A black and white drawing of a building

Description automatically generated

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського» ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

# Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Розрахунково-графічна робота**

з дисципліни **Бази даних і засоби управління**

*на тему: “Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL”*

Виконав:

студент ІІI курсу

групи КВ-23

Зінєдін Шайдін

Перевірив:

Київ – 2024

*Метою роботи* є здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.

*Загальне* *завдання* роботи полягає у наступному:

* Реалізувати функції перегляду, внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
* Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
* Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.
* Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

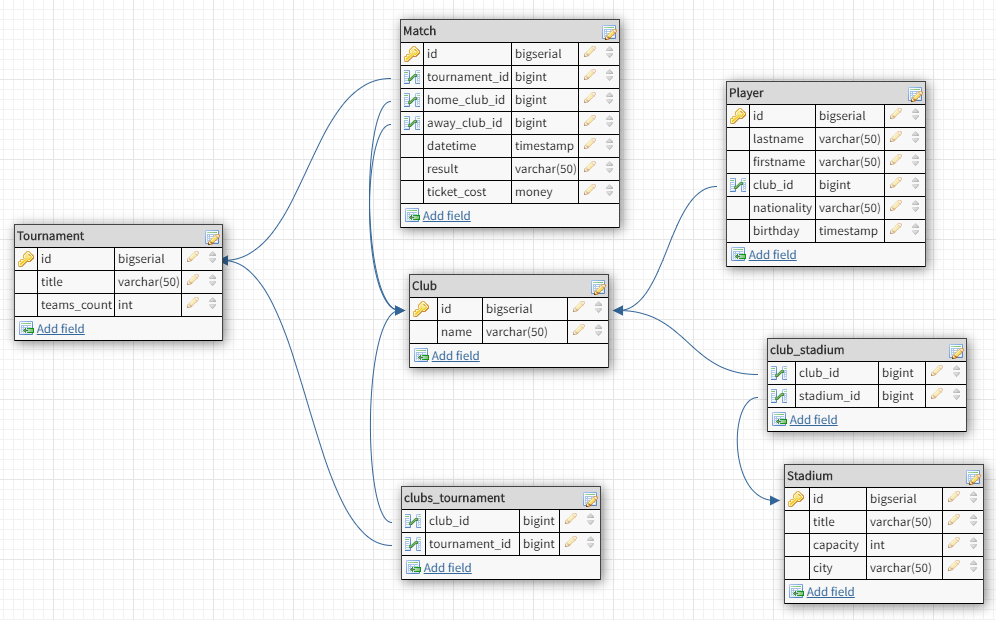
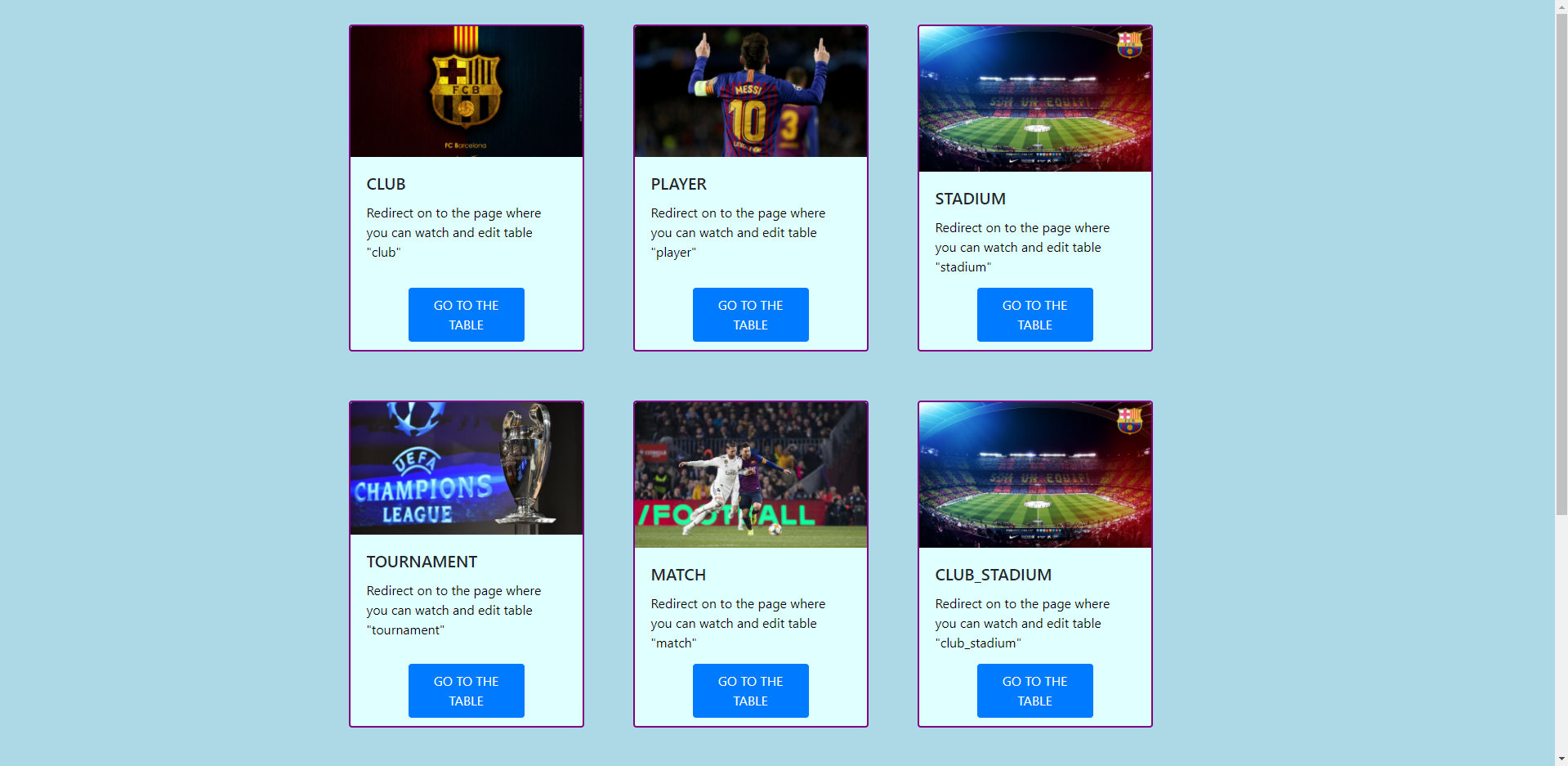
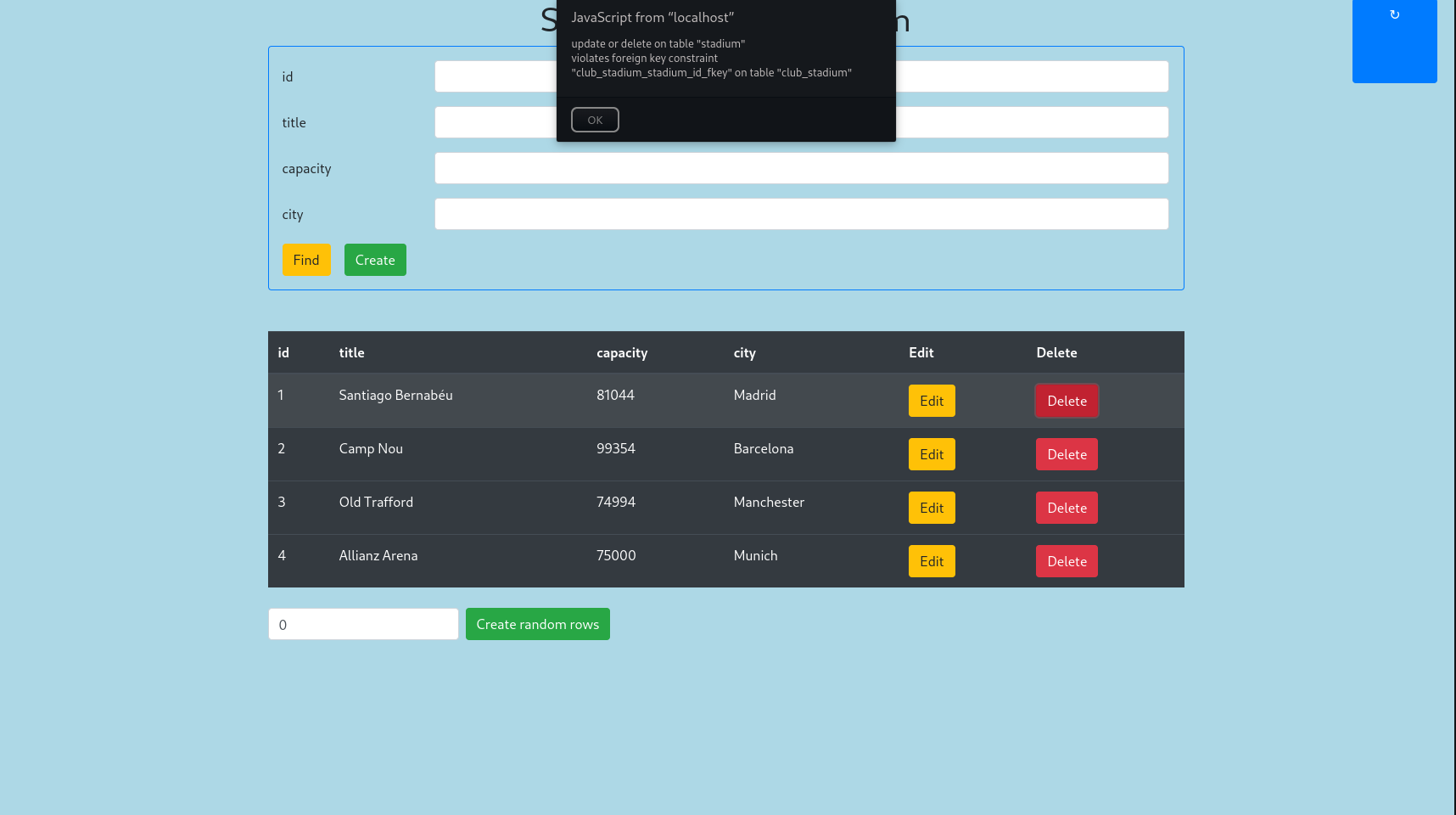


