Базы данных

SQLite

Почти каждая программа, которая работает с данными, требует места для их хранения. В качестве такого хранилища могут выступать обычные текстовые файлы, файлы JSON или XML. Однако для хранения большого количества данных и удобной организации работы с ними существуют базы данных и СУБД.

База данных

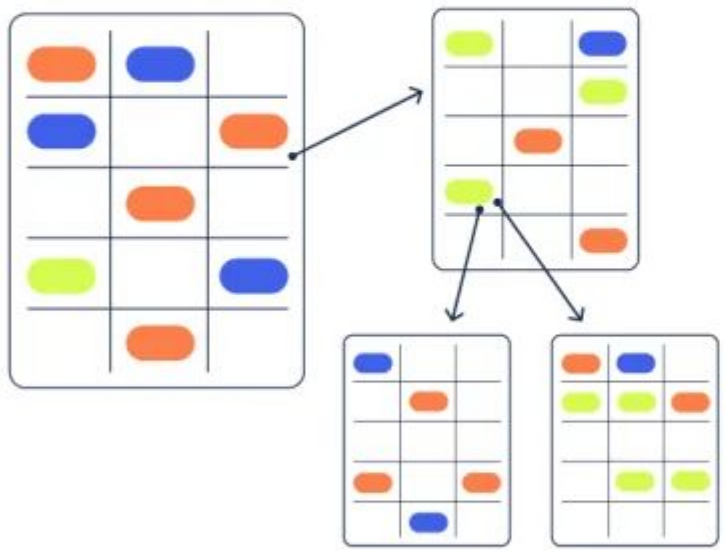
В узком смысле слова, база данных — это некоторый набор данных, необходимых для работы.

Однако данные — это абстракция; никто никогда не видел "просто данные"; они не возникают и не существуют сами по себе. Данные суть отражение объектов реального мира.

В широком смысле слова база данных — это совокупность описаний объектов реального мира и связей между ними, актуальных для конкретной прикладной области.

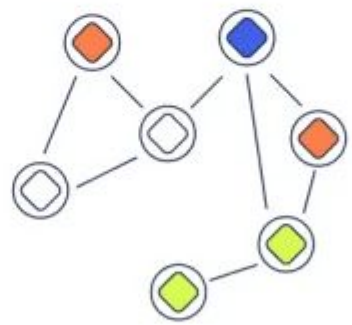
Базы данных

Реляционные

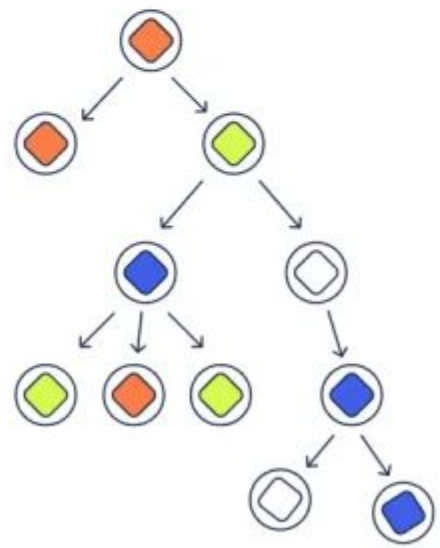


Нереляционные

Сетевые



Иерархические



Реляционные

Реляционная база данных – это набор данных с предопределенными связями между ними. Эти данные организованны в виде набора таблиц, состоящих из столбцов и строк.

В таблицах хранится информация об объектах, представленных в базе данных.

В каждом столбце таблицы хранится определенный тип данных, в каждой ячейке – значение атрибута.

Каждая стока таблицы представляет собой набор связанных значений, относящихся к одному объекту или сущности.

Взаимосвязь таблиц базы данных

В одной таблице содержатся ассоциированные данные, а в разных таблицах одной БД находятся связанные данные.

Сетевые

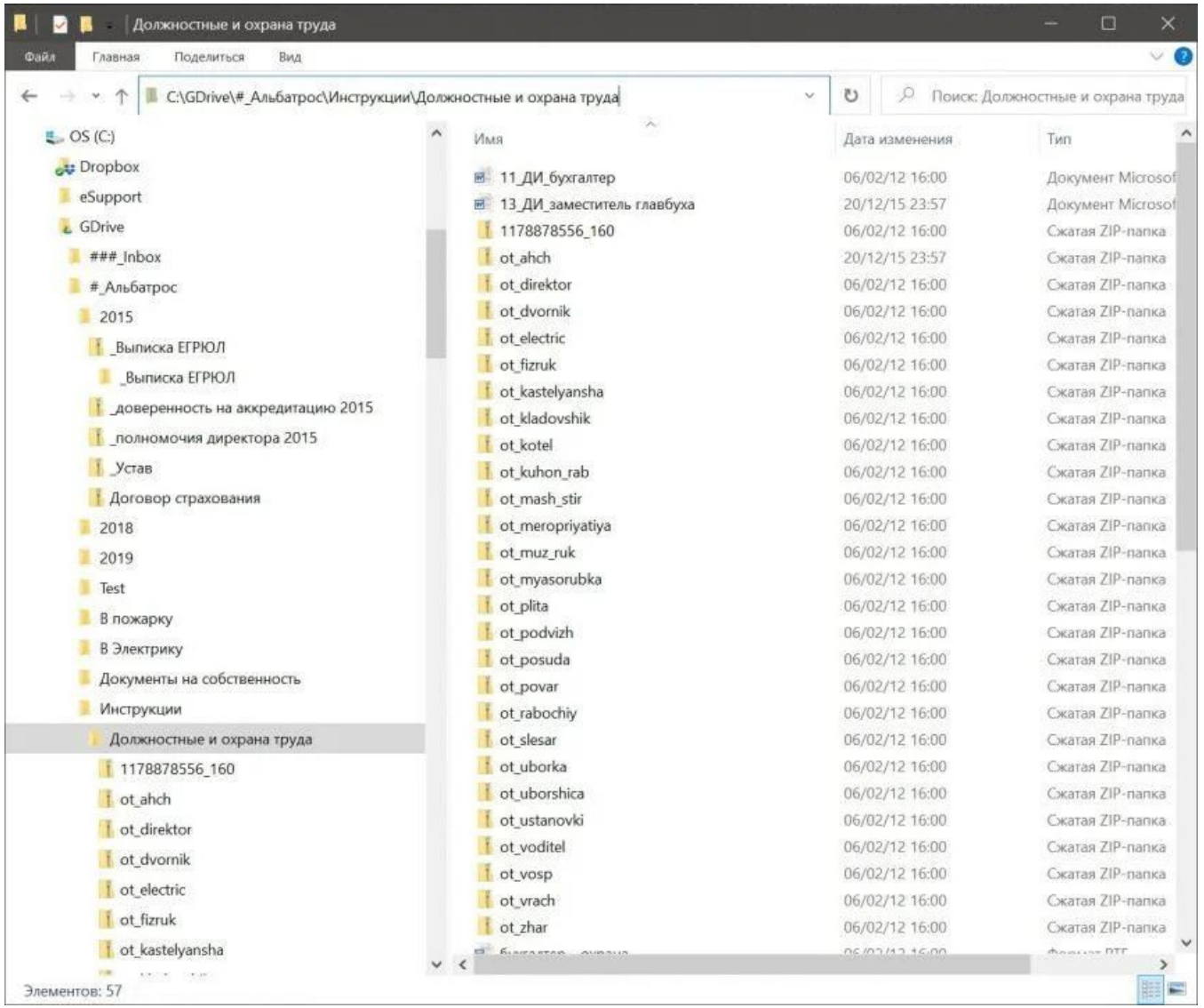
В сетевых базах данных между таблицами и записями может быть несколько разных связей, каждая из который отвечает за что-то своё.

Особенность сетевой базы данных в том, что в ней запоминаются все связи и всё содержимое для каждой связи. Базе не нужно тратить время на поиск нужных данных, потому что вся информация об этом уже есть в специальных индексных файлах. Они показывают, какая запись с какой связана, и быстро выдают результат.

Сетевые базы данных: позволяют хранить много связей между множеством объектов. Например, каталог фильмов: в одном фильме может участвовать много человек, а каждый из них может участвовать во множестве фильмов.

