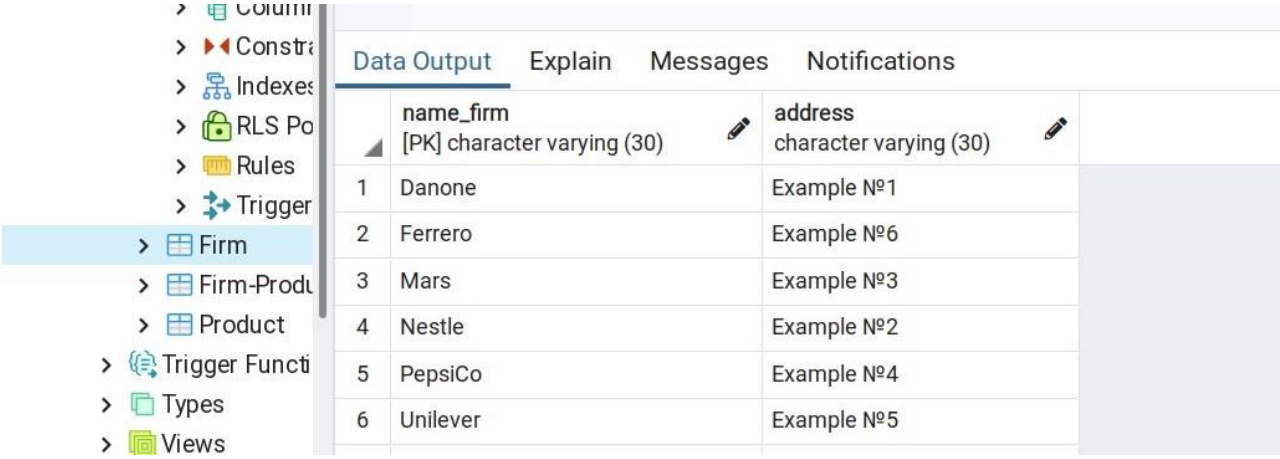
Посилання на репозиторій:

https://github.com/maryna-velychko/My_proj

Структура бази даних з лабораторної №1:

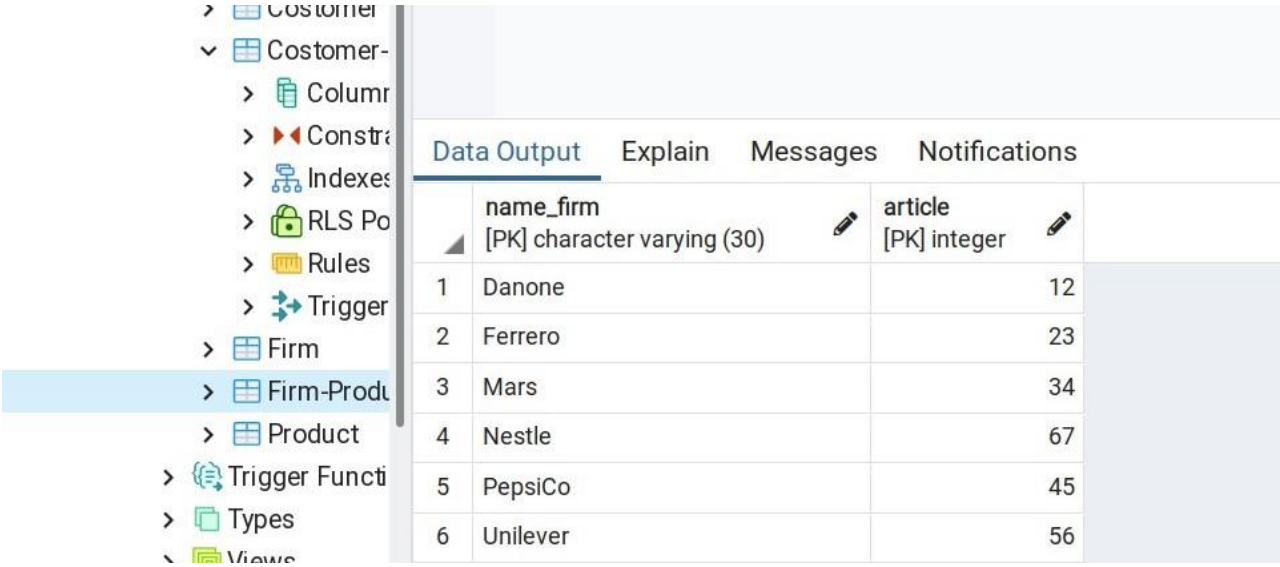
Costomer:

