

9-11 классы

Программирование на Python

Презентация занятия

Базы данных. SQLITE3.



инжинириум®

МГТУ им. Н.Э. Баумана

2022

Программирование
на Python

Теоретическая часть

Базы данных.
SQLITE3.

23 занятие



инжинириум®

МГТУ им. Н.Э. Баумана

2022

Тема: Многофункциональный парсер

Повторение:

```
from bs4 import BeautifulSoup
import requests # Модуль для http-запросов

url = 'https://inginirium.ru/'
req = requests.get(url) #Получение html-страницы
html = BeautifulSoup(req.text, "html5lib") #Преобразование её в дерево
#объектов Python
```

HTTP (англ. *HyperText Transfer Protocol* — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных

HTML (от англ. *HyperText Markup Language* — «язык гипертекстовой разметки») — стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере.

Тема: Многофункциональный парсер

Основные методы BeautifulSoup

```
tag_p = soup.find("p")
all_tags_p = soup.find("p")
all_tag_p_black = soup.find("p", class_="black_text")
# ... soup.find("p", {class_: "black_text", id:}) #Несколько атрибутов

# Методы, которые ходят по тегам одного уровня
next_sib = soup.find("ol").find_next_sibling()
prev_sib = soup.find("ol").find_previous_sibling()

# Метод, который может зайти внутрь тега
next_el = soup.find("ol").find_next()
```

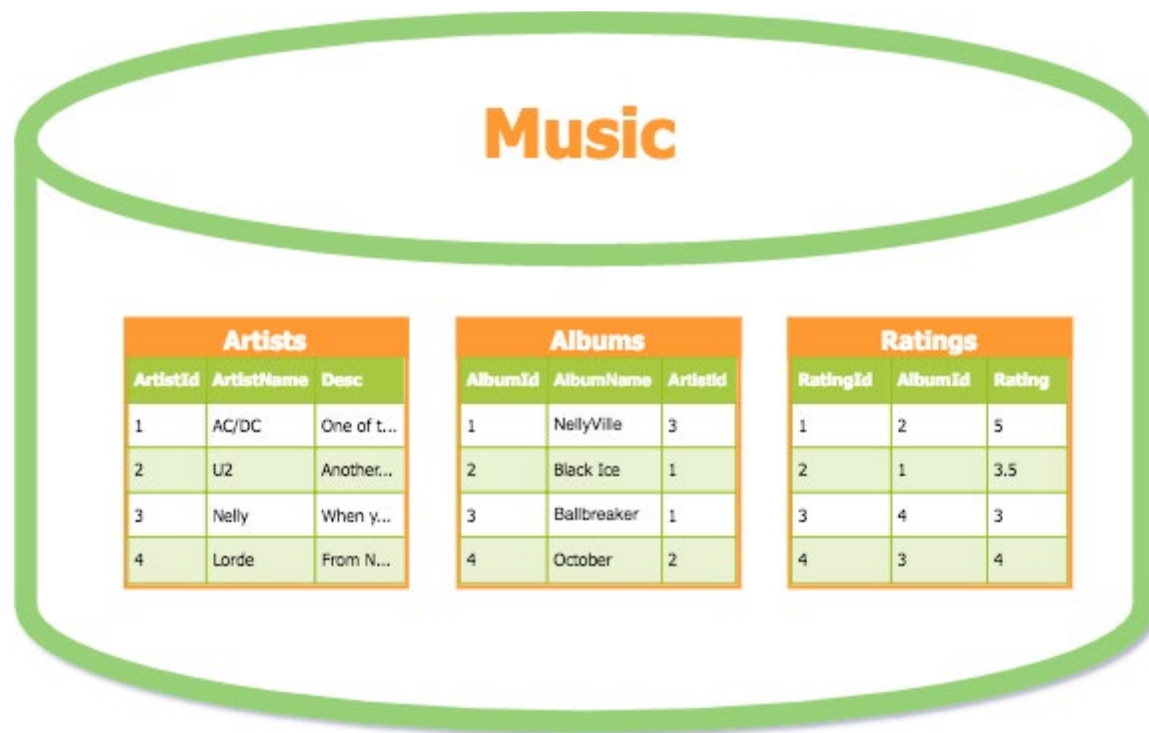
Тема: Базы данных. SQLite3.

База данных — это набор данных, хранящиеся в структурированном виде.