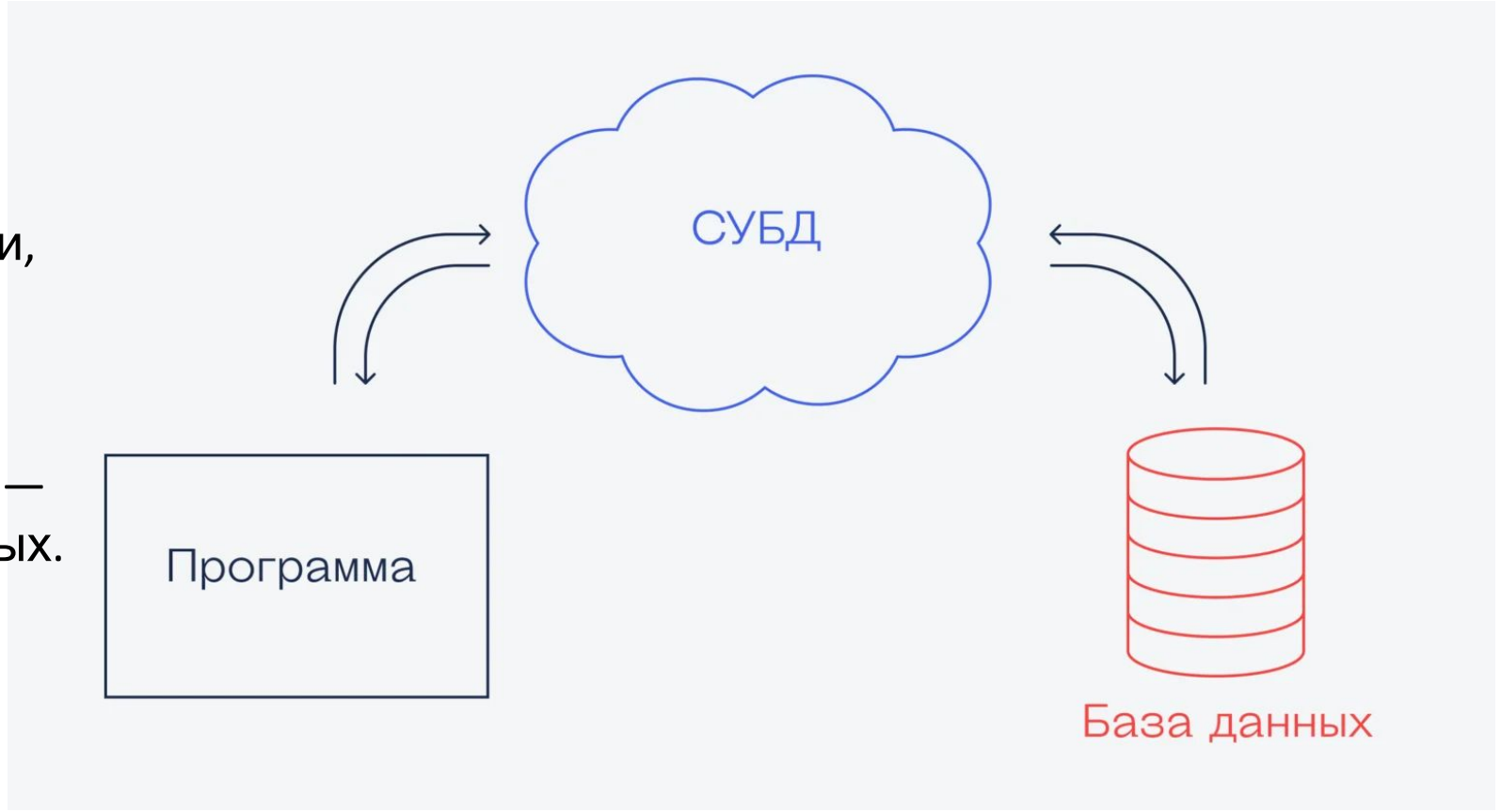
Иерархические

Иерархическая база данных знает, кто кому подчиняется, и поэтому может быстро находить нужную информацию. Но такие базы можно организовать только в том случае, когда у вас есть чёткое разделение в данных, что главнее, а что ему подчиняется.



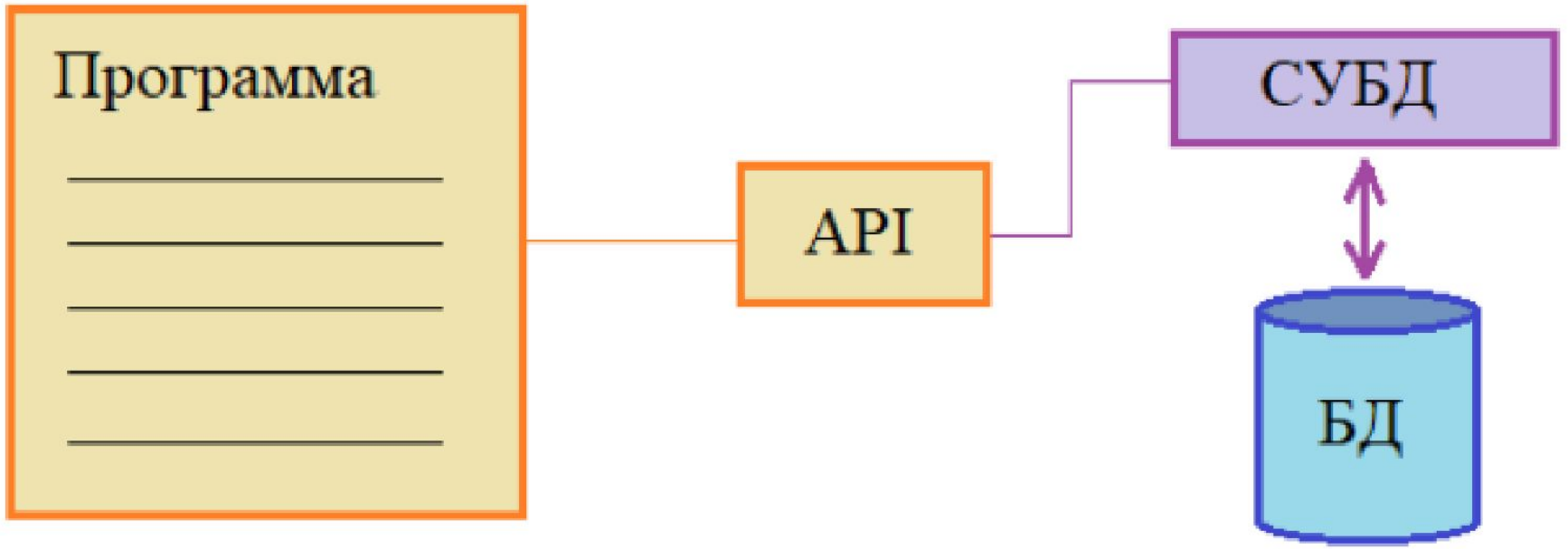
Язык SQL

Чтобы управлять данными в базе, например добавлять новые записи, удалять старые или что-то искать, используют специальный язык запросов к базе — SQL.
Эти запросы обрабатывает СУБД — система управления базами данных. Это как движок для сайтов — он выполняет запросы, работает с базой и отдаёт нам результаты.



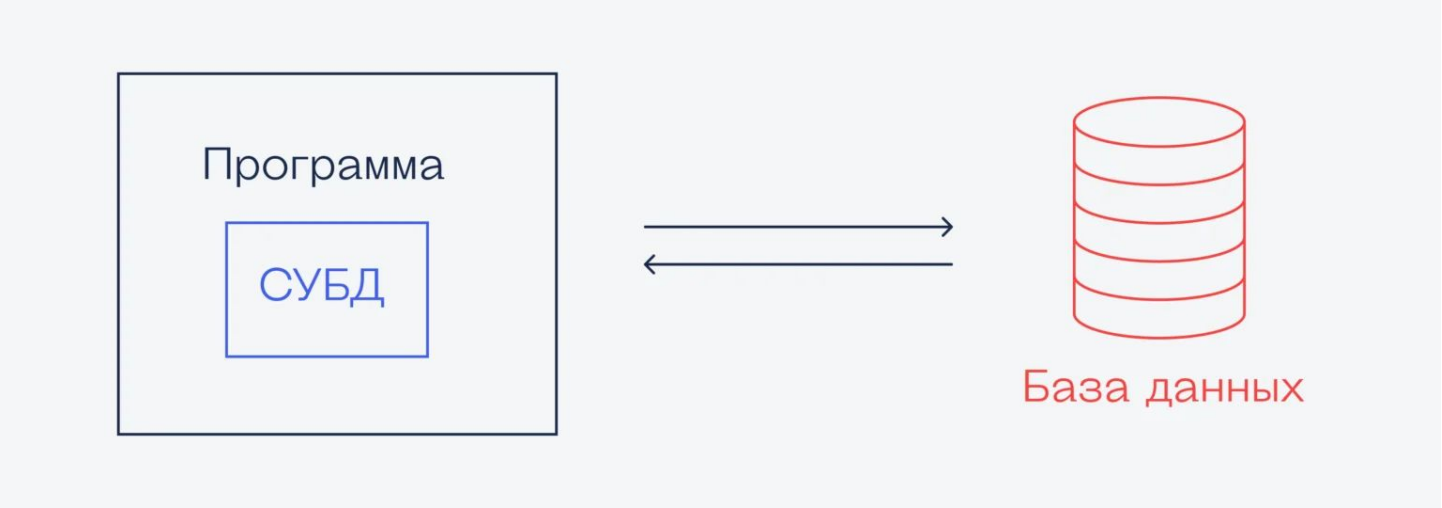
SQL —это язык запросов к базе данных. Название расшифровывается: «язык структурированных запросов» (Structured Query Language). Это значит, что каждый запрос к базе данных формируется по какой-то структуре, а сам язык задаёт правила, как именно сделать такой запрос.

