

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”*

Виконав:

студент ІII курсу

групи КВ-91

Батура В. В.

Перевірив:

Київ – 2021

Метою роботи є здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

Завдання роботи полягає у наступному:

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.

**Модель «сутність-зв’язок» галузі оброблення замовлень магазину**

Обрана предметна галузь передбачає отримання і обробку замовлень з різних інтернет-магазинів. Згідно цієї області для побудови бази даних було виділено наступні сутності:

1. Товар, з атрибутами: код, назва, категорія, ціна. Призначена для того, щоб фіксувати одиницю товару, яку замовили в магазині.
2. Каталог, з атрибутами: номер, назва. Призначена для розподілу і зберігання товарів магазину.
3. Замовлення, з атрибутами: номер, ім’я замовника, дата. Призначена для фіксування факту оформлення замовлення у магазині.
4. Магазин, з атрибутами: код, назва, адреса. Призначена для збереження даних тих магазинів, для яких здійснюється обробка замовлень.

**Опис зв’язків**

Каталог за своєю структурою є деревом, тому ця сутність утворює унарний (рекурсивний) зв’язок 1:N. У одному каталозі є деяка кількість товарів, тому між сутностями Каталог і Товар існує зв’язок 1:N.

Одне замовлення може складатись з декількох товарів, тому між сутностями Замовлення і Товар існує зв’язок 1:N.

У один магазин може надходити деяка кількість замовлень, тому між сутностями Магазин і Замовлення існує зв’язок 1:N. Також, один магазин має дерево каталогів, тому між сутностями Магазин і Каталог існує зв’язок 1:N.

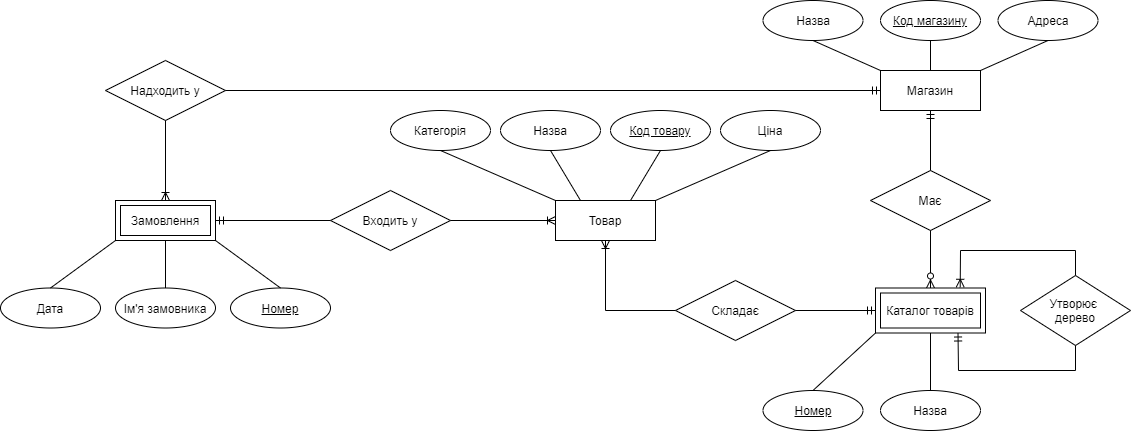


Рисунок 1 - ER-діаграма

**Перетворення моделі у схему бази даних**

Сутність “Товар” було перетворено у таблицю “Product”, а зв’язки 1:N цієї сутності із сутностями “Каталог” і “Замовлення” зумовили появу у ній зовнішніх ключів id\_catalog, id\_order. Сутність “Каталог” було перетворено у таблицю “Catalog”, а зв’язки 1:N цієї сутності із сутностями “Магазин” і “Каталог” (унарний зв’язок) зумовили появу у ній зовнішніх ключів id\_shop, pid\_catalog. Сутність “Каталог” було перетворено у таблицю “Order”, а зв’язок 1:N цієї сутності із сутністю “Магазин” зумовив появу у ній зовнішнього ключа id\_shop. Сутність “Магазин” було перетворено у таблицю “Shop”.

