

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-02

Каленіченко М.В.

Перевірив:

Павловський В. І.

Київ – 2022

*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 занести декілька рядків даних у кожну з таблиць засобами pgAdmin 4.

*Вимоги до ER-моделі:*

1. Сутності моделі предметної галузі мають містити зв’язки типу 1:N або N:M.
2. Кількість сутностей у моделі – 3-4. Кількість атрибутів у кожній сутності: від двох до п’яти.
3. Передбачити наявність зв’язку з атрибутом.

Для побудови ER-діаграм використовувати одну із нотацій: Чена, “Пташиної лапки (Crow’s foot)”, UML.

**Опис предметної галузі «Кав’ярня»**

Обрана предметна галузь передбачає купівлю клієнтом товарів з кав’ярні. Кав’ярня має багато товарів, клієнт може отримувати багато товарів.

**Опис сутностей предметної галузі**

При проектуванні бази даних «Кав’ярня» можна виділити наступні сутності:

* + - 1. Кав’ярня (Cafe), з атрибутами: код кав’ярні (сafe\_id), назва (name), адреса (address).
      2. Товар (Goods), з атрибутами: код товару (goods\_id), назва (name), ціна(price).
      3. Клієнт (Client), з атрибутами: код клієнта (client\_id), ім’я (name).

**Опис зв’язків**

В одній кав’ярні може бути багато товарів, зв’язок 1:N.

Багато товарів може купляти один клієнт, а багато клієнтів може купляти один товарів зв’язок M:N.

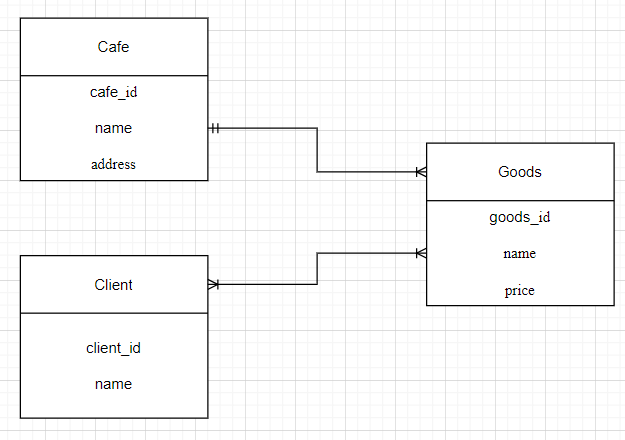


Рисунок 1. ER-діаграма, побудована за нотацією “Пташиної лапки (Crow’s foot)”

**Перетворення концептуальної моделі у схему баз даних**

Для кожної сутності створюється таблиця. Причому кожному атрибуту сутності відповідає стовпець таблиці.

Сутність “Cafe” було перетворено у таблицю “Cafe”, а зв’язки 1:N цієї сутності із сутністю “Goods” зумовили появу у цій сутності зовнішнього ключа cafe\_id.

Сутності “Goods” та “Client” були перетворені у таблиці з відповідними назвами, а зв’язок між ними (M:N) зумовив появу додаткової таблиці “Goods\_Client”.

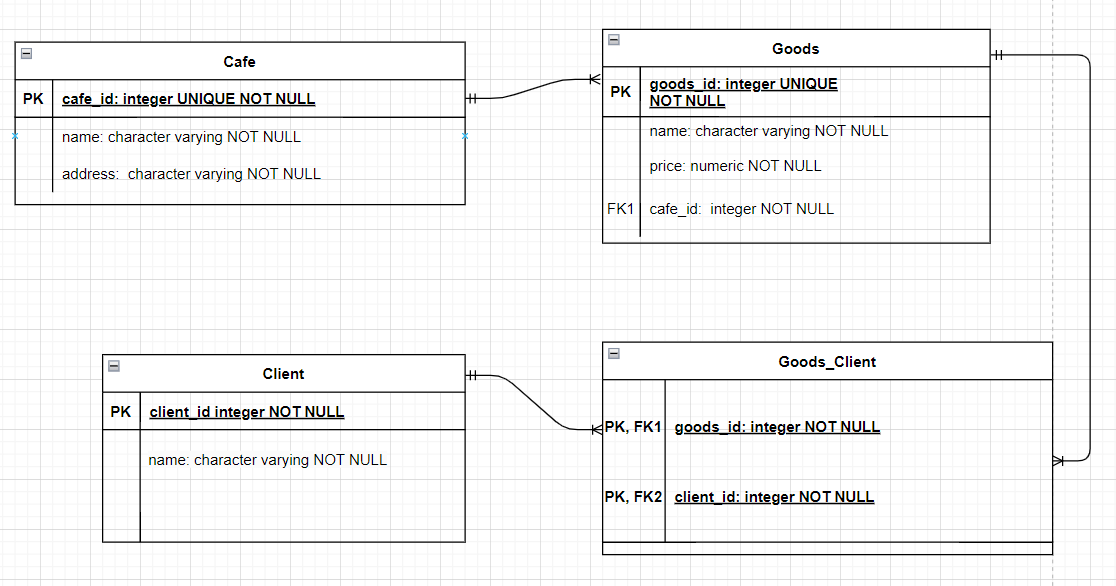


Рисунок 2. Схема бази даних, побудовано у додатку [draw.io](https://app.diagrams.net/).

