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**

**Лабораторна робота №1**

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-34

Вальчишен С.

Перевірив:

Павловский В. І.

**Київ – 2025**

**Мета:** здобуття вмінь проектування бази даних та практичних навичок створення реляційних баз даних за допомогою PostgreSQL.

**Завдання:**

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.

**Опис предметної області**

Предметна область – **Електронна база даних для спортивних змагань з футболу**.

Система має зберігати дані про футбольні змагання, в який день і час вони будуть відбуватись, який тип змагань (чемпіонат, кубкові турніри, юнацькі змагання і тощо), список команд і гравців, а також результати цих змагань.

**Нотація ER-діаграми:** *“Пташина лапка” (Crow’s Foot)*.

**Контакти:**

Репозиторій Git: посилання;

https://github.com/ValchyshenSviatoslav/DB\_lab

1. **Розробка моделі «сутність-зв’язок» обраної предметної галузі**

**Опис сутностей**

1. **Player (Гравець)**

Призначення: зберігання довідкової інформації про гравців.

Основні атрибути:

* player\_id – ідентифікаційний номер гравця;
* name – повне ім’я гравця;
* position – позиція гравця на полі;
* **team\_id – FK → team(team\_id)** – залежність гравця від команди;

1. **Team (Команда)**

Призначення: зберігання довідкової інформації про команди.

Основні атрибути:

* team\_id – ідентифікаційний номер команди;
* name – назва команди;
* city – місто де сформована команда;

1. **Match (Змагання)**

Призначення: зберігання довідкової інформації про змагання.

Основні атрибути:

* match\_id – ідентифікаційний номер змагань;
* **type\_id** – **FK → type(type\_id)** залежність змагань від їх типу;
* date – на який день заплановано проведення;
* time – на яку годину заплановано;
* **home\_team\_id** – **FK → team(team\_id)** залежність змагань від команди один;
* **away\_team\_id** – **FK → team(team\_id)** залежність змагань від команди два;

1. **Type (Тип змагань)**

Призначення: зберігає довідкову інформацію про тип змагань (чемпіонат, кубкові турніри, юнацькі змагання і тощо).

Основні атрибути:

* type\_id– ідентифікаційний номер типу змагань;
* type\_name – назва типу змагань;

1. **Result (Результат)**

Призначення: зберігає довідкову інформацію про результат змагань.

Основні атрибути:

* result\_id – ідентифікаційній номер результатів;
* **match\_id –** **FK → match(match\_id)** залежність від змагань**;**
* home\_team\_goals – кількість голів команди один;
* away\_team\_goals – кількість голів команди два;

**Опис зв’язків між сутностями**

1. **Team – Player: 1:N (один до багатьох, обов’язковий)**

Сенс: кожен гравець належить до певної команди, але одна команда може мати багато гравців.

Кардинальність та обов’язковість:

* **Team → Player:** 1..\* (команда може мати багато гравців);
* **Player → Team:** 1 (гравець належить лише одній команді).

1. **Type – Match: 1:N (один до багатьох, обов’язковий)**

Сенс: кожен матч належить до певного типу змагань, а один тип охоплює багато матчів.

Кардинальність та обов’язковість:

* **Type → Match:** 1..\* (один тип змагань має багато матчів);
* **Match → Type:** 1 (кожен матч має один тип змагань).

1. **Team – Team (через Match): N:M (багато до багатьох, реалізовано через Match)**

Сенс: матч завжди включає дві команди — господаря (*home\_team\_id*) та гостя (*away\_team\_id*).

Таким чином, одна команда може брати участь у багатьох матчах як господар або гість, а один матч пов’язує дві різні команди.

Зв’язок між командами реалізовано через проміжну сутність Match, яка фіксує їхню участь у конкретному змаганні.

Кардинальність та обов’язковість:

* **Team → Match:** 1..\* (кожна команда може брати участь у багатьох матчах);
* **Match → Team:** 2 (у кожному матчі беруть участь рівно дві команди).

1. **Match – Result: 1:1 (один до одного, обов’язковий)**

Сенс: кожен матч має рівно один результат, який зберігає кількість забитих голів обох команд.

Кардинальність та обов’язковість:

* **Match → Result:** 1 (для кожного матчу існує один результат);
* **Result → Match:** 1 (результат належить лише одному матчу).

Графічне подання концептуальної моделі «Сутність-зв’язок» зображено на рисунку 1.