Firm:



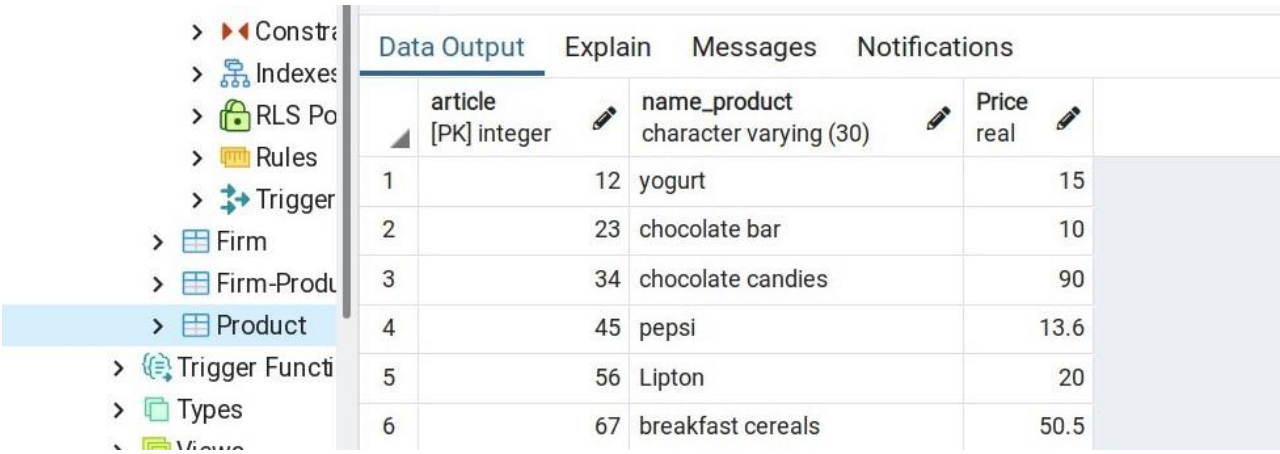
	name_firm [PK] character varying (30)	address character varying (30)
1	Danone	Example №1
2	Ferrero	Example №6
3	Mars	Example №3
4	Nestle	Example №2
5	PepsiCo	Example №4
6	Unilever	Example №5

Firm/Product



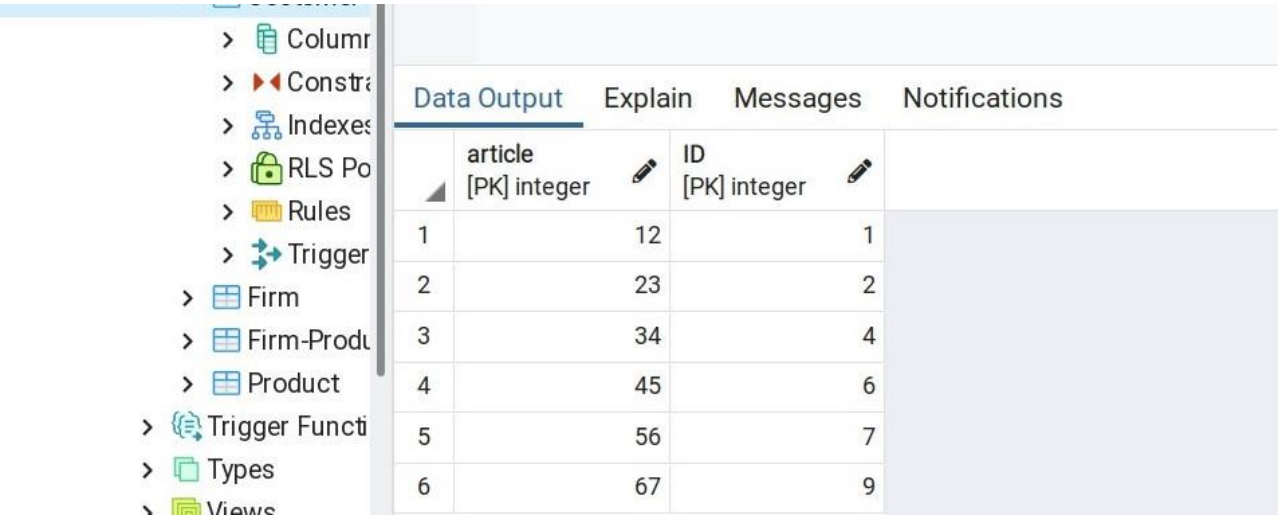
	name_firm [PK] character varying (30)	article [PK] integer
1	Danone	12
2	Ferrero	23
3	Mars	34
4	Nestle	67
5	PepsiCo	45
6	Unilever	56