После выбора определенной СУБД программист получает доступ к ее API (Application Programming Interface) – программному интерфейсу для взаимодействия с СУБД. Фактически, к набору функций, через которые производится работа с базами данных



Что такое SQLite

SQLite — это встраиваемая СУБД, когда система управления встраивается в саму программу. Это значит, что все запросы и команды идут в базу не через посредника, а напрямую из приложения. Чтобы встроить SQLite в код, достаточно подключить нужную библиотеку.



Все данные в SQLite хранятся в одном файле — таблицы, служебные поля, связи и всё остальное. Это упрощает работу с базой и позволяет легко переносить данные из одного места в другое.

Подключение к SQLite в Python

Для установки соединения нужно указать название базы данных, к которой требуется подключиться. Если указать название той, что уже есть на диске, то произойдет подключение. Если же указать другое, то SQLite создаст новую базу данных.

Для подключения к SQLite нужно выполнить следующие шаги

- import sqlite3
- Использовать метод connect() из модуля sqlite3 и передать в качестве аргумента название базы данных – открываем соединение.
- Создать объект cursor для выполнения SQLite-запросов из Python. Вызов метода execute(query) для объекта Cursor выполняет запрос и в случае наличия результата возвращает ответ, который можно получить методом fetchall().
- Закрыть объект cursor после завершения работы.
- Перехватить исключение базы данных, если в процессе подключения произошла ошибка

```
import sqlite3 as sq
con = sq.connect("data.db")
cur = con.cursor()
cur.execute("""
""")
con.close()
```

```
with sq.connect("saper.db") as con:
 cur = con.cursor()
 cur.execute("""
 """)
```

Типы данных

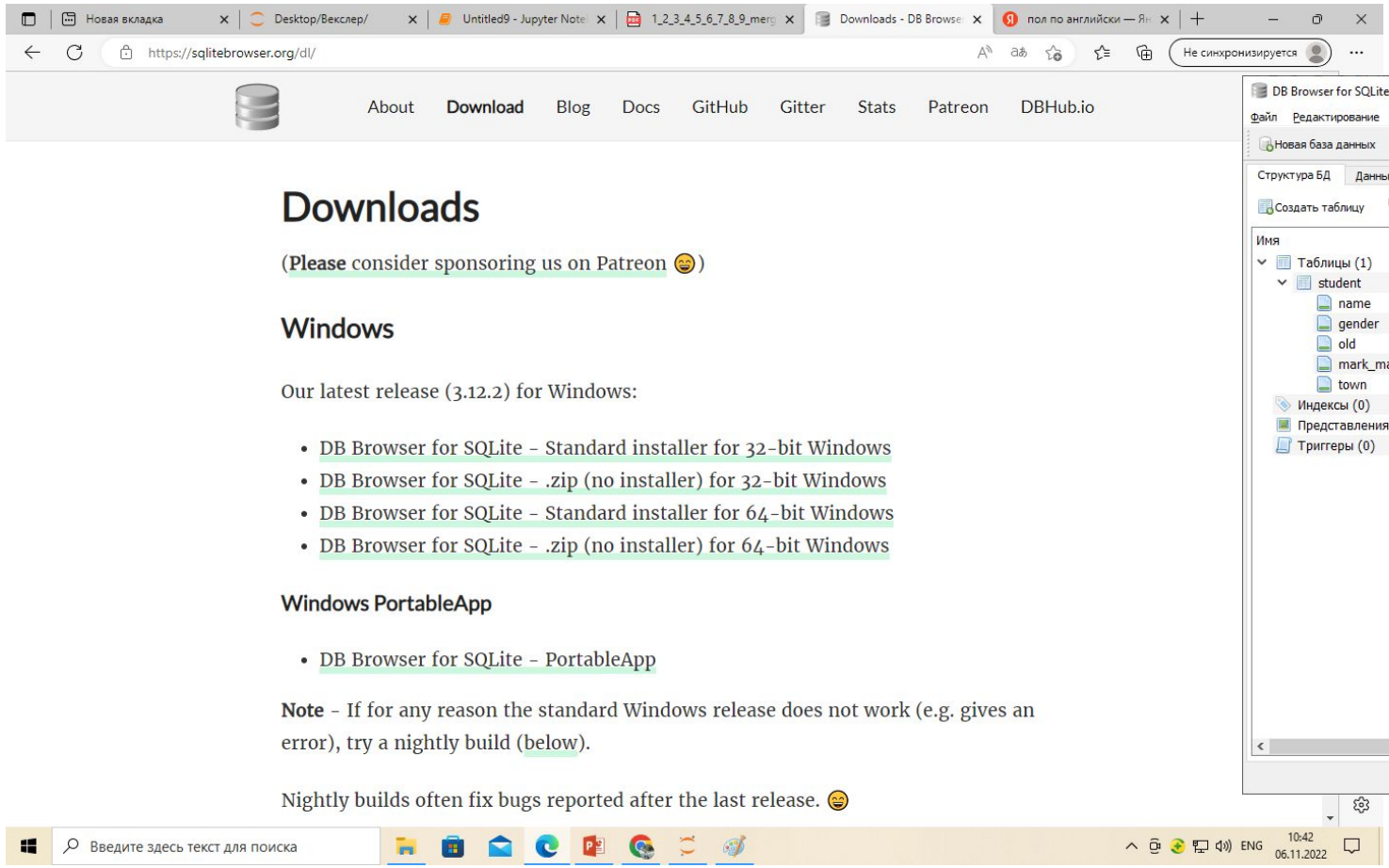
NULL – значение NULL;
INTEGER – целочисленный тип (занимает от 1 до 8 байт);
REAL – вещественный тип (8 байт в формате IEEE);
TEXT – строковый тип (в кодировке данных базы, обычно UTF-8);
BLOB (двоичные данные, хранятся «как есть», например, для небольших изображений).

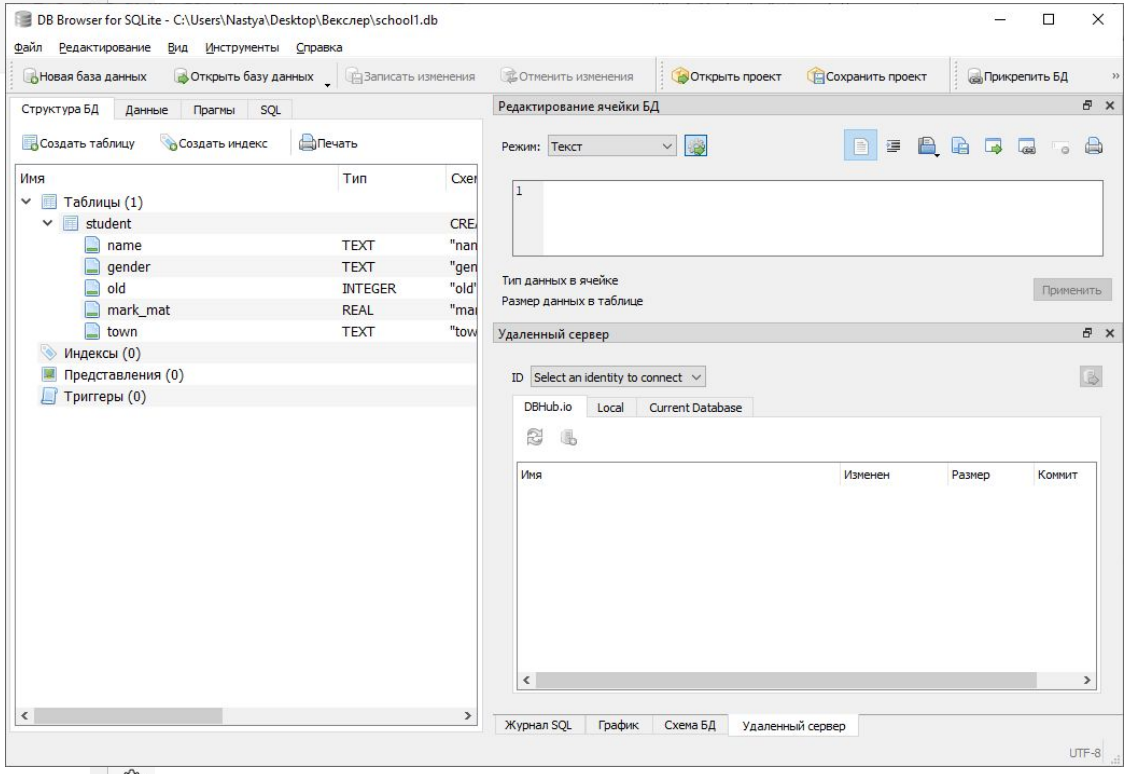
CREATE TABLE (создать таблицу)

