A black and white drawing of a building

Description automatically generated

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського» ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

# Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Лабораторна робота №1**

з дисципліни **Бази даних і засоби управління**

*на тему: “Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL”*

Виконала:

студентка ІІI курсу

групи КВ-21

Оваденко В.М.

Перевірив:

**Київ – 2024**

**Репозиторій GitHub:** <https://github.com/Clemence25/BDZU>

**Мета:** здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

**Завдання:**

1. Розробити модель «сутність-зв’язок» предметної галузі, обраної студентом самостійно, відповідно до пункту «Вимоги до ER-моделі».
2. Перетворити розроблену модель у схему бази даних (таблиці) PostgreSQL.
3. Виконати нормалізацію схеми бази даних до третьої нормальної форми (3НФ).
4. Ознайомитись із інструментарієм PostgreSQL та pgAdmin 4 та внести декілька рядків даних у кожну з таблиць засобами pgAdmin 4.

**Опис предметної області**

Дана предметна область – система управління замовленнями та доставкою косметики. Вона охоплює сукупність процесів і інструментів, які спрямовані на обробку замовлень клієнтів, управління інформацією про товари, організацію доставок та відстеження статусу замовлень. Система забезпечує зручний інтерфейс для клієнтів, де вони можуть замовляти косметичні товари, а також надає інструменти для компаній, щоб контролювати всі етапи від оформлення замовлення до його доставки.

**Опис сутностей**

Для побудови бази даних обраної області, були виділені такі сутності:

1. Клієнт(Customer)

Атрибути: ідентифікатор клієнта, ім’я, електронна скринька, номер телефону.

Призначення: збереження даних щодо клієнтів.

1. Замовлення(Order)

Атрибути: ідентифікатор замовлення, дата.

Призначення: збереження даних щодо замовлень.

1. Продукт(Product)

Атрибути: ідентифікатор товару, назва, ціна

Призначення: збереження даних щодо товарів.

1. Доставка(Delivery)

Атрибути: ідентифікатор доставки, адреса, статус доставки.

Призначення: збереження даних щодо доставок.

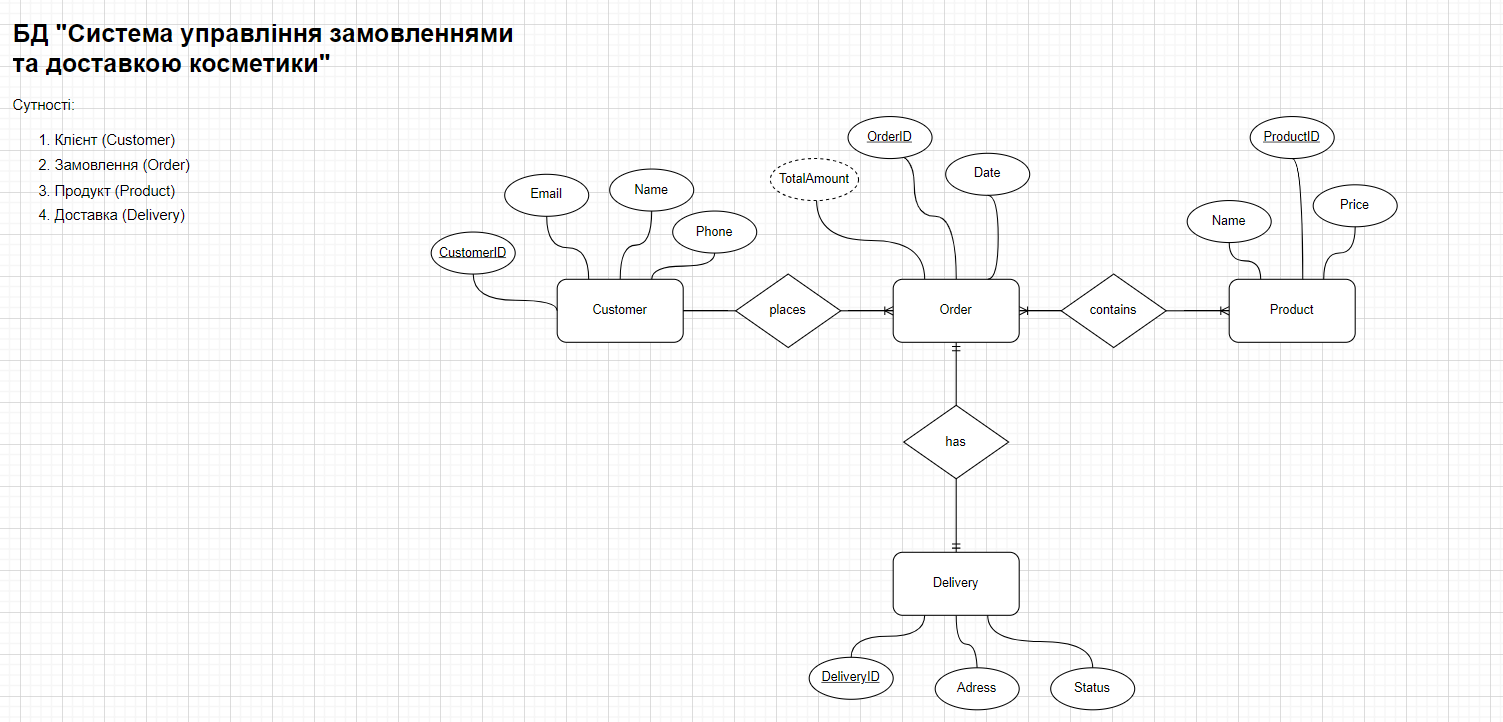
**Опис зв’язків між сутностями**

Зв’язок «Клієнт» - «Замовлення» є зв’язком 1:N. Один клієнт може мати багато замовлень, але кожне замовлення прив’язано лише до одного клієнта.

