

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 ІII курсу

групи КВ-91

Гальфінгер Е. В.

Перевірив:

Павловський В. І.

Київ – 2021

*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 занести декілька рядків даних у кожну з таблиць засобами pgAdmin 4.

*Вимоги до ER-моделі:*

1. Сутності моделі предметної галузі мають містити зв’язки типу 1:N або N:M.
2. Кількість сутностей у моделі – 3-4. Кількість атрибутів у кожній сутності: від двох до п’яти.
3. Передбачити наявність зв’язку з атрибутом.

Для побудови ER-діаграм використовувати одну із нотацій: Чена, “Пташиної лапки (Crow’s foot)”, UML.

**Опис предметної галузі**

Тема: Ігрова індустрія(player, game, competition)

При проектуванні бази даних “Ігрова індустрія” можна виділити наступні сутності: відомості про гравця, відомості про команду, відомості про гру, відомості про змагання (турнір).

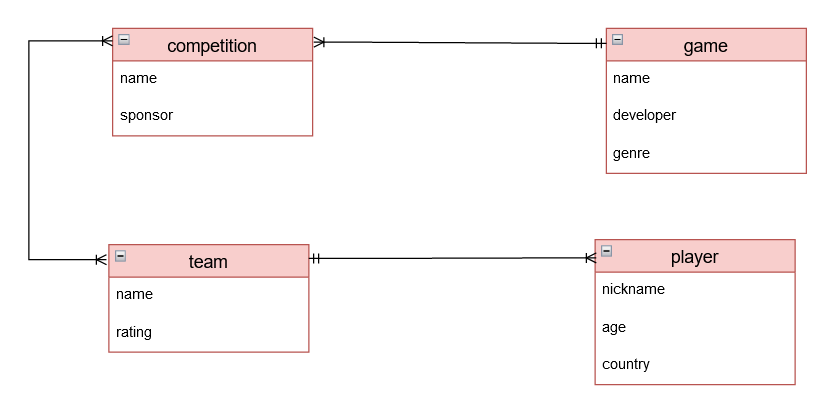
Атрибути заданих сутностей:

* + - 1. player: nickname, age, country
      2. team: name, rating
      3. game: name, developer, genre
      4. competition: name, sponsor

Опис зв’язків

В певній команді можуть бути декілька гравців, отже між сутностями player і team зв’язок 1:N.

Певний турнір відбувається по одній конкретній грі, однак по певній грі можуть відбуватися різні турніри, отже між сутностями game і competition зв’язок 1:N.

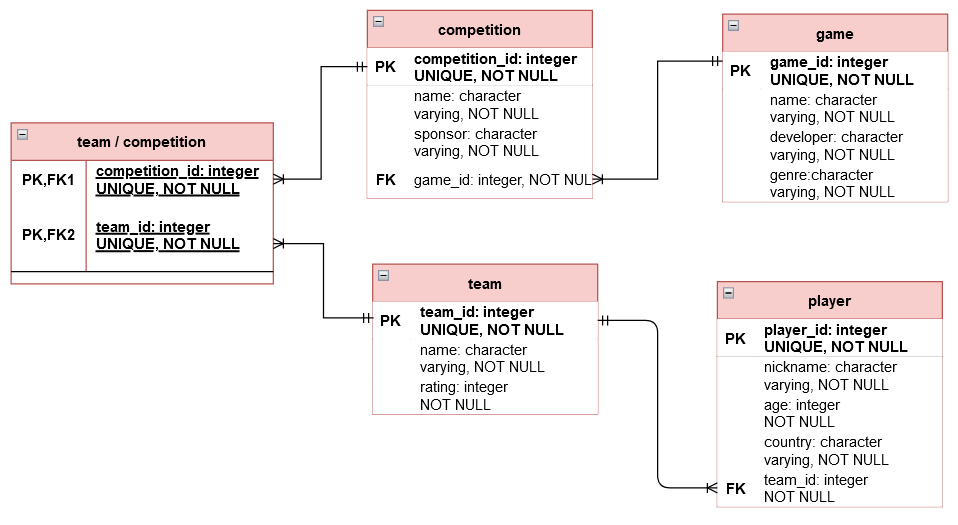
 Певна команда може брати участь в багатьох турнірах, також в певному турнірі можуть брати участь багато команд, тож між сутностями team і competition зв’язок N:M.

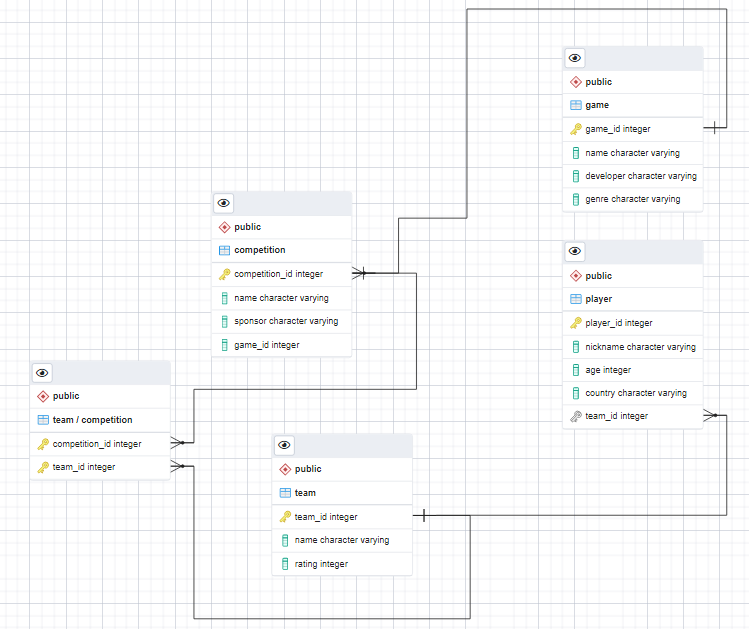
**Перетворення концептуальної моделі у схему баз даних**