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""CREATE TABLE student ( name TEXT,
    gender TEXT, old INTEGER, mark_mat REAL, town TEXT )""")
```

DB Browser for SQLite

https://sqlitebrowser.org/





Повторный запуск –
ошибка (таблица есть)

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""CREATE TABLE student ( name TEXT,
    gender TEXT, old INTEGER, mark_mat REAL, town TEXT )""")
```

```
---------------------------------------------------------------------------
OperationalError                          Traceback (most recent call last)
Input In [9], in <cell line: 1>()
      1 with sq.connect("school1.db") as con:
      2     cur = con.cursor()
----> 3     cur.execute("""CREATE TABLE student ( name TEXT,
      4     gender TEXT, old INTEGER, mark_mat REAL, town TEXT )""")

OperationalError: table student already exists
```

Решение

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""CREATE TABLE IF NOT EXISTS student ( name TEXT,
    gender TEXT, old INTEGER, mark_mat REAL, town TEXT )""")
```

Создавать таблицу только если она не существует. Теперь, запуская программу, никаких ошибок появляться не будет и, кроме того, мы точно будем уверены, что таблица student присутствует в нашей БД.

INSERT – добавление записи в таблицу

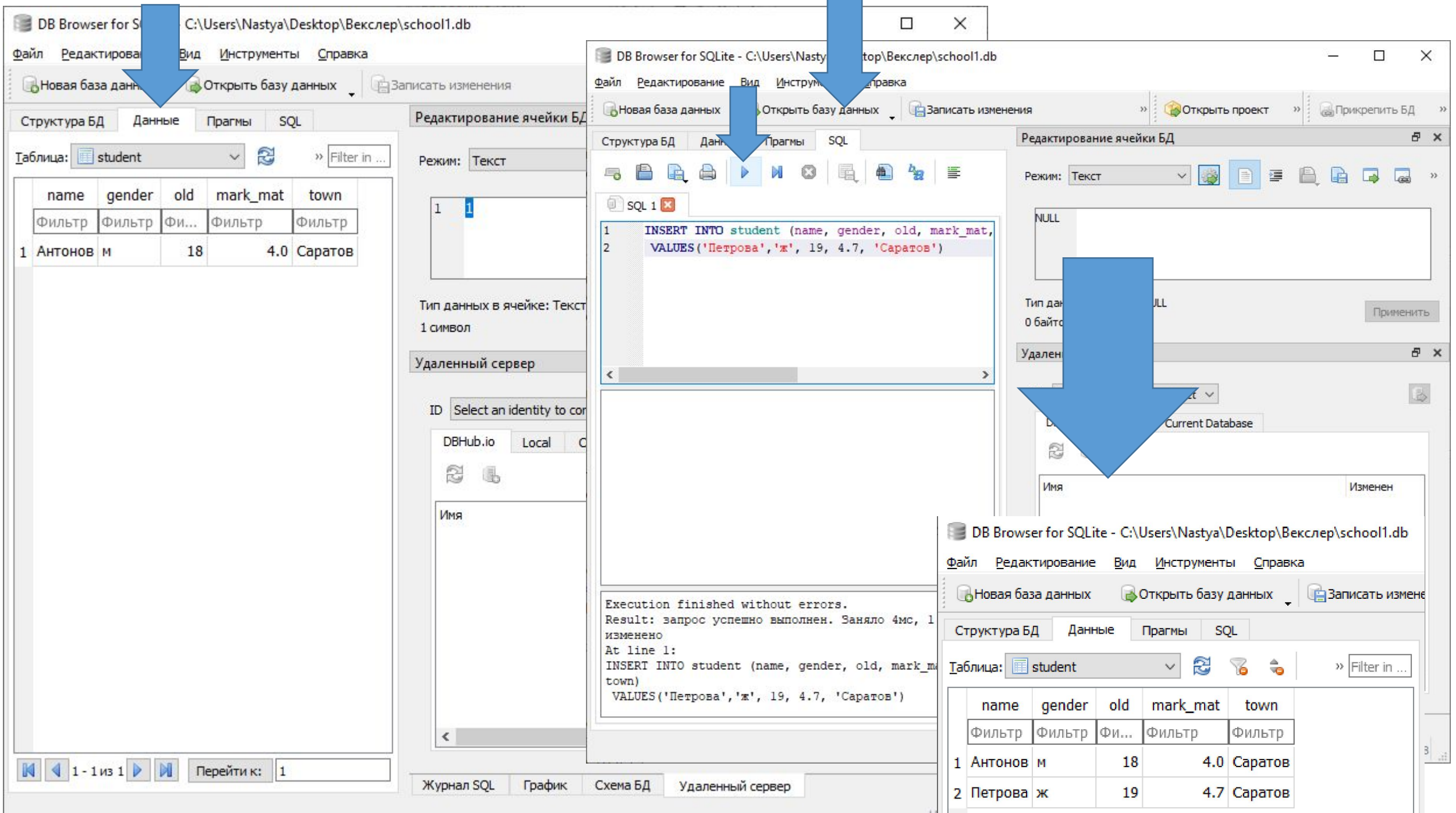
INSERT INTO <table_name> (<column_name1>, <column_name2>, ...)
VALUES (<value1>, <value2>, …)

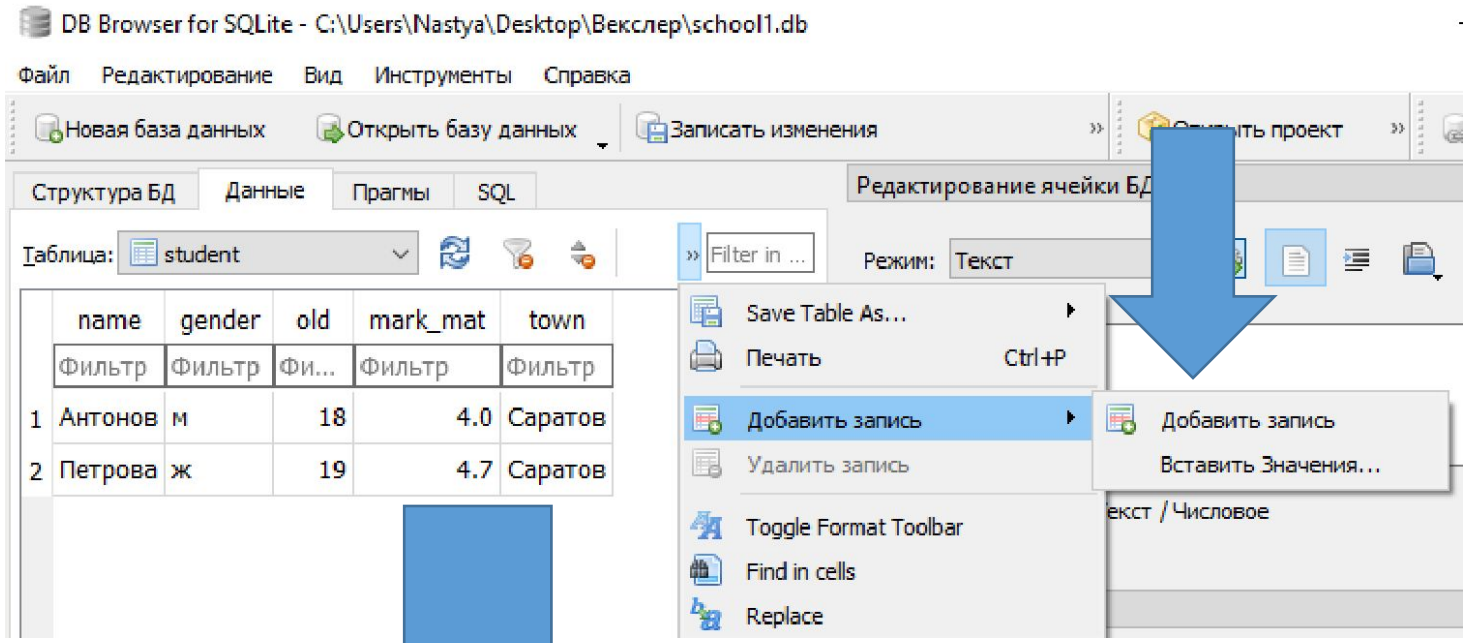
или так:

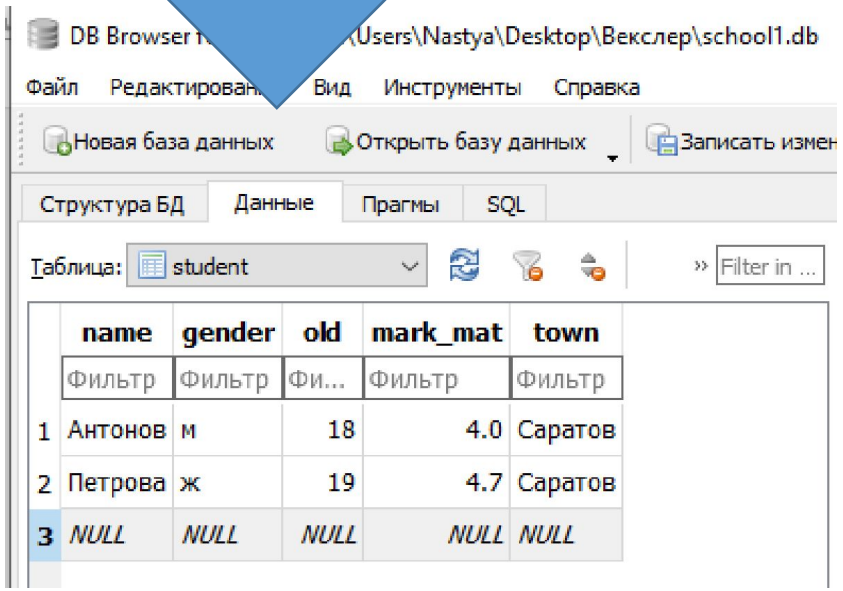
INSERT INTO <table_name> VALUES (<value1>, <value2>, …)

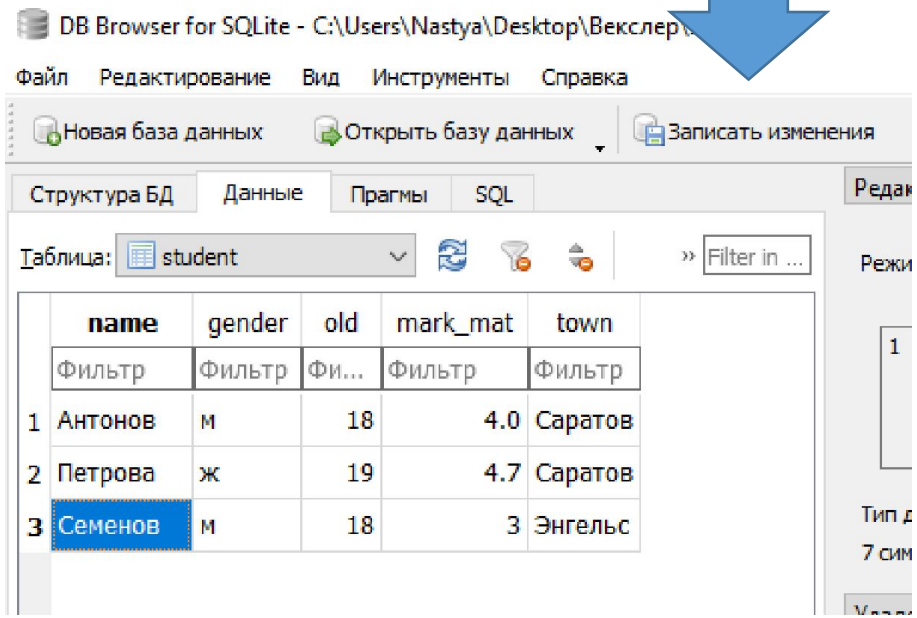
```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""INSERT INTO student (name, gender, old, mark_mat, town)
    VALUES('Антонов','м', 18, 4, 'Саратов')
    """)
    result = cur.fetchall()
    print(result)
```