Product:



	article [PK] integer	name_product character varying (30)	Price real
1	12	yogurt	15
2	23	chocolate bar	10
3	34	chocolate candies	90
4	45	pepsi	13.6
5	56	Lipton	20
6	67	breakfast cereals	50.5

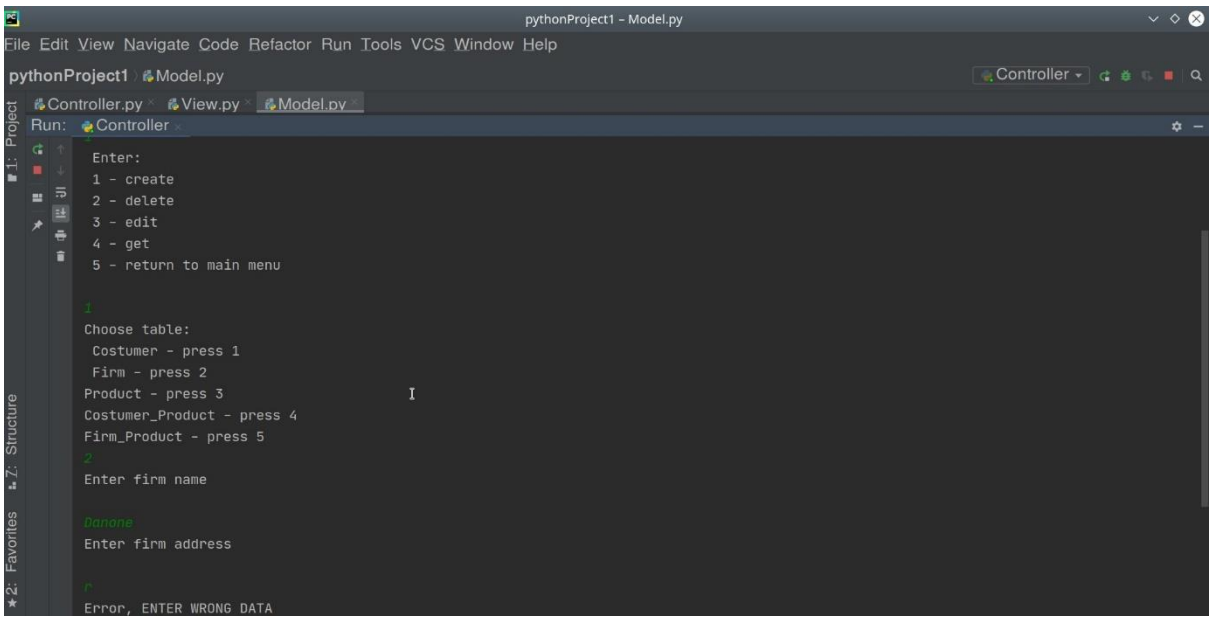
Costomer/Product:



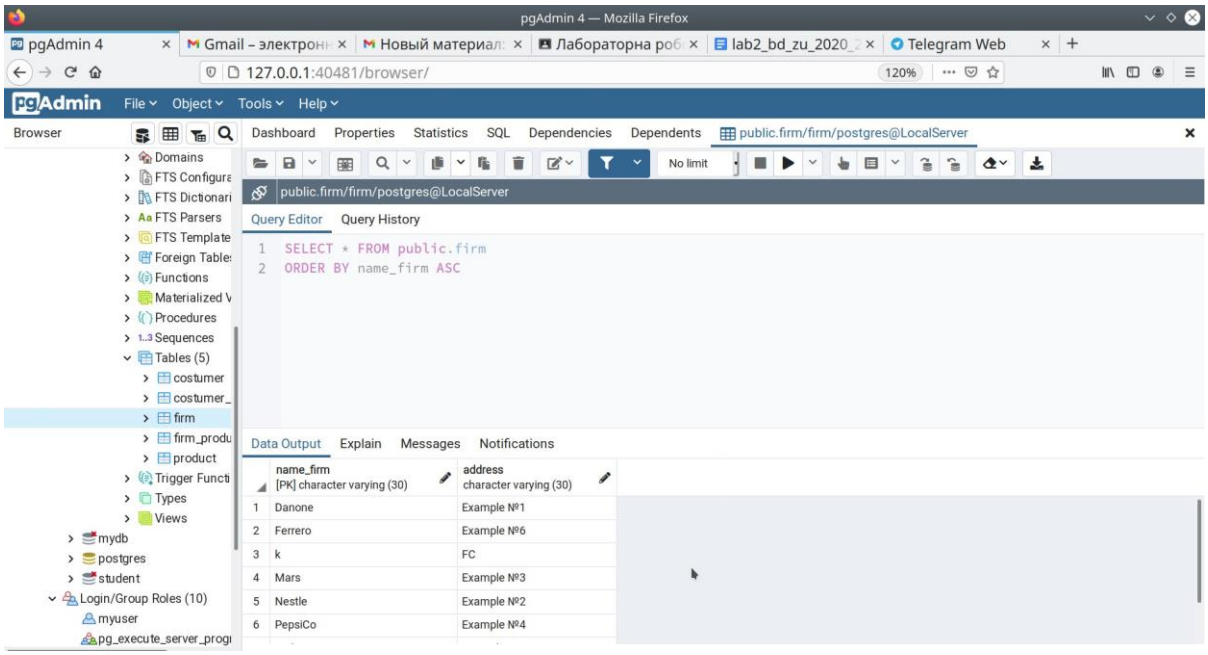
Вимоги до пункту №1 деталізованого завдання:

Ілюстрації обробки виняткових ситуацій (помилок) при уведенні/вилучення даних:

Помилка при додаванні даних до таблиці.



Атрибут name_firm є головним ключем таблиці costumer, значення Danone вже існує)



Для інших таблиць також передбачені подібні помилки:

Для редагування:

```
def edit(cursor, table, parameter_1, parameter_2, parameter_3):
    try:
        if table == 1:
            cursor.execute("UPDATE costumer SET name_costumer = %s, email = %s WHERE id = %s", (parameter_1, parameter_2, [parameter_3]
        elif table == 2:
            cursor.execute("UPDATE firm SET name_firm = %s, address = %s WHERE name_firm = %s", (parameter_1, parameter_2, parameter_3,)
        elif table == 3:
            cursor.execute("UPDATE product SET name_product = %s, price = %s WHERE article = %s", (parameter_1, float(parameter_2), para
        elif table == 4:
            cursor.execute("UPDATE costumer_product SET article = %s, id = %s WHERE article = %s", (parameter_1, parameter_2, parameter_
        elif table == 5:
            cursor.execute("UPDATE firm_product SET name_firm = %s, article = %s WHERE article = %s", (parameter_1, parameter_2, paramet
    except errors.ForeignKeyViolation:
        print("Error, ENTER WRONG DATA\n")
```

Для додавання:

```
def create(cursor, table, parameter_1, parameter_2):
    try:
        if table == 1:
            cursor.execute("INSERT INTO costumer (name_costumer, email) VALUES (%s, %s)", (parameter_1, parameter_2))
        elif table == 2:
            cursor.execute("INSERT INTO firm (name_firm, address) VALUES (%s, %s)", (parameter_1, parameter_2))
        elif table == 3:
            cursor.execute("INSERT INTO product (name_product, price) VALUES (%s, %s)", (parameter_1, float(parameter_2)))
        elif table == 4:
            cursor.execute("INSERT INTO costumer_product (id, article) VALUES (%s, %s)", ([parameter_1], [parameter_2]))
        elif table == 5:
            cursor.execute("INSERT INTO firm_product (name_firm, article) VALUES (%s, %s)", (parameter_1, [parameter_2]))
    except errors.ForeignKeyViolation:
        print("Error, ENTER WRONG DATA\n")
```

