Министерство цифрового развития, связи и
массовых коммуникаций Российской Федерации

Сибирский государственный университет телекоммуникаций и информатики

Лабораторно-практическая работа №2

по дисциплине: «Программирование и обработка графического интерфейса»

«Работа с базами данных»

Выполнил: Сливинский Р.О

Группа: ДИМ-35

Вариант: 4

Проверил: Голованчиков С.А.

Новосибирск, 2025

Цель работы:

- получение навыков работы с базами данных на языке C#;
- знакомство с принципами построения SQL-запросов.

Задание:

Разработать WPF-приложение с графическим интерфейсом и реализовать следующие функции:
1) ввод данных о студентах: уникальный номер, ФИО, оценка по физике, оценка по математике;
2) добавление данных в базу данных SQLite (далее - БД) через интерфейс приложения;
3) чтение данных из БД и отображение их в окне приложения;
4) редактирование данных в БД через интерфейс приложения;
5) удаление данных из таблиц.

БД должна содержать две таблицы, связанные через уникальный номер:

1. таблица, содержащая уникальный номер и ФИО;
2. таблица, содержащая уникальный номер и оценки.

Дополнительное задание: реализовать хранение и редактирование даты рождения студента.

Описание разработанной программы:

Программа представляет собой WPF-приложение для управления базой данных студентов с использованием SQLite.

Основные функции:

- Добавление, редактирование, удаление студентов.

- Хранение данных: ФИО, дата рождения, оценки по физике и математике.

- Просмотр списка студентов в табличном виде.

1. Структура программы

1.1. База данных (SQLite)

Используется две таблицы, связанные по StudentId:

Таблица Students	Таблица Grades
Id (PK)	Id(PK)
StudentId (уникальный)	StudentId (FK > Students)
FullName (текст)	PhysicsGrade (целое)
BirthDate (текст, Null)	MathGrade (целое)

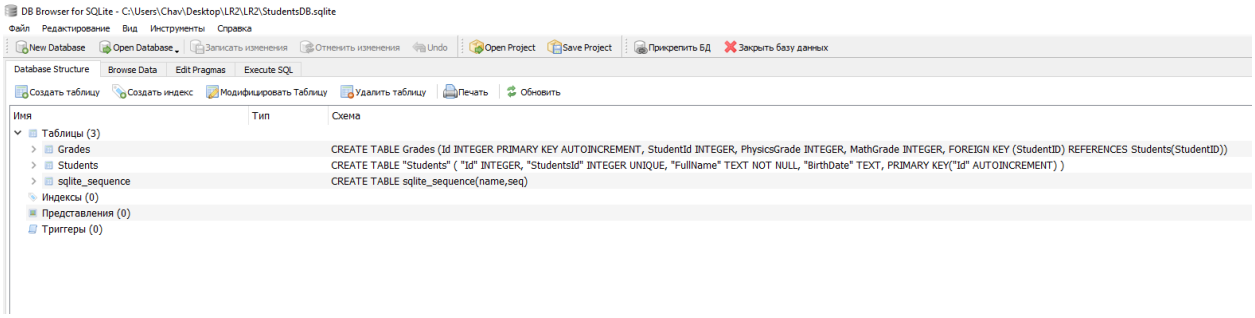

Рис 1. Скриншот структуры БД в DB Browser for SQLite

2. Алгоритмы работы

2.1 Добавление студента (AddStudent)

Описание:

- Открывается окно для ввода данных.
- Данные сохраняются в БД в двух таблицах (Students и Grades) в рамках одной транзакции.

```
public void AddStudent(int studentId, string fullName, DateTime? birthDate, int physicsGrade, int mathGrade)
{
    using (var connection = new SQLiteConnection(_connectionString))
    {
        connection.Open();
        using (var transaction = connection.BeginTransaction())
```

```
        {
            try
            {
                // 1. Добавляем в таблицу Students
                string insertStudent = "INSERT INTO Students (...) VALUES (...)";
                // 2. Добавляем в таблицу Grades
                string insertGrades = "INSERT INTO Grades (...) VALUES (...)";
                transaction.Commit(); // Подтверждаем изменения
            }
            catch
            {
                transaction.Rollback(); // Откат при ошибке
                throw;
            }
        }
    }
}
```