```
[]
```

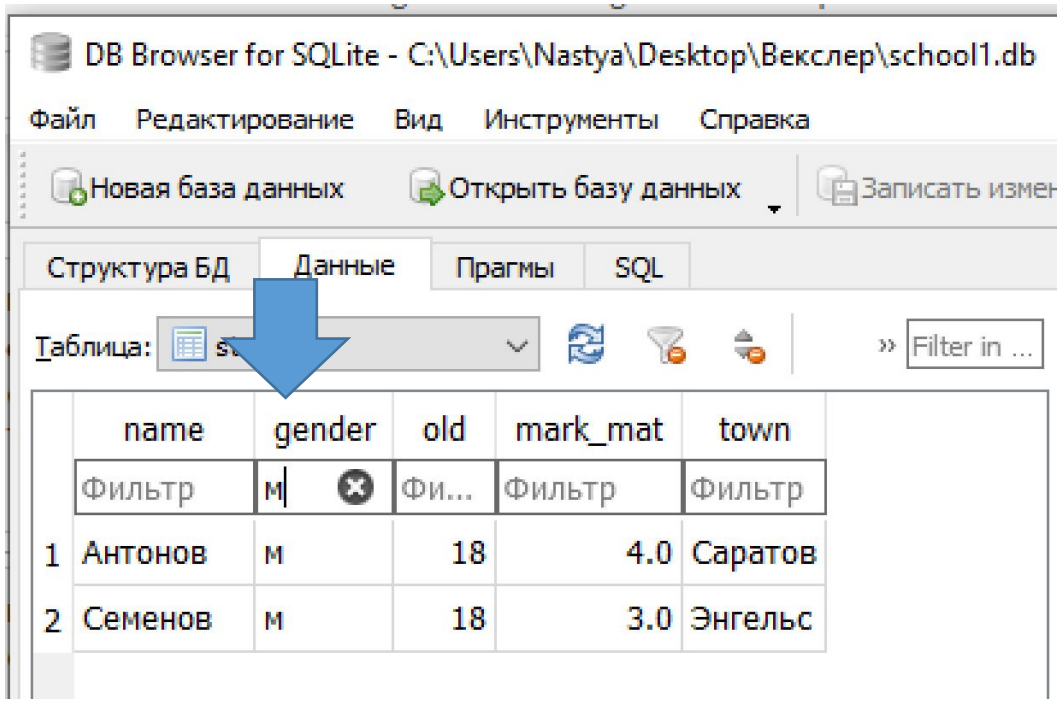

DB Browser for SQLite - C:\Users\Nastya\Desktop\Векслер\school1.db
Файл Редактирование Вид Инструменты Справка
Новая база данных
Открыть базу данных
Записать изменения
Открыть проект
Прикрепить БД
Структура БД
Данные
Прагмы
SQL
Таблица: student
Filter in ...
name gender old mark_mat town
Фильтр Фильтр Фи... Фильтр Фильтр
1 Антонов м 18 4.0 Саратов
Редактирование ячейки БД
Режим: Текст
Тип данных в ячейке: Текст
1 символ
Удаленный сервер
ID
DBHub.io
Local
Имя
1 - 1 из 1
Перейти к:
Журнал SQL
График
Схема БД
Удаленный сервер
SQL 1
INSERT INTO student (name, gender, old, mark_mat,
VALUES('Петрова','ж', 19, 4.7, 'Саратов')
Execution finished without errors.
Result: запрос успешно выполнен. Заняло 4мс, 1
изменено
At line 1:
INSERT INTO student (name, gender, old, mark_ma
town)
VALUES('Петрова','ж', 19, 4.7, 'Саратов')
NULL
Применить
Current Database
Имя
Изменен
2 Петрова ж 19 4.7 Саратов

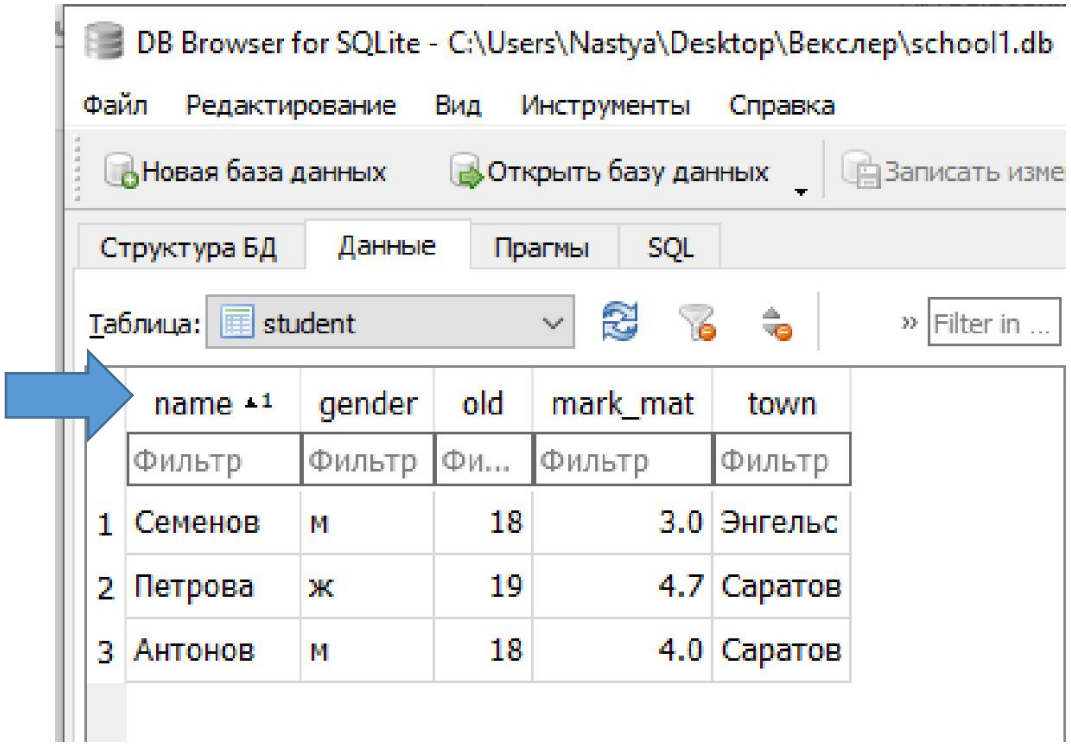

DB Browser for SQLite - C:\Users\Nastya\Desktop\Векслер\school1.db
Файл Редактирование Вид Инструменты Справка
Новая база данных
Открыть базу данных
Записать изменения
Структура БД
Данные
Прагмы
SQL
Редактирование ячейки БД
Таблица: student
Filter in ...
Режим: Текст
name gender old mark_mat town
Фильтр Фильтр Фи... Фильтр Фильтр
1 Антонов м 18 4.0 Саратов
2 Петрова ж 19 4.7 Саратов
Save Table As...
Печать Ctrl+P
Добавить запись
Удалить запись
Toggle Format Toolbar
Find in cells
Replace
Добавить запись
Вставить Значения...