Тема: Базы данных. SQLite3.



Тема: Базы данных. SQLite3.

Система управления базами данных — это система, позволяющая создавать базы данных и манипулировать сведениями из них.

1	M130069	UAP1	5	3	6	3	4	12
2	M130069	UAP1	5	3	6	3	4	12
3	M130069	UAP1	5	3	6	3	4	12
4	M130069	UAP1	5	3	6	3	4	12
5	M130069	UAP1	5	3	6	3	4	12
6	M130069	UAP1	5	3	6	3	4	12
7	M130069	UAP1	5	3	6	3	4	12
8	M130069	UAP1	5	3	6	3	4	12
9	M130069	UAP1	5	3	6	3	4	12
10	M130069	UAP1	5	3	6	3	4	12

Тема: Базы данных. SQLite3.

- Первичный ключ. Первичный ключ состоит из набора значений, которые однозначно определяют запись базовой таблицы.
- Внешние ключи. Внешний ключ — это столбец, значения которого соответствуют значениям первичного ключа другой связанной таблицы.

```
films_db.sqlite
├── films
│   ├── id : integer
│   ├── title : text
│   ├── year : int
│   ├── genre : int
│   └── duration : int
└── genres
    ├── id : integer
    └── title : text
```



Тема: Базы данных. SQLite3.



films_db.sqlite

- films
 - id : integer (primary key)
 - title : text
 - year : int
 - genre : int
 - duration : int
- genres
 - id : integer (primary key)
 - title : text

id	title
1	комедия
2	драма
3	мелодрама
4	детектив
5	документальный
6	ужасы
7	музыка

id	title	year	genre	duration
1	А был ли Каротин	2000	123	154
2	А в России опять окаянные дни	1990	2	133
3	А вдруг это любовь?	2007	1	90
4	А вот и моя крошка	1994	2	96
5	А вот и Полли	2004	1	90





Тема: Базы данных. SQLite3.



SQL - Structured Query Language (язык структурированных запросов) - язык взаимодействия с базами данных (БД).



Тема: Базы данных. SQLite3.

7 из 10 рейтинговых баз данных—реляционные, запросы к которым осуществляются с помощью SQL. Данные факты, раскрывают его значимость и повсеместность.



1. Oracle
2. MySQL
3. Microsoft SQL Server
4. PostgreSQL
5. MongoDB
6. IBM Db2
7. Redis
8. Microsoft Access
9. Cassandra
10. SQLite

Тема: Базы данных. SQLite3.

Одна из основных функций SQL — получение данных из СУБД. Для построения всевозможных запросов к базе данных используется оператор SELECT. Он позволяет выполнять сложные проверки и обработку данных.

```
SELECT [DISTINCT | ALL] поля_таблиц  
FROM список_таблиц  
[WHERE условия_на_ограничения_строк]  
[GROUP BY условия_группировки]  
[HAVING условия_на_ограничения_строк_после_группировки]  
[ORDER BY порядок_сортировки [ASC | DESC]]  
[LIMIT ограничение_количества_записей]
```

Structured Query Language



В описанной структуре запроса необязательные параметры указаны в квадратных скобках.

Тема: Базы данных. SQLite3.

Параметры оператора

DISTINCT используется для исключения повторяющихся строк из результата

ALL (по умолчанию) используется для получения всех данных, в том числе и повторений

FROM перечисляет используемые в запросе таблицы из базы данных

WHERE — это условный оператор, который используется для ограничения строк по какому-либо условию

GROUP BY используется для группировки строк

HAVING применяется после группировки строк для фильтрации по значениям агрегатных функций

ORDER BY используется для сортировки. У него есть два параметра:

ASC (по умолчанию) используется для сортировки по возрастанию

DESC — по убыванию

LIMIT используется для ограничения количества строк для вывода

Тема: Базы данных. SQLite3.

```
SELECT столбцы FROM имя_таблицы  
WHERE условие
```



```
films_db.sqlite  
└─ films  
    ├── id : integer  
    ├── title : text  
    ├── year : int  
    ├── genre : int  
    └── duration : int
```

```
SELECT title FROM Films  
WHERE year = 2010
```

	title
1	Алиса в стране чудес
2	Железный человек 2
3	Ноттингем
4	Утомленные солнцем: Предстояние

