

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 Шевчук Я.О.

Перевірив: Петрашенко А. В.

Оцінка:

**Київ – 2023**

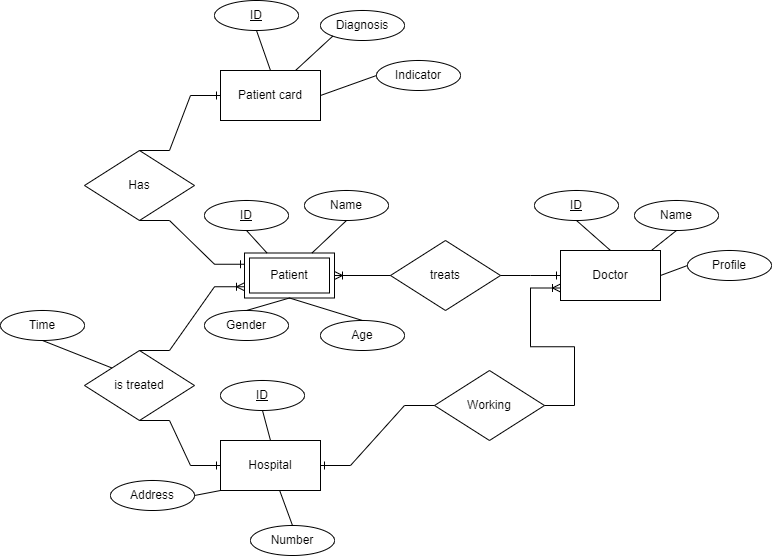
*Метою роботи* є здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

*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 «Patient health monitoring system». Предметна галузь - «Система відстеження стану здоров’я пацієнтів».



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

**Сутності з описом призначення:**

Предметна галузь «Patient health monitoring system» включає в себе 4 сутності, кожна сутність містить декілька атрибутів:

1. Patient (Id, name, gender, age).
2. Patient card (Id, diagnosis, indicator).
3. Hospital (Id, number, address).
4. Doctor (id, name, profile).

Сутність Patient описує пацієнта, який відвідує лікарню. Кожен пацієнт має свій ідентифікатор Id, а також має ім’я, стать та вік.

Сутність Patient card описує стан здоров’я пацієнта. Кожна карта пацієнта має свій ідентифікатор, діагноз пацієнта та показник про його здоров’я(у відсотках).

Сутність Hospital описує лікарню, яку відвідує пацієнт. Кожна лікарня має свій ідентифікатор, адресу і кількість пацієнтів, які її відвідують.

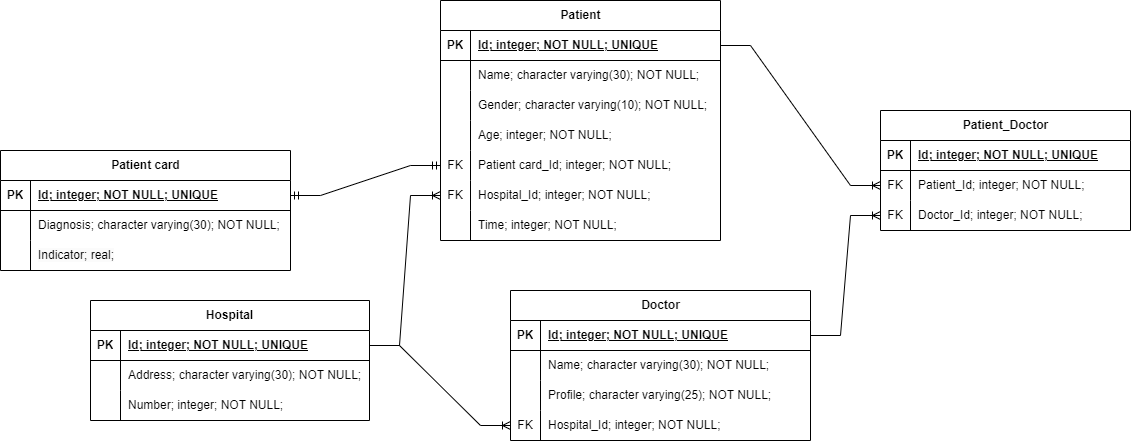
Сутність Doctor описує лікаря до якого ходить пацієнт. Кожен лікар має свій ідентифікатор, ім’я та спеціалізацію(профіль).

