Лабораторная работа №4:
Доступ к базе данных

Цель:

Целью данной работы является получение навыков работы с базами данных на языке высокого уровня C# в среде программирования Microsoft Visual Studio 2022Community

Примечание:

В рамках лабораторной работы, будет использован data base engine SQLite: https://www.sqlite.org/index.html

Создание и просмотр базы данных лучше всего осуществлять при помощи стороннего ПО. Например, DB Browser for SQLite: http://sqlitebrowser.org/

Добавить поддержку баз данных SQLite в Microsoft Visual Studio можно установив соответственный nu-get package: https://www.nuget.org/packages/System.Data.SQLite

Задание:

Разработать программное приложение WPF, имеющее графический интерфейс и реализующее следующие функции:

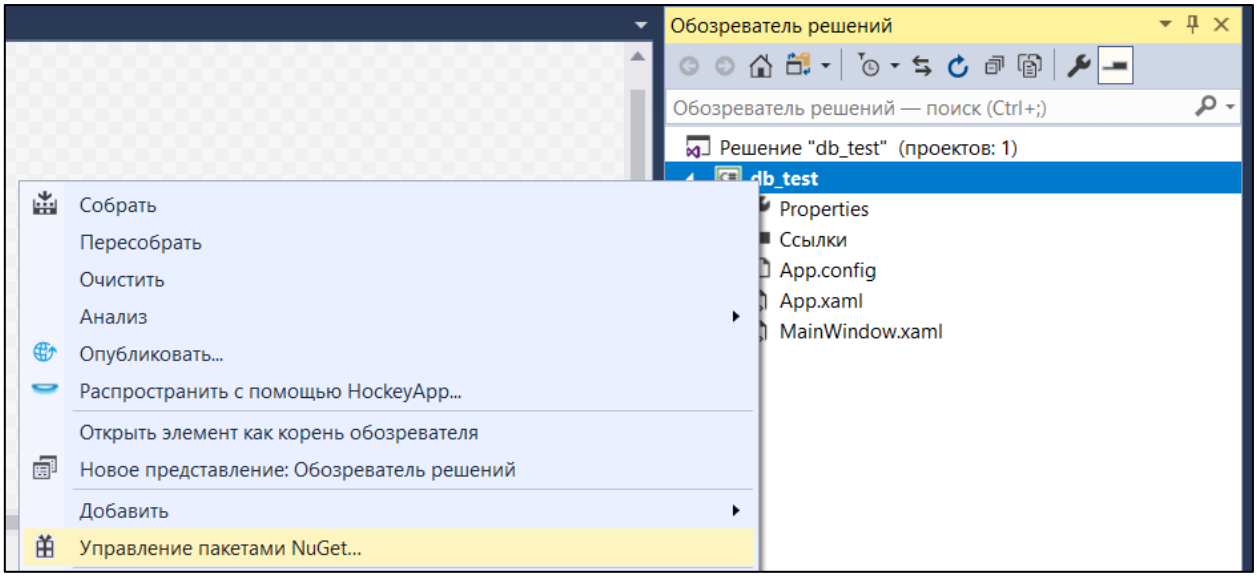
1) Ввод данных о студентах: уникальный номер, фио, оценка по физике, оценка по математике.
2) Добавление данных в таблицу базы данных SQLite через интерфейс приложения.
3) Чтение данных из базы данных SQLite и отображение их на форме приложения.
4) Редактирование данных в базе данных SQLite через интерфейс приложения.
5) Удаление данных из таблиц

База данных должна содержать 2 таблицы, связанные через уникальный номер:

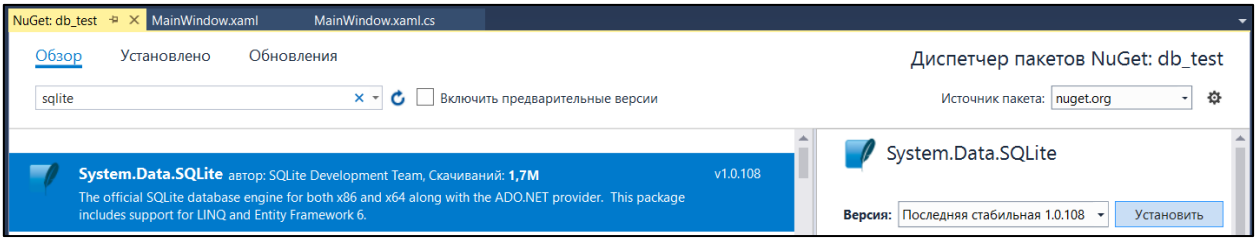
1. Таблица, содержащая уникальный номер и фио.
2. Таблица, содержащая уникальный номер и оценки.

