

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»**

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

Виконав: студент ІІI курсу

ФПМ групи КВ-84

Мелюх В. В.

Перевірив:Петрашенко А.В.

Київ – 2020

**Загальне завдання:**

*У звіті щодо пункту №1 завдання має бути:*

* перелік сутностей з описом їх призначення;
* графічний файл розробленої моделі «сутність-зв’язок»;
* назва нотації.

*У звіті щодо пункту №2 завдання має бути:*

* опис процесу перетворення (наприклад, “сутність А було перетворено у таблицю А, а зв’язок R (M:N) зумовив появу додаткової таблиці R1 тощо);
* схему бази даних у графічному вигляді **з назвами таблиць (!) та зв’язками між ними.**

*У звіті щодо пункту №3 завдання має бути:*

* пояснення (**обґрунтування!**) щодо відповідності схеми бази даних нормальним формам НФ1, НФ2 та НФ3. Пояснення ***полягає у наведенні функціональних залежностей***, що демонструють висновки. У випадку невідповідності надати опис необхідних змін у схемі;
* У випадку проведення змін у схемі бази даних надати оновлену версію схеми, інакше - не наводити схему.

*У звіті щодо пункту №4 завдання має бути:*

* навести копії екрану з pgAdmin4, що відображають назви, типи та обмеження на стовпці (доступне у закладці “Columns” та “Constraints” властивостей “Properties” таблиць дерева об’єктів у pgAdmin4);
* навести копії екрану з pgAdmin4, що відображають вміст таблиць бази даних у PostgreSQL. Таблиці на зображенні обов'язково **повинні мати назву**!

**Варіант завдання:**

*Командне спортивне змагання з футболу.*

**Завдання №1:**

**Перелік сутностей з описом їх призначення:**

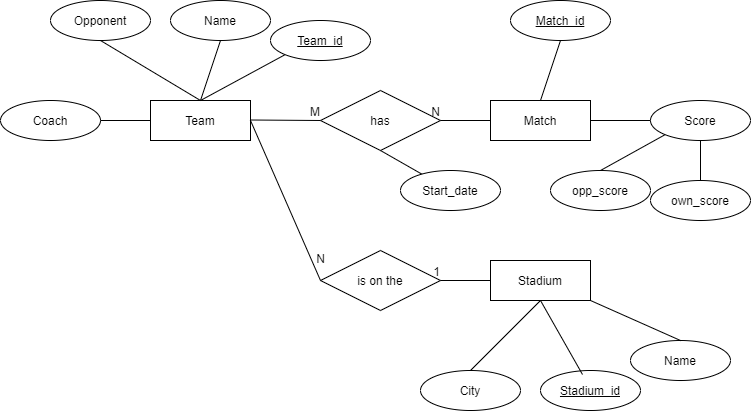
В даній роботі у нас є 3 сутності: Team, Match і Stadium.

Перша сутність “Team” призначена для ведення обліку команд шляхом їх унікальної ідентифікації. Також для визначення назви команди та їх тренера і з’ясування назви команди опонента.

Друга сутність “Match” призначена для ведення обліку матчів шляхом їх унікальної ідентифікації та визначення власного рахунку й рахунку опонентів.

Третя сутність “Stadium” призначена для ведення обліку стадіонів шляхом їх унікальної ідентифікації, визначення назви стадіону та для з’ясування в якому місті він розміщується.

**Графічний файл розробленої моделі «сутність-зв’язок»:**



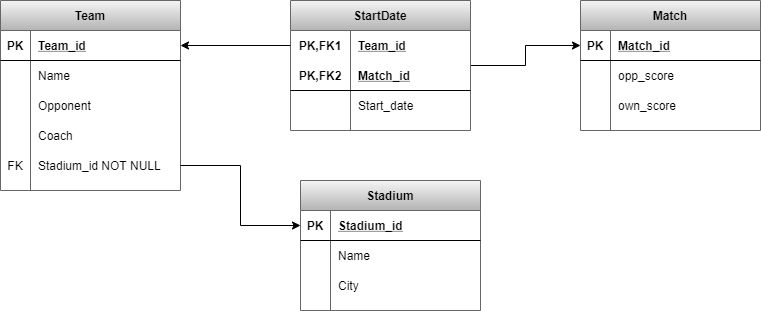
Використана нотація Чена

**Завдання №2:**

**Опис процесу перетворення:**

Сутність “Team” було перетворено у таблицю “Team”, сутність “Match” було перетворено у таблицю “Match”, сутність “Stadium” було перетворено у таблицю “Stadium”. Переносимо всі атрибути, які належали сутностям у відповідні таблиці. В таблицю “Match” замість композитної сутності Score записуємо прості атомарні атрибути opp\_score та own\_score. Усі первинні ключі сутностей стають первинними ключами таблиць. Зв’язок 1:N “is on the” є обов’язковим, тому первинний ключ сутності на стороні 1 додається атрибутом до сутності на стороні N та стає зовнішнім ключем. А зв’язок “has” зумовлює появу додаткової таблиці “StartDate”. Вона має ключі обох сутностей та атрибут “Start\_date”, який з’явився через наявність зв’язку із атрибутом.