екст / Числовое

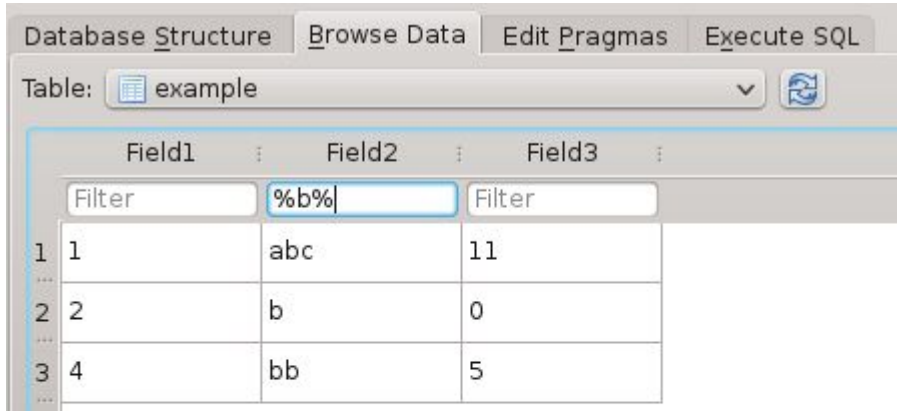

DB Browser
Users\Nastya\Desktop\Векслер\school1.db
Файл Редактирование Вид Инструменты Справка
Новая база данных
Открыть базу данных
Структура БД
Данные
Прагмы
SQL
Таблица: student
Filter in ...
name gender old mark_mat town
Фильтр Фильтр Фи... Фильтр Фильтр
1 Антонов м 18 4.0 Саратов
2 Петрова ж 19 4.7 Саратов
3 NULL NULL NULL NULL NULL

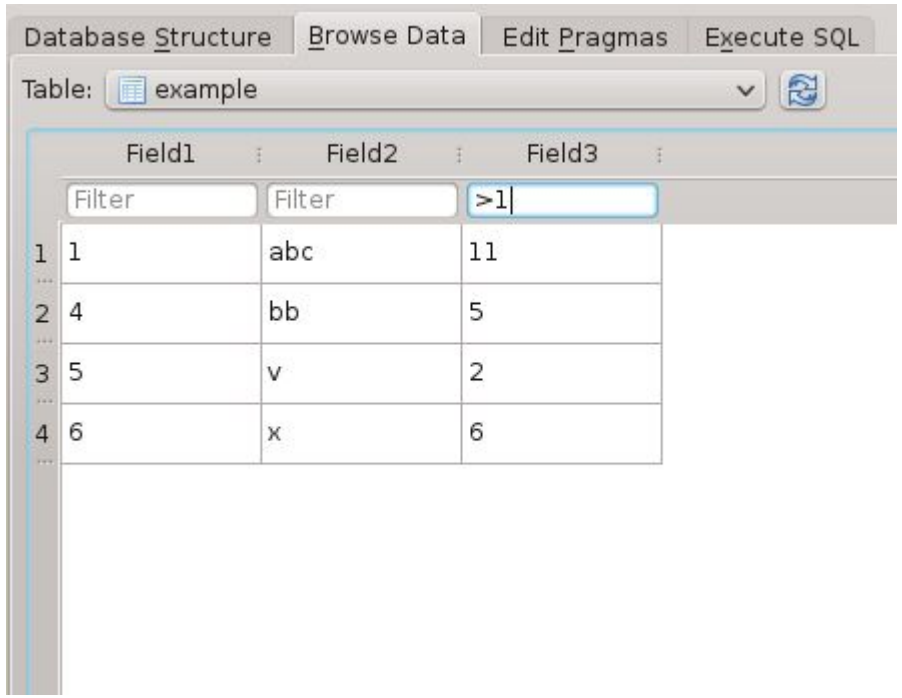

DB Browser for SQLite - C:\Users\Nastya\Desktop\Векслер
Файл Редактирование Вид Инструменты Справка
Новая база данных
Открыть базу данных
Записать изменения
Структура БД
Данные
Прагмы
SQL
Таблица: student
Filter in ...
name gender old mark_mat town
Фильтр Фильтр Фи... Фильтр Фильтр
1 Антонов м 18 4.0 Саратов
2 Петрова ж 19 4.7 Саратов
3 Семенов м 18 3 Энгельс

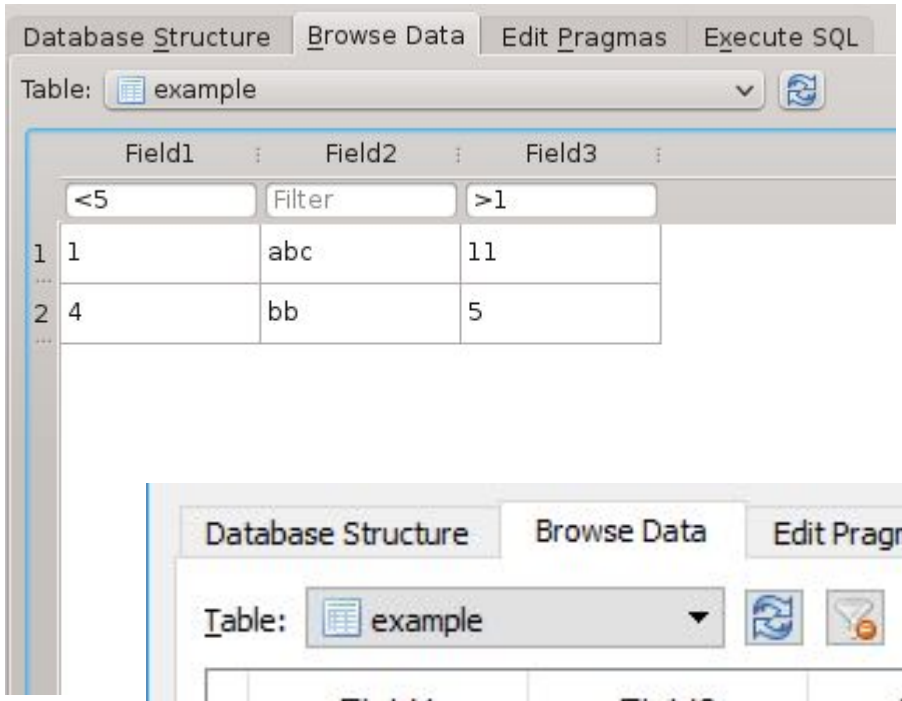
Фильтры, сортировки в программе

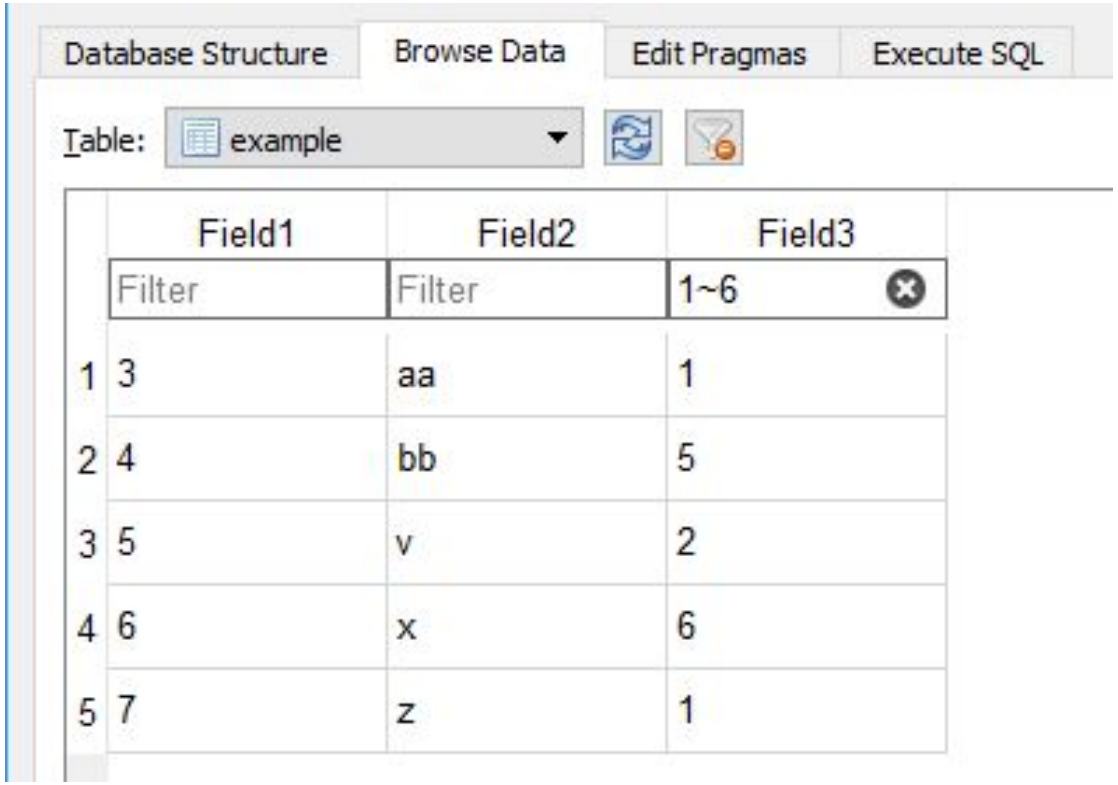





Database Structure
Browse Data
Edit Pragmas
Execute SQL
Table: example
Field1
Field2
Field3
Filter
%b%
Filter
1 1 abc 11
2 2 b 0
3 4 bb 5


Database Structure
Browse Data
Edit Pragmas
Execute SQL
Table: example
Field1
Field2
Field3
Filter
Filter
>1
1 1 abc 11
2 4 bb 5
3 5 v 2
4 6 x 6


Database Structure
Browse Data
Edit Pragmas
Execute SQL
Table: example
Field1
Field2
Field3
<5
Filter
>1
1 1 abc 11
2 4 bb 5


Database Structure
Browse Data
Edit Pragmas
Execute SQL
Table: example
Field1
Field2
Field3
Filter
Filter
1~6
1 3 aa 1
2 4 bb 5
3 5 v 2
4 6 x 6
5 7 z 1

Operator	Description	Notes
>	Greater than	