Зв’язок «Замовлення» - «Продукт» є зв’язком N:M. Одне замовлення може містити багато товарів, і один товар може входити до складу багатьох замовлень.

Зв’язок «Замовлення» - «Доставка» є зв’язком 1:1. Кожне замовлення може мати тільки одну доставку, і кожна доставка відповідає лише одному замовленню.

Графічне подання концептуальної моделі «Сутність-зв’язок» зображено на рисунку 1.



*Рисунок 1 – ER-діаграма, побудована за нотацією Пташиної лапки (Crow’s foot)*

**Перетворення концептуальної моделі у логічну модель та схему бази даних**

Сутність Customer перетворено в таблицю Customer з первинним ключем customer\_id та атрибутами name, email, phone.

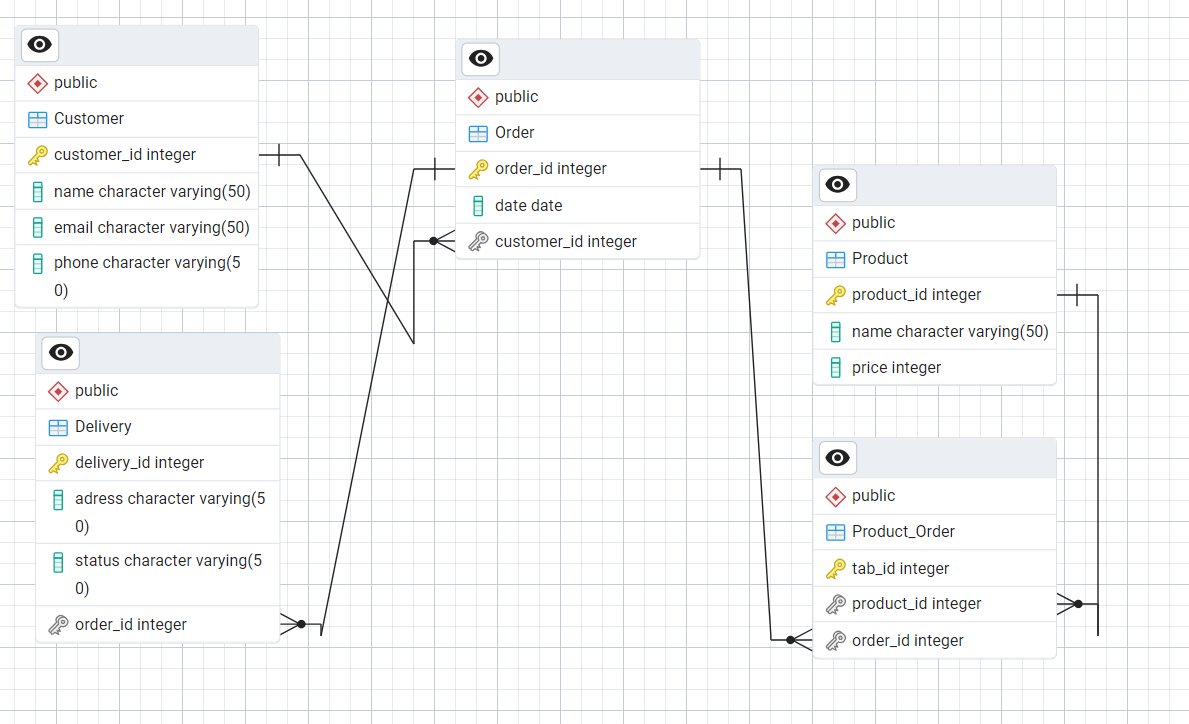
Сутність Order перетворено в таблицю Order з первинним ключем order\_id та атрибутом date та зовнішнім ключем customer\_id.

Сутність Delivery перетворено в таблицю Delivery з первинним ключем delivery\_id та атрибутами adress, status, та зовнішнім ключем order\_id.

Сутність Product перетворено в таблицю Product з первинним ключем product\_id та атрибутами name, price.

Оскільки в логічній моделі безпосередній зв’язок N:M є неможливим, а в концептуальній моделі він існує між сутностями Product i Order, то для його реалізації було створено таблицю Product\_Order, з первинним ключем tab\_id, та зовнішніми ключами product\_id i teacher\_id.

Графічне подання логічної моделі «Сутність-зв’язок» зображено на рисунку 2.



*Рисунок 2 – Схема бази даних*

Таблиця 1 ілюструє детальний перехід від однієї моделі до іншої.

*Таблиця 1 – Опис об’єктів бази даних*

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| *Customer* – містить дані про клієнтів | *customer\_id* – унікальний ідентифікатор студента **(UNIQUE, NOT NULL)** | *integer* (число) |
| *name* – ім’я клієнта **(NOT NULL)** | *character varying* (рядок) |
| *email* – електронна скринька клієнта **(NOT NULL)** | *character varying* (рядок) |
| *phone -* мобільний телефон клієнта **(NOT NULL)** | *character varying* (рядок) |
| *Order* – містить дані замовлень | *order\_id* – унікальний ідентифікатор замовлення **(UNIQUE, NOT NULL)** | *integer* (число) |
| *date* – дата **(NOT NULL)** | *date* (дата) |
| *Delivery* – містить дані доставок | *delivery\_id* – унікальний ідентифікатор доставки **(UNIQUE, NOT NULL)** | *integer* (число) |
| *address* – адреса для доставки **(NOT NULL)** | *character varying* (рядок) |
| *status* – статус доставки **(NOT NULL)** | *character varying* (рядок) |
| *Product* – містить дані про товари | *product\_id* – унікальний ідентифікатор товару **(UNIQUE, NOT NULL)** | *integer* (число) |
| *name* – назва товару **(NOT NULL)** | *character varying* (рядок) |
| *price* – ціна **(NOT NULL)** | *integer* (число) |
| *Product\_Order* – містить інформацію про відповідність товару та замовлення | *tab\_id* – унікальний ідентифікатор відповідності **(UNIQUE, NOT NULL)** | *integer* (число) |
| *product\_id* – ідентифікатор товару **(NOT NULL)** | *integer* (число) |
| *order\_id* – ідентифікатор замовлення **(NOT NULL)** | *integer* (число) |