Рисунок 1 – Схема бази даних

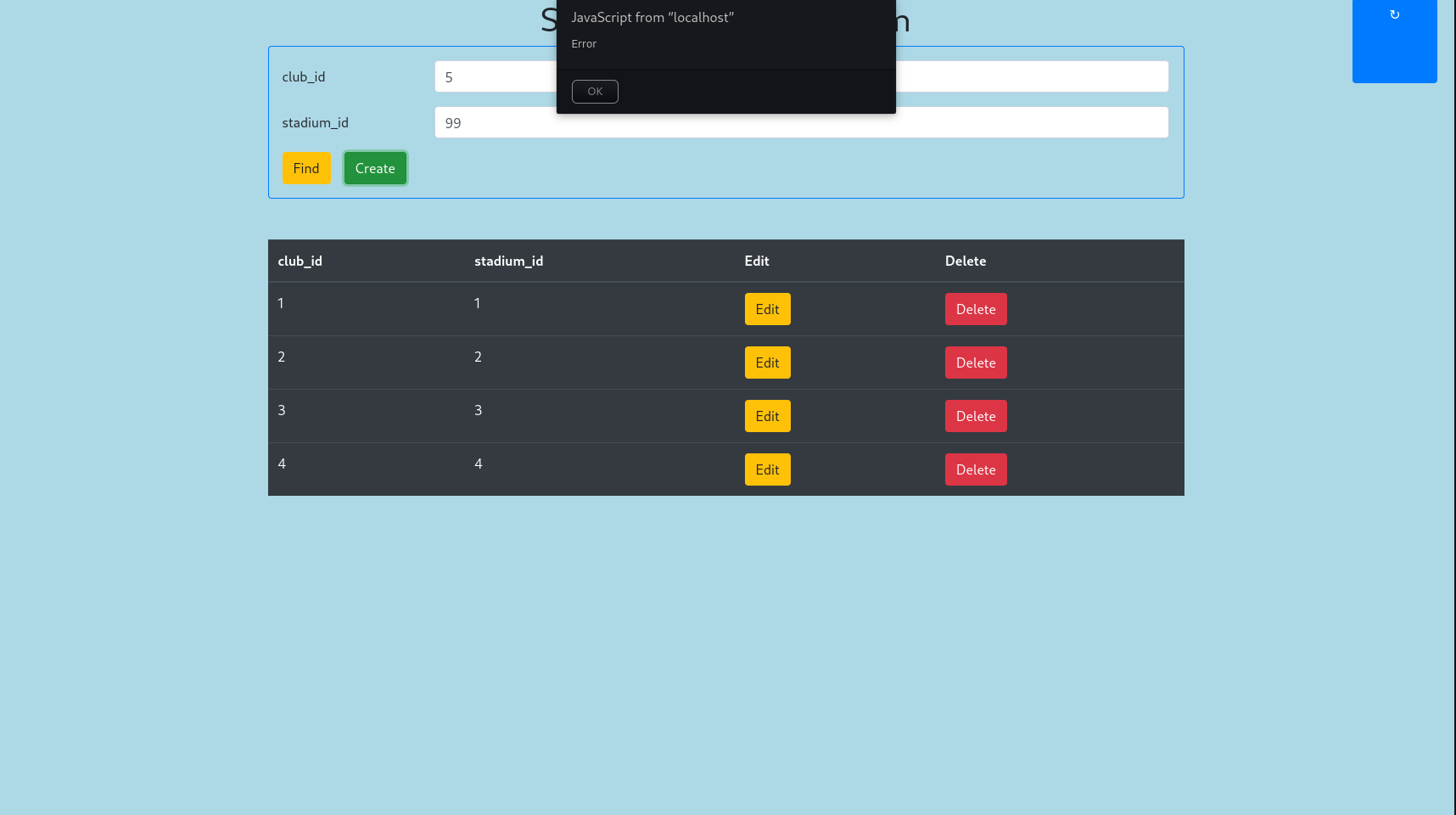
Бібліотеки, що були викорситані: pg, express, cors

Загальне меню користувача має наступний вигляд:  
  


*1. Функції перегляду, внесення, редагування та вилучення даних у таблицях бази даних*