**Зв’язки між сутностями:**

Зв’язок між Patient та Patient card:  
Кожен пацієнт має свою картку пацієнта, де міститься інформація про його стан здоров’я. Оскільки, пацієнт має тільки одну картку і одна карта відповідає лише одному пацієнту, то зв’язок 1:1.  
Зв’язок між Patient та Hospital:  
Кожен пацієнт для лікування повинен відвідувати лікарню до якої приписаний. Оскільки, в одній лікарні лікується багато пацієнтів, а один пацієнт відвідує одну лікарню, то зв’язок 1:N.  
Зв’язок між Patient та Doctor:  
Кожен пацієнт лікується у свого сімейного лікаря. Оскільки в сімейного лікаря своїх пацієнтів багато, але в одного пацієнта лише один сімейний лікар, то зв’язок 1:N.  
Зв’язок між Doctor та Hospital:  
Кожен лікар працює у лікарні. Оскільки, в лікарні багато лікарів, але один лікар працює лише в одній лікарні, то зв’язок 1:N.

**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 Patient було перетворено на таблицю Patient. Первинний ключ (ідентифікатор) Id, атрибути: Name, Gender, Age, Time(час відвідування лікарні), Patient card\_Id, Hospital\_Id.

Сутність Patient card було перетворено на таблицю Patient card. Первинний ключ (ідентифікатор) Id, атрибути: Diagnosis, Indicator.

Сутність Hospital було перетворено на таблицю Hospital. Первинний ключ (ідентифікатор) Id, атрибути: Address, Number.

Сутність Doctor було перетворено на таблицю Doctor. Первинний ключ (ідентифікатор) Id, атрибути: Name, Profile, Hospital\_Id.

Було також створено таблицю Patient\_Doctor. Первинний ключ (ідентифікатор) Id, атрибути: Patient \_Id, Doctor \_Id

Таблиця Patient\_Doctor була створена для того, щоб утворювати пару лікар – пацієнт. Можна було б додати в таблицю Patient поле Doctor\_Id, яке теоретично б пов’язувало пацієнта з лікарем, який його лікує. АЛЕ: якщо б один і той самий пацієнт лікувався в двох різних лікарів, то довелось мати б два майже однакові записи в таблиці Patient. Єдиною відмінністю було б поле Doctor\_Id. Отже, знову була б виділена пам’ять на всі інші поля (Id, Name, Gender, Age, Time, Patient card\_Id, Hospital\_Id) Це вимагало б додаткової пам’яті. А так, завдяки такій таблиці Patient\_Doctor, реалізована можливість відвідування одного і того ж лікаря декількома пацієнтами. Це також реалізовує НФ1, оскільки 1 нормальна форма вимагає відсутні груп полів, які повторюються. В таблиці Patient\_Doctor створено два зовнішніх ключа: FK\_ Patient та FK\_ Doctor. Вони посилаються на поля Id в таблицях Patient та Doctor відповідно.

В ER-діаграмі був зазначений атрибут «time». В таблиці БД він перетворився на поле типу time в таблиці Patient, що означає загальну кількість років, яку пацієнт відвідує даного лікаря.

Кожний пацієнт має свою лікарню та свою картку пацієнта. В таблиці БД це було обумовлено завдяки зовнішнім ключам FK\_Hospital та FK\_PatientCard, зв’язки «is treated» (1:N) та «has» (1:1) відповідно.  
Кожен лікар працює в якійсь лікарні. Для отримання цього зв’язку було створено зовнішній ключ FK\_Hospital. В таблиці Doctor створено поле, яке посилається на поле (первинний ключ) Id в таблиці Hospital. (1:N)

**Завдання № 3**

**Функціональні залежності:**

Patient (*Id*, name, gender, age).

*Id* → name

*Id* → gender

*Id* → age

Patient card (*Id*, diagnosis, indicator).

*Id* → diagnosis  
*Id* → indicator

Hospital (*Id*, number, address).

*Id* → number

*Id* → address

Doctor (*Id*, name, profile).