<	Less than	
>=	Equal to or greater	
<=	Equal to or less	
=	Equal to	Unlike the default behaviour this performs an 100% exact search, i.e. case-dependent and '%' characters not treated as wild cards but as normal characters.
<>	Unequal	This is the opposite of the '=' operator, that is no wild cards allowed and case-dependent search.

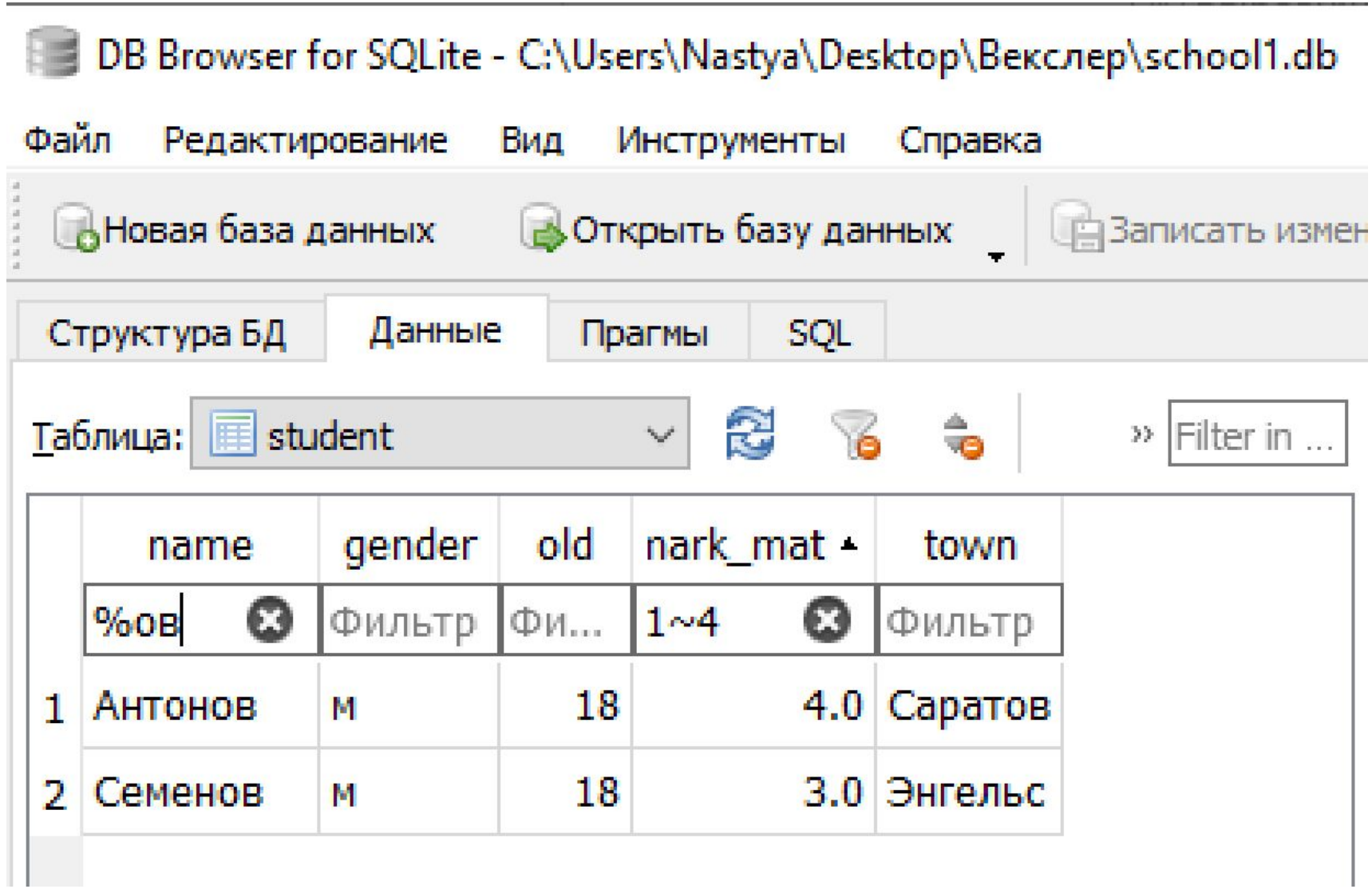
Regular-expression filter

You can filter the column using a regular expression. You only have to enclose the regular expression in '/' characters, like in `/regexp/`. Examples:

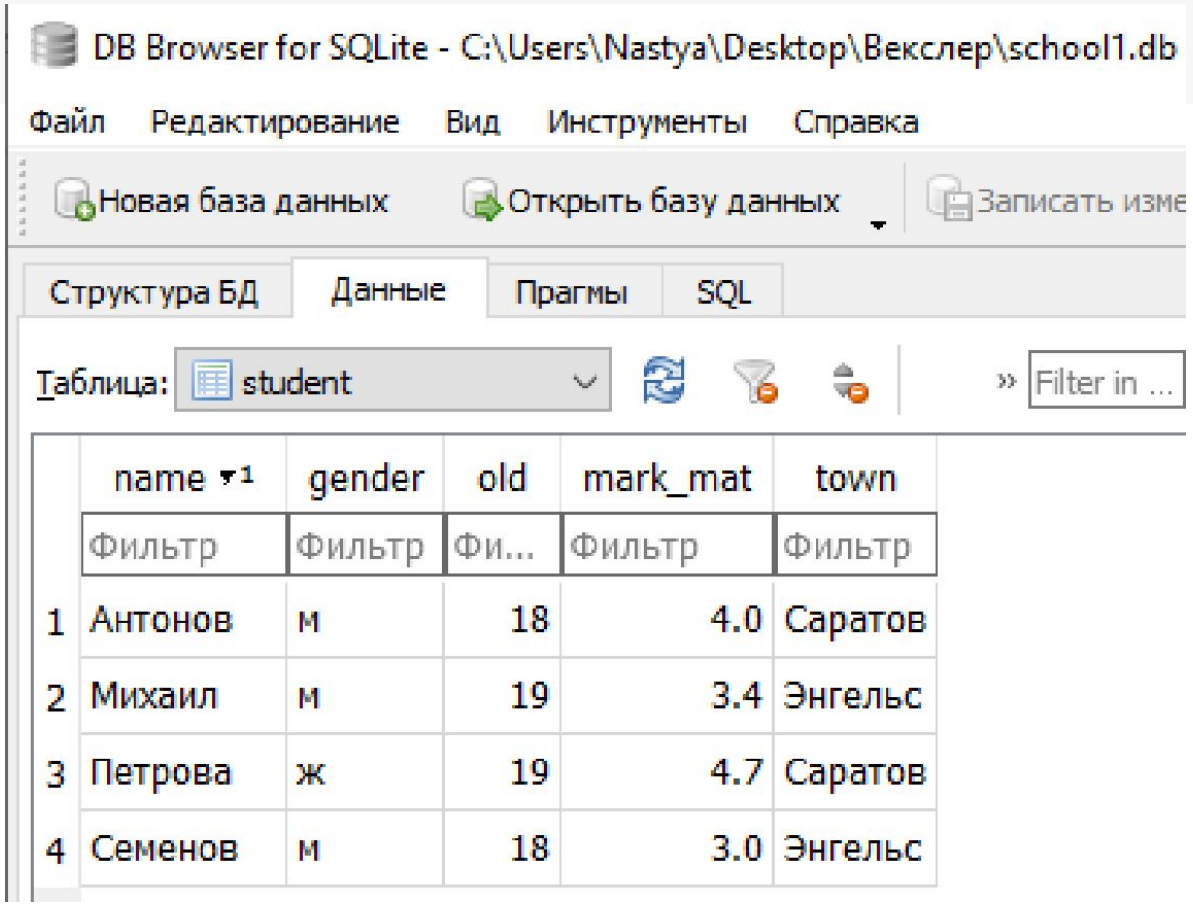
Example	Description
`/[a-zA-Z]/`	Contains any letter
`/^[a-zA-Z]*$/`	Contains only letters or is empty
`/Sat|Sun/`	Contains the word `Sat` or the word `Sun`
`/^Sat/`	Begins with `Sat`
`/Sun$/`	Ends with `Sun`
`/^(?!^.*US.*$)/`	Exclude all cells containing `US`
`/^((?!^Microsoft Office$\|^Adobe Reader$).)*$/`	Exclude all cells that don't contain either (`Microsoft Office` OR `Adobe Reader`)

If you want to use a "Containing" filter which has to include `/something/`, just use `\/something\/` to escape the character `/` and avoid the regular-expression interpretation.

NOTE - Release 3.12


DB Browser for SQLite - C:\Users\Nastya\Desktop\Векслер\school1.db
Файл
Редактирование
Вид
Инструменты
Справка
Новая база данных
Открыть базу данных
Записать измен
Структура БД
Данные
Прагмы
SQL
Таблица:
student
Filter in ...
name
gender
old
nark_mat
town
%ов
Фильтр
Фи...
1~4
Фильтр
1
Антонов
м
18
4.0
Саратов
2
Семенов
м
18
3.0
Энгельс