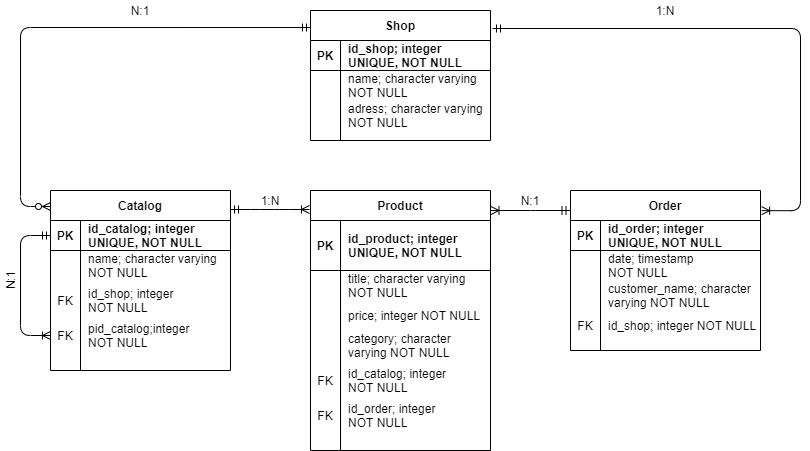
****

Рисунок 2 - Схема бази даних

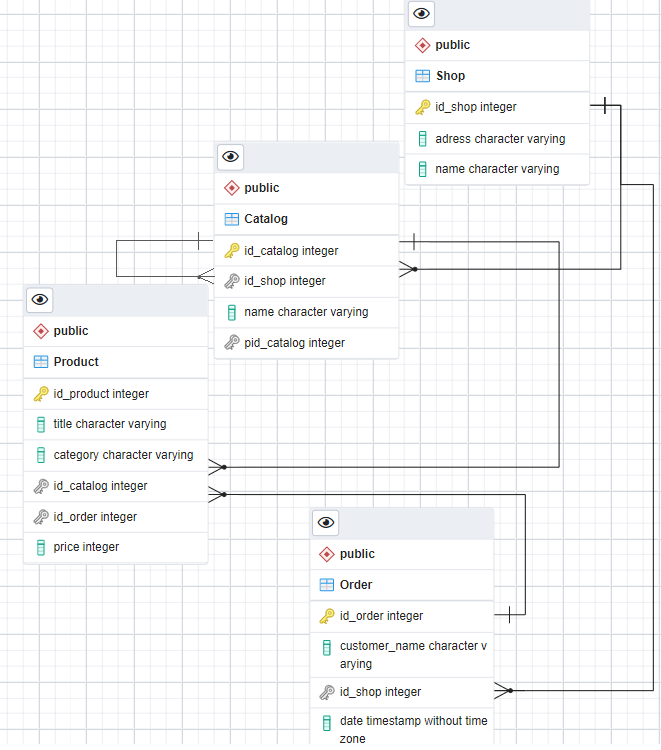


Рисунок 3 - Схема бази даних у pgAdmin 4

Таблиця 1. Опис структури БД

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| **Product** - містить дані про замовлений товар | **id\_product** - унікальний ідентифікатор товару  **title** - назва товару  **price** - ціна товару  **category** - категорія, до якої належить товар  **id\_catalog** - ідентифікатор каталогу, в якому знаходиться товар  **id\_order** - ідентифікатор замовлення, до якого додано товар | **integer** (числовий)  **character varying** (рядок)  **integer** (числовий)  **character varying** (рядок)  **integer** (числовий)  **integer** (числовий) |
| **Catalog** - містить у собі всі товари магазину | **id\_catalog** - унікальний ідентифікатор каталогу  **name** - назва каталогу  **id\_shop** - ідентифікатор магазину, до якого відноситься даний каталог  **pid\_catalog** - посилання на батьківський каталог | **integer** (числовий)  **character varying** (рядок)  **integer** (числовий)  **integer** (числовий) |
| **Order** - зберігає дані про замовлення | **id\_order** - унікальний ідентифікатор замовлення  **customer\_name** - ім’я замовника  **date** - мітка часу, коли було оформлено замовлення  **id\_shop** - ідентифікатор магазину, в який надійшло замовлення | **integer** (числовий)  **character varying** (рядок)  **timestamp without time zone** (мітка часу)  **integer** (числовий) |
| **Shop** - містить інформацію про магазин | **id\_order** - унікальний ідентифікатор магазину  **name** - назва магазину  **address** - фізична адреса магазину | **integer** (числовий)  **character varying** (рядок)  **character varying** (рядок) |