*Id* → name

*Id* → profile

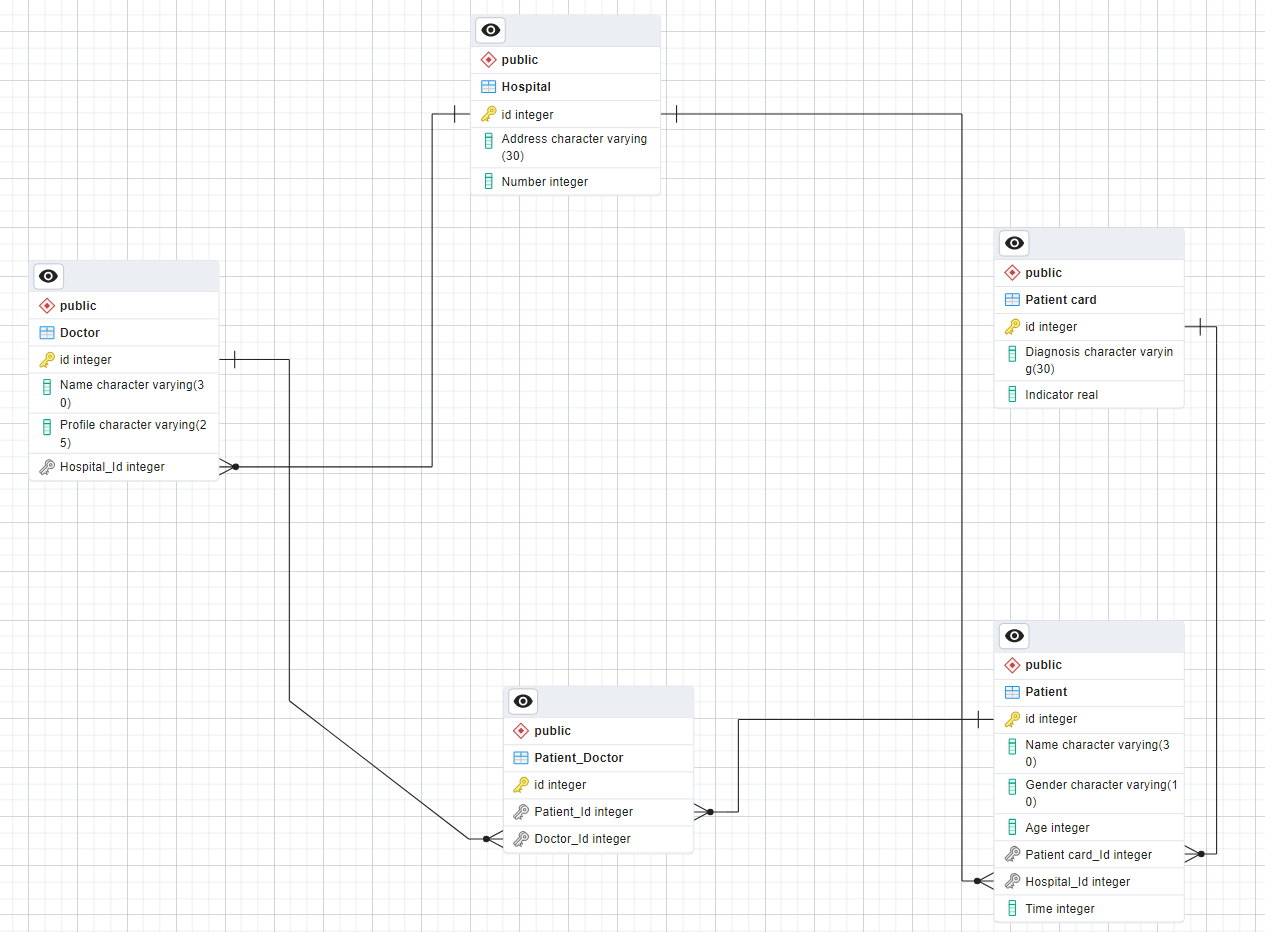
Схема бази даних відповідає 1НФ, тому що значення в кожній комірці таблиці є атомарними, кожне поле таблиці є неподільним, кожен рядок є унікальним, немає повторень рядків.

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.

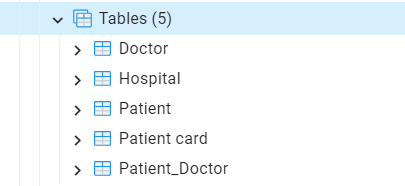
Схема бази даних відповідає 3НФ, тому що вона відповідає 2НФ та кожен неключовий атрибут не є транзитивно залежним від кожного кандидатного ключа. В таблицях нема не ключового поля, яке залежить від значення іншого не ключового поля.

