

Міністерство освіти і науки України Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп’ютерних систем

Лабораторна робота № 1

*з дисципліни*

«Бази даних та засоби управління»

**Тема: «Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL»**

Виконав студент групи:

КВ-11 Чебан М. Д.

Перевірив: Петрашенко А. В.

Оцінка:

**Київ – 2023**

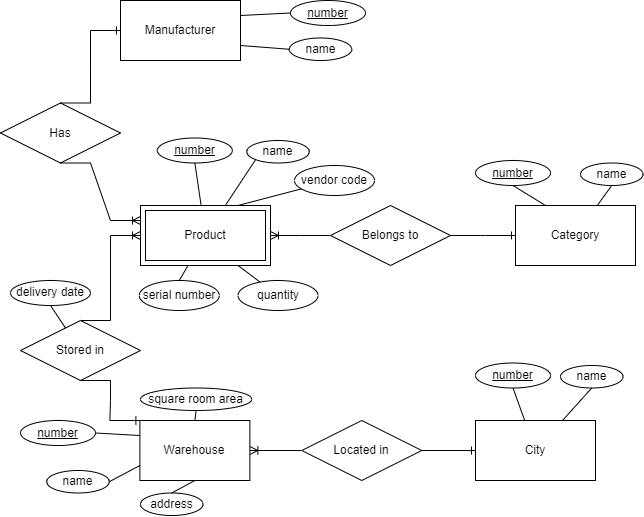
*Метою роботи* є здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

*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.

**Завдання № 1**

Розробка моделі «сутність-зв’язок» предметної галузі для проектування бази даних «Inventory of warehouse accounting». Предметна галузь - «Інвентаризація складського обліку».



Малюнок 1. ER-діаграма побудована за нотацією «Crow`s foot»

**Сутності з описом призначення:**

Предметна галузь «Inventory of warehouse accounting» включає в себе 5 сутностей, кожна сутність містить декілька атрибутів:

1. Product (Id, name, vendor code, quantity, serial number).
2. Manufacturer (Id, name).
3. Category (Id, name).
4. Warehouse (id, name, address, square room area).
5. City (Id, name).

Сунтість Product описує продукти, якиі зберігається на складі. Кожний продукт має свій ідентифікатор Id, а також містить інформацію про свою назву, код постачальника, серійний номер та кількість продукту(товару).

Сутність Manufacturer описує виробника продукту. Кожний виробник має свій ідентифікатор та назву.

Сутність Category описує категорію продукту. Кожна категорія має свій ідентифікатор та назву.

Сутність Warehouse описує склади, на яких будуть зберігатись продукти. Кожний склад має свій ідентифікатор, назву, адресу, розмір площі у м.кв.

Сутність City відповідає за міста, в яких розташовані склади. Кожне місто має свій ідентифікатор та назву.

**Зв’язки між сутностями:**

Зв’язок між Product та Category:

Кожний товар відноситься до певної категорії. Наприклад: «Cell phones» – мобільні телефони, «Appliances» - побутова техніка, «Furniture» - меблі і т.д. Це може бути потрібно, наприклад, для того, щоб сортувати та розподіляти товар по певних групах на складах. Зв'язок 1:N – до однієї категорії може належати багато різних товарів.

Зв’язок між Product та Manufacturer:

Кожний товар має свого виробника. Простіше кажучи, свою фірму. Зв'язок 1:N – один виробник може виготовляти багато різних товарів. Товар без виробника бути не може.