**Нормалізація схеми бази даних до третьої нормальної форми**

**Функціональні залежності для кожної таблиці**

PRODUCT:

id\_product → title, price, category

id\_product → title (назва конкретного товару залежить від його коду)

id\_product → price (ціна товару не залежить від назви або категорії, тільки від унікального коду)

id\_product → category (категорія товару визначається його приналежністю до певного каталогу, а ця приналежність - кодом товару)

title → category

id\_product → title → category (транзитивне відношення, але воно включає в себе ключовий атрибут)

CATALOG:

id\_catalog → name, pid\_catalog

id\_catalog → name (назва гілки каталогу залежить від її номеру)

id\_catalog → pid\_catalog (номер батьківської гілки каталогу - це фактично номер гілки, що включає в себе поточну)

ORDER:

id\_order → customer\_name, date

id\_order → customer\_name (ім’я людини, що зробила замовлення прив’язане до його номеру)

id\_order → date (дата замовлення формується після створення замовлення і прив’язується до його ідентифікатору)

SHOP:

id\_shop → address, name

id\_shop → address (адреса магазину зв’язана з його унікальним кодом)

id\_shop → name (назва магазину також пов’язується з його ідентифікатором)

address → name

name → address

id\_shop → address → name (транзитивне відношення, але воно включає в себе ключовий атрибут)

id\_shop → name → address (транзитивне відношення, але воно включає в себе ключовий атрибут)

Схема відповідає 1НФ, тому що:

1) Кожна таблиця має основний ключ (Primary key, PK) та мінімальний набір атрибутів, які ідентифікують запис;

2) Кожен атрибут має лише одне значення, а не множину значень.

Схема відповідає 2НФ, тому що:

1) Вона відповідає вимогам 1НФ;

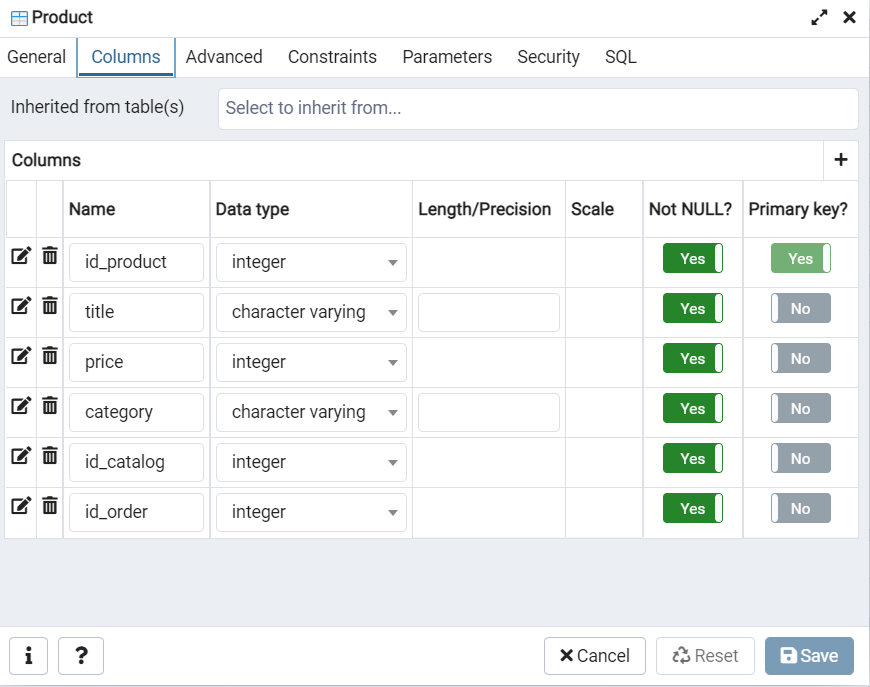
2) Кожен не ключовий атрибут функціонально залежить від основного ключа.

