

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		Перевірив	
Студент I курсу	·	" 20 p.	
групи КП-01		викладач	
Беліцький Олександр Сергійович	Рад	ченко Констянтин Олександрович	

Мета роботи

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

Постановка завдання

- 1. Реалізувати функції внесення, редагування та вилучення 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-поданняконтролер).

Приклади результатів

1. Робота команди INSERT:

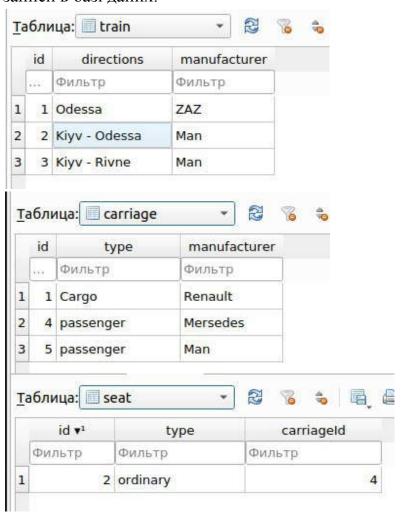
• з валідними даними:

```
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:



• з неправильними даними:

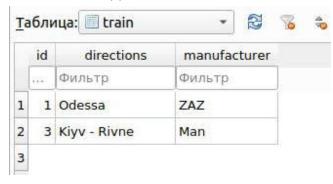
alexandr@neophyte:~/University/Radchenko/Database1/Lab2\$ dotnet run
Enter command: insert traineee
Error: unknown command

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

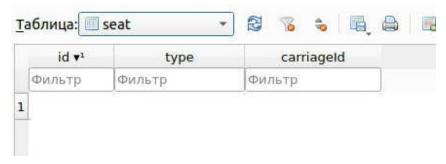
Робота команди DELETE*:

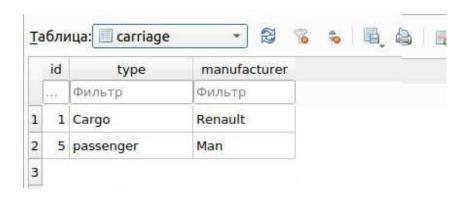
Enter command: delete train
Enter id: 2
Operation successful
Enter command: delete carriage
Enter id: 4
Operation successful

записи в базі даних:

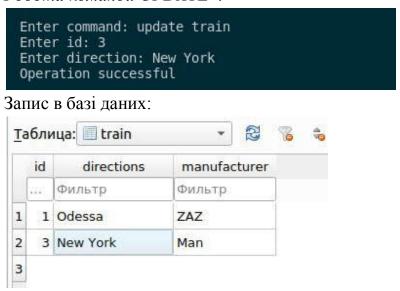


Таблиця Seat буде пустою, бо ми видалили вагон, в якому було записане наше місце:





Робота команди UPDATE*:



Оновлення даних доступно для типу сидіння.

- *) *Примітка*. Валідація даних проходить по одному і тому ж принципу: перевірки чи іd ϵ цілим числом та чи існу ϵ воно в базі даних та перевірку чи рядок не ϵ 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 п(визначає користувач) записів з рядків по 5 літер.

Функція та приклад роботи для n=5:

```
Ireference
public long InsertRandom(int n)

connection.Open();
for (int i = 0; i < n; i++)
{
    Random rnd = new Random();
    String alphabet = "OWERTYUIOPASDFGHJKLZXCVBNMqwertyuiopasdfghjklzxcvbnm";
    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]);
    }

    SqliteCommand 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;
</pre>
```

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

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

```
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 ід 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
п		
Carriage.cs Controller	lab2	3 minutes ago
CarriageRepository.cs <i>Model</i>	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 <i>Model</i>	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-контролерпредставлення), а враховуючі правила чистого коду для С#: кожен файл відповідає одній своїй глобальній функції. Наприклад, файл-репозиторії відповідають за роботу з "своєю" 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 ϵ основою цього додатку.

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