```python
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""INSERT INTO student VALUES('Михаил', 'м', 19, 3.4, 'Энгельс')
    """)
    result = cur.fetchall()
    print(result)
```



	name	gender	old	mark_mat	town
	Фильтр	Фильтр	Фи...	Фильтр	Фильтр
1	Антонов	м	18	4.0	Саратов
2	Михаил	м	19	3.4	Энгельс
3	Петрова	ж	19	4.7	Саратов
4	Семенов	м	18	3.0	Энгельс

SELECT – выборка данных из таблиц

SELECT col1, col2, … FROM <table_name>

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""SELECT name, old, mark_mat FROM student """)
    result = cur.fetchall()
    print(result)
    cur.execute("""SELECT * FROM student """)
    result = cur.fetchall()
    print(result)
```

```
[('Антонов', 18, 4.0), ('Петрова', 19, 4.7), ('Семенов', 18, 3.0), ('Михаил', 19, 3.4)]
[('Антонов', 'м', 18, 4.0, 'Саратов'), ('Петрова', 'ж', 19, 4.7, 'Саратов'), ('Семенов', 'м
'м', 19, 3.4, 'Энгельс')]
```

Оператор WHERE

Если нам нужно добавить фильтр для выбираемых записей, то это делается с помощью ключевого слова WHERE, которое записывается после имени таблицы:
SELECT col1, col2, … FROM <table_name> WHERE <условие>
после слова WHERE записывается логическое выражение и в качестве сравнения можно использовать следующие операторы: = или ==, >, =, <=, !=, BETWEEN

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""SELECT name, old, mark_mat FROM student WHERE gender == 'м'""")
    result = cur.fetchall()
    print(result)
    cur.execute("""SELECT * FROM student WHERE  mark_mat BETWEEN 3 AND 4""")
    result = cur.fetchall()
    print(result)
```

```
[('Антонов', 18, 4.0), ('Семенов', 18, 3.0), ('Михаил', 19, 3.4)]
[('Антонов', 'м', 18, 4.0, 'Саратов'), ('Семенов', 'м', 18, 3.0, 'Энгельс'), ('Михаил', 'м', 19, 3.4, 'Энгельс')]
```

Составные условия

Часто при описании фильтра требуется учитывать значения сразу нескольких столбцов:

- AND – условное И: exp1 AND exp2. Истинно, если одновременно истинны exp1 и exp2.
- OR – условное ИЛИ: exp1 OR exp2. Истинно, если истинно exp1 или exp2 или оба выражения.
- NOT – условное НЕ: NOT exp. Преобразует ложное условие в истинное и, наоборот, истинное – в ложное.
- IN – вхождение во множество значений: col IN (val1, val2, …)
- NOT IN – не вхождение во множество значений: col NOT IN (val1, val2, …)

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""SELECT * FROM student WHERE  (old IN(18, 19) OR gender = 'м') AND mark_mat > 4""")
    result = cur.fetchall()
    print(result)
```

```
[('Петрова', 'ж', 19, 4.7, 'Саратов')]
```

Сортировка ORDER BY

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""SELECT name, old, mark_mat FROM student WHERE gender == 'м' ORDER BY mark_mat""")
    result = cur.fetchall()
    print(result)
```

```
[('Семенов', 18, 3.0), ('Михаил', 19, 3.4), ('Антонов', 18, 4.0)]
```

Если нужно отсортировать данные по убыванию, то после имени поля следует указать флаг DESC:

SELECT * FROM users WHERE score < 1000 ORDER BY old DESC

Ограничение выборки LIMIT

```
with sq.connect("school1.db") as con:
    cur = con.cursor()
    cur.execute("""SELECT name, old, mark_mat FROM student WHERE gender == 'м' AND town='Энгельс'""")
    result = cur.fetchall()
    print(result)
    cur.execute("""SELECT * FROM student WHERE  mark_mat BETWEEN 3 AND 4 OR old IN (17,18)""")
    result = cur.fetchall()
    print(result)
```

```
[('Семенов', 18, 3.0), ('Михаил', 19, 3.4)]
[('Антонов', 'м', 18, 4.0, 'Саратов'), ('Семенов', 'м', 18, 3.0, 'Энгельс'), ('Михаил', 'м', 19, 3.4, 'Энгельс')]
```