**Схема бази даних у графічному вигляді:**



**Завдання №3:**

**Пояснення щодо відповідності схеми бази даних нормальним формам:**

Схема бази даних задовольняє умовам першої нормальної форми, оскільки всі атрибути в таблиці є атомарними, відсутні багатозначні та композитні атрибути, кожна таблиця має Primary key.

Схема бази даних задовольняє умовам другої нормальної форми, оскільки всі неключові атрибути повинні залежати від повного первинного ключа. Щоб бути в другій нормальній формі, всі неключові атрибути повинні залежати від цілого ключа. Таким чином, кожне відношення, яке знаходиться в першій нормальній формі з одним ключем атрибута, автоматично переходить у другу нормальну форму. В цьому випадку присутній складений ключ, який складається з двох атрибутів, тому усі неключові атрибути повинні залежати від усіх компонентів ключа. А оскільки Start\_date залежить від команди, яка має матч, тому можна стверджувати те, що дана схема перебуває у 2НФ.

Схема бази даних задовольняє умовам третьої нормальної форми, оскільки вона задовольняє умовам 2НФ та атрибути не залежать від інших атрибутів, які не є первинними ключами, тобто немає транзитивних функціональних залежностей неключових атрибутів від ключових.

StartDate(Start\_date, Match\_id, Team\_id)

Primary keys: Match\_id, Team\_id.

Functional dependencies:

Match\_id, Team\_id - > Start\_date

Team(Team\_id, Name, Opponent, Coach, Stadium\_id)

Primary keys: Team\_id.

Functional dependencies:

Team\_id - > Name

Team\_id - > Opponent

Team\_id - > Coach

Team\_id - > Stadium\_id

Team\_id - > Name, Opponent, Coach, Stadium\_id

Coach - > Name

Name - > Coach

Match(Match\_id, opp\_score, own\_score)

Primary keys: Match\_id.

Functional dependencies:

Match\_id - > opp\_score, own\_score

Match\_id - > opp\_score

Match\_id - > own\_score

Stadium(Stadium\_id, Name, City)

Primary keys: Stadium\_id

Functional dependencies:

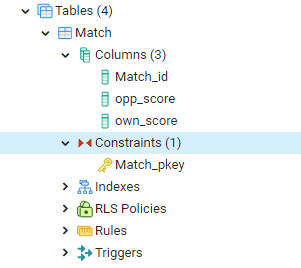
Stadium\_id - > Name, City

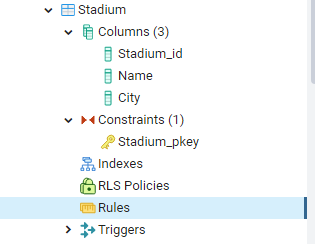
Stadium\_id - > Name

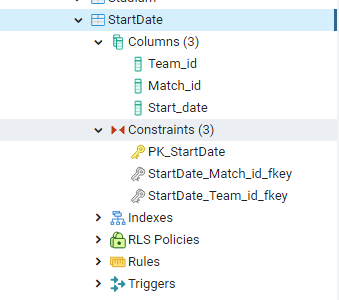
Stadium\_id - > City

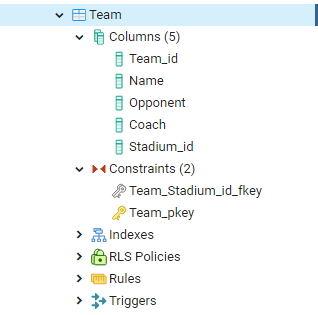
**Завдання №4:**

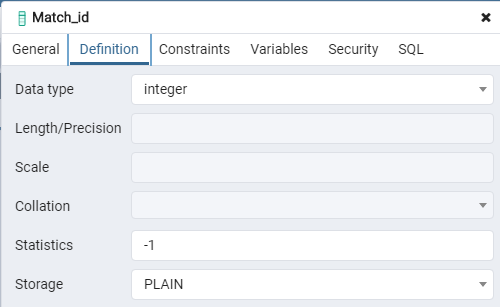
**Копії екрану з pgAdmin4, що відображають назви та типи стовпців та обмеження на стовпці:**