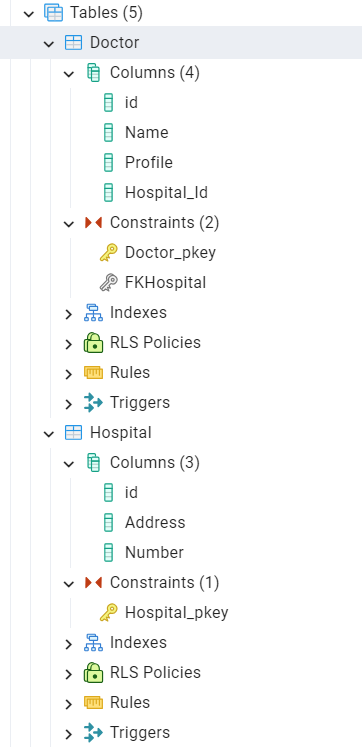
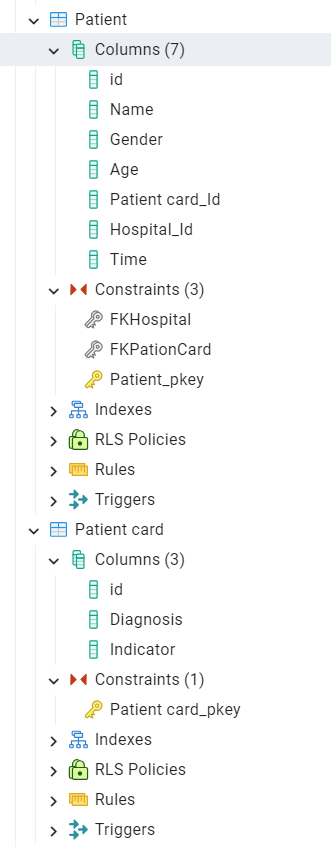
**Завдання № 4**

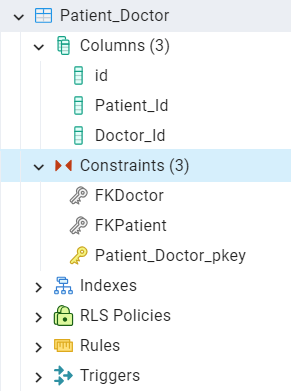
Ознайомлення із інструментарієм PostgreSQL та pgAdmin 4 та внесення даних у кожну з таблиць засобами pgAdmin 4.



