



МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики  
Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота № 2**  
з дисципліни «Бази даних та засоби управління»  
тема “Створення додатку бази даних, орієнтованого на взаємодію з СУБД”

Виконав  
Студент I курсу  
групи КП-01

Беліцький Олександр Сергійович

Перевірів  
“ \_\_\_\_ ” “ \_\_\_\_ ” 20\_\_ р.  
викладач

Радченко Константин  
Олександрович

## **Мета роботи**

Метою роботи є здобуття вмінь програмування прикладних додатків баз даних SQLite.

## **Постановка завдання**

1. Реалізувати функції внесення, редагування та видалення даних у таблицях бази даних, створених у лабораторній роботі No1, 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/False, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

## Приклади результатів

### 1. Робота команди *INSERT*:

- з валідними даними:

```
Enter command: insert train
Enter directions: Kiyv - Rivne
Enter manufacturer: Man
Operation successful

Enter command: insert carriage
Enter type: cargo
Enter manufacturer: Renault
Operation successful

Enter command: insert seat
Enter type: odrinary
Enter carriageId: 4
Operation successful
```

записи в базі даних:

Таблиця: train			
	id	directions	manufacturer
	...	Фільтр	Фільтр
1	1	Odessa	ZAZ
2	2	Kiyv - Odessa	Man
3	3	Kiyv - Rivne	Man

Таблиця: carriage			
	id	type	manufacturer
	...	Фільтр	Фільтр
1	1	Cargo	Renault
2	4	passenger	Mersedes
3	5	passenger	Man

Таблиця: seat			
	id ▼ <sup>1</sup>	type	carriageId
	Фільтр	Фільтр	Фільтр
1	2	ordinary	4

- з неправильними даними:

```
alexandr@neophyte:~/University/Radchenko/Database1/Lab2$ dotnet run
Enter command: insert traineee
Error: unknown command
```

```
Enter command: insert seat
Enter type: odrinary
Enter carriageId: 1
Operation unsuccessful: this train is cargo type
```

*Робота команди DELETE\*:*

```
Enter command: delete train
Enter id: 2
Operation successful

Enter command: delete carriage
Enter id: 4
Operation successful
```

записи в базі даних:

Таблица: train

	id	directions	manufacturer
	...	Фильтр	Фильтр
1	1	Odessa	ZAZ
2	3	Kiyv - Rivne	Man
3			

Таблица Seat буде пустою, бо ми видалили вагон, в якому було записане наше місце:

Таблица: seat

	id ▼ <sup>1</sup>	type	carriageId
	Фильтр	Фильтр	Фильтр
1			

Таблиця: carriage

	id	type	manufacturer
	...	Фільтр	Фільтр
1	1	Cargo	Renault
2	5	passenger	Man
3			

Робота команди UPDATE\*:

```
Enter command: update train
Enter id: 3
Enter direction: New York
Operation successful
```

Запис в базі даних:

Таблиця: train

	id	directions	manufacturer
	...	Фільтр	Фільтр
1	1	Odessa	ZAZ
2	3	New York	Man
3			

Оновлення даних доступно для типу сидіння.

\*) **Примітка.** Валідація даних проходить по одному і тому ж принципу: перевірки чи id є цілим числом та чи існує воно в базі даних та перевірку чи рядок не є пустим.

2. На жаль, SQLite, на відмінну від PostgreSQL, не може виконувати генерації даних за допомогою лише SQL-запитів. Тому довелося створювати власноруч рандомайзер рядків для запису в базу даних. Дана функція дозволяє створити n(визначає користувач) записів з рядків по 5 літер.




Функція та приклад роботи для n=5:

```
1 reference
public long InsertRandom(int n)
{
    connection.Open();
    for (int i = 0; i < n; i++)
    {
        Random rnd = new Random();
        string alphabet = "QWERTYUIOPASDFGHJKLZXCVBNMqwertyuiopasdfghjklzxcvbnm";
        StringBuilder sb = new StringBuilder(5);
        int position = 0;
        for (int j = 0; j < 5; j++)
        {
            position = rnd.Next(0, alphabet.Length - 1);
            sb.Append(alphabet[position]);
        }

        SqlCommand command = connection.CreateCommand();
        command.CommandText =
            @"
                INSERT INTO train (directions, manufacturer)
                VALUES ($directions, $manufacturer);

                SELECT last_insert_rowid();
            ";
        command.Parameters.AddWithValue("$directions", sb.ToString());
        command.Parameters.AddWithValue("$manufacturer", sb.ToString());

        long newId = (long)command.ExecuteScalar();
    }
    connection.Close();
    return 0;
}
```

Таблица:  train   

	id	directions	manufacturer
	...	Фільтр	Фільтр
1	1	Odessa	ZAZ
2	3	New York	Man
3	4	icZPg	icZPg
4	5	LRnPG	LRnPG
5	6	lvljZ	lvljZ
6	7	uIFvT	uIFvT
7	8	UVMzB	UVMzB
8	9	PhBAA	PhBAA

3. Команда пошуку потягів від початкового id до кінцевого id:

```
Enter command: find trains
Enter front id: 2
Enter back id: 4
3 New York Man
4 icZPg icZPg
```

Використаний SQL-запит:

```
SELECT * FROM train WHERE id BETWEEN $frontId and $endId
```

Пошук вагонів за типами:

```
Enter command: find carriageTypes
Enter type: passenger

5 passenger Man
```

```
SELECT * FROM carriage WHERE type LIKE $type
```

Пошук місць за id вагону:

```
Enter command: find seatsByCarriage
Enter id: 5

Tip: can't find seats
```

```
SELECT * FROM seat WHERE carriageId = $carriageId
```

Значення, що записані шаблоном \$value, програмно підставляються користувачем. Такий запис запобігає потраплянню неправильних даних до бази даних та покращує захищенність бази даних.

#### 4. Розбиття коду:

BelitskyiAlexandr Update README.md		5c0c2a0 32 seconds ago	History
..			
Carriage.cs	Controller	lab2	3 minutes ago
CarriageRepository.cs	Model	lab2	3 minutes ago
Lab2.csproj		lab2	3 minutes ago
Program.cs	VIEW	lab2	3 minutes ago
README.md		Update README.md	32 seconds ago
Seat.cs	Controller	lab2	3 minutes ago
SeatRepository.cs	Model	lab2	3 minutes ago
Train.cs	Controller	lab2	3 minutes ago
TrainRepository.cs	Model	lab2	3 minutes ago
train	Database	lab2	3 minutes ago

В даному випадку я розбив код не на 3 файли (чітко модель-контролер-представлення), а враховуючі правила чистого коду для С#: кожен файл відповідає одній своїй глобальній функції. Наприклад, файл-репозиторії відповідають за роботу з "свою" конкретною таблицею.



## **Висновки**

Виконавши дану лабораторну роботу було вивчено CRUD операції, їх правильна реалізація через SQL запити без втручання мови програмування до бази даних. При виконанні операцій забезпечувались дії, що підтримували базу даних у третій нормальній формі.

Також було проведено створення консольного додатку для керування базою даних. Оброблення виключних ситуацій та валідація даних є основою цього додатку.

Робота була виконана в операційній системі Linux дистрибутив Ubuntu за допомогою програм Visual Studio Code та DB Browser for SQLite.