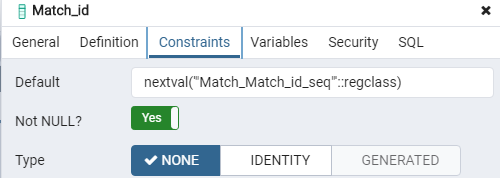


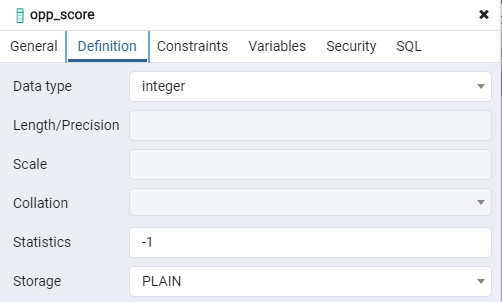


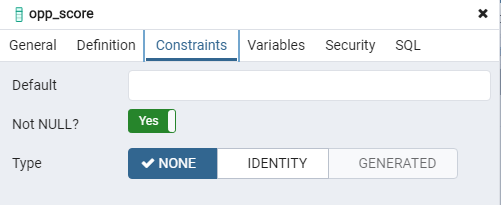


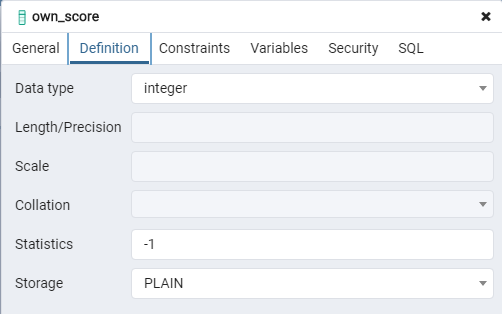


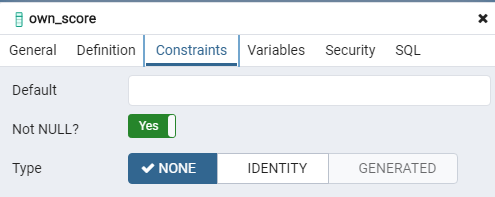


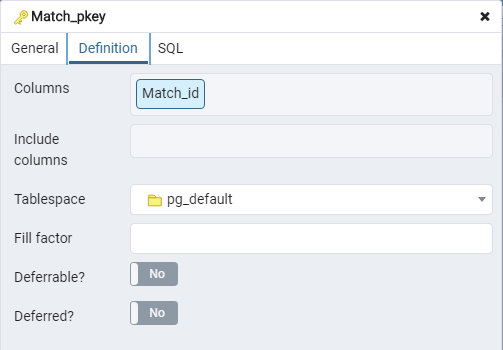


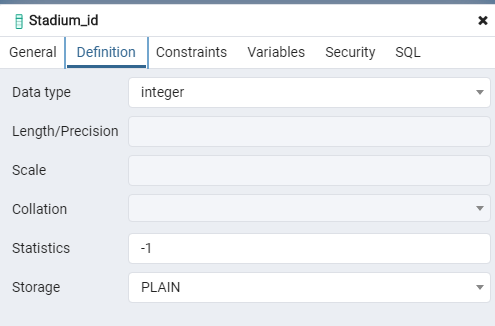


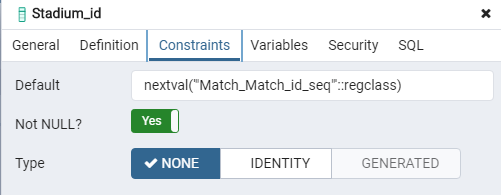


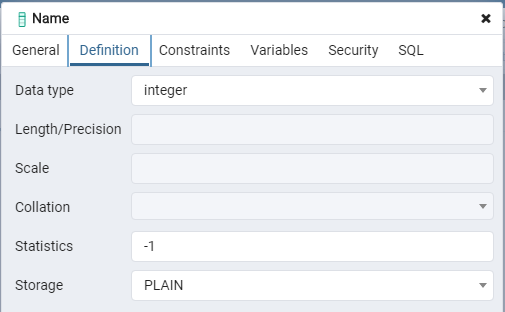


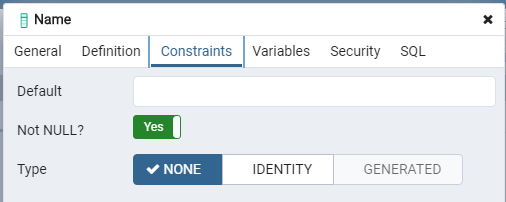


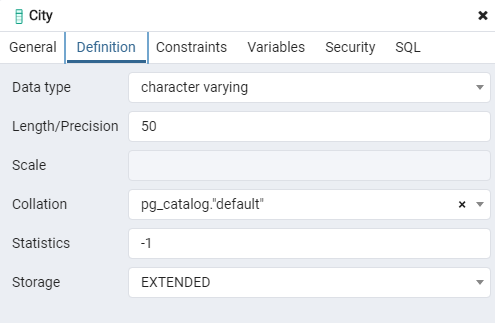


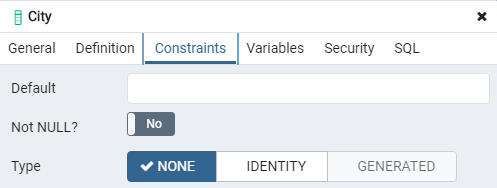