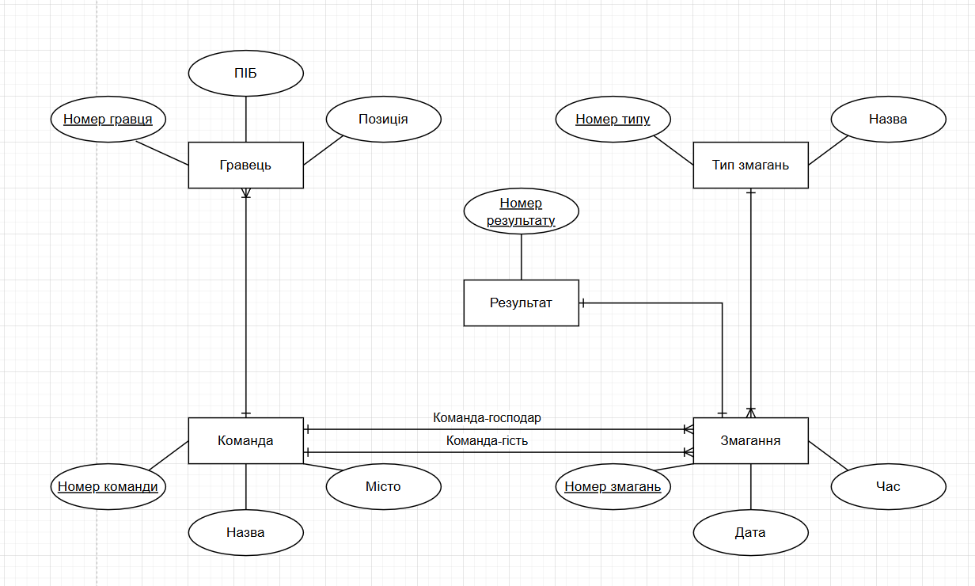


Рисунок 1 - ER-діаграма

1. **Перетворення концептуальної моделі у логічну модель та схему бази даних**

Таблиця 1 ілюструє детальний перехід від однієї моделі до іншої.

*Таблиця 1 – Опис об’єктів бази даних*

|  |  |  |
| --- | --- | --- |
| Сутність | Атрибут | Тип атрибуту |
| Player | player\_id | integer |
| player\_name | varchar (100) |
| position | varchar (50) |
| team\_id | integer |
| Team | team\_id | integer |
| team\_name | varchar (80) |
| city | varchar (80) |
| Match | match\_id | integer |
| type\_id | integer |
| date | date |
| time | time |
| home\_team\_id | integer |
| away\_team\_id | integer |
| Type | type\_id | integer |
| type\_name | varchar (100) |
| Result | result\_id | integer |
| match\_id | integer |
| home\_team\_goals | integer |
| away\_team\_goals | integer |

**Функціональні залежності для кожної таблиці**

**Team**

* team\_id → team\_name, city

**Player**

* player\_id → name, position, team\_id
* team\_id → team (через зовнішній ключ, залежність від *Team*)

**Type**

* type\_id → type\_name

**Match**

* match\_id → type\_id, date, time, home\_team\_id, away\_team\_id
* type\_id → type (через зовнішній ключ, залежність від *Type*)
* home\_team\_id → home\_team (через зовнішній ключ, залежність від таблиці *Team*)
* away\_team\_id → away\_team (через зовнішній ключ, залежність від таблиці *Team*)

**Result**

* result\_id → match\_id, home\_team\_goals, away\_team\_goals
* match\_id → match (через зовнішній ключ, залежність від *Match*)

1. **Виконання нормалізації схеми бази даних до третьої нормальної форми (3НФ).**

Графічне подання логічної моделі «Сутність-зв’язок» зображено на рисунку 2.