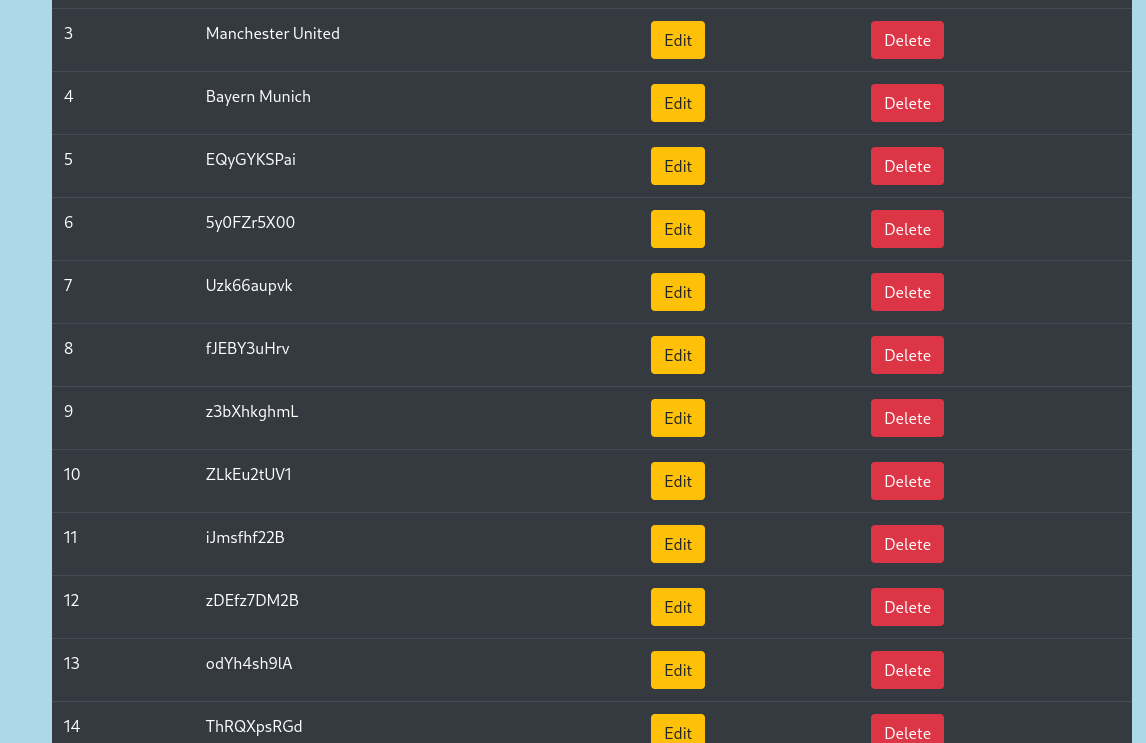
**

*update or delete on table "stadium" violates foreign key constraint "club\_stadium\_stadium\_id\_fkey" on table "club\_stadium"*

**

*Error: can not insert foreign key that does not exist*

*2. Автоматичне пакетне генерування «рандомізованих» даних у базі.*

  
INSERT INTO club(name) select getrandomstring(10) as name from generate\_series(1, ${count})

CREATE OR REPLACEFUNCTIONgetrandomstring(lengthINT)

RETURNS TEXT AS $$

DECLARE

   characters TEXT := 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';

   result TEXT := '';

   i INT := 0;

BEGIN

   FOR i IN 1..length LOOP

       result := result || substring(characters FROM trunc(random() \* length(characters) + 1)::int FOR 1);

   END LOOP;

   RETURN result;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION randomnum(length INT)

RETURNS BIGINT AS $$

DECLARE

   result BIGINT := 0;

   i INT := 0;

BEGIN

   IF length < 1 THEN

       RAISE EXCEPTION 'Length must be greater than 0';

   END IF;

   FOR i IN 1..length LOOP

       result := result \* 10 + trunc(random() \* 10)::int;