Тема: Базы данных. SQLite3.

```
SELECT столбцы FROM имя_таблицы  
WHERE условие
```

```
SELECT * FROM Films  
WHERE year = 2010
```

	id	title	year	genre	duration
1	248	Алиса в стране чудес	2010	13	
2	4382	Железный человек 2	2010	11	
3	9138	Ноттингем	2010	11	
4	15495	Утомленные солнцем: Предстояние	2010	2	



Тема: Базы данных. SQLite3.

films_db.sqlite

films

- id : integer
- title : text
- year : int
- genre : int
- duration : int

`SELECT` столбцы `FROM` имя_таблицы
`WHERE` условие

`SELECT title, duration * FROM Films`
`WHERE duration IN (45, 90)`

	id	title	year	genre	duration	
1	3	А вдруг это любовь?	2007	1	90	
2	39	Абсурдистан	2008	1	88	
3	148	Адреналин	2006	11	87	
4	160	Адский бункер	2008	11	90	
5	173	Азиат	2008	11	90	
6	174	Азирис Нуна	2006	16	90	
7	232	Александра	2007	2	90	
8	414	Андрей Миронов. Обыкновенное чудо	2006	5	52	
9	448	Антидурь	2007	11	90	
10	529	Астерикс и викинги	2006	13	78	



Тема: Базы данных. SQLite3.





Тема: Базы данных. SQLite3.

```
1  import sqlite3
2
3  # подключение к базе данных films_db.sqlite
4  # создание СОЕДИНЕНИЯ с БД
5  con = sqlite3.connect('films_db.sqlite')
6
7  # создание КУРСОРА через СОЕДИНЕНИЕ
8  # КУРСОР и будет общаться с БД
9  cur = con.cursor()
10
11 # создание запроса к БД через КУРСОР
12 # query - запрос (SQL - Structured Query Language)
13 que = '''
14 SELECT title, duration FROM films
15 WHERE year = 2010
16 '''
17
18 # просим КУРСОР выполнить (execute) запрос и
19 # ПОЛУЧИТЬ (fetch - синоним get - получать)
20 result = cur.execute(que)
21 data = result.fetchall()
22 for line in data:
23     print(line)
24
25 con.close()
```

```
('Алиса в стране чудес', 201)
('Железный человек 2', 287)
('Ноттингем', 188)
('Утомленные солнцем: Предстояние', 139)
```



Программирование
на Python

Практическая часть

Базы данных.
SQLITE3.

23 занятие



инжинириум®

МГТУ им. Н.Э. Баумана

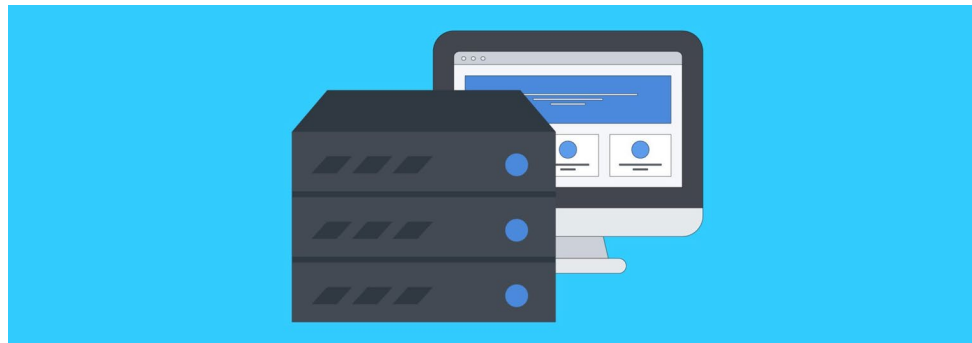
2022

Тема: Базы данных. SQLite3.

Первым этапом создания таблицы БД является задание ее структуры, т.е. определение количества и типа полей. Вторым этапом является ввод и редактирование записей в таблицу. БД считается созданной, даже если она пустая.

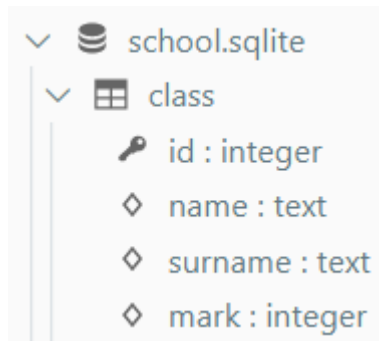
Поля таблицы просто определяют ее структуру и групповые свойства данных, записываемых в ячейках. Рассмотрим основные свойства полей БД.