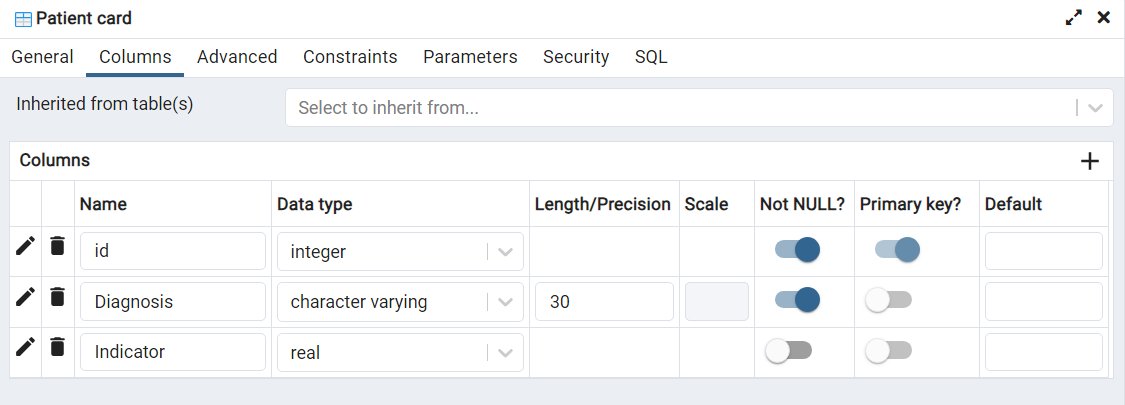
Малюнок 3. Схема бази даних у pgAdmin4

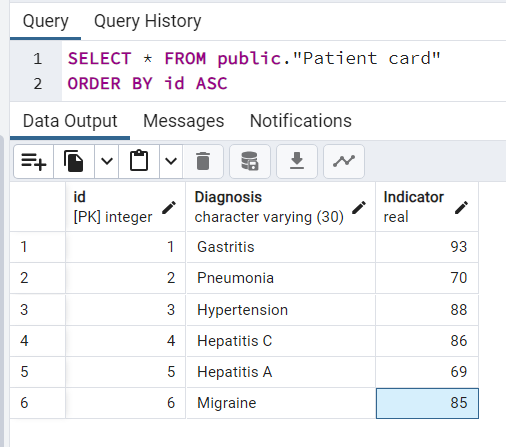


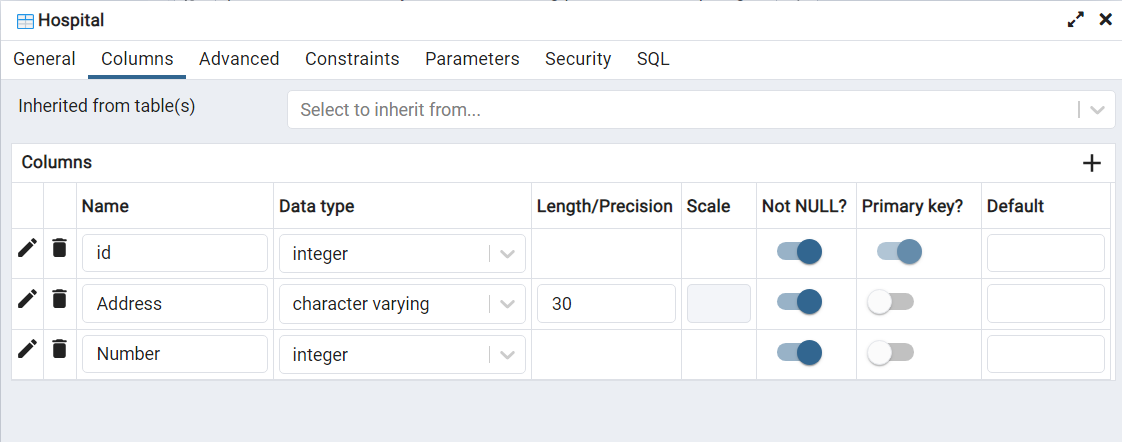


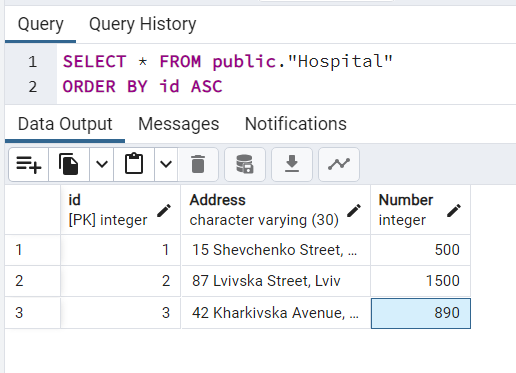
**Таблиця Patient Card:**



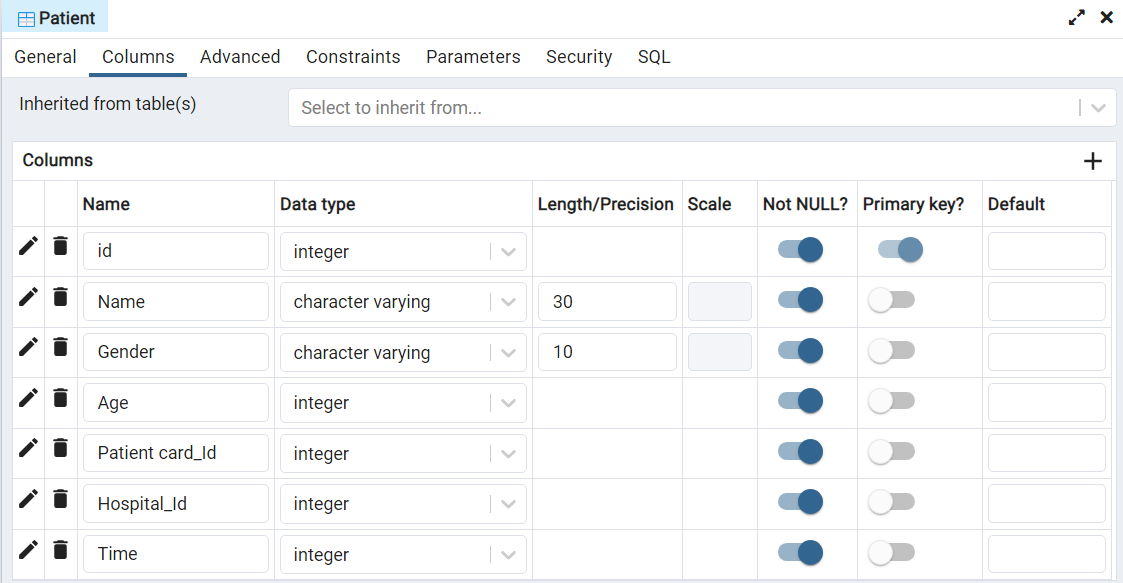