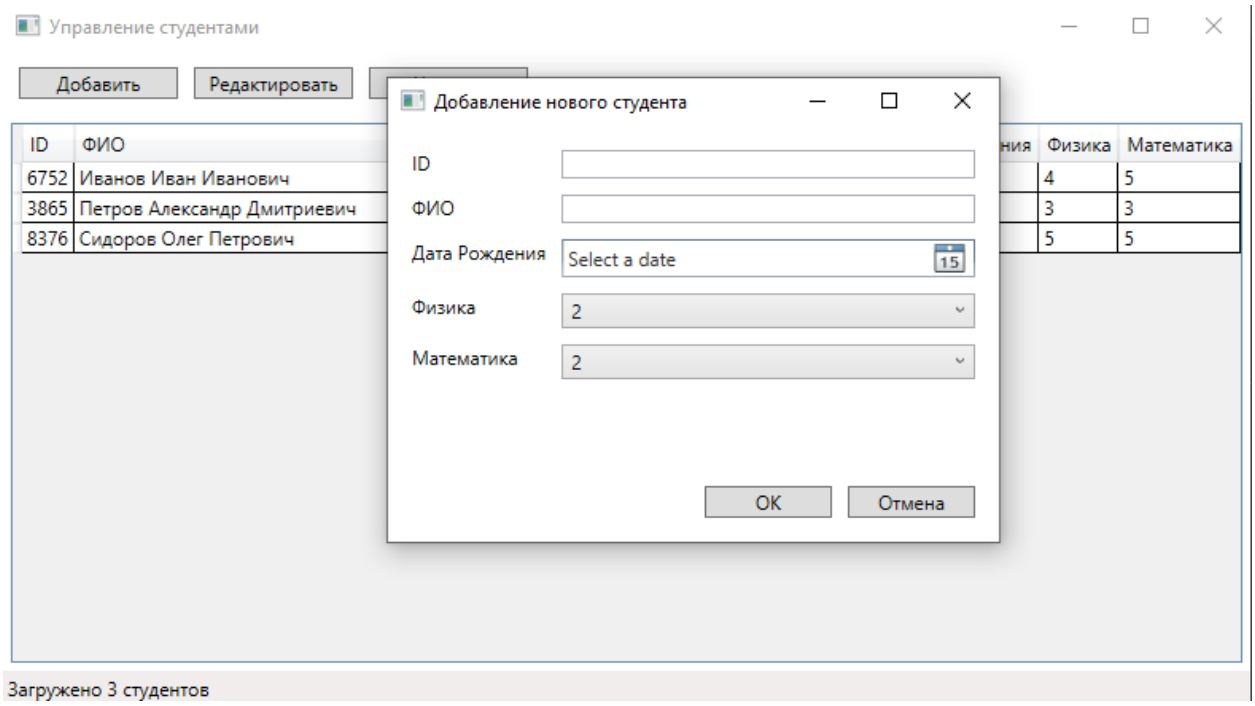

Рис 2. Окно для ввода данных

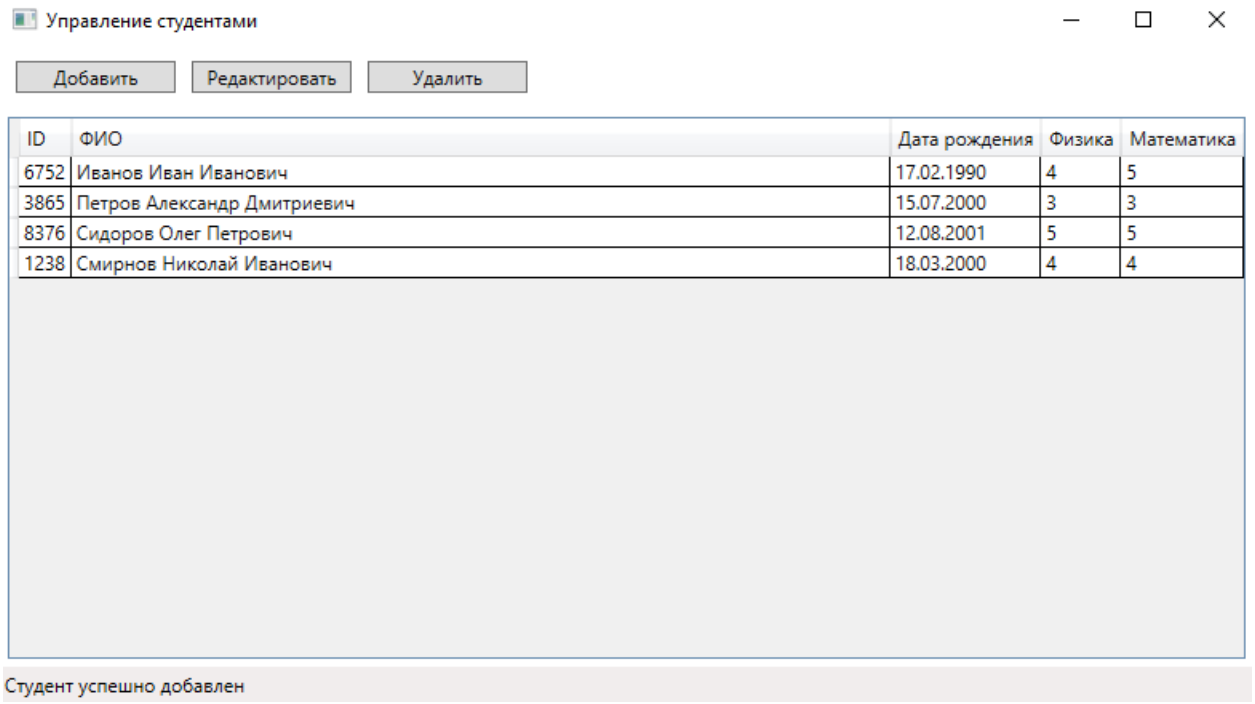

Рис3. Данные студента сохранились в двух таблицах

2.2 Чтение данных (GetAllStudents)

Описание:

- SQL-запрос с JOIN объединяет данные из двух таблиц.
- Результаты преобразуются в список объектов Student.

```
public List<Student> GetAllStudents()
{
