8-11 классы

Программирование
на Python

Презентация занятия

Создание первой реляционной бд. Заполнение

52 занятие

инжиниум®
МГТУ им. Н.Э. Баумана

2023

Python: Создание первой реляционной бд.

В текущем занятии необходимо заполнить нашу базу данных при помощи генерации случайных чисел и ООП. Для заполнения добавим следующие библиотеки:

```
import sqlite3
import random as rd
```

Python: Создание первой реляционной бд.

Описание работы класса student_list:
Класс student_list должен хранить словарь, где в качестве ключей будет id, а в качестве значений будет объект класса student. Также класс должен иметь возможность случайно генерировать данные студентов, с учетом дубликатов и возможность занесения данных в файл с расширением db

Python: Создание первой реляционной бд.

Создадим класс student_list для работы с списком учеников, также при вызове конструктора будет открываться нужный файл и создаваться таблица :

```
class student_list(student):
    def __init__(self):
        self.data={}
        self.connection = sqlite3.connect('database.db')
        self.cursor = self.connection.cursor()
        self.cursor.execute('''create table if not exists students(
            id int,
            name text,
            lastname text,
            age int,
            group_id text,
            gpa float)''')
```

Python: Создание первой реляционной бд.

Для генерации случайной пары имени и фамилии создадим метод generate_once:

```
def generate_once(self):
        name_m = ['Sasha', 'Timur', 'Ivan', 'Nikolay', 'Dima', 'Andrey', 'Alex']
        lastname_m = ['Korolev', 'Tretyakov', 'Zaicev', 'Kotov', 'Kiselev', 'Ivanov',
'Alehin']
        name_f = ['Anya', 'Inna', 'Sveta', 'Lena', 'Vika', 'Veronica', 'Olya']
        lastname_f = ['Koroleva', 'Tretyakova', 'Solovieva', 'Shukina', 'Pychkova',
'Zhukova', 'Smirnova']
        if rd.randint(0, 1) == 1:
            return (name_m[rd.randint(0,6)], lastname_m[rd.randint(0,6)])
        return (name_f[rd.randint(0,6)], lastname_f[rd.randint(0,6)])
```

Python: Создание первой реляционной бд.

Для генерации списка студентов реализуем метод generate, принимающий размер списка, который будет сгенерирован, необходимо убедиться, чтобы имя и фамилии не повторялись:

```
def generate(self, n):
        a = [self.generate_once() for i in range(n)]
        groups = ['IT', '3D', 'ROBO', 'BIO', 'SOC', 'MATH', 'PYSH']
        res = []
        for i in a:
            if i not in res:
                res.append(i)
        for i in range(len(res)):
            self.data[i+1] = student(res[i][0], res[i][1], rd.randint(10,
30), rd.choice(groups), rd.randint(20, 50)/10)
```

Python: Создание первой реляционной бд.

Для успешного вывода и работы с списком реализуем метод get_list и перегрузим оператор __str__:

```
def get_list(self):
    return [(id,)+self.data[id].get_list() for id in self.data]
def __str__(self):
    return '\n'.join(map(str, self.get_list()))
```

Python: Создание первой реляционной бд.

Для записи данных в таблицу students реализуем метод put:

```
def put(self):
        for new in self.get_list():
            if new not in self.cursor.execute('select * from students'):
                self.cursor.execute('insert into students values (?,?,?,?,?,?)', new)
                self.connection.commit()
                print(f'Student {new} added')
            else:
                print(f'Student {new} already in the database')
                pass
```

Python: Создание первой реляционной бд.

Для проверки генерации данных необходимо скрестить пальцы и запустить следующий код:

```
def test():
    connection = sqlite3.connect('database.db')
    cursor = connection.cursor()
    for i in cursor.execute('select * from students'):
        print(i)
data = student_list()
rd.seed(100)
data.generate(90)
#print(data)
data.put()
test()
```