Зв’язок між Product та Warehouse:

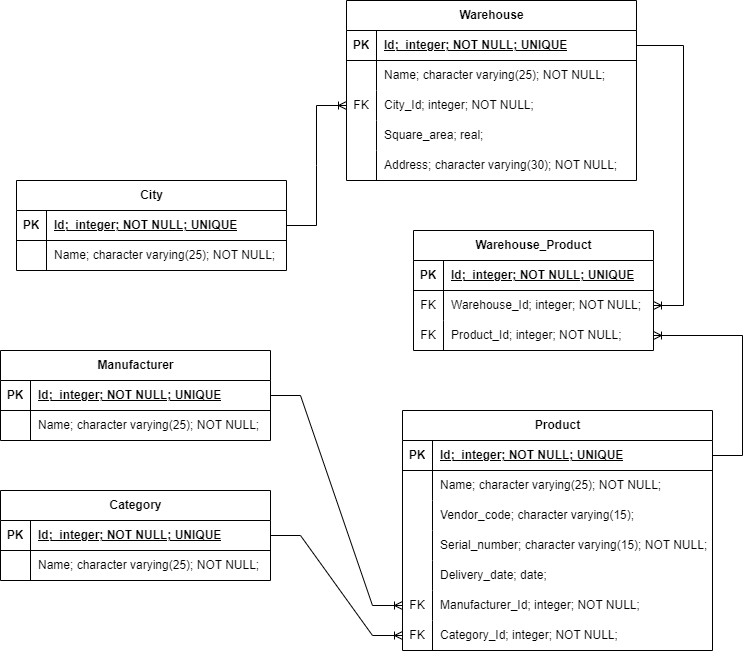
Продукти зберігаються на складі. Оскільки на одному складі зберігаються багато продуктів, зв’язок 1:N.

Зв’язок між Warehouse та City:

Склади розташовані в певних містах або селищах. Зв'язок 1:N – в одному місті може бути багато складів. Склад має обов'язково бути розташованим в якомусь місті.

**Завдання № 2**

Перетворення розробленої моделі «сутність-зв’язок» у схему бази даних PostgreSQL



Малюнок 2. Схема бази даних у графічному вигляді

**Опис процесу перетворення:**

Сутність Product було перетворено на таблицю Product. Первинний ключ (ідентифікатор) Id, атрибути: Name, Vendor\_code, Serial\_number, Delivery\_date, Manufacturer\_Id, Category\_Id.

Сутність Warehouse було перетворено на таблицю Warehouse. Первинний ключ (ідентифікатор) Id, атрибути: Name, City\_Id, Square\_area, Address.

Сутність Category було перетворено на таблицю Category. Первинний ключ (ідентифікатор) Id, атрибути: Name.

Сутність Manufacturer було перетворено на таблицю Manufacturer. Первинний ключ (ідентифікатор) Id, атрибути: Name.

Сутність Category було перетворено на таблицю Category. Первинний ключ (ідентифікатор) Id, атрибути: Name.

Було також створено таблицю Warehouses\_Products. Первинний ключ (ідентифікатор) Id, атрибути: Warehouse\_Id, Product\_Id

Таблиця Warehouses\_Products булла створена для того, щоб утворювати пару склад – товар. Можна було б додати в таблицю Product поле Warehouse\_Id, яке теоретично б пов’язувало товар зі складом, на якому зберігається цей товера. АЛЕ: якщо б 1 і той самий товар зберігався в двох різних складах, то довелося би мати 2 майже однакові записи в таблиці Product. Єдиною відмінністю було б поле Warehouse\_Id. Отже, знову була б виділена пам’ять на всі інші поля (Id, Name, Vendor\_code, Serial\_number, Delivery\_date, Manufacturer\_Id). Це вимагало б додаткової пам’яті. А так, завдяки такій таблиці Warehouses\_Products, реалізована можливість зберігання одного і того ж самого товара на різних складах. Це також реалізовує НФ1, оскільки 1 нормальна форма вимагає відсутні груп полів, які повторюються. В таблиці Warehouses\_Products створено два зовнішніх ключа: FK\_Warehouse та FK\_Product. Вони посилаються на поля Id в таблицях Warehouse та Product відповідно.

В ER-діаграмі був зазначений атрибут «delivery date». В таблиці БД він перетворився на поле типу date в таблиці Product.