**Функціональні залежності для кожної таблиці**

1. Customer

**Primary key:** customer\_id

**Candidate key:** email

**Candidate key:** phone

customer\_id → {name, email, phone}

email → {customer\_id, name, phone}

phone → {customer\_id, name, email}

email → phone

phone → email

1. Order

**Primary key:** order\_id

**Foreign key:** customer\_id

order\_id → {date, customer\_id}

1. Product

**Primary key:** product\_id

**Candidate key:** name

product\_id → {name, price}

name → {product\_id, price}

1. Delivery

**Primary key:** delivery\_id

**Foreign key:** order\_id

delivery\_id → {adress, status, order\_id}

order\_id → {delivery\_id, adress, status}

1. Product\_Order

**Primary key:** tab\_id

**Foreign keys:** product\_id, order\_id

tab\_id → {product\_id, order\_id}

Ці функціональні залежності вказують на те, які атрибути в кожній таблиці є функціонально залежними від ключових атрибутів. Це важливо для нормалізації бази даних, оскільки дозволяє уникати дублювання даних і забезпечувати цілісність інформації.

Функціональні залежності допомагають визначити, які атрибути можуть бути унікально ідентифіковані через інші атрибути. Наприклад, у таблиці **Customer**ідентифікатор *customer\_id* визначає такі атрибути, як ім'я, електронна пошта та телефон.

**Відповідність схеми нормальним формам**

1. Щоб задовольнити умови 1НФ, кожен атрибут у таблиці має бути атомарним, тобто:
   * Кожна клітинка містить єдине значення;
   * Кожен запис є унікальним.

У цій схемі всі атрибути, такі як name, email, phone, adress тощо, є атомарними. Кожен запис має унікальний ідентифікатор (наприклад, customer\_id, order\_id), що забезпечує унікальність рядків. Отже, схема *відповідає 1НФ*.

1. Щоб схема відповідала 2НФ, повинні виконуватись умови:
   * Схема перебуває в 1НФ;
   * Кожен неключовий атрибут функціонально залежить від цілого ключа.

У цій схемі кожна таблиця має унікальний ідентифікатор (ключ). Наприклад, у таблиці Customer неключові атрибути: name, email, phone функціонально залежать від первинного ключа customer\_id. Оскільки, кожний атрибут у кожній таблиці залежить від цього одного унікального ідентифікатора, ця схема *відповідає 2НФ*.

3. Щоб схема відповідала 3НФ, повинні виконуватись умови:

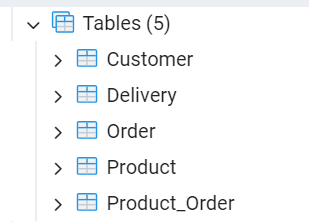
* Схема перебуває в 2НФ;
* Кожен неключовий атрибут функціонально залежить від усього ключа, а не його частини.

Дана схема в НФ2, і хоча присутні транзитивні залежності, вони стосуються кандидатних ключів. Тому їх можна не враховувати. Таким чином, схема *відповідає 3НФ*.

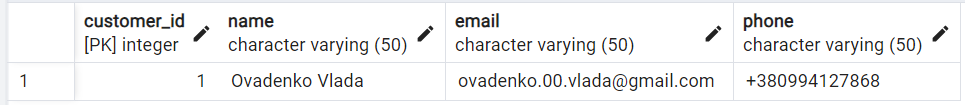
Висновок:

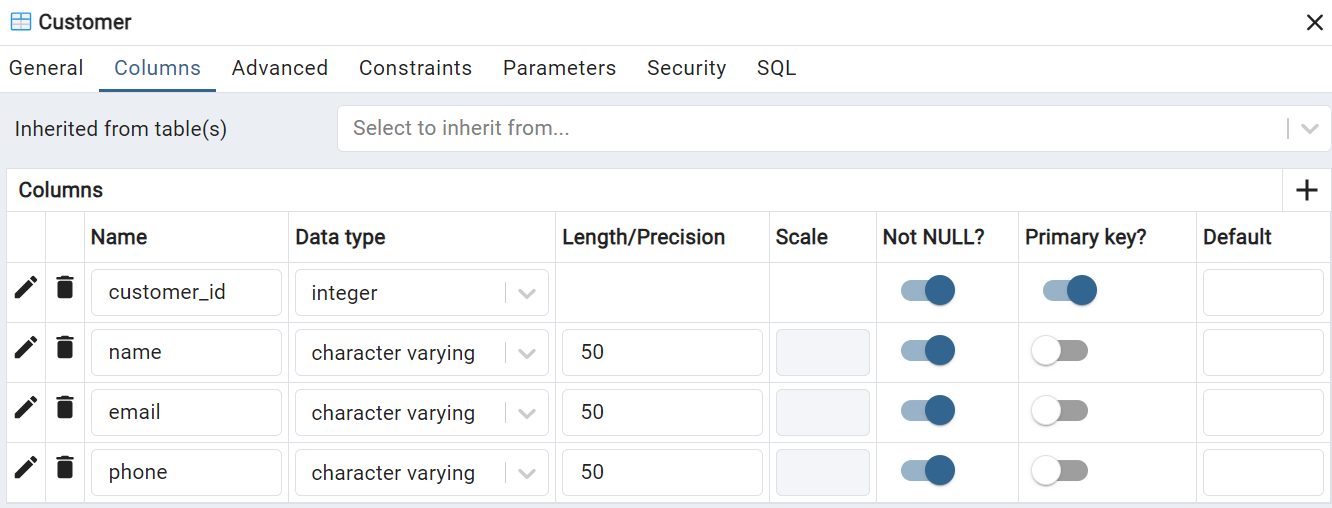
Схема бази даних відповідає нормальним формам 1НФ, 2НФ та 3НФ. Вона добре структурована і нормалізована, що сприяє ефективному та надійному зберіганню та обробці даних.

