НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

**Кафедра системного програмування та спеціалізованих комп’ютерних**

**систем**

**Лабораторна робота №2**

з дисципліни

**«Бази даних і засоби управління»**

Тема: «Ознайомлення з базовими операціями СУБД PostgreSQL»

Виконав: студент III курсу

ФПМ групи КП-83

Бойчук В. А.

Київ – 2020

*Мета роботи:* здобуття практичних навичок проектування та побудови реляційних баз даних та створення прикладних програм з базами даних.

*Загальне завдання роботи полягає у наступному*:

1. Реалізувати функції внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. 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/Fаlse, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

*Деталізоване завдання*:

1. Забезпечити можливість уведення/редагування/вилучення даних у таблицях бази даних з можливістю контролю відповідність типів даних атрибутів таблиць (рядків, чисел, дати/часу). Для контролю пропонується два варіанти: контроль при введенні (валідація даних) та перехоплення помилок (try..except) від сервера PostgreSQL при виконанні відповідної команди SQL. Особливу увагу варто звернути на дані таблиць, що мають зв’язок 1:N. При цьому з боку батьківської таблиці необхідно контролювати **вилучення** рядків за умови наявності даних у підлеглій таблиці. З точки зору підлеглої таблиці варто контролювати наявність відповідності рядка у батьківській таблиці при виконанні **внесення** нових даних. Унеможливити виведення програмою системних помилок на екрані шляхом їх перехоплення і адекватної обробки. Внесення даних виконується у консольному вікні програми.
2. Забезпечити можливість автоматичної генерації великої кількості даних у таблицях за допомогою вбудованих у PostgreSQL функцій роботи з псевдовипадковими числами. Дані мають бути згенерованими **не мовою програмування, а відповідним SQL-запитом**!
3. Для реалізації пошуку необхідно підготувати 3 запити, включають дані з декількох таблиць і фільтрують рядки за 3-4 атрибутами цих таблиць. Забезпечити можливість введення конкретних значень констант для фільтрації з клавіатури користувачем. Крім того, після виведення даних необхідно вивести час виконання запиту у мілісекундах. Перевірити швидкодію роботи запитів на попередньо згенерованих даних.
4. Програмний код організувати згідно шаблону Model-View-Controller(MVC). При цьому модель, подання та контролер мають бути реалізовані у окремих файлах. Для доступу до бази даних використовувати **лише мову SQL** (без ORM).

**Логічна модель бази даних**

Нижче (Рис. 1) наведено логічну модель бази даних:

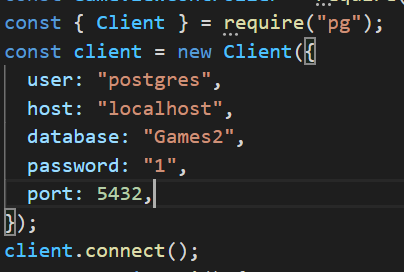
Рис. 1 – Логічна модель бази даних

Зміни у порівнянні з першою лабораторною роботою:

**Середовище розробки та налаштування підключення до бази даних**

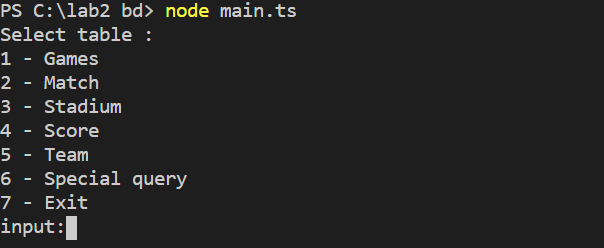
Для виконання лабораторної роботи використовувалась мова програмування TypeScript та текстовий редактор Visual Studio Code.