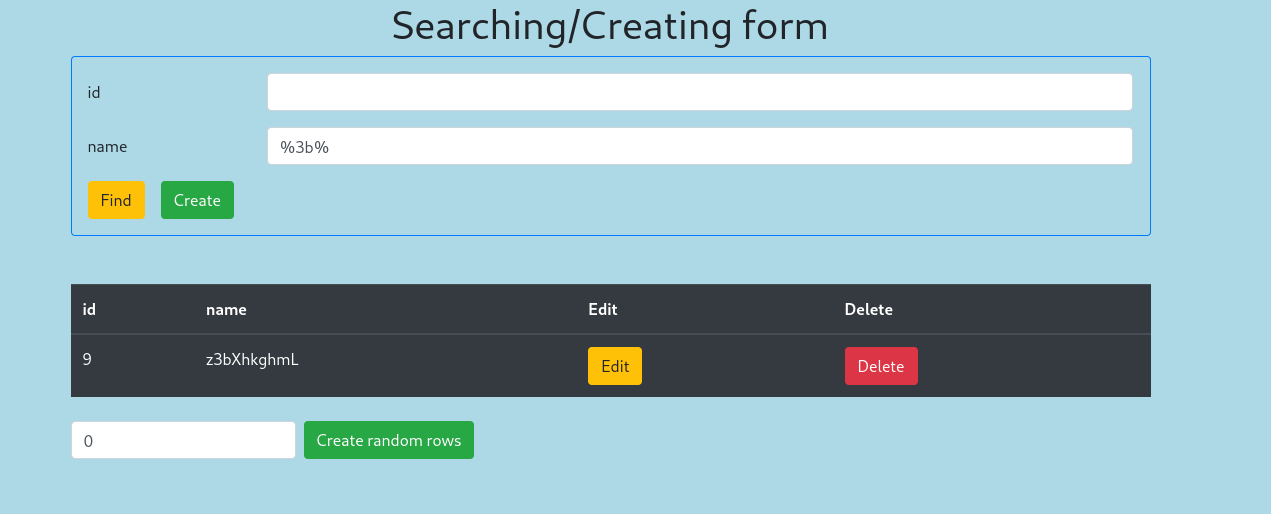
   END LOOP;

   RETURN result;

END;

$$ LANGUAGE plpgsql;

*3. Реалізація пошуку за декількома атрибутами з двох та більше сутностей одночасно*



let qu = `SELECT \* FROM stadium WHERE TRUE `

       if(title !== undefined && title.length !== 0) {

           qu += `AND LOWER(title) LIKE \'%${title}%\' `

       }

       if(city !== undefined && city.length !== 0) {

           qu += `AND LOWER(city) LIKE \'%${city}%\' `

       }

       if(capacity !== undefined && capacity.length !== 0) {

           qu += `AND capacity = ${capacity} `

       }

       qu += ';'

       const response = await pool.query(qu)