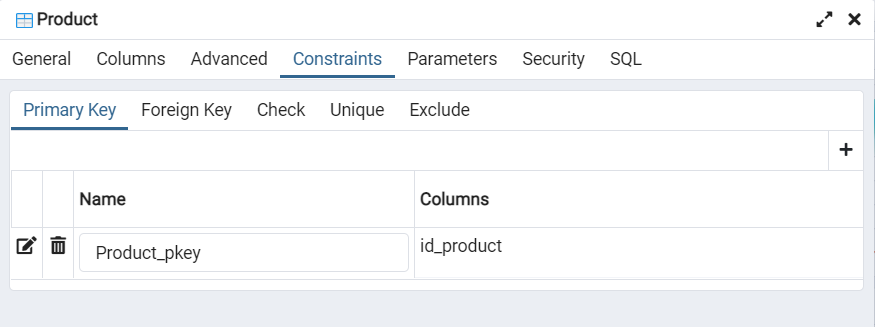
Схема відповідає 3НФ, а саме:

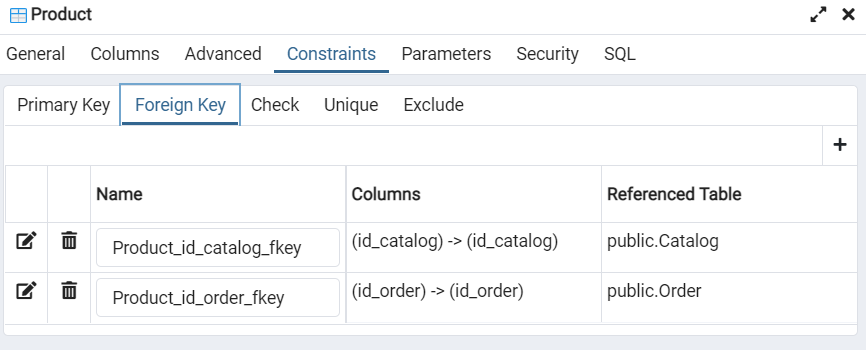
1) Вона відповідає вимогам 2НФ;

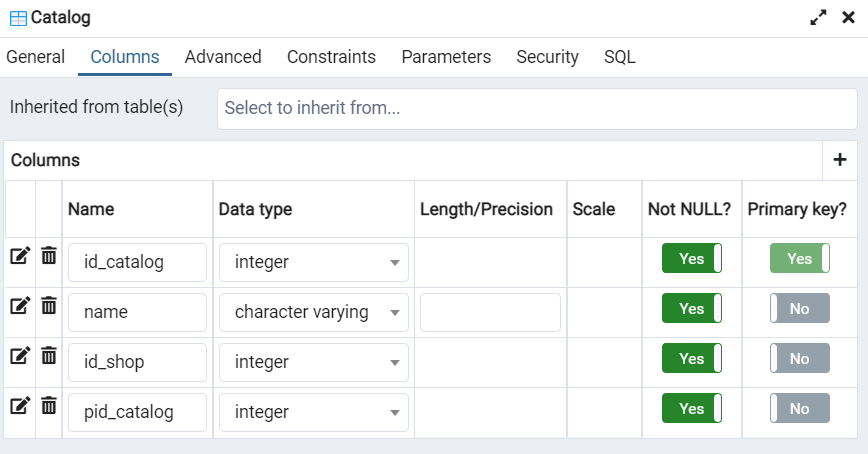
2) Дані в таблиці залежать винятково від основного ключа