1. **Имя поля** — определяет как надо обращаться к данным поля (имена используются как заголовки таблиц).
2. **Тип поля** — определяет тип данных, которые могут содержаться в данном поле (текстовые, числовые, и др.).



Тема: Базы данных. SQLite3.

Создадим БД (school.sqlite) с одной таблицей, которую назовём class.
Структура class представлена на фото ниже.



school.sqlite	
class	
id	integer
name	text
surname	text
mark	integer

```
1  import sqlite3
2
3  con = sqlite3.connect('school.sqlite')
4  cur = con.cursor()
5
6  que_create = '''
7  CREATE TABLE IF NOT EXISTS class (
8      id INTEGER PRIMARY KEY,
9      name TEXT,
10     surname TEXT,
11     mark INTEGER
12 )
13 '''
14
15 cur.execute(que_create)
16 con.commit()
17
18 con.close()
```





Тема: Базы данных. SQLite3.

Занесём в class пару строк.

```
1  import sqlite3
2
3  con = sqlite3.connect('test.sqlite3')
4  cur = con.cursor()
5
6  que_insert = '''
7  INSERT INTO school (name, surname, mark) VALUES
8      ('Василий', 'Пупкин', 3),
9      ('Денис', 'Синицын', 4),
10     ('Ангелина', 'Соколова', 5),
11     ('Саша', 'Петров', 2)
12  '''
13
14  cur.execute(que_insert)
15  con.commit()
16
17  con.close()
```

id	name	surname	mark
1	Василий	Пупкин	3
2	Денис	Синицын	4
3	Ангелина	Соколова	5
4	Саша	Петров	2





Тема: Базы данных. SQLite3.

Занесём в class ещё пару строк.

```
1  import sqlite3
2  from random import choice
3
4  con = sqlite3.connect('school.sqlite')
5  cur = con.cursor()
6
7  que_insert = '''
8  INSERT INTO class (name, surname, mark) VALUES
9  |   ('{}', '{}', {})
10 |   '''
11
12  pool_name = ('Василий', 'Денис', 'Костя', 'Саша')
13  pool_surname = ('Синицин', 'Соколов', 'Петров', 'Крылов')
14  pool_nums = tuple(range(2, 6))
15
16  for i in range(5):
17      name_insert = choice(pool_name)
18      surname_insert = choice(pool_surname)
19      mark_insert = choice(pool_nums)
20      cur.execute(que_insert.format(name_insert, surname_insert, mark_insert))
21
22  con.commit()
23  con.close()
```

id	name	surname	mark
1	Василий	Пупкин	3
2	Денис	Синицин	4
3	Ангелина	Соколова	5
4	Саша	Петров	2
5	Денис	Синицин	3
6	Костя	Соколов	5
7	Костя	Синицин	3
8	Костя	Петров	3
9	Денис	Синицин	4



Тема: Базы данных. SQLite3.

```
que_select_cols = '''SELECT name,mark FROM class'''
que_select_all = '''SELECT * FROM class WHERE mark > 4'''

cur.execute(que_select_all)
res = cur.fetchall() #Получение всех записей таблицы
# .fetchone - получение одной записи
# .fetchmany(N) - получение N-го количества записей
print(res)

con.commit()
con.close()
```


Тема: Базы данных. SQLite3.

Операторы:

- AND - условное И
- OR - условное ИЛИ
- IN - вхождение в множество
- NOT IN - невхождение в множество
- NOT - условное НЕ

=, ==, >, <, <=, >=, !=, BETWEEN

```
'''SELECT * FROM class WHERE mark BETWEEN 2 and 4'''
```



Тема: Базы данных. SQLite3.

Задание 1.

Создайте базу данных game, с таблицей score и полями name, score_points.

* Занесите 100 случайных значений

** Выведите в терминал 5 лучших результатов из таблицы



Тема: Базы данных. SQLite3.

Задание 2.