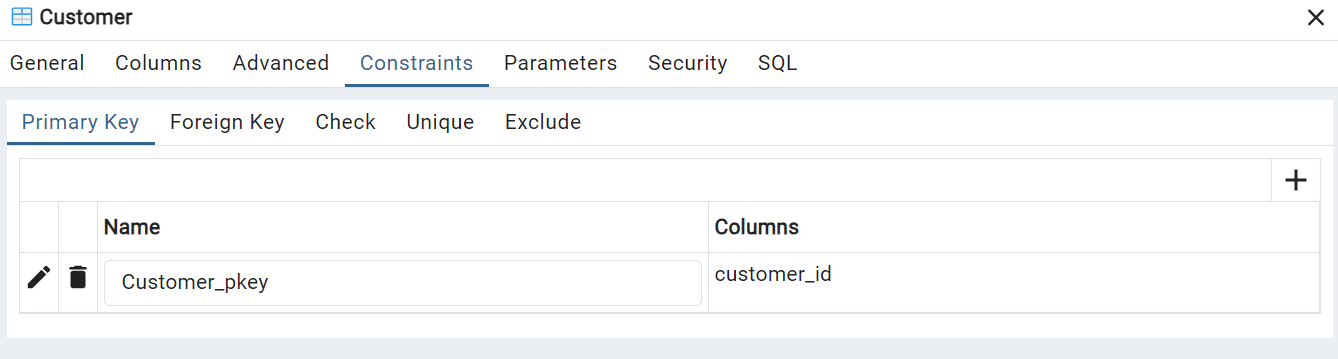
**Таблиці бази даних у pgAdmin4**

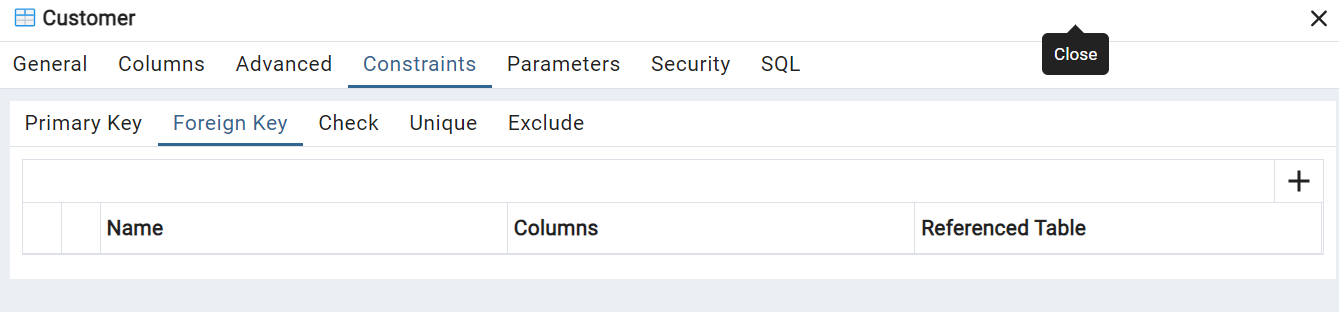
******

***Customer***

******

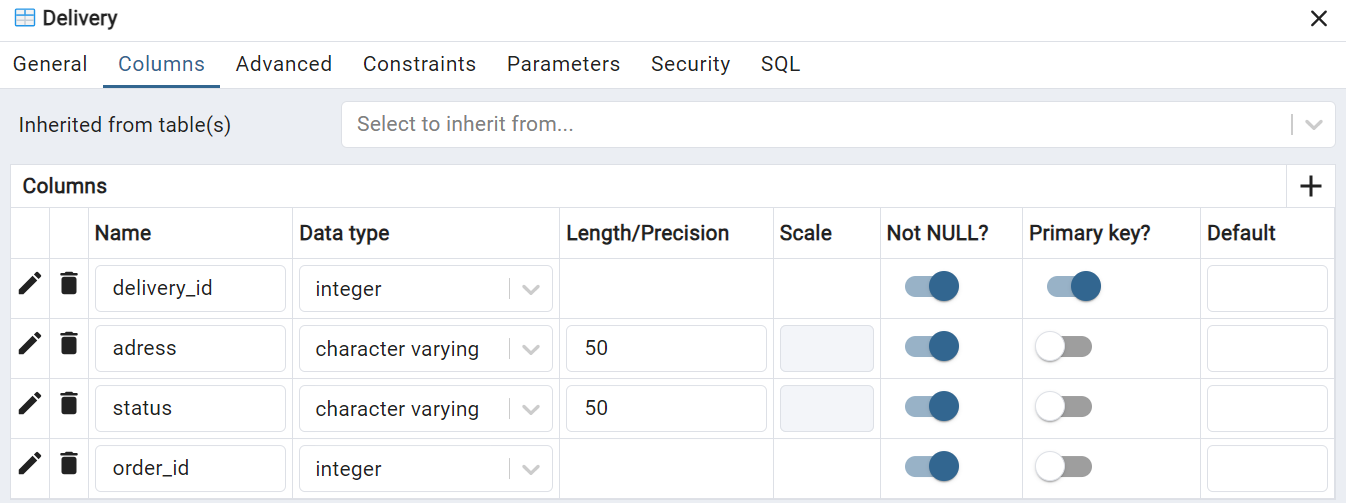


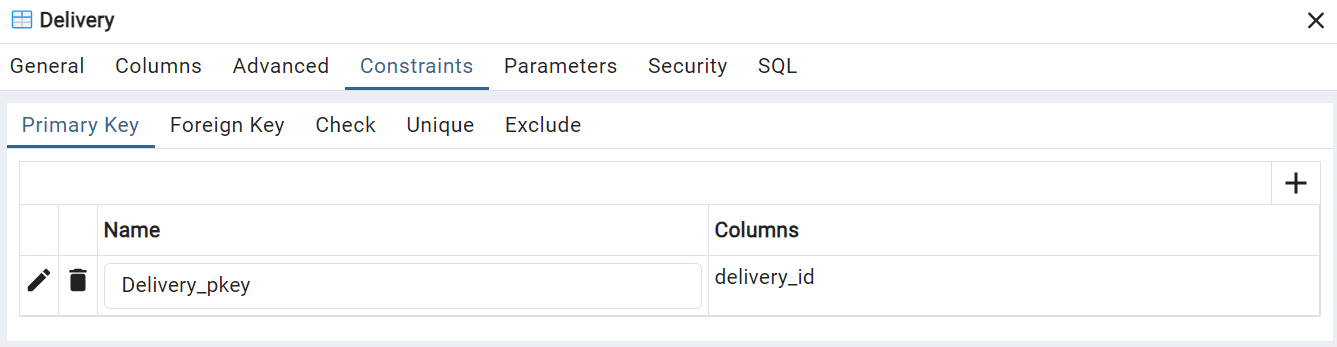


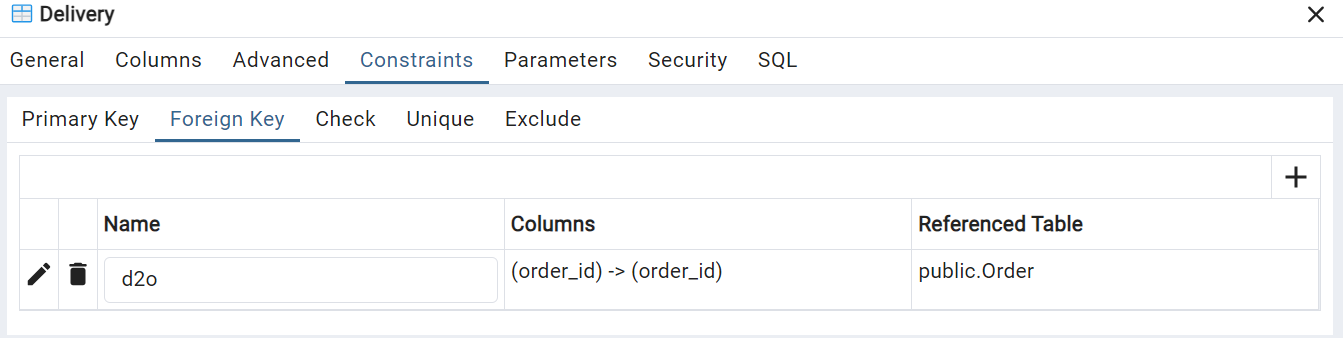


***Delivery***

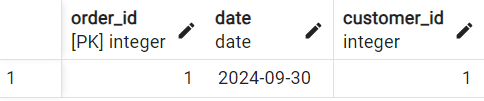
******

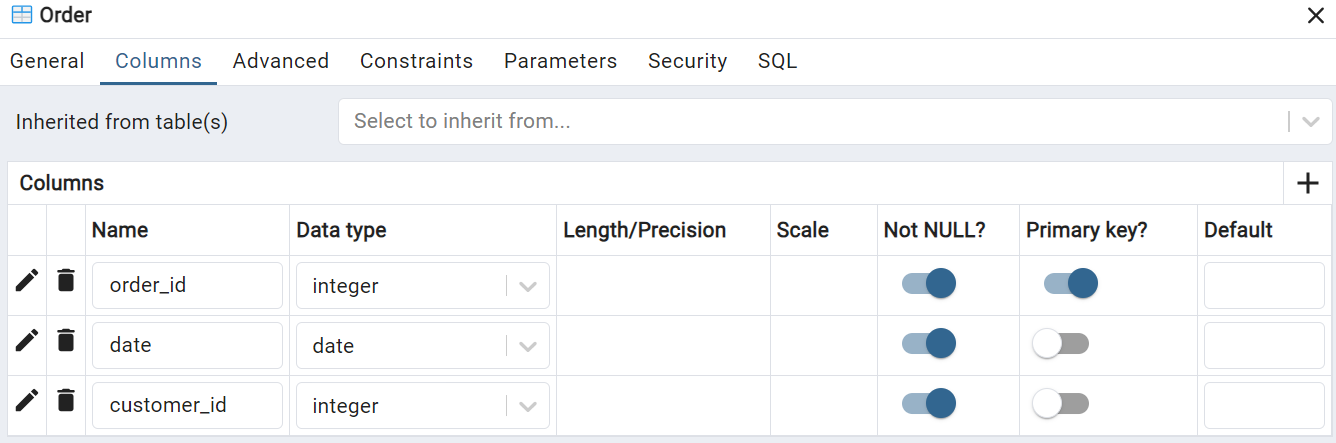


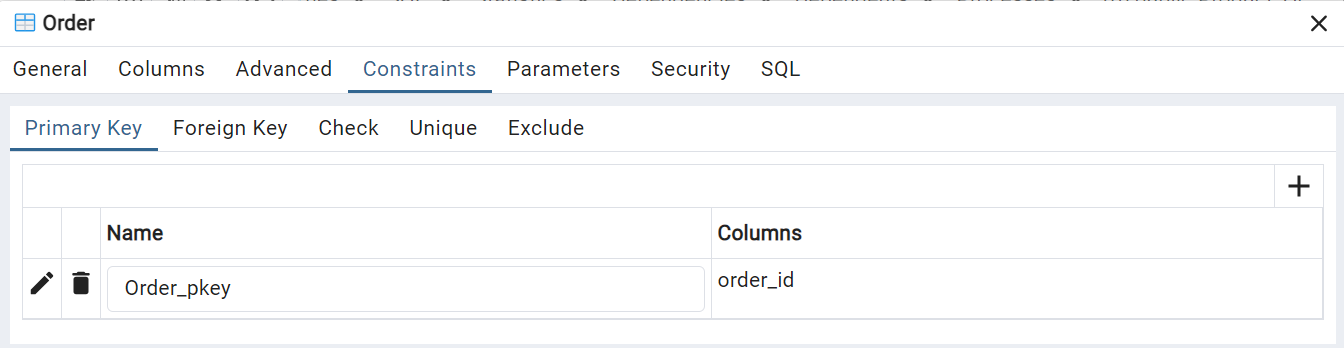


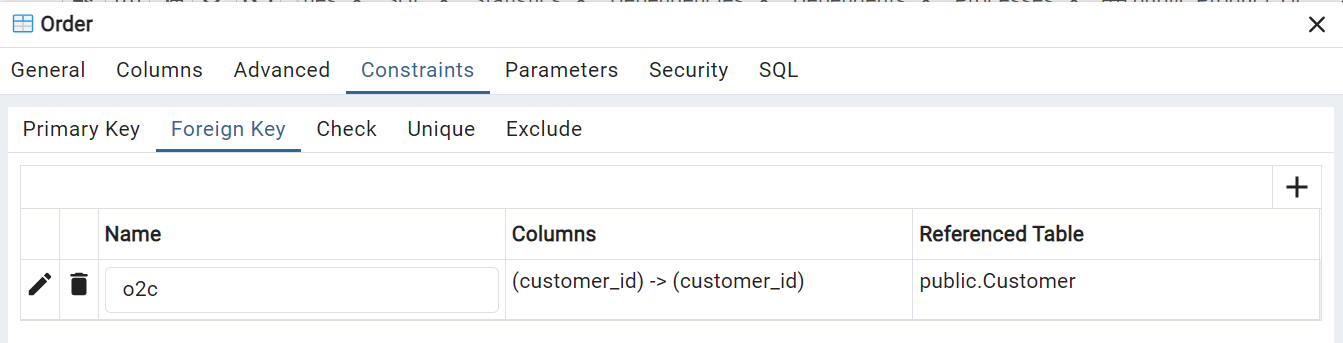


***Order***

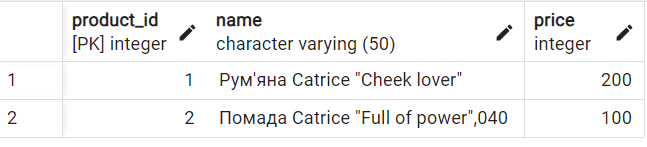
******

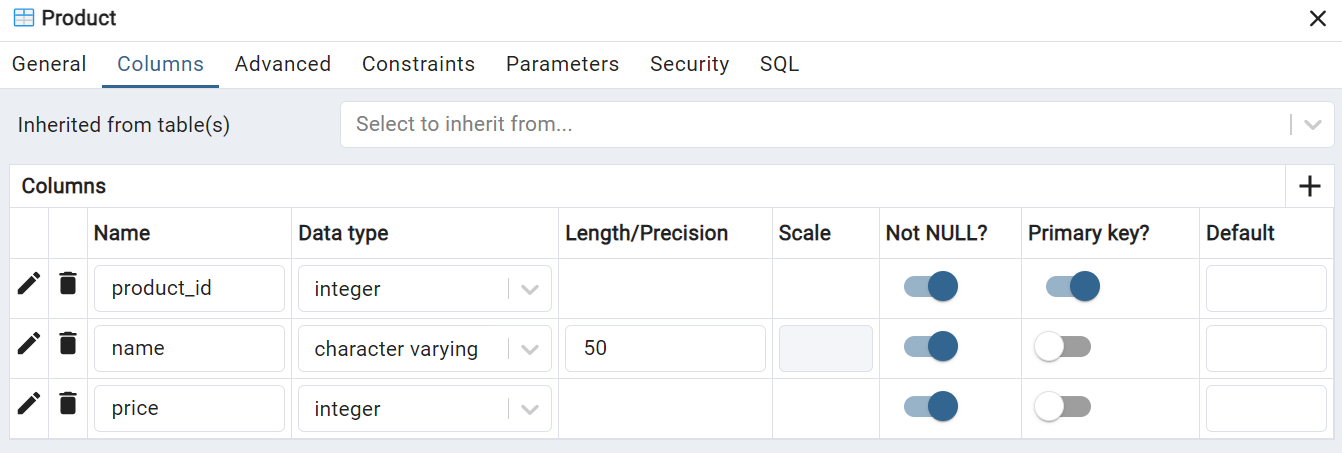
******

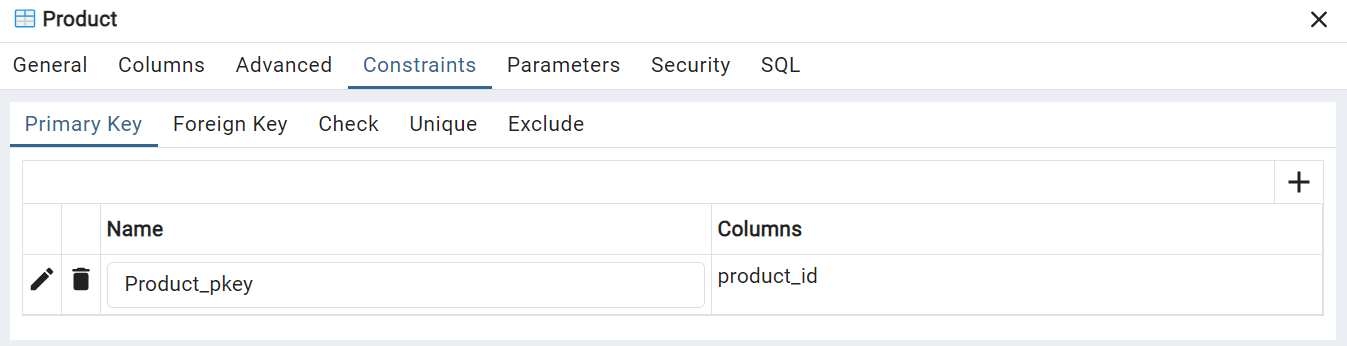