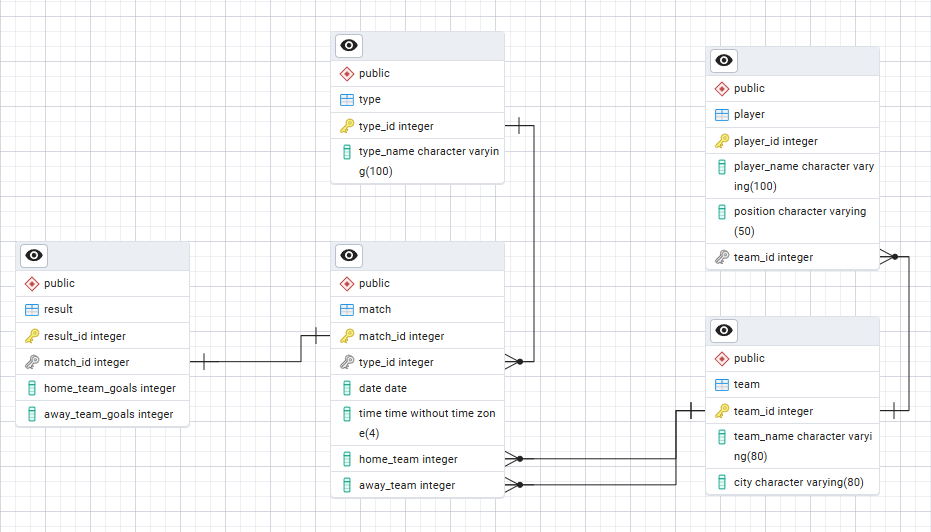


Рисунок 2 - Схема бази даних

**Перша нормальна форма (1НФ).**

Усі атрибути таблиць є атомарними, тобто містять лише неподільні значення. Для кожної сутності визначено унікальний первинний ключ (*player\_id*, *team\_id*, *match\_id*, *type\_id*, *result\_id*), який однозначно ідентифікує записи. Повторювані групи відсутні, а кожен атрибут зберігає єдине значення (наприклад, у таблиці *Match* вказано дві команди через окремі зовнішні ключі — *home\_team\_id* і *away\_team\_id*). Таким чином, усі таблиці відповідають вимогам 1НФ.

**Друга нормальна форма (2НФ).**

У всіх таблицях первинні ключі є простими, тому кожен неключовий атрибут повністю залежить від свого первинного ключа. Часткових залежностей не існує: у таблиці *Player* усі атрибути залежать лише від *player\_id*, у *Match* — лише від *match\_id*, де вказані команди-господар і гість. У таблицях *Type*, *Team* і *Result* аналогічно всі атрибути залежать тільки від своїх первинних ключів. Схема бази даних задовольняє вимогам другої нормальної форми.

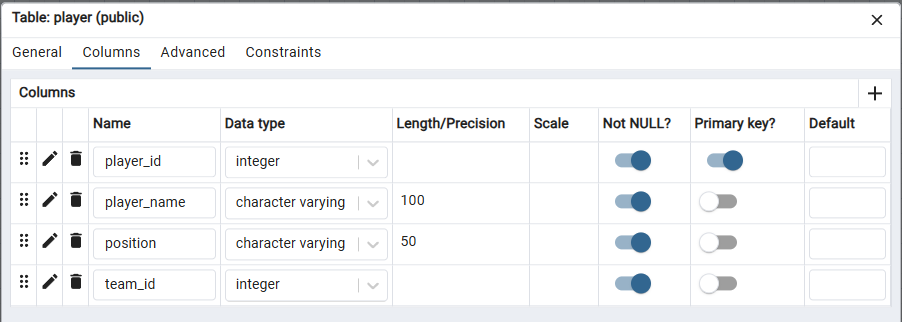
**Третя нормальна форма (3НФ).**

Усі неключові атрибути залежать безпосередньо від первинних ключів і не мають транзитивних залежностей. Дані про команди, типи змагань, гравців та результати розміщені в окремих таблицях, що виключає дублювання. Таблиця *Match* реалізує зв’язок між двома екземплярами сутності *Team* через зовнішні ключі *home\_team\_id* і *away\_team\_id*, що забезпечує правильне моделювання участі команд у матчах без надлишкових даних. Таким чином, схема бази даних повністю відповідає вимогам третьої нормальної форми (3НФ) — усі залежності визначені правильно, а структура є логічно узгодженою та мінімальною за надлишковістю.

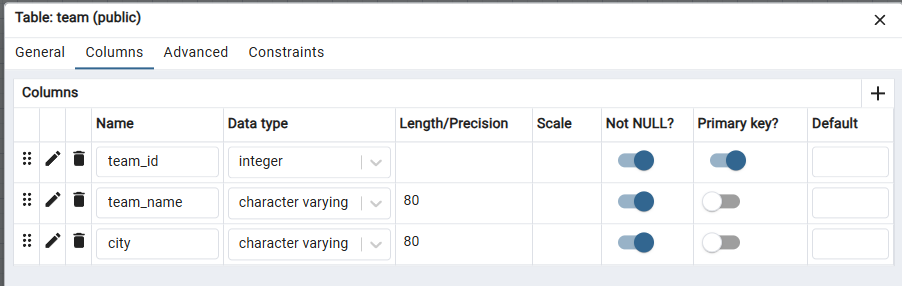
1. **Ознайомлення та розробка таблиць баз даних у pgAdmin4**