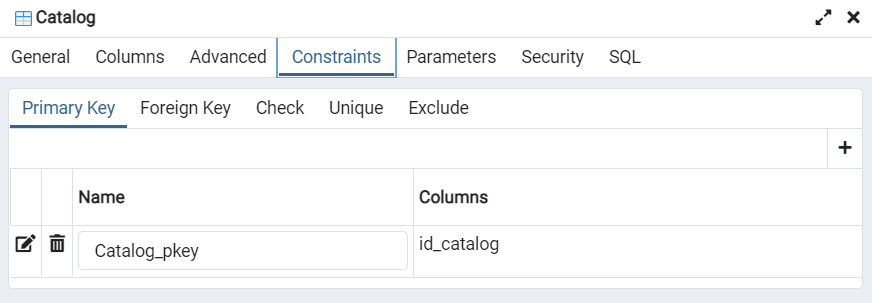
**Таблиці бази даних у pgAdmin4**

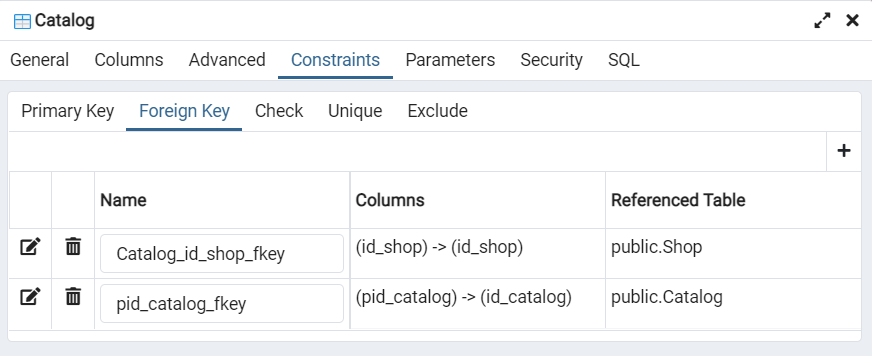
Product

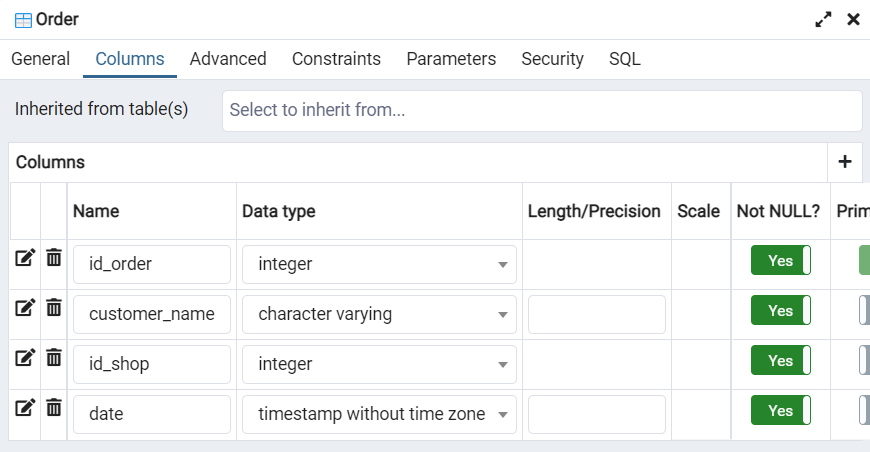


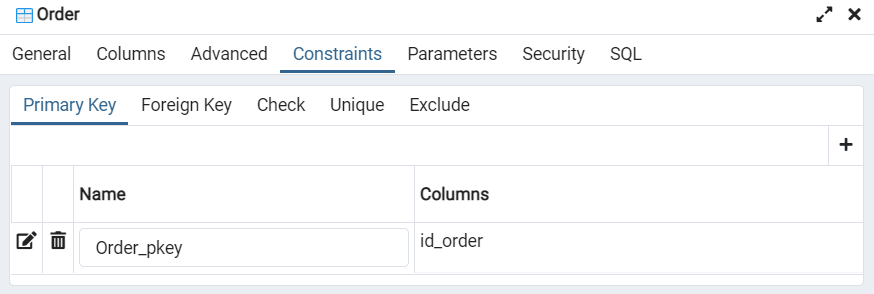


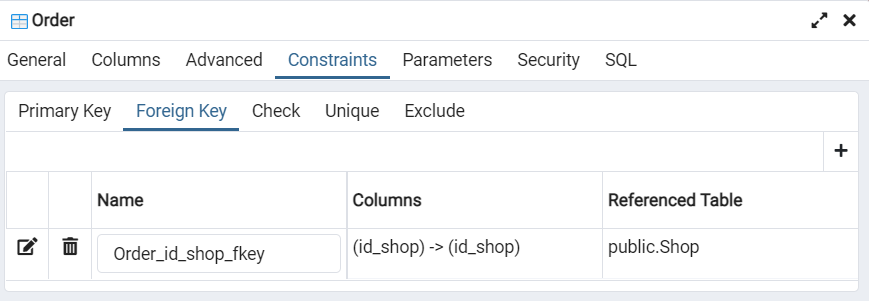
Catalog

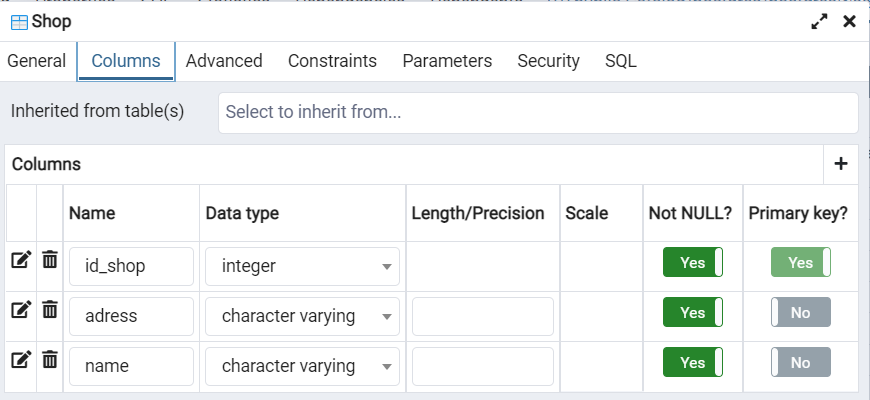


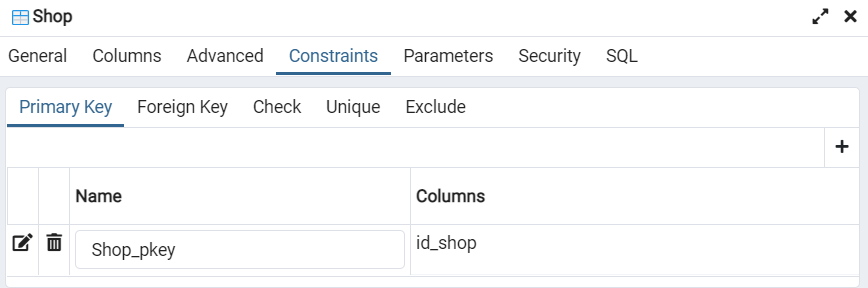


Order

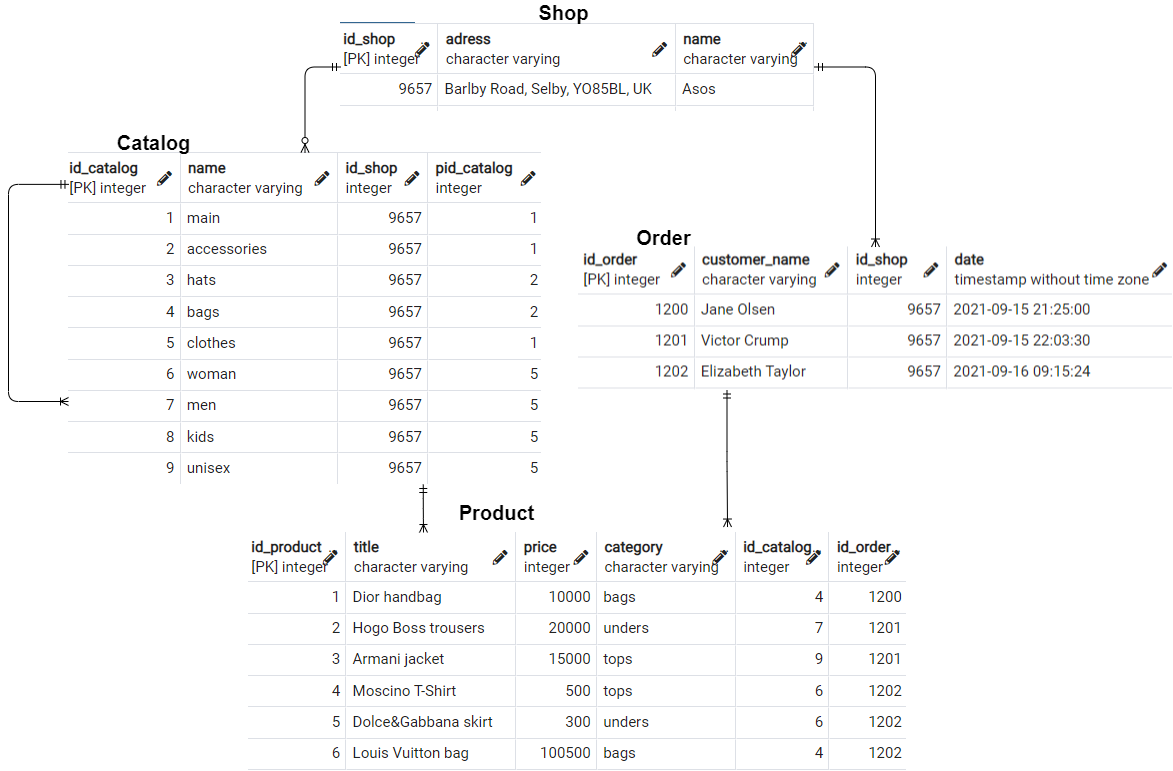




Shop



**Таблиці заповнені даними (уривки знімків екрану pgAdmin4)**

****

**SQL текст**

**product.sql**

-- Table: public.Product

-- DROP TABLE public."Product";

CREATE TABLE IF NOT EXISTS public."Product"

(

id\_product integer NOT NULL,

title character varying COLLATE pg\_catalog."default" NOT NULL,

price integer NOT NULL,

category character varying COLLATE pg\_catalog."default" NOT NULL,

id\_catalog integer NOT NULL,

id\_order integer NOT NULL,

CONSTRAINT "Product\_pkey" PRIMARY KEY (id\_product),

CONSTRAINT "Product\_id\_catalog\_fkey" FOREIGN KEY (id\_catalog)