Для кожної сутності створюється таблиця. Причому кожному атрибуту сутності відповідає стовпець таблиці. В даній моделі перетворення в схему баз даних відбувалося за такими правилами:

Якщо зв'язок типу 1: N і клас приналежності сутності на стороні N є обов'язковим, то необхідно побудувати таблицю для кожної сутності. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Первинний ключ сутності на стороні 1 додається як атрибут в таблицю для сутності на стороні N. Даний зв’язок можна спостерігати між сутностями dormitory і room, dormitory і inmate, room\_type і room.

Якщо зв'язок типу N: M, то необхідно побудувати три таблиці - по одній для кожної сутності і одну для зв'язку. Первинний ключ сутності повинен бути первинним ключем відповідної таблиці. Таблиця для зв'язку серед своїх атрибутів повинна мати ключі обох сутностей. Даний зв’язок можна спостерігати між сутностями room і inmate.





**Відповідність схеми бази даних до третьої нормальної форми**

Схема відповідає 1НФ, тому що:

1. В таблиці немає дубльованих рядків.
2. В кожній комірці зберігається атомарне (скалярне) значення.
3. В кожному стовпці зберігаються дані одного типу.

Схема відповідає 2НФ, тому що:

1. Вона відповідає 1НФ.
2. Має первинний ключ, а всі не ключові стовпці таблиці залежать від первинного ключа.

Схема відповідає 3НФ, тому що:

1. Вона відповідає 2НФ.
2. Всі не ключові атрибути таблиці залежать винятково від усього первинного ключа, а не його частини. Тобто кожен неключовий атрибут нетранзитивно (без посередника) залежить від первинного ключа

SQL TEXT

**-- Database: cybersport**

**-- DROP DATABASE cybersport;**

**CREATE DATABASE cybersport**

**WITH**

**OWNER = postgres**

**ENCODING = 'UTF8'**

**LC\_COLLATE = 'Ukrainian\_Ukraine.1251'**

**LC\_CTYPE = 'Ukrainian\_Ukraine.1251'**

**TABLESPACE = pg\_default**

**CONNECTION LIMIT = -1;**

**-- Table: public.competition**

**-- DROP TABLE public.competition;**

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

**(**

**competition\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),**

**name character varying COLLATE pg\_catalog."default" NOT NULL,**

**sponsor character varying COLLATE pg\_catalog."default" NOT NULL,**

**game\_id integer NOT NULL,**

**CONSTRAINT competition\_pkey PRIMARY KEY (competition\_id),**

**CONSTRAINT fk\_game\_id FOREIGN KEY (competition\_id)**

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

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**)**

**TABLESPACE pg\_default;**

**ALTER TABLE public.competition**

**OWNER to postgres;**

**-- Table: public.game**

**-- DROP TABLE public.game;**

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

**(**

**game\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),**

**name character varying COLLATE pg\_catalog."default" NOT NULL,**

**developer character varying COLLATE pg\_catalog."default" NOT NULL,**

**genre character varying COLLATE pg\_catalog."default" NOT NULL,**

**CONSTRAINT game\_pkey PRIMARY KEY (game\_id)**

**)**

**TABLESPACE pg\_default;**

**ALTER TABLE public.game**

**OWNER to postgres;**

**-- Table: public.player**

**-- DROP TABLE public.player;**

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

**(**

**player\_id integer NOT NULL,**

**nickname character varying COLLATE pg\_catalog."default" NOT NULL,**

**age integer NOT NULL,**

**country character varying COLLATE pg\_catalog."default" NOT NULL,**

**team\_id integer NOT NULL,**

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

**CONSTRAINT fk\_team\_id FOREIGN KEY (team\_id)**

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

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**NOT VALID**

**)**

**TABLESPACE pg\_default;**

**ALTER TABLE public.player**

**OWNER to postgres;**

**-- Table: public.team**

**-- DROP TABLE public.team;**

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

**(**

**team\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),**

**name character varying COLLATE pg\_catalog."default" NOT NULL,**

**rating integer NOT NULL,**

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

**)**

**TABLESPACE pg\_default;**

**ALTER TABLE public.team**

**OWNER to postgres;**

**-- Table: public.team / competition**

**-- DROP TABLE public."team / competition";**

**CREATE TABLE IF NOT EXISTS public."team / competition"**

**(**

**competition\_id integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),**

**team\_id integer NOT NULL,**

**CONSTRAINT "team / competition\_pkey" PRIMARY KEY (competition\_id, team\_id),**

**CONSTRAINT fk\_competition\_id FOREIGN KEY (competition\_id)**

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

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION,**

**CONSTRAINT fk\_team\_id FOREIGN KEY (team\_id)**

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

**ON UPDATE NO ACTION**

**ON DELETE NO ACTION**

**)**

**TABLESPACE pg\_default;**

**ALTER TABLE public."team / competition"**

**OWNER to postgres;**

Таблиці бази даних у pgAdmin 4