**Таблиці бази даних у pgAdmin4**

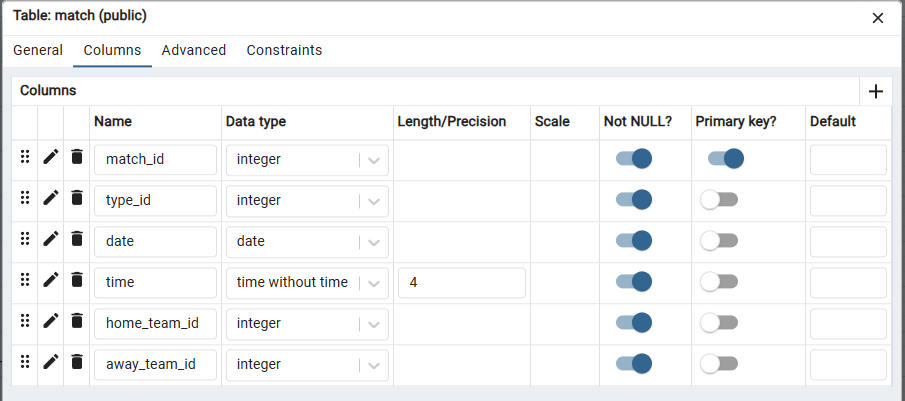
Таблиця Player



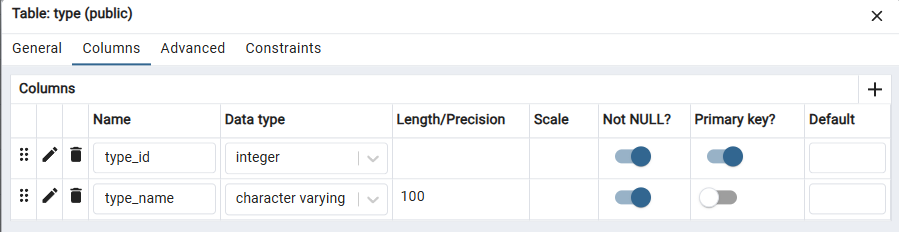
Таблиця Team



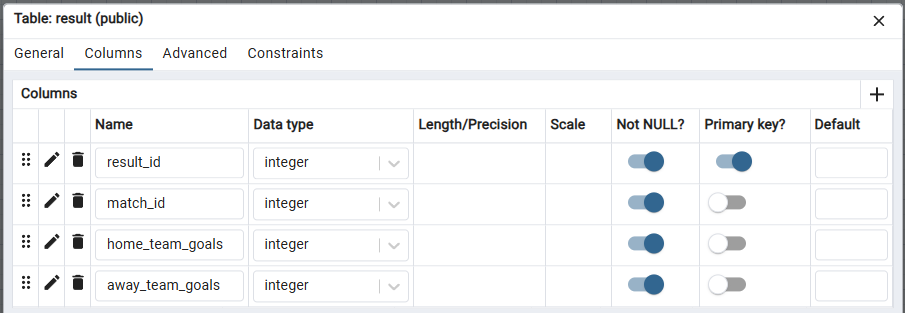
Таблиця Match



Таблиця Type

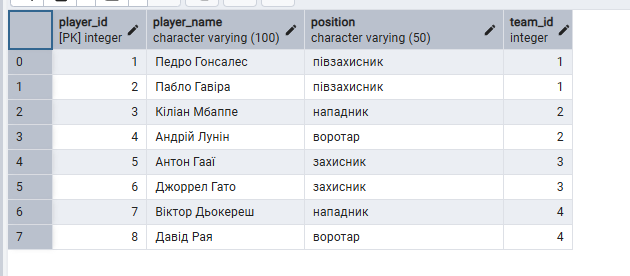


Таблиця Result

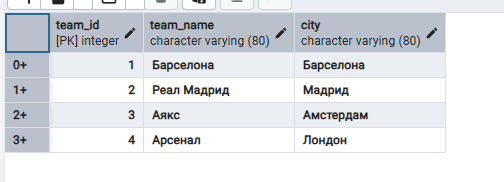


До таблиць внесено по декілька рядків інформації.