    var students = new List<Student>();
    string query = @"
        SELECT s.StudentId, s.FullName, s.BirthDate, g.PhysicsGrade, g.MathGrade
        FROM Students s
        JOIN Grades g ON s.StudentId = g.StudentId";

    // ... выполнение запроса и заполнение списка
    return students;
}
```

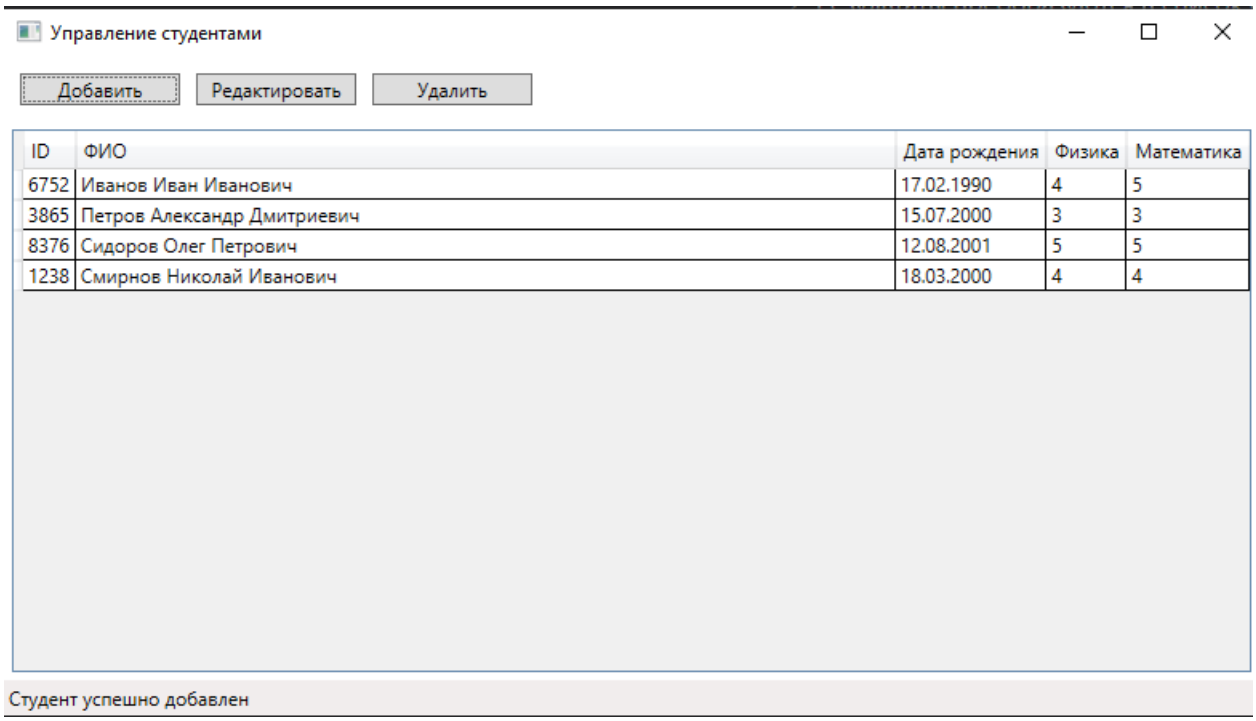

Рис 4. Чтение данных.