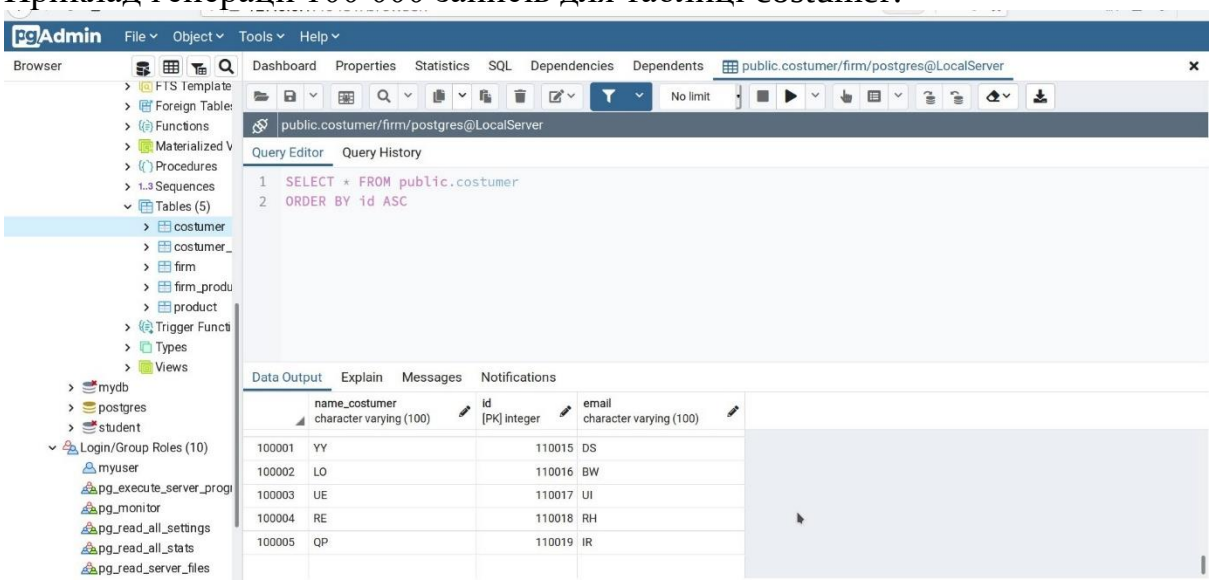
Ілюстрації валідації даних при уведенні користувачем:

```
Run: Controller
 5 - return to main menu

1
Choose table:
 Costumer - press 1
 Firm - press 2
Product - press 3
Costumer_Product - press 4
Firm_Product - press 5
1
Enter costumer name

Polly
Enter costumer e-mail

p.@example
 Enter:
 1 - create
 2 - delete
 3 - edit
 4 - get
 5 - return to main menu
```

```
Controller.py  View.py  Model.py
Run: Controller

4
Enter name table:
 costumer
 firm
product
costumer_product
firm_product
costumer
('Jonny', 1, '1@example.com')
('Mary', 2, '5@example.com')
('John', 9, '8@example.com')
('Ann\n', 7, '0@example.com\n')
('Polly', 19, 'p.@example')
 Enter:
 1 - create
 2 - delete
 3 - edit
 4 - get
 5 - return to main menu
```

Вимоги до пункту №2 деталізованого завдання:

Копії екрану (ілюстрації) з фрагментами згенерованих даних таблиць
Приклад генерації 100 000 записів для таблиці costumer:



Копії SQL-запитів:

Кількість значень, що згенерують функції задається користувачем (параметр number)

```python
def random_generation(cursor, table, number):
    try:
        if table == 1:
            cursor.execute("INSERT INTO costumer (name_costumer, email) "
                           " SELECT chr(trunc(65 + random()*25)::int)|| chr(trunc(65 + random()*25)::int),"
                           " chr(trunc(65 + random()*25)::int)|| chr(trunc(65 + random()*25)::int)"
                           " FROM generate_series(1, %s)", [number])
        elif table == 2:
            cursor.execute("INSERT INTO firm (name_firm, address) "
                           " SELECT chr(trunc(65 + random()*25)::int)|| chr(trunc(65 + random()*25)::int),"
                           " chr(trunc(65 + random()*25)::int)|| chr(trunc(65 + random()*25)::int)"
                           " FROM generate_series(1, %s)", [number])
        elif table == 2:
            cursor.execute("INSERT INTO firm (name_firm, address) "
                           " SELECT chr(trunc(65 + random()*25)::int)|| chr(trunc(65 + random()*25)::int),"
                           " chr(trunc(65 + random()*25)::int)|| chr(trunc(65 + random()*25)::int)"
                           " FROM generate_series(1, %s)", [number])
        elif table == 3:
            cursor.execute("INSERT INTO product (name_product, price) "
                           " SELECT chr(trunc(65 + random()*25)::int)|| chr(trunc(65 + random()*25)::int),"
                           " trunc(random()*100 + random())::real FROM generate_series(1, %s)",
                           [number])
        elif table == 4:
            cursor.execute("INSERT INTO costumer_product (article, id) SELECT article, id FROM product TABLESAMPLE"
                           " bernoulli(%s), costumer TABLESAMPLE bernoulli(%s)", (number, number,))
```

random_generation() › try › if table == 1

Користувач задає процент вибірки значень з ключових таблиць:

```python
        elif table == 4:
            cursor.execute("INSERT INTO costumer_product (article, id) SELECT article, id FROM product TABLESAMPLE"
                           " bernoulli(%s), costumer TABLESAMPLE bernoulli(%s)", (number, number,))
        elif table == 5:
            cursor.execute("INSERT INTO firm_product (name_firm, article) SELECT name_firm, article"
                           " FROM firm TABLESAMPLE"
                           " bernoulli(%s), costumer TABLESAMPLE bernoulli(%s)", (number, number,))
    except psycopg2.errors.UniqueViolation:
        print("Error\n")


def select_function(cursor, parameter, item):
```

random_generation() › try › if table == 1

9: Git TODO 6: Problems Terminal Python Console Event Log

62:68 LF UTF-8 4 spaces Python 3.8 (pythonProject1) master

Вимоги до пункту №3 деталізованого завдання:

Ілюстрації уведення пошукового запиту та результатів виконання запитів:
Запит 1(Вивести ім’я клієнта, що придбав товар з заданим артикулом):

```
 Enter:
 1 - for testing task
 2 - for task 2
 3 - for 3
 4 - exit

3
 Enter: 1, 2 or 3 for testing select
1
Enter the product article to display the name of the customer who bought it
12
Name costumer
('Jonny',)
Request processing time  0.035886386999865936
 Enter:
 1 - for testing task
 2 - for task 2
 3 - for 3
 4 - exit
```

Запит 2(Вивести адресу фірми, що постачає товар з заданим артикулом):

```
 Enter:
 1 - for testing task
 2 - for task 2
 3 - for 3
 4 - exit

3
 Enter: 1, 2 or 3 for testing select
2
Enter the product article to display the address of the firm who produces it
12
Name firm Address
('Example №1',)
Request processing time  0.004365266000604606
 Enter:
 1 - for testing task
 2 - for task 2
 3 - for 3
 4 - exit
```

Запит 3(Виведення загальної суми покупок клієнта з вказаним айді)

```
 Enter:
 1 - for testing task
 2 - for task 2
 3 - for 3
 4 - exit

3
 Enter: 1, 2 or 3 for testing select
3
Enter the customer ID to find out the total cost of his purchases
2
Price
(53.7,)
Request processing time  0.2125439940000433
 Enter:
 1 - for testing task
 2 - for task 2
 3 - for 3
 4 - exit
```

Копії SQL-запитів:

```
cursor.execute("with table_1 as (SELECT id FROM costumer_product"
               " where costumer_product.article = %s)"
               " SELECT name_costumer FROM costumer inner join table_1 on costumer.id = table_1.id",
               [parameter])
```

```
105        cursor.execute("with table_1 as (SELECT name_firm FROM firm_product"
106                       " where firm_product.article = %s)"
107                       " SELECT address FROM firm inner join table_1 on firm.name_firm = table_1.name_firm",
108                       [parameter])
```

```
116        cursor.execute("with table_1 as (SELECT article FROM costumer_product"
117                       " where costumer_product.id = %s)"
118                       " SELECT sum(price) FROM product inner join table_1 on product.article = table_1.article",
119                       [parameter])
```