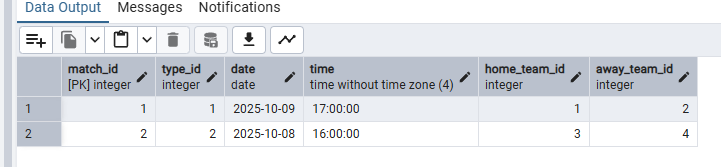
Таблиця Player



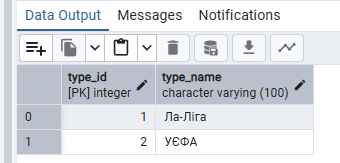
Таблиця Team



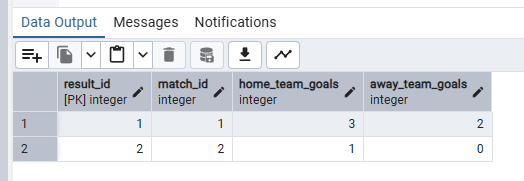
Таблиця Match



Таблиця Type



Таблиця Result



**Таблиці в коді SQL**

**BEGIN;**

**CREATE TABLE IF NOT EXISTS public.team**

**(**

**team\_id integer NOT NULL,**

**team\_name character varying(80) NOT NULL,**

**city character varying(80) NOT NULL,**

**CONSTRAINT teamid\_pkey PRIMARY KEY (team\_id)**

**);**

**CREATE TABLE IF NOT EXISTS public.player**

**(**

**player\_id integer NOT NULL,**

**player\_name character varying(100) NOT NULL,**

**"position" character varying(50) NOT NULL,**

**team\_id integer NOT NULL,**

**CONSTRAINT playerid\_pkey PRIMARY KEY (player\_id)**

**);**

**CREATE TABLE IF NOT EXISTS public.type**

**(**

**type\_id integer NOT NULL,**

**type\_name character varying(100) NOT NULL,**

**CONSTRAINT typeid\_pkey PRIMARY KEY (type\_id)**

**);**

**CREATE TABLE IF NOT EXISTS public.match**

**(**

**match\_id integer NOT NULL,**

**type\_id integer NOT NULL,**

**date date NOT NULL,**

**"time" time(4) without time zone NOT NULL,**

**home\_team\_id integer NOT NULL,**

**away\_team\_id integer NOT NULL,**

**CONSTRAINT matchid\_pkey PRIMARY KEY (match\_id)**

**);**

**CREATE TABLE IF NOT EXISTS public.result**

**(**

**result\_id integer NOT NULL,**

**match\_id integer NOT NULL,**

**home\_team\_goals integer NOT NULL,**

**away\_team\_goals integer NOT NULL,**

**CONSTRAINT resultid\_pkey PRIMARY KEY (result\_id),**

**UNIQUE (match\_id)**

**);**

**ALTER TABLE IF EXISTS public.player**

**ADD FOREIGN KEY (team\_id)**

**REFERENCES public.team (team\_id) MATCH SIMPLE**

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**NOT VALID;**

**ALTER TABLE IF EXISTS public.match**

**ADD FOREIGN KEY (type\_id)**

**REFERENCES public.type (type\_id) MATCH SIMPLE**

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**NOT VALID;**

**ALTER TABLE IF EXISTS public.match**

**ADD FOREIGN KEY (team\_one\_id)**

**REFERENCES public.team (team\_id) MATCH SIMPLE**

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**NOT VALID;**

**ALTER TABLE IF EXISTS public.match**

**ADD FOREIGN KEY (team\_two\_id)**

**REFERENCES public.team (team\_id) MATCH SIMPLE**

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**NOT VALID;**

**ALTER TABLE IF EXISTS public.result**

**ADD FOREIGN KEY (match\_id)**

**REFERENCES public.match (match\_id) MATCH SIMPLE**

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**NOT VALID;**

**END;**

**Висновок**

У ході виконання лабораторної роботи було розроблено концептуальну, логічну та фізичну моделі бази даних для предметної області **«**Електронна база даних для спортивних змагань з футболу**»**.

Було створено ER-діаграму з використанням нотації *Crow’s Foot*, визначено сутності, атрибути та типи зв’язків між ними.

Схему бази даних перетворено у формат, сумісний із PostgreSQL, де кожна таблиця має первинний ключ, а зв’язки реалізовано за допомогою зовнішніх ключів.

Проведено нормалізацію бази даних до третьої нормальної форми (3НФ), що дозволило усунути надлишковість даних та забезпечити логічну цілісність структури.

За допомогою інструментів pgAdmin 4 було створено таблиці, внесено приклади даних і перевірено коректність зв’язків між ними.

Таким чином, поставлену мету досягнуто — отримано практичні навички проектування, нормалізації та реалізації реляційних баз даних у середовищі PostgreSQL.