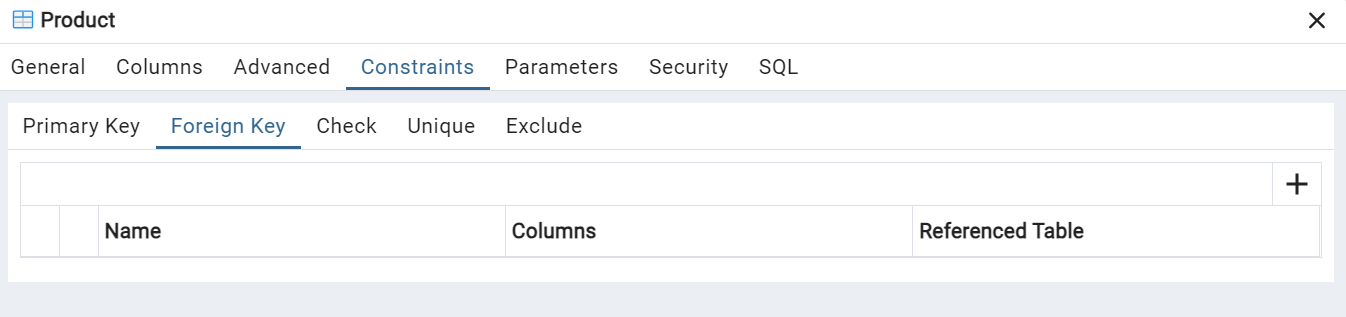


***Product***

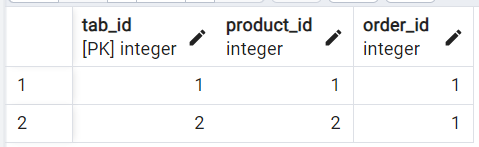
******

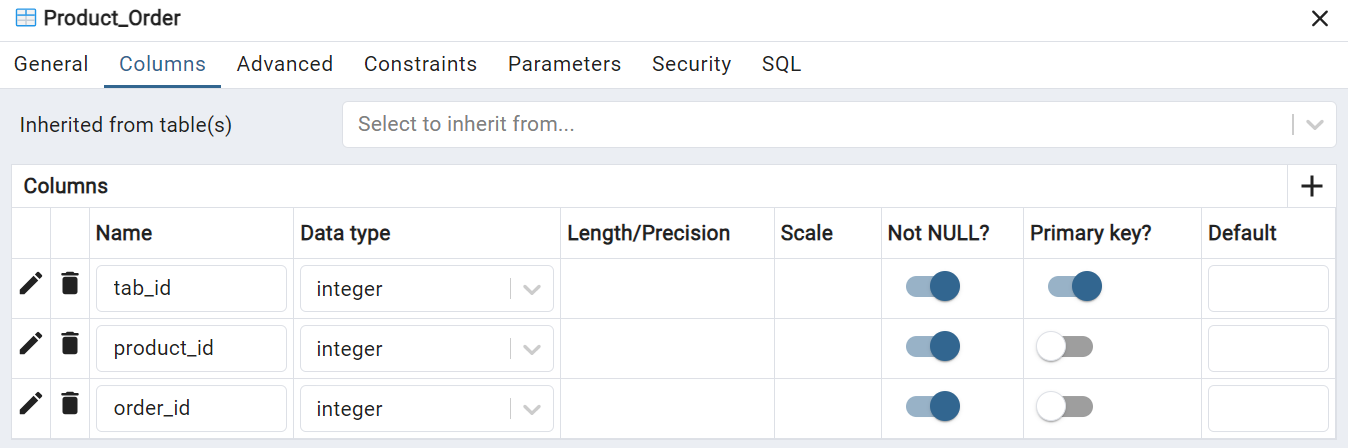


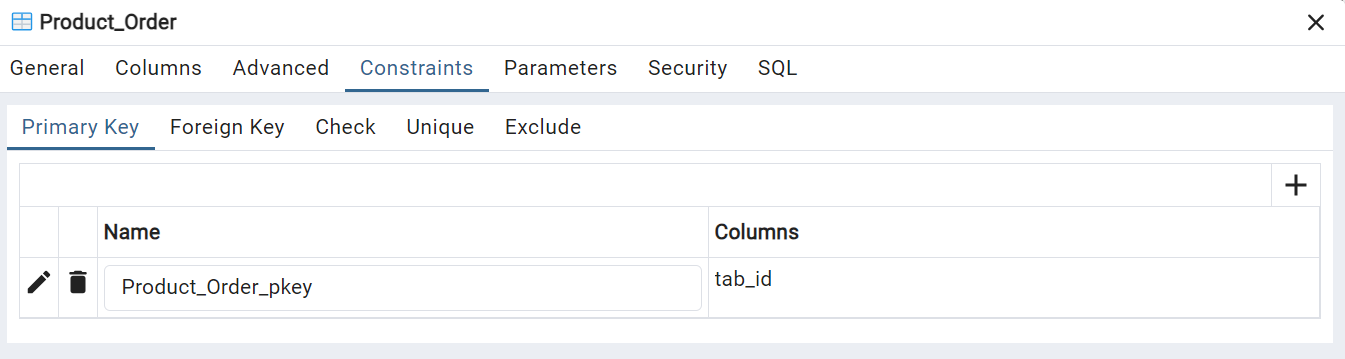


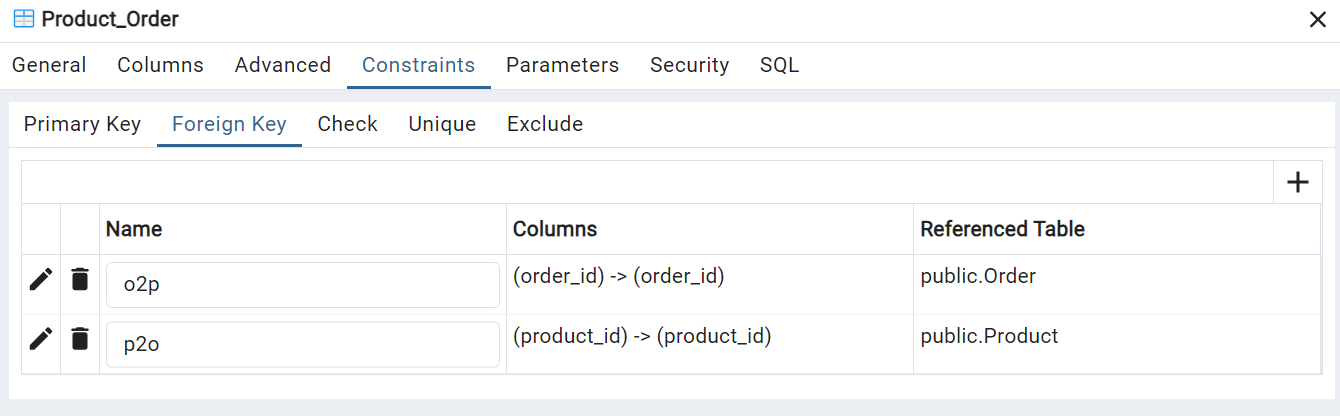


***Product\_Order***

******







**Таблиці в коді SQL**

-- This script was generated by the ERD tool in pgAdmin 4.

-- Please log an issue at https://github.com/pgadmin-org/pgadmin4/issues/new/choose if you find any bugs, including reproduction steps.

BEGIN;

CREATE TABLE IF NOT EXISTS public."Customer"

(

customer\_id integer NOT NULL,

name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