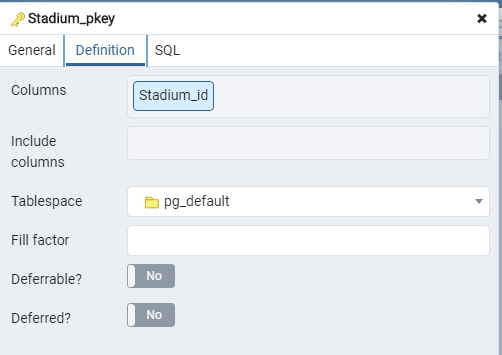


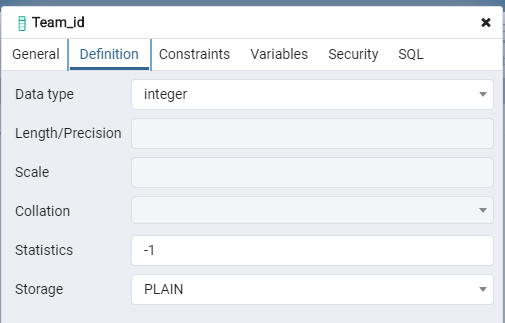


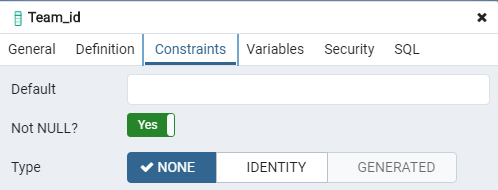


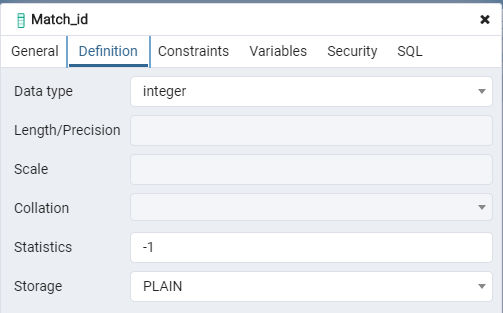


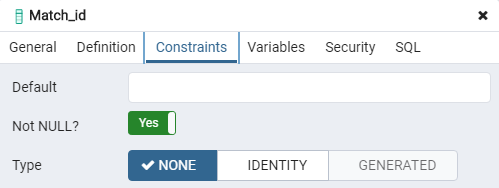


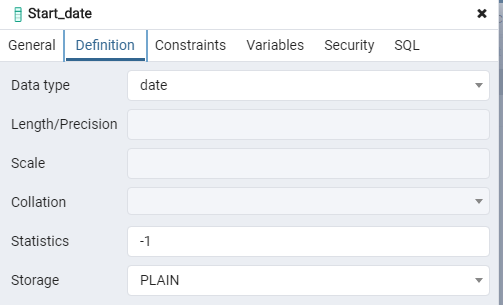


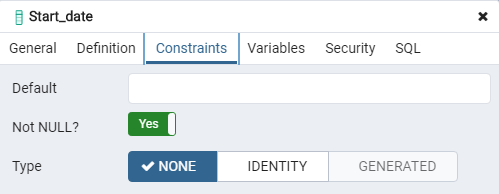


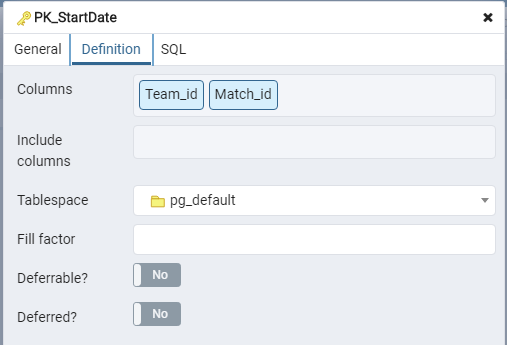


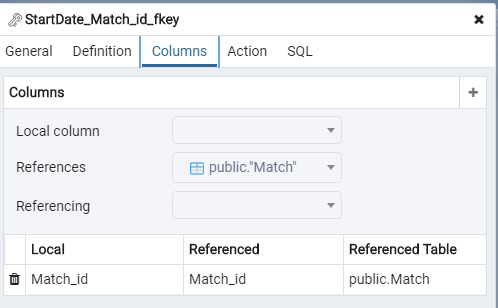


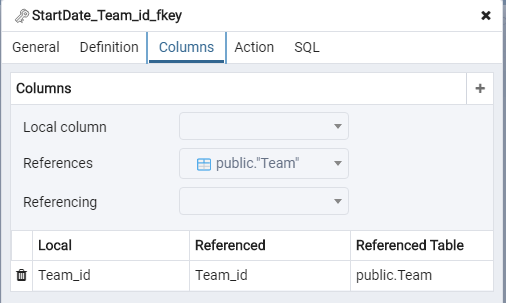


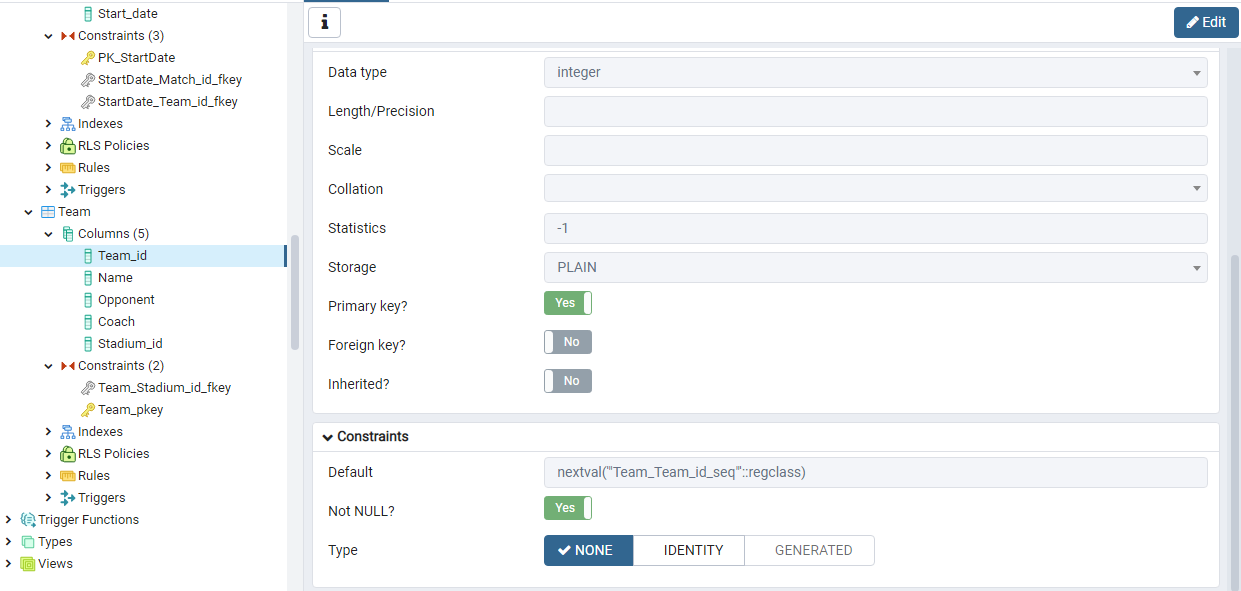


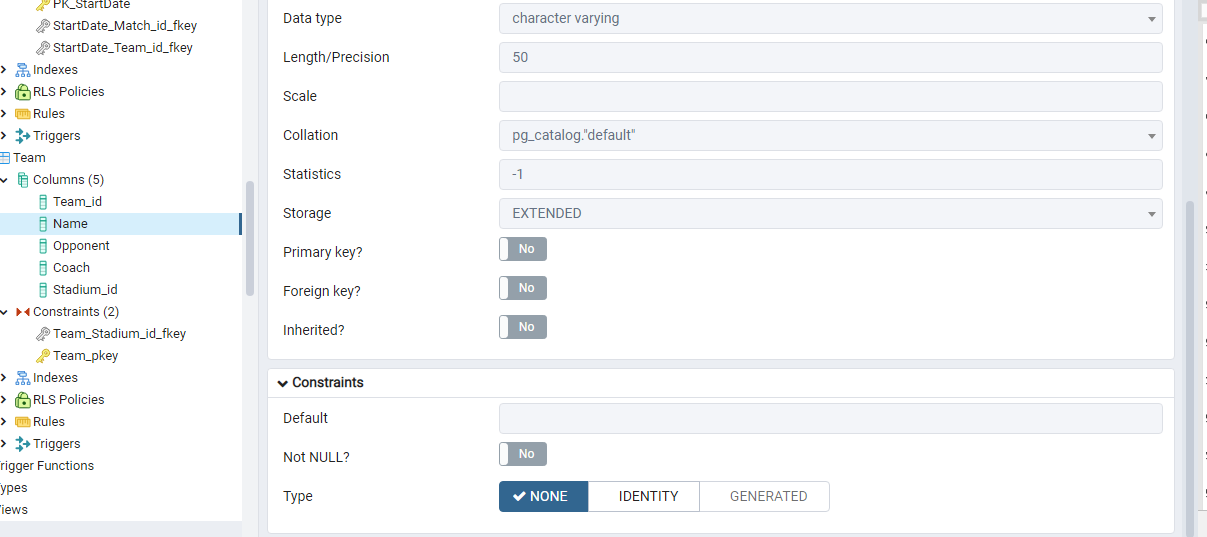