let {lastname, firstname, club\_id, birthday} = req.body

       let qu = `${selectAll} player WHERE TRUE `

       if(lastname !== undefined && lastname.length !== 0) {

           qu += `AND LOWER(lastname) LIKE \'%${lastname}%\' `

       }

       if(firstname !== undefined && firstname.length !== 0) {

           qu += `AND LOWER(firstname) LIKE \'%${firstname}%\' `

       }

       if(club\_id !== undefined && club\_id.length !== 0) {

           qu += `AND club\_id = ${club\_id} `

       }

       if(birthday !== undefined && birthday.length !== 0) {

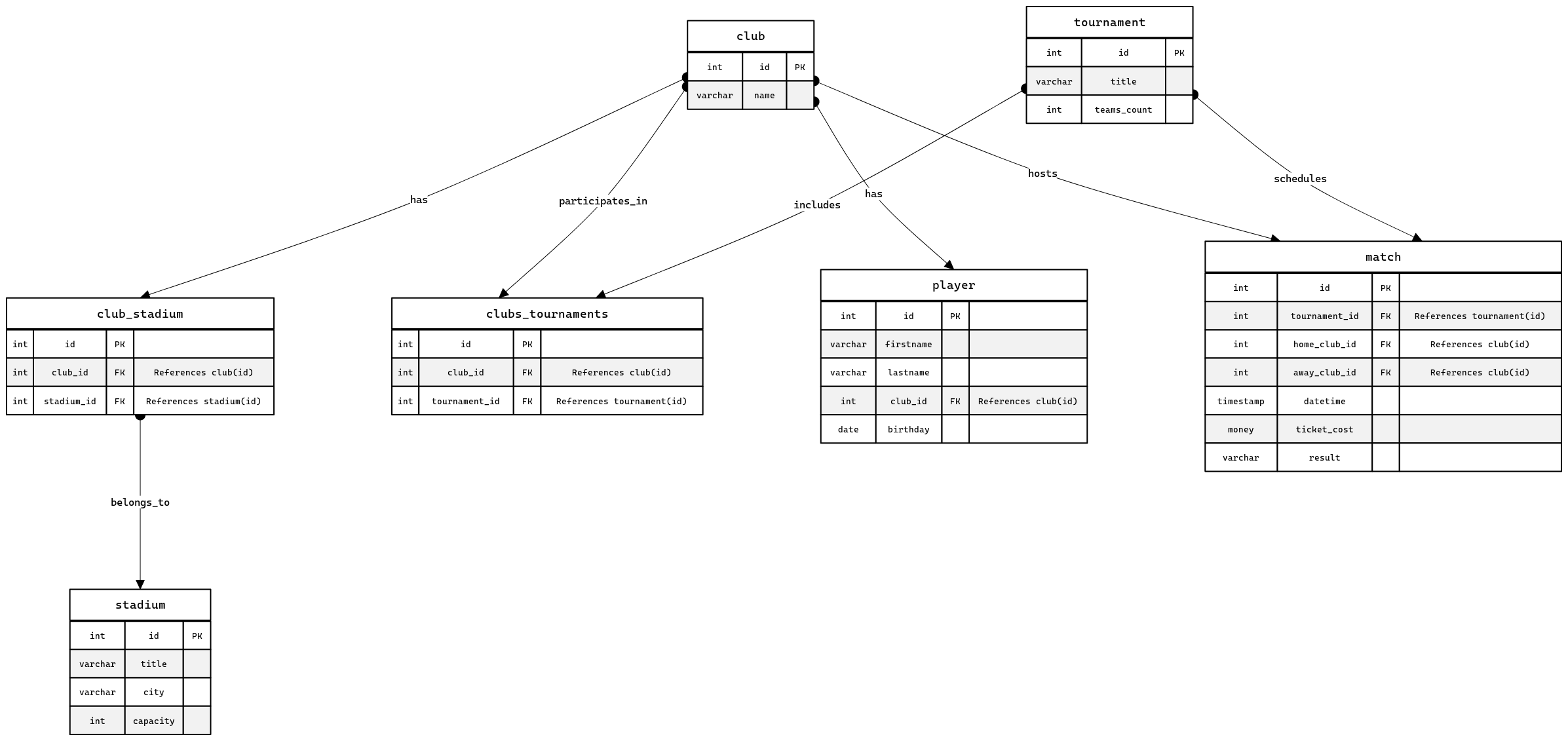
           qu += `AND birthday = '${birthday}'::date`

       }

       qu += ';'

       const response = await pool.query(qu)

*4. Код згідно шаблону MVC (модель-подання-контролер)*

**

**Модель** у цьому проєкті є ключовим шаром даних та бізнес-логіки в архітектурі **MVC (Model-View-Controller)**. Вона відповідає за визначення схеми бази даних, взаємозв'язків між сутностями та основних операцій з даними.

Основні функції Моделі включають:

* Обробку запитів до бази даних, включаючи **CRUD-операції** (створення, читання, оновлення, видалення).
* Забезпечення логічної цілісності даних шляхом перевірки правил та залежностей.
* Представлення даних у структурованому вигляді для передачі в інші шари застосунку (Контролер або Відображення).
* Забезпечення зв'язків між сутностями, таких як відносини "один до багатьох", "багато до багатьох" тощо.

Цей модуль також відіграє критичну роль у реалізації бізнес-логіки, обробляючи спеціальні правила та обчислення, що стосуються футбольних клубів, турнірів, стадіонів, гравців та матчів. Він є фундаментом для надійного та ефективного функціонування всієї системи.

*Посилання на репозиторій GitHub:* *https://github.com/9koto/RGR\_Zinedin\_Shaidin*