email character varying(50) COLLATE pg\_catalog."default" NOT NULL,

phone character varying(50) COLLATE pg\_catalog."default" NOT NULL,

CONSTRAINT "Customer\_pkey" PRIMARY KEY (customer\_id)

);

CREATE TABLE IF NOT EXISTS public."Delivery"

(

delivery\_id integer NOT NULL,

adress character varying(50) COLLATE pg\_catalog."default" NOT NULL,

status character varying(50) COLLATE pg\_catalog."default" NOT NULL,

order\_id integer NOT NULL,

CONSTRAINT "Delivery\_pkey" PRIMARY KEY (delivery\_id)

);

CREATE TABLE IF NOT EXISTS public."Order"

(

order\_id integer NOT NULL,

date date NOT NULL,

customer\_id integer NOT NULL,

CONSTRAINT "Order\_pkey" PRIMARY KEY (order\_id)

);

CREATE TABLE IF NOT EXISTS public."Product"

(

product\_id integer NOT NULL,

name character varying(50) COLLATE pg\_catalog."default" NOT NULL,

price integer NOT NULL,

CONSTRAINT "Product\_pkey" PRIMARY KEY (product\_id)

);

CREATE TABLE IF NOT EXISTS public."Product\_Order"

(

tab\_id integer NOT NULL,

product\_id integer NOT NULL,

order\_id integer NOT NULL,

CONSTRAINT "Product\_Order\_pkey" PRIMARY KEY (tab\_id)

);

ALTER TABLE IF EXISTS public."Delivery"

ADD CONSTRAINT d2o FOREIGN KEY (order\_id)

REFERENCES public."Order" (order\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE IF EXISTS public."Order"

ADD CONSTRAINT o2c FOREIGN KEY (customer\_id)

REFERENCES public."Customer" (customer\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE IF EXISTS public."Product\_Order"

ADD CONSTRAINT o2p FOREIGN KEY (order\_id)

REFERENCES public."Order" (order\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION;

ALTER TABLE IF EXISTS public."Product\_Order"

ADD CONSTRAINT p2o FOREIGN KEY (product\_id)

REFERENCES public."Product" (product\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION;

END;