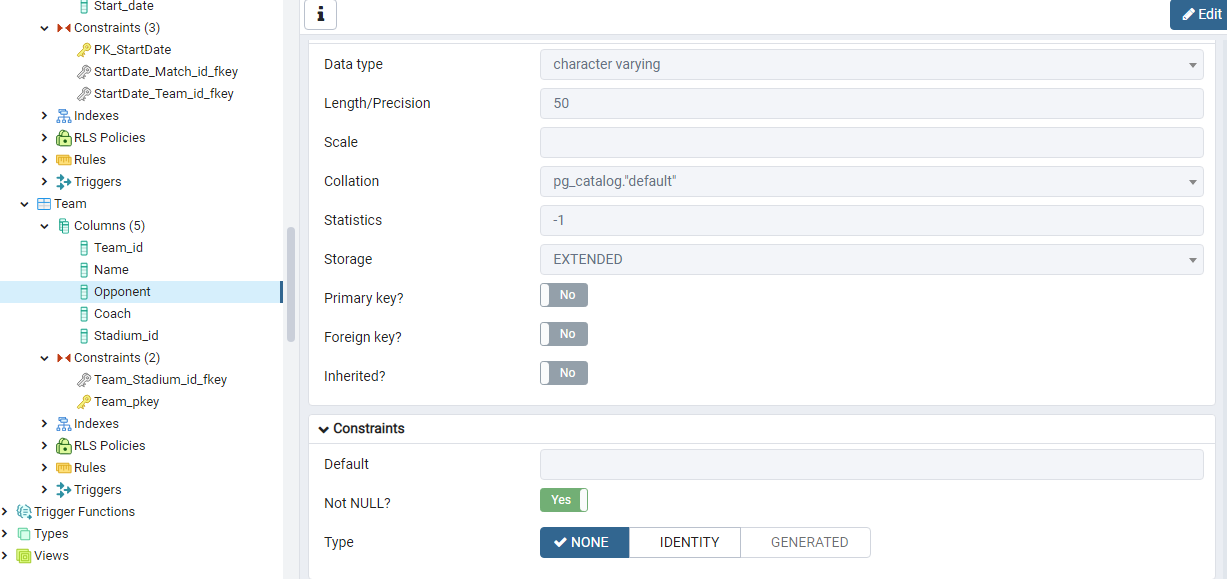


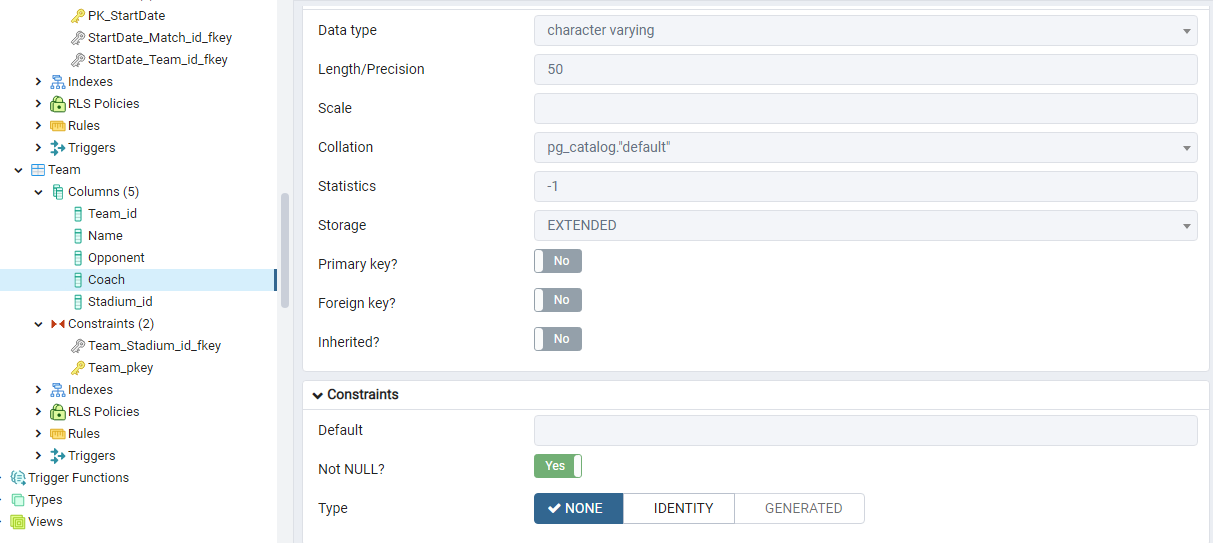


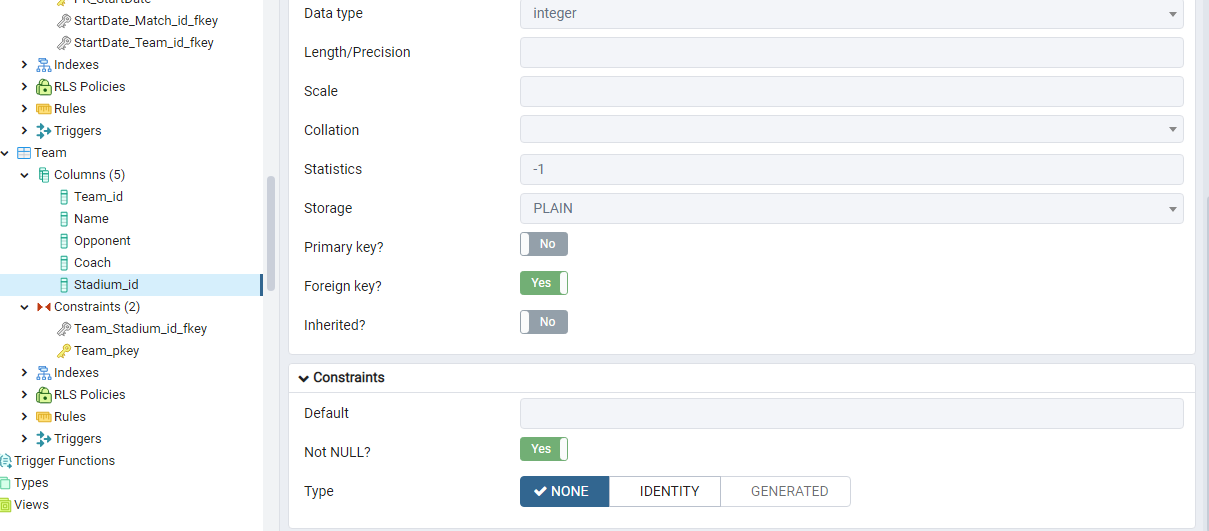


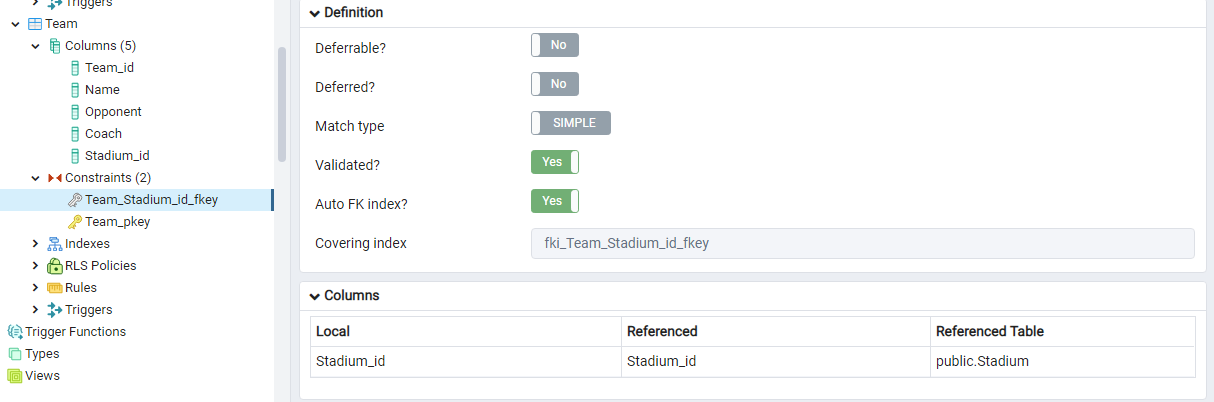


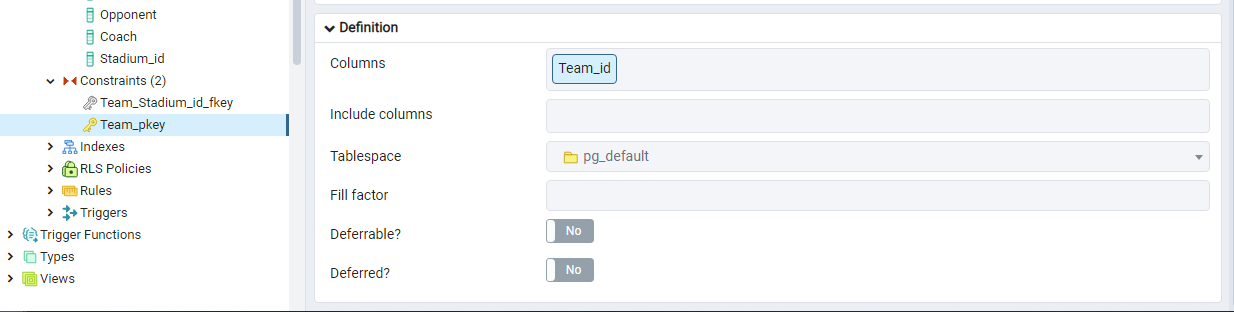




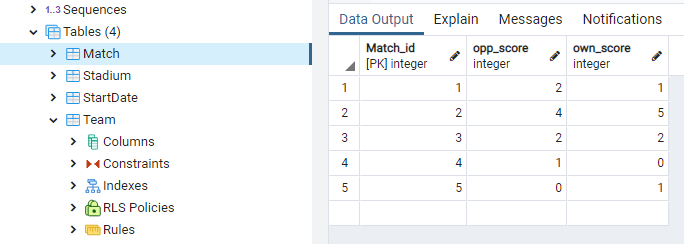


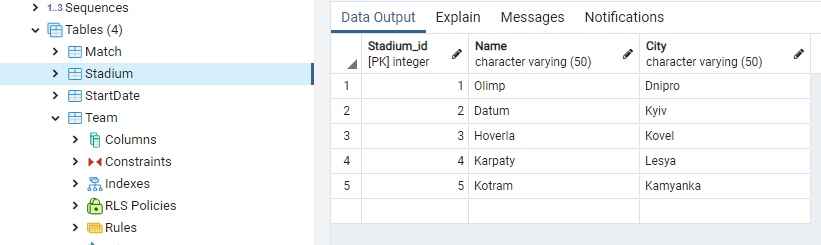


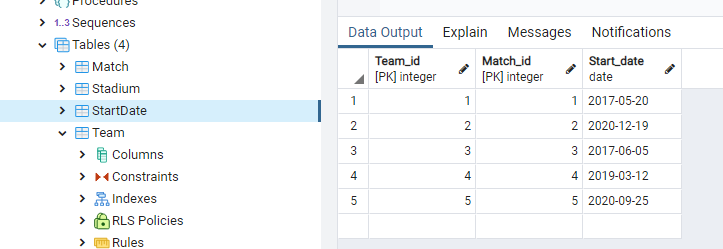


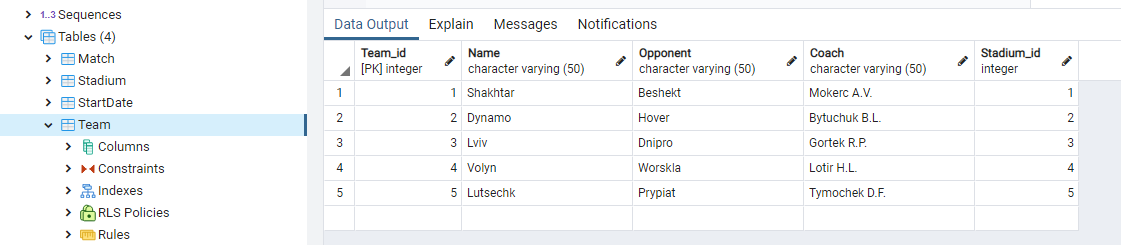


**Копії екрану з pgAdmin4, що відображають вміст таблиць бази даних у PostgreSQL:**









**Відповіді на контрольні запитання:**

1. Сформулювати призначення діаграм типу «сутність-зв’язок».