REFERENCES public."Catalog" (id\_catalog) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID,

CONSTRAINT "Product\_id\_order\_fkey" FOREIGN KEY (id\_order)

REFERENCES public."Order" (id\_order) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE public."Product"

OWNER to postgres;

COMMENT ON COLUMN public."Product".id\_order

IS '(UNIQUE)';

**catalog.sql**

-- Table: public.Catalog

-- DROP TABLE public."Catalog";

CREATE TABLE IF NOT EXISTS public."Catalog"

(

id\_catalog integer NOT NULL,

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

id\_shop integer NOT NULL,

pid\_catalog integer NOT NULL,

CONSTRAINT "Catalog\_pkey" PRIMARY KEY (id\_catalog),

CONSTRAINT "Catalog\_id\_shop\_fkey" FOREIGN KEY (id\_shop)

REFERENCES public."Shop" (id\_shop) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID,

CONSTRAINT pid\_catalog\_fkey FOREIGN KEY (pid\_catalog)

REFERENCES public."Catalog" (id\_catalog) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE public."Catalog"

OWNER to postgres;

**order.sql**

-- Table: public.Order

-- DROP TABLE public."Order";

CREATE TABLE IF NOT EXISTS public."Order"

(

id\_order integer NOT NULL,

customer\_name character varying COLLATE pg\_catalog."default" NOT NULL,

id\_shop integer NOT NULL,

date timestamp without time zone NOT NULL,

CONSTRAINT "Order\_pkey" PRIMARY KEY (id\_order),

CONSTRAINT "Order\_id\_shop\_fkey" FOREIGN KEY (id\_shop)

REFERENCES public."Shop" (id\_shop) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID

)

TABLESPACE pg\_default;

ALTER TABLE public."Order"

OWNER to postgres;

**shop.sql**

-- Table: public.Shop

-- DROP TABLE public."Shop";

CREATE TABLE IF NOT EXISTS public."Shop"

(

id\_shop integer NOT NULL,

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

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

CONSTRAINT "Shop\_pkey" PRIMARY KEY (id\_shop)

)

TABLESPACE pg\_default;

ALTER TABLE public."Shop"

OWNER to postgres;