# Лабораторная работа: Подключение базы данных к десктопному приложению на Tkinter

## Цель работы:

Научиться подключать базы данных к десктопным приложениям, построенным на библиотеке Tkinter, для выполнения операций CRUD (Create, Read, Update, Delete).

## Этапы выполнения работы

### 1. Подготовка окружения

1. Установите Python, если он еще не установлен.
2. Установите библиотеку для работы с базой данных SQLite (уже включена в стандартную библиотеку Python).
3. Установите дополнительные модули, если вы используете MySQL или PostgreSQL:
 ```bash
 pip install pymysql psycopg2
 ```

### 2. Создание структуры приложения

1. Создайте файл проекта, например, `app.py`.
2. Подключите библиотеку Tkinter:
 ```python
 from tkinter import \*
 from tkinter import messagebox
 import sqlite3
 ```

### 3. Настройка базы данных

1. Создайте функцию для подключения к SQLite и инициализации таблицы:
 ```python
 def connect\_db():
 conn = sqlite3.connect("app\_database.db")
 cursor = conn.cursor()
 cursor.execute("""
 CREATE TABLE IF NOT EXISTS users (
 id INTEGER PRIMARY KEY AUTOINCREMENT,
 name TEXT NOT NULL,
 age INTEGER NOT NULL
 )
 """)
 conn.commit()
 conn.close()
 ```

2. Вызовите функцию при запуске программы:
 ```python
 if \_\_name\_\_ == "\_\_main\_\_":
 connect\_db()
 ```

### 4. Создание графического интерфейса

1. Настройте основное окно:
 ```python
 root = Tk()
 root.title("Работа с базой данных")
 root.geometry("400x400")
 ```

2. Добавьте элементы ввода данных:
 ```python
 Label(root, text="Имя").grid(row=0, column=0)
 name\_entry = Entry(root)
 name\_entry.grid(row=0, column=1)

 Label(root, text="Возраст").grid(row=1, column=0)
 age\_entry = Entry(root)
 age\_entry.grid(row=1, column=1)
 ```

### 5. Реализация операций с базой данных

* \*\*Добавление записи:\*\*

```python
def add\_user():
 name = name\_entry.get()
 age = age\_entry.get()

 if name and age:
 conn = sqlite3.connect("app\_database.db")
 cursor = conn.cursor()
 cursor.execute("INSERT INTO users (name, age) VALUES (?, ?)", (name, age))
 conn.commit()
 conn.close()
 messagebox.showinfo("Успех", "Пользователь добавлен!")
 name\_entry.delete(0, END)
 age\_entry.delete(0, END)
 view\_users()
 else:
 messagebox.showerror("Ошибка", "Все поля должны быть заполнены!")
```

* \*\*Отображение записей:\*\*

```python
def view\_users():
 conn = sqlite3.connect("app\_database.db")
 cursor = conn.cursor()
 cursor.execute("SELECT \* FROM users")
 rows = cursor.fetchall()
 conn.close()

 # Очищаем список перед обновлением
 listbox.delete(0, END)
 for row in rows:
 listbox.insert(END, f"ID: {row[0]}, Имя: {row[1]}, Возраст: {row[2]}")
```

* \*\*Удаление записи:\*\*

```python
def delete\_user():
 selected\_item = listbox.curselection()
 if selected\_item:
 user\_id = listbox.get(selected\_item).split(",")[0].split(":")[1].strip()
 conn = sqlite3.connect("app\_database.db")
 cursor = conn.cursor()
 cursor.execute("DELETE FROM users WHERE id=?", (user\_id,))
 conn.commit()
 conn.close()
 messagebox.showinfo("Успех", "Пользователь удален!")
 view\_users()
 else:
 messagebox.showerror("Ошибка", "Выберите запись для удаления!")
```

* \*\*Обновление записи:\*\*

```python
def update\_user():
 selected\_item = listbox.curselection()
 if selected\_item:
 user\_id = listbox.get(selected\_item).split(",")[0].split(":")[1].strip()
 name = name\_entry.get()
 age = age\_entry.get()

 if name and age:
 conn = sqlite3.connect("app\_database.db")
 cursor = conn.cursor()
 cursor.execute("UPDATE users SET name=?, age=? WHERE id=?", (name, age, user\_id))
 conn.commit()
 conn.close()
 messagebox.showinfo("Успех", "Данные обновлены!")
 name\_entry.delete(0, END)
 age\_entry.delete(0, END)
 view\_users()
 else:
 messagebox.showerror("Ошибка", "Все поля должны быть заполнены!")
 else:
 messagebox.showerror("Ошибка", "Выберите запись для обновления!")
```

### 6. Добавление кнопок и списка

1. Добавьте кнопки для выполнения операций:
 ```python
 Button(root, text="Добавить", command=add\_user).grid(row=2, column=0, pady=10)
 Button(root, text="Удалить", command=delete\_user).grid(row=2, column=1)
 Button(root, text="Обновить", command=update\_user).grid(row=2, column=2)
 ```

2. Добавьте список для отображения записей:
 ```python
 listbox = Listbox(root, width=50)
 listbox.grid(row=3, column=0, columnspan=3, pady=10)
 ```

3. Загрузите записи при запуске:
 ```python
 view\_users()
 ```

4. Запустите основной цикл приложения:
 ```python
 root.mainloop()
 ```

### 7. Итог

После выполнения работы вы:
- Создали базу данных SQLite.
- Подключили базу данных к десктопному приложению.
- Реализовали функции добавления, удаления, обновления и отображения данных.
- Построили графический интерфейс с использованием Tkinter.

### Дополнительное задание

1. Подключите базу данных MySQL или PostgreSQL вместо SQLite.
2. Добавьте поле для поиска по имени.
3. Улучшите интерфейс с использованием библиотеки `ttk` для более стильных элементов управления.