Кожний товар має свого виробника та свою категорію. В таблиці БД це було обумовлено завдяки зовнішнім ключам FK\_Manufacturer та FK\_Category, зв'язки «has» (1:N) та «belongs to» (1:N) відповідно.

Кожний склад розташований в якомусь місті. Для дотримання цього зв’язку було створено зовнішній ключ FK\_City. В таблиці Warehouse створено поле, яке посилається на поле (первинний ключ) Id в таблиці City. (1:N)

**Завдання № 3**

**Функціональні залежності:**

Product (*Id*, name, vendor code, quantity, serial number).

*Id* → name

*Id* → vendor code

*Id* → quantity

*Id* → serial number

Manufacturer (*Id*, name).

*Id* → name

Category (*Id*, name).

*Id* → name

Warehouse (*id*, name, address, square room area).

*Id* → name

*Id* → address

*Id* → square room area

City (*Id*, name).

*Id* → name

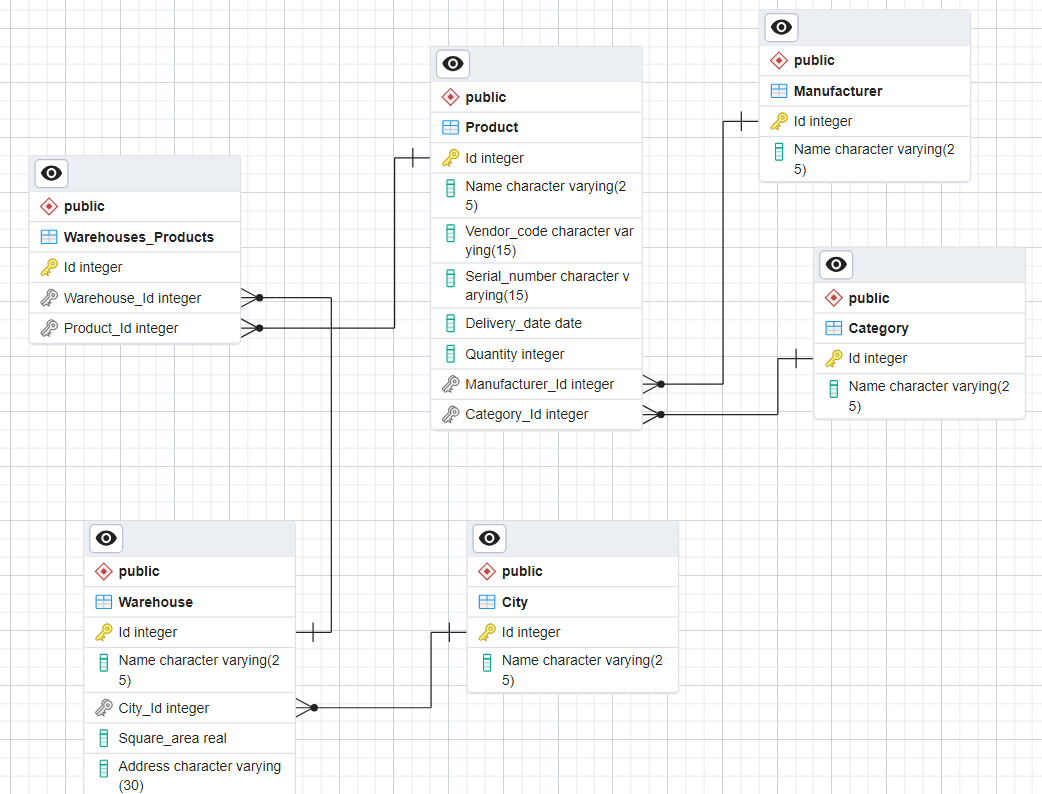
Схема бази даних відповідає 1НФ, тому що значення в кожній комірці таблиці є атомарними, кожне поле таблиці є неподільним, кожен рядок є унікальним, немає повторень рядків.

Схема бази даних відповідає 2НФ, бо вона відповідає 1НФ та кожен неключовий атрибут залежить від первинного повного ключа, отже первинний ключ одразу визначає запис та не є надмірним.