Після розробки моделі предметної галузі «Кав’ярня» та перетворення її у схему бази даних, було створено дану базу даних у додатку pgAdmin 4

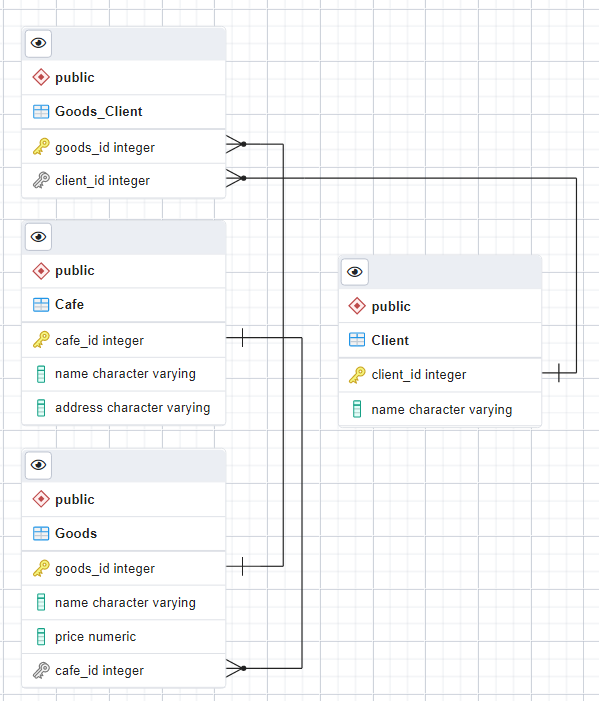


Рисунок 3. Схема бази даних, побудовано у [p](https://app.diagrams.net/)gAdmin 4.

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

Схема відповідає 1НФ, тому що:

1. В таблиці немає дубльованих рядків.
2. В кожній комірці зберігається атомарне значення.
3. В кожному стовпці зберігаються дані одного типу.

Схема відповідає 2НФ, тому що:

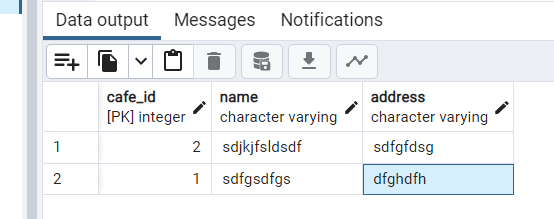
1. Вона відповідає 1НФ.
2. Має первинний ключ, а всі не ключові стовпці таблиці залежать від первинного ключа.

Схема відповідає 3НФ, тому що:

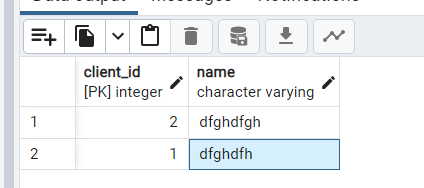
1. Вона відповідає 2НФ.
2. Всі не ключові атрибути таблиці залежать винятково від усього первинного ключа, а не його частини.

**Таблиці бази даних у pgAdmin 4**

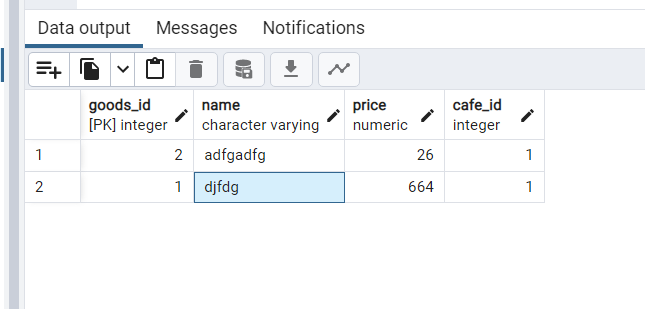
**Café**

****

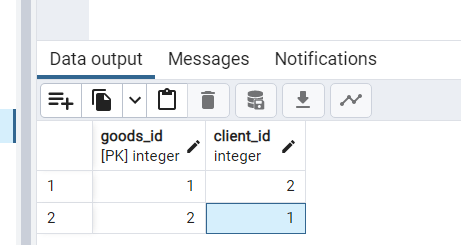
**Client**

****

**Goods**

****

**Goods\_Client**

****

**SQL-текст опису БД «Кав’ярня»**

CREATE TABLE public."Cafe"  
(  
 cafe\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY,  
 name character varying NOT NULL,  
 address character varying NOT NULL,  
 PRIMARY KEY (cafe\_id)  
);  
  
ALTER TABLE IF EXISTS public."Cafe"  
 OWNER to postgres;  
  
CREATE TABLE public."Goods"  
(  
 goods\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY,  
 name character varying NOT NULL,  
 price numeric NOT NULL,  
 cafe\_id integer NOT NULL,  
 PRIMARY KEY (goods\_id),  
 FOREIGN KEY (cafe\_id)  
 REFERENCES public."Cafe" (cafe\_id) MATCH SIMPLE  
 ON UPDATE NO ACTION  
 ON DELETE NO ACTION  
 NOT VALID  
);  
  
ALTER TABLE IF EXISTS public."Goods"  
 OWNER to postgres;  
  
CREATE TABLE public."Client"  
(  
 client\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY,  
 name character varying NOT NULL,  
 PRIMARY KEY (client\_id)  
);  
  
ALTER TABLE IF EXISTS public."Client"  
 OWNER to postgres;  
  
CREATE TABLE public."Goods\_Client"  
(  
 goods\_id integer NOT NULL,  
 client\_id integer NOT NULL,  
 PRIMARY KEY (goods\_id),  
 FOREIGN KEY (goods\_id)  
 REFERENCES public."Goods" (goods\_id) MATCH SIMPLE  
 ON UPDATE NO ACTION  
 ON DELETE NO ACTION  
 NOT VALID,  
 FOREIGN KEY (client\_id)  
 REFERENCES public."Client" (client\_id) MATCH SIMPLE  
 ON UPDATE NO ACTION  
 ON DELETE NO ACTION  
 NOT VALID  
);  
  
ALTER TABLE IF EXISTS public."Goods\_Client"  
 OWNER to postgres;