2.3 Удаление студента (DeleteStudent)

Описание:

- Сначала удаляются оценки из Grades (из-за FOREIGN KEY).
- Затем удаляется запись из Students.

```
public void DeleteStudent(int studentId)
{
    // 1. Удаляем из Grades
    string deleteGrades = "DELETE FROM Grades WHERE StudentId =
@studentId";
    // 2. Удаляем из Students
    string deleteStudent = "DELETE FROM Students WHERE StudentId =
@studentId";
}
```

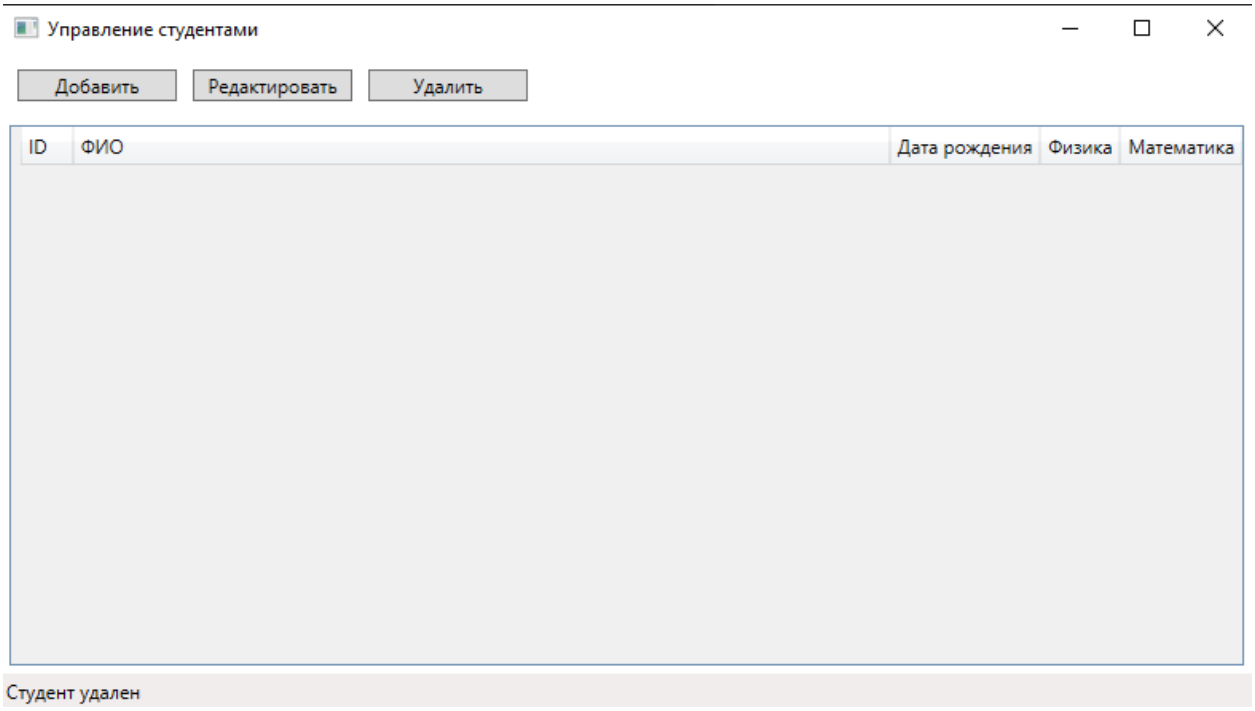

Рис 5. Скриншот после удаления всех студентов

3. Вывод

Разработанная программа представляет собой WPF-приложение для управления базой данных студентов, реализующее основные CRUD-операции (добавление, чтение, обновление и удаление записей) с использованием SQLite. Приложение хранит информацию о студентах (ФИО, дата рождения) и их оценках по физике и математике в двух связанных таблицах, обеспечивая целостность данных через транзакции и внешние ключи. Интерфейс включает главное окно с табличным представлением данных и диалоговые окна для редактирования.

Программа демонстрирует:
- Работу с SQLite в C# (CRUD-операции).
- Использование транзакций для целостности данных.
- Связь между таблицами через FOREIGN KEY.