Справочная информация:

Для того, чтобы получить возможность подключения к базам, необходимо добавить поддержку SQLite в Microsoft Visual Studio. Сделать это можно нажав правой кнопкой мыши на название проекта, после чего выбрав “Управление пакетами NuGet…”:

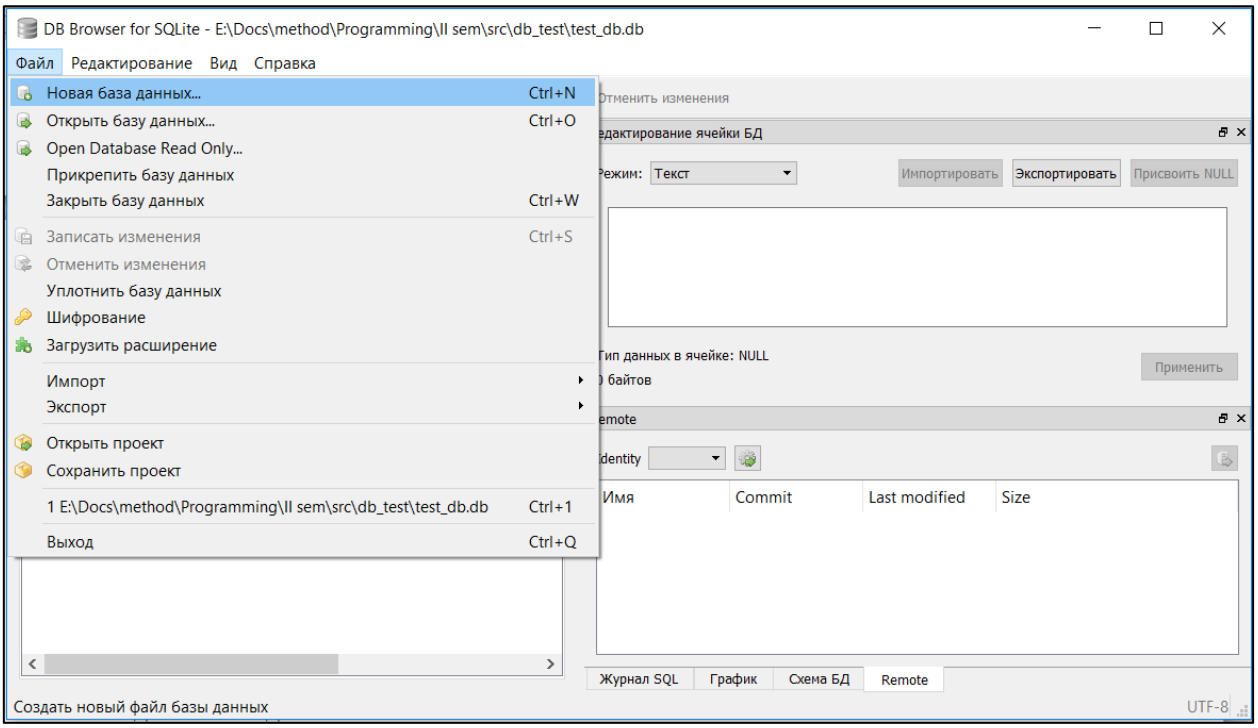


Перейдите в раздел “Обзор”, в строке поиска введите “sqlite”, после чего выберите System.Data.SQLite и нажмите “Установить”:

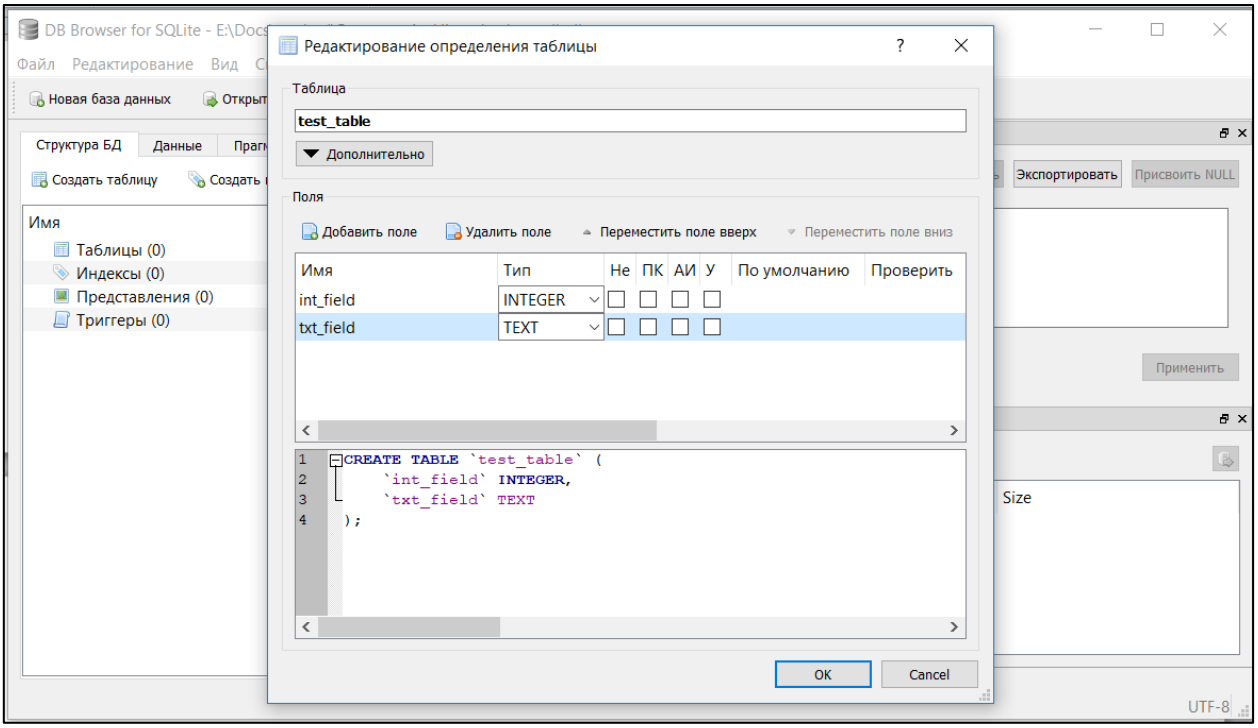


Создание таблиц при помощи DB Browser for SQLite:

После запуска DB Browser for SQLite, выберите “Файл” -> ”Новая база данных”:



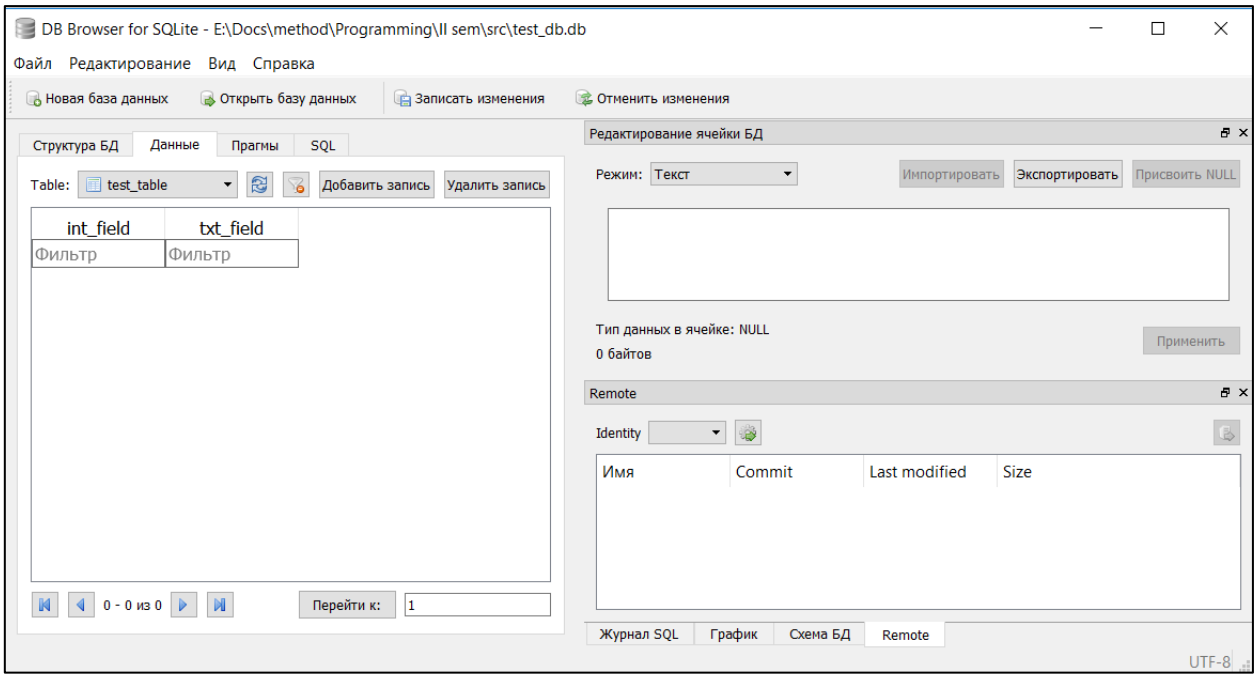
После создания базы данных, будет предложено создать первую таблицу. Сделать это можно указав имя таблицы, и добавив поля:



Поля могут иметь один из 5 типов, а так же дополнительные свойства:

- Первичный ключ – поле используется для связи таблиц.
- Авто инкремент – значение увеличивается автоматически.
- Уникальное – значение должно быть уникальным.

Просмотреть или отредактировать таблицу, можно на вкладке “Данные”:



Некоторые функции для работы с базами данных SQLite:

Подключение библиотеки:

```
using System.Data.SQLite;
```

Создание базы данных:

```
SQLiteConnection.CreateFile("D:\\MyDocs\\method\\intsys\\MyDatabase.sqlite");
```

Добавление таблицы:

```
string sql = "CREATE TABLE test (name VARCHAR(20), score INT)";
SQLiteCommand command = new SQLiteCommand(sql, m_dbConnection);
command.ExecuteNonQuery();
```

Подключение к уже существующей базе данных:

```
//имя базы данных
string db_name = dlg.FileName;

SQLiteConnection m_dbConnection;
m_dbConnection = new SQLiteConnection("Data Source=" + db_name + ";Version=3;");
//открытие соединения с базой данных
m_dbConnection.Open();

//выполнение запросов

//закрытие соединения с базой данных
m_dbConnection.Close();
```

Запись данных в уже существующую таблицу:

```
//формирование запроса на добавление данных в поля типа INTEGER и TEXT
//обратите внимание, что в текстовое поле, данные добавляются в формате ‘data’
string sql = "INSERT INTO test_table (int_field, txt_field) VALUES (" + tb1.Text + ",'" + tb2.Text +
"')";
SQLiteCommand command = new SQLiteCommand(sql, m_dbConnection);
//извлечение запроса
command.ExecuteNonQuery();
```

Чтение данных из существующей таблицы:

```
string sql = "SELECT * FROM test_table ORDER BY int_field";
SQLiteCommand command = new SQLiteCommand(sql, m_dbConnection);
SQLiteDataReader reader = command.ExecuteReader();
while (reader.Read())
    list.Items.Add(reader["int_field"] + " : " + reader["txt_field"]);
```

Для того, что бы получить номер последней добавленной записи, можно выполнить SELECT запрос следующего вида:

```
string sql = "SELECT int_field FROM test_table ORDER BY int_field DESC LIMIT 1";
```

Язык запросов SQL:

Для выполнения лабораторной работы, вам понадобятся всего четыре типа запросов: INSERT, UPDATE, SELECT и DELETE.

Шаблон запроса **INSERT** выглядит следующим образом:

```
INSERT INTO название_таблицы (поле_1 [, поле_2, поле_3 ... ]) VALUES (значение_1 [, значение_2,
значение_3 ... ])
```

Шаблон запроса **UPDATE** выглядит следующим образом:

```
UPDATE название_таблицы SET название_поля = значение [,название_поля = значение ...] [WHERE условие]
```

Пример:

```
UPDATE test_table SET txt_field = ‘ten’ WHERE int_field = 10
```

Шаблон запроса **SELECT** выглядит следующим образом:

```
SELECT
определение, какие именно поля требуется выбрать (* - выбрать все поля)
FROM имя_таблицы
[WHERE условие_выбора]
[GROUP BY условие_группирования]
[HAVING условие_наличия]
[ORDER BY условие_сортировки]
```

Пример запроса по двум таблицам:

```
SELECT * FROM table_1, table_2 WHERE table_1.id = table_2.user_id
```