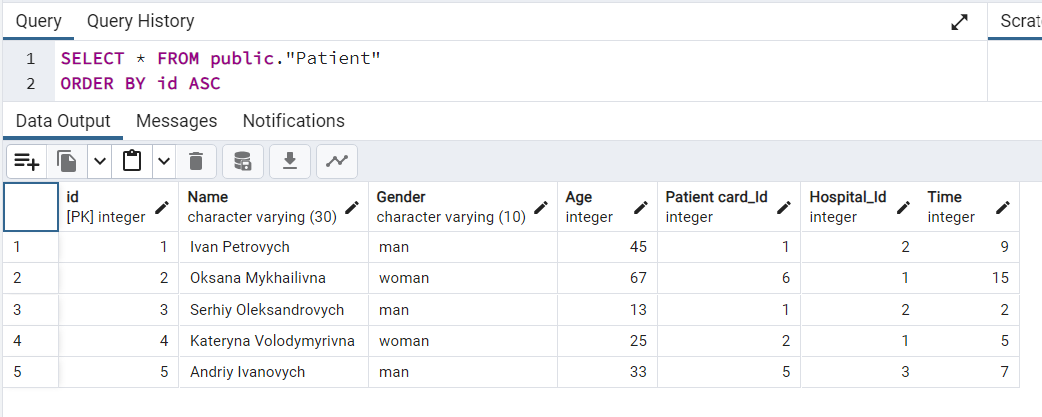
**Таблиця Hospital:**



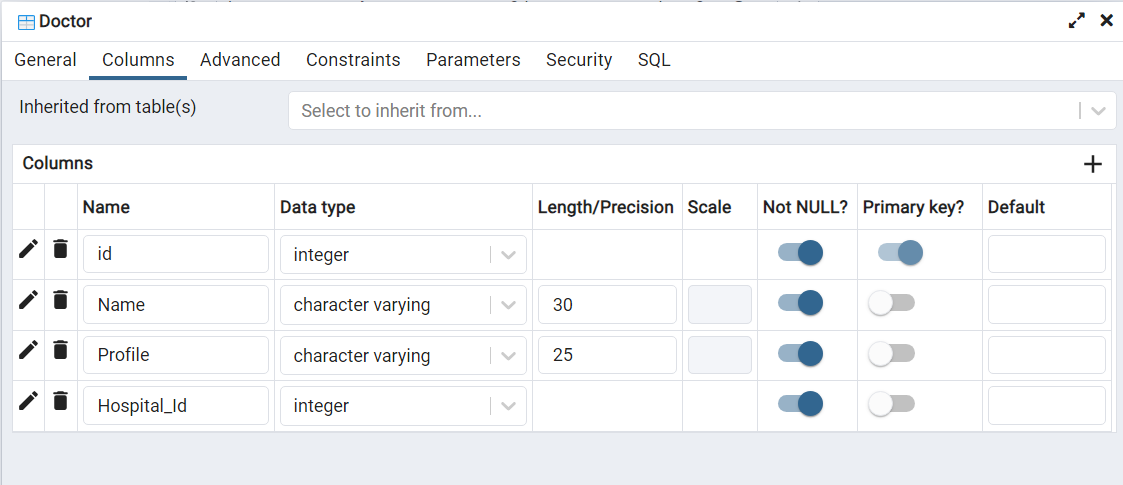


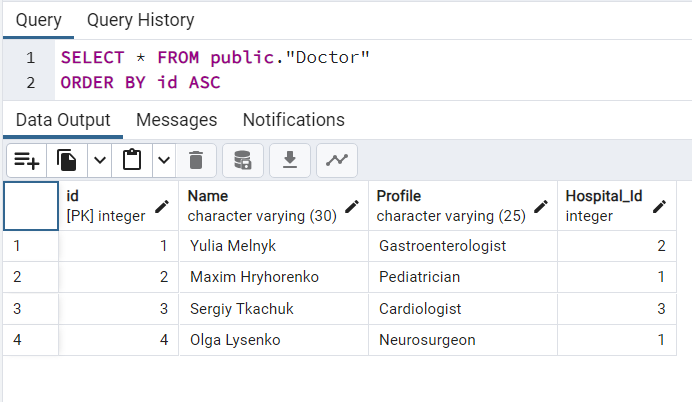
**Таблиця Patient:**





**Таблиця Doctor:**





**Таблиця Patient\_Doctor:**