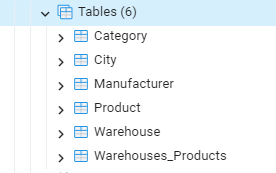
Схема бази даних відповідає 3НФ, тому що вона відповідає 2НФ та кожен неключовий атрибут не є транзитивно залежним від кожного кандидатного ключа. В таблицях нема не ключового поля, яке залежить від значення іншого не ключового поля.

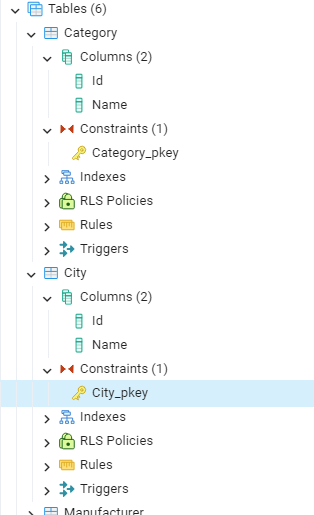
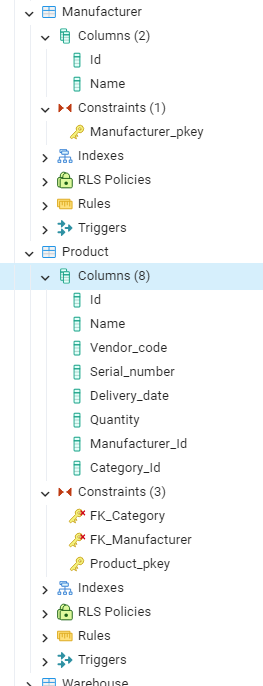
**Завдання № 4**

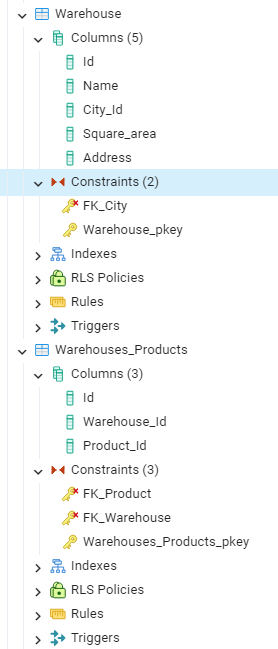
Ознайомлення із інструментарієм PostgreSQL та pgAdmin 4 та внесення даних у кожну з таблиць засобами pgAdmin 4.