Для підключення до серверу бази даних PostgreSQL використовувався **npm** **пакет** **pg**. Для цього він підключився до проекту, з нього було «дістано» класс **Client** на його основі було створено змінну для керування базою даних:

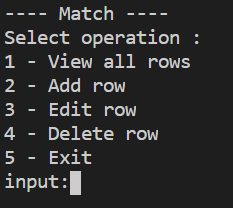


**Структура меню програми**

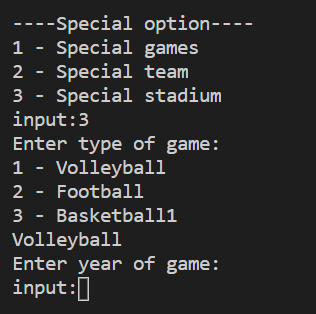
Головне меню



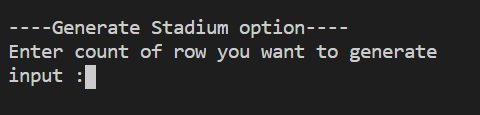
Меню для таблиці



Меню для вибору динамічних запитів



Меню вибору кількості рядків для генерації



**Посилання для навігації**

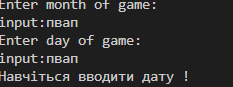
1. Ілюстрації обробки виняткових ситуацій (помилок) при введенні/вилучення даних;

ілюстрації валідації даних при введенні користувачем.

1. Копії екрану (ілюстрації) з фрагментами згенерованих даних таблиць.
2. Ілюстрації введення пошукового запиту та результатів виконання запитів.
3. Ілюстрації програмного коду з репозиторію Git.

**1)Ілюстрації обробки виняткових ситуацій (помилок) при введенні/вилучення даних;ілюстрації валідації даних при введенні користувачем**

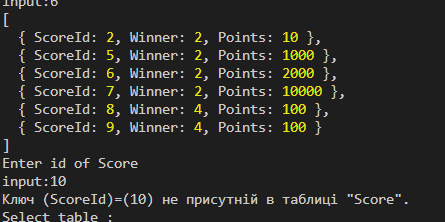
Валідація дати:



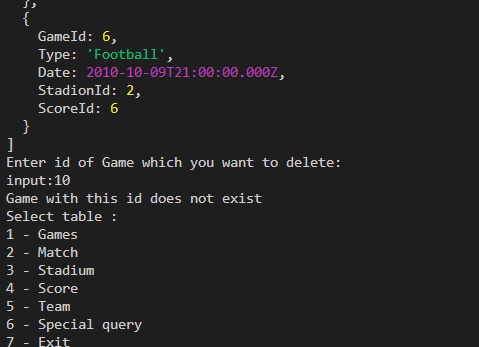
Валідація чисел:



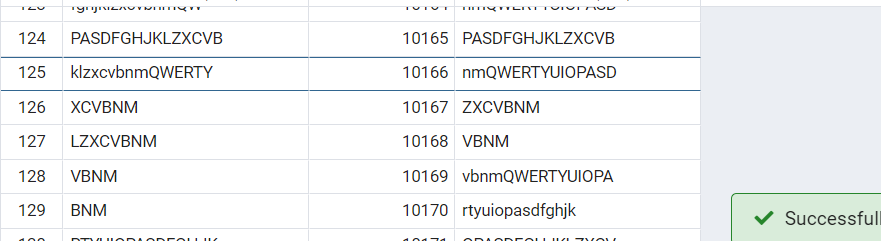
Валідація foreign key для всіх додавань/видалень/редагувань:

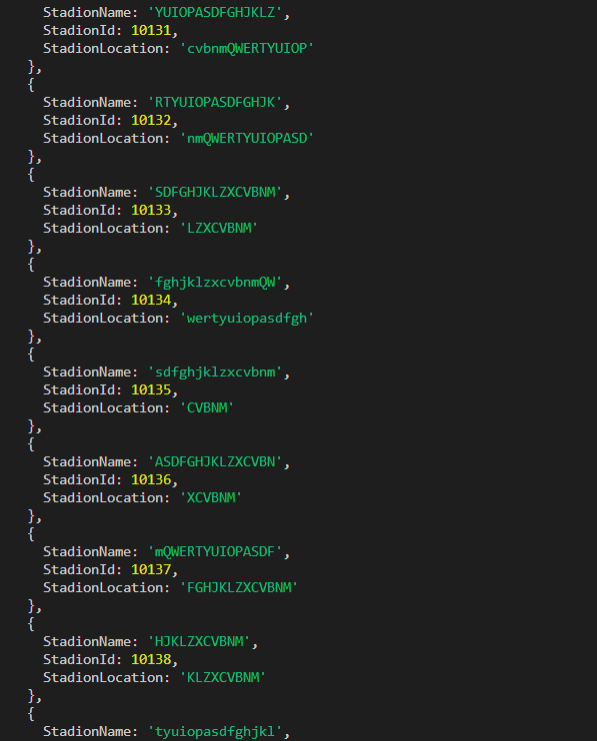


Валідація primary key для видалень/редагувань:

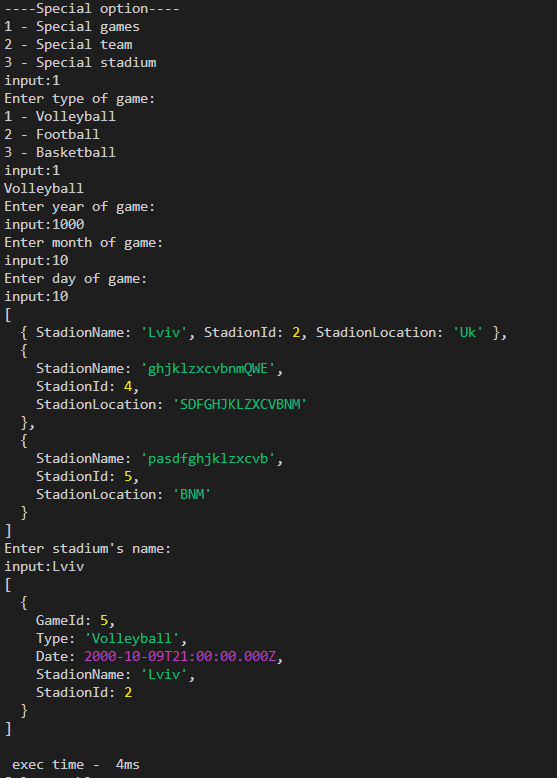


**2)Копії екрану (ілюстрації) з фрагментами згенерованих даних таблиць.**

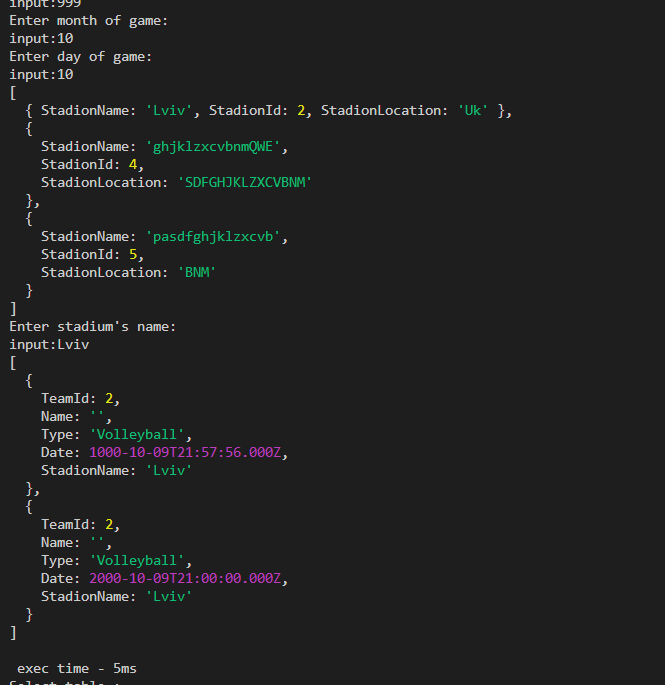




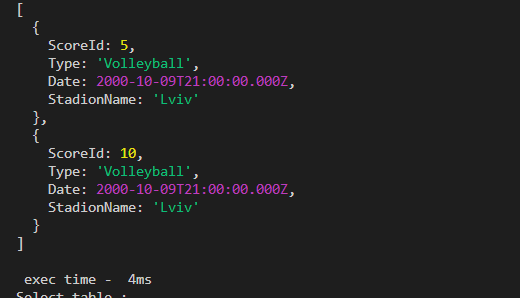
**3)Ілюстрації введення пошукового запиту та результатів виконання запитів.**

****

Даний запит виводить всі волейбольні ігри проведені на стадіоні Львів пізніше певної дати

****

Даний запит виводить всі команди які грали гру волейбол на стадіоні Львів пізніше певної дати

****

Даний запит виводить всі рахунки які грали гру волейбол на стадіоні Львів пізніше певної дати

**Ілюстрації програмного коду на Github**