- С помощью парсинга сайта “Инжинириум” (см. прошлый урок) составьте базу данных, в которой будет отмечено способность вести основные курсы “Инжинириума”: 3D-моделирование, Робототехника, Биотехнологии, Композиты, Курс Юного(Молодого) Инженера, Python, C++, Scratch, Web-программирование.
- С помощью SQL-запроса вывести всех преподавателей IT-направления, посчитать их количество.

name	modeling	wedo	arduino	bio
Соколов Григорий В...	-	-	-	-
Григорьева Елизаве...	-	-	-	+
Нурапкин Антон Евг...	-	+	-	-
Тектуманидзе Алекс...	-	-	-	-
Сайдашева Асия Ни...	-	-	-	+
Касьяненко Анна Се...	-	-	-	-
Рыкова Валентина С...	-	-	-	+





Тема: Базы данных. SQLite3.

```
import sqlite3
import requests
from bs4 import BeautifulSoup

# Парсинг данных(см. прошлый урок)
url2 = 'https://inginirium.ru/about/'
req = requests.get(url2)
html = BeautifulSoup(req.text, "html5lib")

# Парсинг преподавателей
teachers = html.find_all('div', class_="col-12 course__teacher_name")
# Извлечение текста
for i in range(len(teachers)):
    teachers[i] = teachers[i].find('h3').text

# Парсинг курсов преподавателей
courses_of_teachers = html.find_all('div', class_="col-12 course__teacher_courselist")
# Извлечение текста из тегов
for i in range(len(courses_of_teachers)):
    courses_of_teachers[i] = courses_of_teachers[i].find_all('a')
    for j in range(len(courses_of_teachers[i])):
        courses_of_teachers[i][j] = courses_of_teachers[i][j].text
```



Тема: Базы данных. SQLite3.

```
# На данный момент есть список с именами из 57 элементов
# Есть двумерный список из 57 списков, внутри которых
# находятся курсы, которые ведут преподаватели
# У некоторых они могут быть пустыми, так как на сайте нет информации

# Названия основных курсов
courses_names = ['web', 'c++', 'python', 'scratch', 'ev3', 'моделирование',
                  'wedo', 'arduino', 'био', 'композит', 'инженер']
# Создания матрицы для удобства представления
array = [['-' for j in range(len(courses_names))] for i in range(len(teachers))]

# Здесь мы смотрим, есть ли название курса
# в списке предметов преподавателей
# Если есть, то ставим плюс
for i in range(len(teachers)):
    for j in range(len(courses_names)):
        for k in range(0, len(courses_of_teachers[i])):
            if courses_names[j] in courses_of_teachers[i][k].lower():
                array[i][j] = "+"
                break
```





Тема: Базы данных. SQLite3.

#Создание БД

```
con = sqlite3.connect('teachers.sqlite')
```

```
cur = con.cursor()
```

```
cur.execute('''CREATE TABLE IF NOT EXISTS teachers(  
    id INTEGER PRIMARY KEY,  
    name TEXT,  
    web TEXT,  
    cpp TEXT,  
    python TEXT,  
    scratch TEXT,  
    ev3 TEXT,  
    modeling TEXT,  
    wedo TEXT,  
    arduino TEXT,  
    bio TEXT,  
    composites TEXT,  
    engineer TEXT  
)''')
```





Тема: Базы данных. SQLite3.

```
que_insert = '''INSERT INTO teachers(name,web,cpp,python,scratch,ev3,modeling,  
VALUES ('{}',{})'''
```

Составление SQL-запроса

```
for i in range(len(teachers)):
```

Циклом ниже создаем строку,

которая будет вставлена в запрос

```
string = ''
```

```
for j in range(len(courses_names)):
```

```
    if j == 0:
```

```
        string = f''' {{array[i][j]}}' '''
```

```
    else:
```

```
        string = string + "," + f''' {{array[i][j]}}' '''
```

```
cur.execute(que_insert.format(teachers[i],string))
```

```
con.commit()
```

```
con.close()
```



Тема: Базы данных. SQLite3.

Вывод данных быстрее будет сделать в отдельном файле
`import sqlite3`

```
con = sqlite3.connect('teachers.sqlite')  
cur = con.cursor()
```

```
que_select = '''SELECT name FROM teachers  
|         |         |         |         WHERE web="+" OR cpp="+" OR python = "+" OR scratch = "+"'''
```

```
cur.execute(que_select)  
names = cur.fetchall()  
for name in names:  
|     print(*name)  
print("Всего преподавателей в IT:", len(names))  
  
con.commit()  
con.close()
```

Всего преподавателей в IT: 12
Новиков Андрей Дмитриевич
Андреев Данил Алексеевич
Левина Наталья Александровна
Оганян Диана Вардановна
Приёмко Кирилл Сергеевич