Діаграми типу “сутність-зв’язок” використовуються для створення дизайну бази даних. Це дає змогу побачити модель бази даних на більш високому рівні ще до її кінцевої реалізації. Що дозволяє переконатися в тому, що в діаграмі було зафіксовано усе що потрібно та налаштувати її ще на логічному рівні до того, коли ви перейдете до складніших процесів створення бази даних. Можна буде просто показати дизайн нетехнічному клієнту, щоб зрозуміти чи всі вимоги були виконані правильно. До того ж цей вид діаграм широко використовується в проектуванні або аналізу баз даних в бізнес процесах.

1. Назвати основні об’єкти схеми PostgreSQL.

У PostgreSQL схема є простором імен, що містить іменовані об'єкти бази даних, такі як таблиці, подання, індекси, типи даних, функції, збережені процедури та оператори.

1. Навести приклади різних типів зв’язків у базах даних (1:1, 1:N, N:M).

- Один до одного: одна сутність асоціюється з іншою сутністю. Наприклад: Кожна людина може мати лише один паспорт, або ж шлюб чоловіка та жінки. Вчителька молодших класів має лише один клас.

- Один до багатьох: одна сутність асоціюється з багатьма іншими сутностями. Наприклад: користувач має багато замовлень, вчителька має багато учнів, курс складається з багатьох груп.

- Багато до багатьох: Багато сутностей пов'язані з багатьма іншими сутностями. Наприклад: покупець може купити більш ніж один продукт, а продукт може бути куплений більш ніж один раз. Викладач може викладати багатьом групам, а група може вчитися у багатьох викладачів.