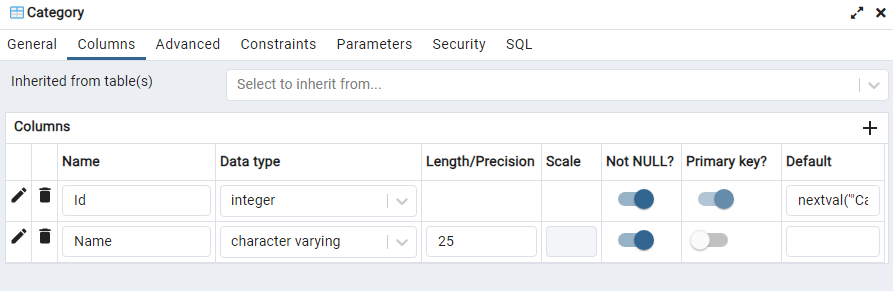
Малюнок 3. Схема бази даних у pgAdmin4

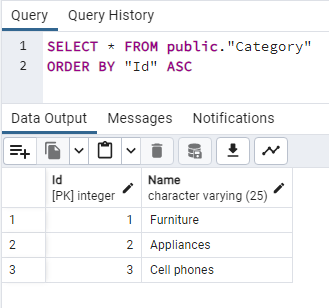


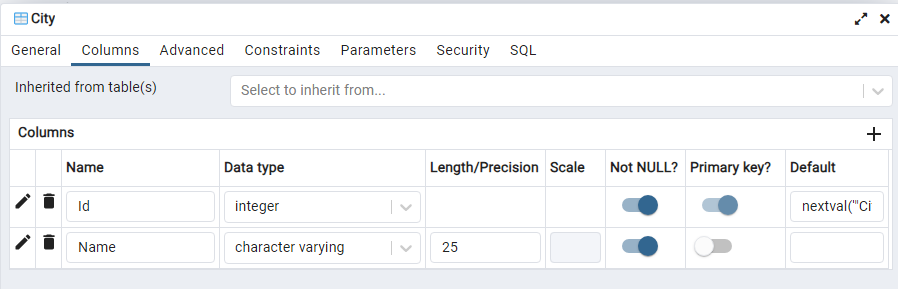


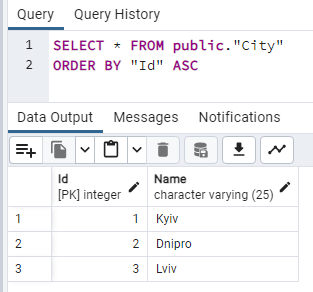
**Таблиця Category:**



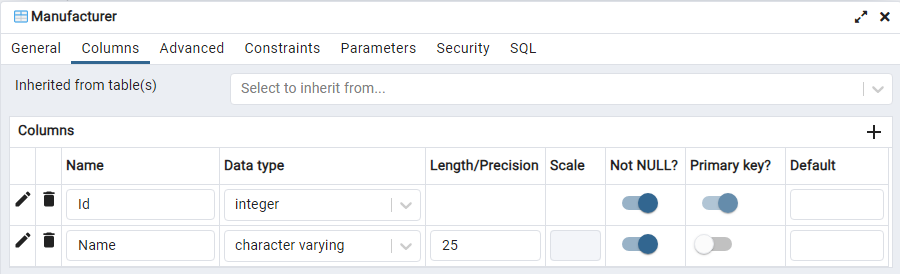


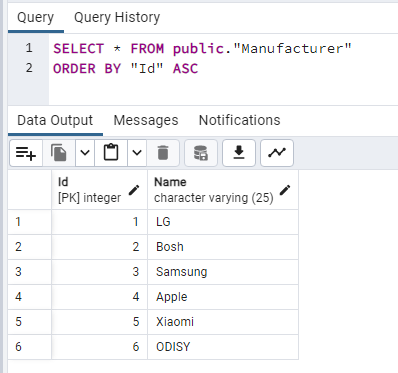