Шаблон запроса **DELETE** выглядит следующим образом:

```
DELETE FROM имя_таблицы [WHERE условие];
```

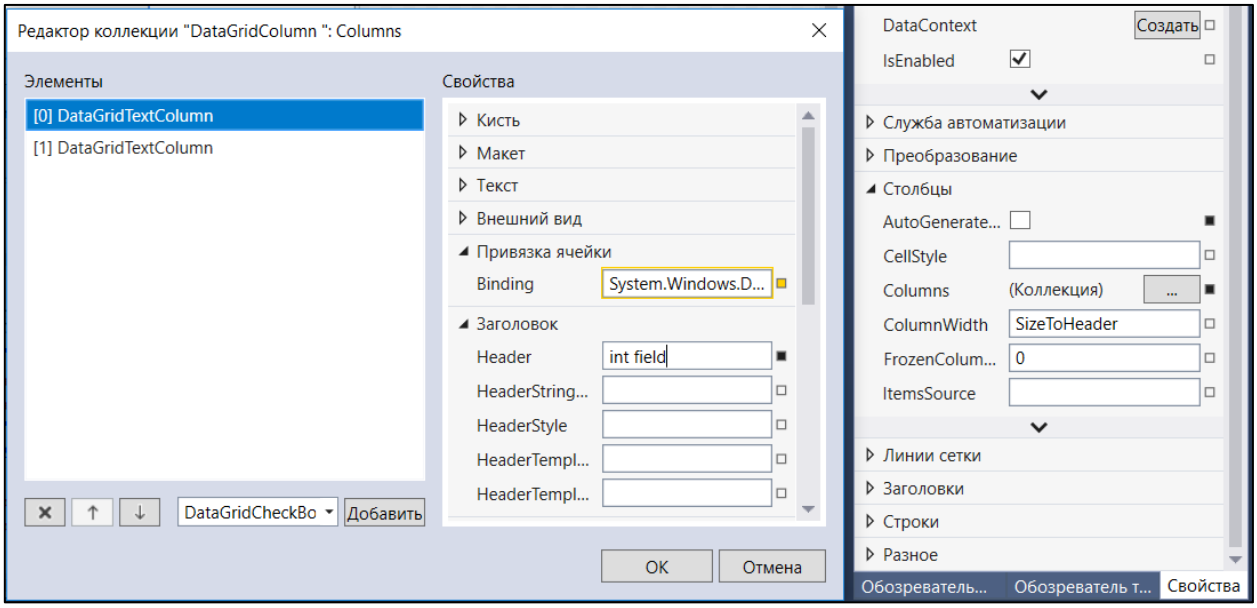
Пример:

```
DELETE FROM test_table WHERE int_field = 10;
```

Компонент DataGrid.

Для того, что бы отобразить содержимое базы данных в виде таблицы, можно использовать компонент DataGrid.

Добавить столбцы можно, следующим образом:



1. Перейдите в раздел “Столбцы”.
2. Нажмите Коллекция.
3. Выберите компонент “DataGridTextColumn”.
4. Укажите имя столбца в разделе “Заголовок”
5. Свяжите столбец с данными в разделе “Привязка ячейки”, в данном случае, в поле “Binding”, нужно прописать `{Binding int_field}`

Следующий шаг, создание структуры данных, описывающих строку таблицы:

```
public class CTest
{
    public int int_field { get; set; }
    public string txt_field { get; set; }
}
```

Имена полей структуры, должны совпадать с теми, что были указаны в “Binding”.

Для того, что бы добавить в DataGrid строку, считанную из таблицы можно использовать следующий код:

```
//создание строки
var data = new CTest { int_field = int.Parse(reader["int_field"].ToString()), txt_field =
reader["txt_field"].ToString() };
//добавление строки в DataGrid
datgrid.Items.Add(data);
```

Для того чтобы получить данные выбранные в DataGrid, можно использовать:

```
private void datgrid_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
    //получение строки из DataGrid
    CTest test = (CTest)datgrid.SelectedItem;

    MessageBox.Show(test.int_field.ToString() + " " + test.txt_field);
}
```

Список литературы:

Компонент DataGrid:

https://msdn.microsoft.com/ru-ru/library/system.windows.controls.datagrid(v=vs.110).aspx

Справочник с примерами по языку SQL: http://sql.itsoft.ru/

Основы проектирования реляционных баз данных:

http://citforum.ru/database/dbguide/index.shtml