**Лабораторная работа № 6**

**РАЗРАБОТКА ПРИЛОЖЕНИЯ РАБОТЫ С БАЗОЙ ДАННЫХ**

**Цель работы:** изучить возможности взаимодействия Python с реляционными базами данных с помощью DB-API 2.0

**Краткая теория**

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 СУБД. Здесь будут рассмотрены механизмы для работы с базами данных с помощью Python.

**Задание.** **Вариант 3.**

Написать функцию, которая возвращает наименование всех таблиц связки, которые находятся в базе данных, если они существуют. Стоит отметить, что в данном задании предполагается, что таблицы связки именуются склеиванием имени двух таблиц через знак нижнего подчеркивания «\_».

import sqlite3  
  
def create\_database(db\_path):  
 connection = sqlite3.connect(db\_path)  
 cursor = connection.cursor()  
  
 # Создание таблицы  
 cursor.execute('''  
 CREATE TABLE IF NOT EXISTS Table1 (  
 id INTEGER PRIMARY KEY,  
 name TEXT NOT NULL  
 );  
 ''')  
  
 # Создание второй таблицы  
 cursor.execute('''  
 CREATE TABLE IF NOT EXISTS Table2 (  
 id INTEGER PRIMARY KEY,  
 value INTEGER NOT NULL  
 );  
 ''')  
  
 # Пример создания таблицы связки  
 cursor.execute('''  
 CREATE TABLE IF NOT EXISTS Table1\_Table2 (  
 id INTEGER PRIMARY KEY,  
 table1\_id INTEGER,  
 table2\_id INTEGER,  
 FOREIGN KEY (table1\_id) REFERENCES Table1(id),  
 FOREIGN KEY (table2\_id) REFERENCES Table2(id)  
 );  
 ''')  
  
 connection.commit()  
 connection.close()  
  
def get\_foreign\_key\_tables(db\_path):  
 connection = sqlite3.connect(db\_path)  
 cursor = connection.cursor()  
  
 cursor.execute("SELECT name FROM sqlite\_master WHERE type='table';")  
 all\_tables = cursor.fetchall()  
  
 foreign\_key\_tables = []  
 for table in all\_tables:  
 table\_name = table[0]  
 if "\_" in table\_name:  
 foreign\_key\_tables.append(table\_name)  
  
 connection.close()  
 return foreign\_key\_tables  
  
# Пример использования  
db\_path = "your\_database.db"  
create\_database(db\_path)  
result = get\_foreign\_key\_tables(db\_path)  
print(result)

Рисунок 1.1 – код программы.

Результат работы программы представлен на рисунке 1.2

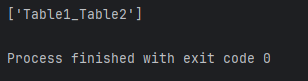


Рисунок 1.2 – результат работы программы.

**Вывод**: в ходе выполненной лабораторной работы были изучены возможности взаимодействия Python с реляционными базами данных с помощью DB-API 2.0

|  |  |
| --- | --- |
| Выполнил | Семеняк В.А. |
| Проверил | Елкин Н.С. |