**Таблиця City:**



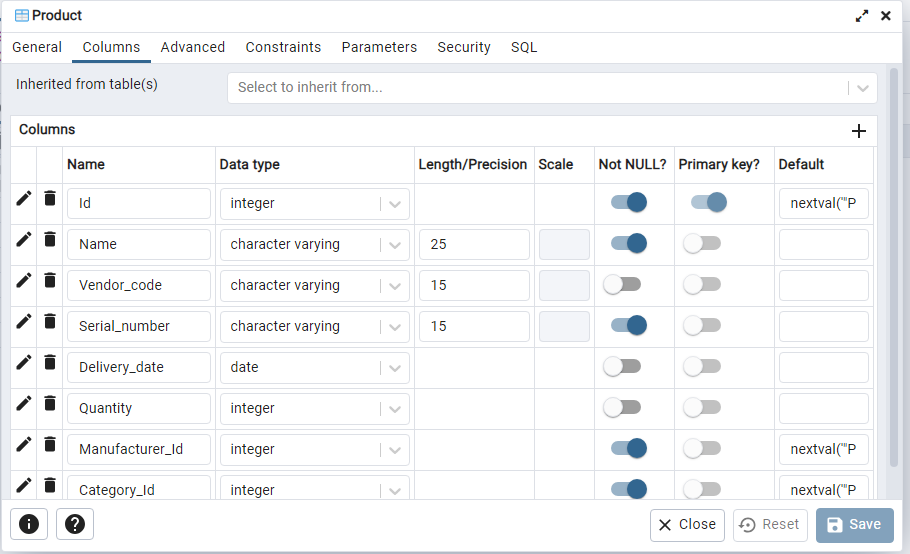


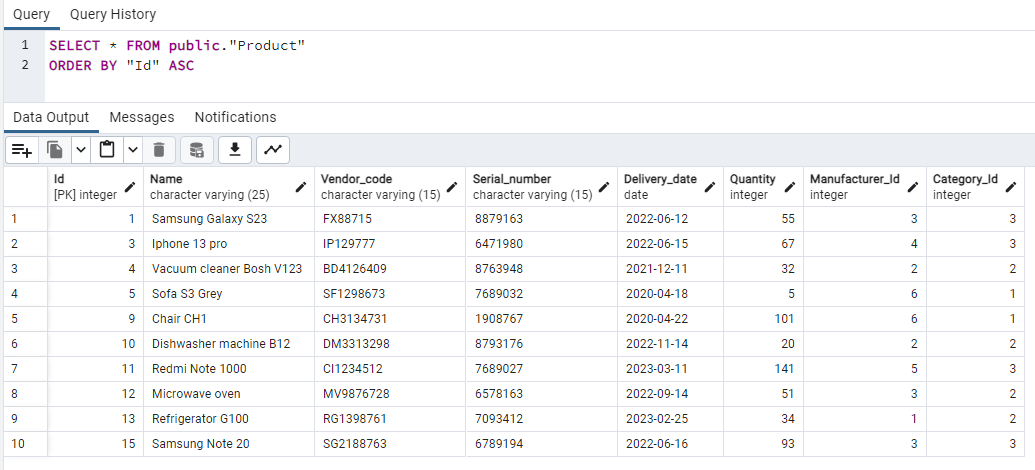
**Таблиця Manufacturer:**



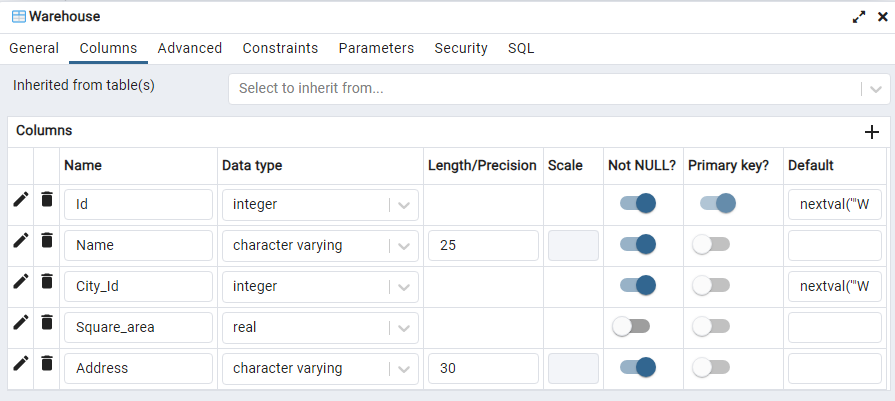


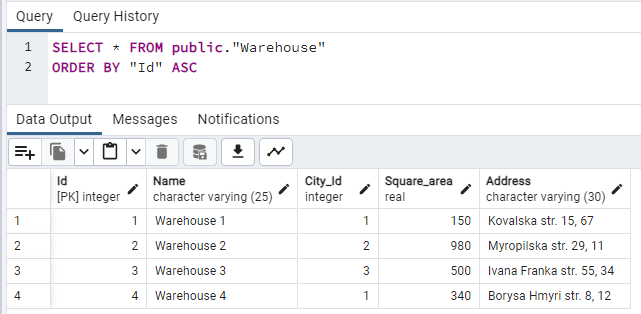
**Таблиця Product:**





**Таблиця Warehouse:**





**Таблиця Warehouses\_Products:**

