Институт цифрового развития
Кафедра инфокоммуникаций

«Взаимодействие с базами данных SQLite3 с помощью языка программирования Python»

ОТЧЕТ
по лабораторной работе №21
дисциплины
«Основы программной инженерии»

Выполнил:
Мизин Глеб Егорович
2 курс, группа ПИЖ-б-о-21-1,
011.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и сопровождение программного обеспечения», очная форма обучения

_____
(подпись)

Проверил:
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Пример:

```python
def create_db(database_path: Path) -> None:
    """
    Создать базу данных.
    """
    conn = sqlite3.connect(database_path)
    cursor = conn.cursor()

    # Создать таблицу с информацией о должностях.
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS posts (
            post_id INTEGER PRIMARY KEY AUTOINCREMENT,
            post_title TEXT NOT NULL
        )
        """
    )

    # Создать таблицу с информацией о работниках.
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS workers (
            worker_id INTEGER PRIMARY KEY AUTOINCREMENT,
            worker_name TEXT NOT NULL,
            post_id INTEGER NOT NULL,
            worker_year INTEGER NOT NULL,
            FOREIGN KEY(post_id) REFERENCES posts(post_id)
        )
        """
    )
```

Индивидуальное задание:

```python
def create_db(database_path: Path) -> None:
    """
    Создать базу данных.
    """
    conn = sqlite3.connect(database_path)
    cursor = conn.cursor()

    # Создать таблицу с информацией об Трансферах.
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS Transfers (
            Transfer_id INTEGER PRIMARY KEY AUTOINCREMENT,
            Transfer_ammount INTEGER
        )
        """
    )

    # Создать таблицу с информацией об аккаунтах получателя и отправителя .
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS Accounts (
            Sender_id INTEGER PRIMARY KEY AUTOINCREMENT,
            Sender INTEGER,
            Receiver_id INTEGER,
            Receiver INTEGER,
            FOREIGN KEY(Sender_id) REFERENCES Transfers(Transfer_id)
        )
        """
    )

```

Контрольные вопросы

1. Каково назначение модуля sqlite3 ?

Модуль sqlite3 предназначен для работы с базой данных SQLite3.

2. Как выполняется соединение с базой данных SQLite3? Что такое курсор базы данных?

Соединение с базой данных SQLite3 выполняется через функцию connect(). Курсор базы данных - это объект, который используется для выполнения операций с базой данных.

3. Как подключиться к базе данных SQLite3, находящейся в оперативной памяти компьютера?

Чтобы подключиться к базе данных SQLite3, находящейся в оперативной памяти компьютера, нужно передать имя файла ":memory:" при вызове функции connect().

```
def sql_connection():
    try:
        con = sqlite3.connect(':memory:')
        print("Connection is established: Database is created in memory")
```

4. Как корректно завершить работу с базой данных SQLite3?

Для корректного завершения работы с базой данных SQLite3 нужно закрыть курсор и соединение с базой данных.

5. Как осуществляется вставка данных в таблицу базы данных SQLite3?

Данные в таблицу базы данных SQLite3 можно вставить с помощью метода execute(), используя SQL-запрос с командой INSERT.

```
import sqlite3


con = sqlite3.connect('mydatabase.db')


def sql_insert(con, entities):
    cursor_obj = con.cursor()
    cursor_obj.execute(
        """
        INSERT INTO employees(id, name, salary, department, position, hireDate)
        VALUES(?, ?, ?, ?, ?, ?)
        """,
        entities
    )
    con.commit()
```

```
entities = (2, 'Andrew', 800, 'IT', 'Tech', '2018-02-06')
sql_insert(con, entities)
```

6. Как осуществляется обновление данных таблицы базы данных SQLite3?

Данные в таблице базы данных SQLite3 можно обновить с помощью метода execute(), используя SQL-запрос с командой UPDATE.

```
import sqlite3


con = sqlite3.connect('mydatabase.db')


def sql_update(con):
    cursor_obj = con.cursor()
    cursor_obj.execute(
        "UPDATE employees SET name = 'Rogers' where id = 2"
    )

    con.commit()


sql_update(con)
```

7. Как осуществляется выборка данных из базы данных SQLite3?

Данные из базы данных SQLite3 можно получить с помощью метода execute(), используя SQL-запрос с командой SELECT.

```
import sqlite3


con = sqlite3.connect('mydatabase.db')


def sql_fetch(con):
    cursor_obj = con.cursor()
    cursor_obj.execute("SELECT * FROM employees")

    rows = cursorObj.fetchall()
    for row in rows:
        print(row)


sql_fetch(con)
```

8. Каково назначение метода rowcount?

Метод rowcount возвращает количество строк, затронутых последней операцией.

9. Как получить список всех таблиц базы данных SQLite3?

Список всех таблиц базы данных SQLite3 можно получить с помощью метода execute(), используя SQL-запрос с командой SELECT.

```
import sqlite3


con = sqlite3.connect('mydatabase.db')


def sql_fetch(con):
    cursor_obj = con.cursor()
    cursor_obj.execute(
        "SELECT name from sqlite_master where type='table'"
    )

    print(cursor_obj.fetchall())


sql_fetch(con)
```

10. Как выполнить проверку существования таблицы как при ее добавлении, так и при ее удалении?

Существование таблицы можно проверить с помощью метода execute(), используя SQL-запрос с командой CREATE или DROP.

11. Как выполнить массовую вставку данных в базу данных SQLite3?

Массовую вставку данных в базу данных SQLite3 можно выполнить с помощью метода executemany().

12. Как осуществляется работа с датой и временем при работе с базами данных SQLite3?

Для работы с датой и временем в базе данных SQLite3 есть функции и определенные форматы данных. Например, можно использовать функцию strftime() для форматирования даты и времени.