1. Opis funkcjonalny

Aplikacja jest prostym systemem do zarządzania bazą danych zawierającą informacje o osobach. Główne funkcjonalności aplikacji obejmują:

Dodawanie nowych rekordów (osób) do bazy danych.

Usuwanie istniejących rekordów z bazy danych.

Edytowanie istniejących rekordów.

Przeglądanie listy rekordów.

Wyszukiwanie rekordów po imieniu i nazwisku.

Sortowanie rekordów według różnych kryteriów.

Filtrowanie rekordów po wieku.

2. Opis struktury bazy danych

Aplikacja wykorzystuje bazę danych SQLite o nazwie "moja_baza_danych.db". Baza danych zawiera jedną tabelę o nazwie "moja_tabela" o następującej strukturze:

id (INTEGER) - Unikalny identyfikator rekordu.

imie (TEXT) - Imię osoby.

nazwisko (TEXT) - Nazwisko osoby.

wiek (INTEGER) - Wiek osoby.

3. Diagram ERD

4. Kod SQL tworzący bazę danych

```sql
CREATE TABLE IF NOT EXISTS moja_tabela (
    id INTEGER PRIMARY KEY,
    imie TEXT,
    nazwisko TEXT,
    wiek INTEGER
);
```

5. Kod SQL wypełniający bazę danych przykładowymi danymi

```
import sqlite3
import random

# Lista przykładowych imion i nazwisk
imiona = ["Jan", "Anna", "Piotr", "Marta", "Tomasz", "Katarzyna",
"Marcin", "Magdalena"]
nazwiska = ["Kowalski", "Nowak", "Zieliński", "Wiśniewska", "Lis",
"Szymański", "Wójcik", "Kowalczyk"]

# Tworzenie lub łączenie z bazą danych
connection = sqlite3.connect('moja_baza_danych.db')
cursor = connection.cursor()

# Tworzenie tabeli, jeśli nie istnieje
cursor.execute('''CREATE TABLE IF NOT EXISTS moja_tabela (
                  id INTEGER PRIMARY KEY,
                  imie TEXT,
                  nazwisko TEXT,
                  wiek INTEGER)''')

# Wypełnianie tabeli losowymi danymi
for _ in range(10):  # Możesz zmienić ilość rekordów
    imie = random.choice(imiona)
    nazwisko = random.choice(nazwiska)
    wiek = random.randint(18, 70)  # Losowy wiek od 18 do 70 lat

    cursor.execute("INSERT INTO moja_tabela (imie, nazwisko, wiek) VALUES
(?, ?, ?)", (imie, nazwisko, wiek))

connection.commit()
connection.close()

print("Baza danych została wypełniona losowymi danymi.")
```

6. Kod źródłowy aplikacji wraz z komentarzami

```
# Importowanie niezbędnych modułów
import tkinter as tk
from tkinter import ttk
import sqlite3

# Funkcja do dodawania rekordów do bazy danych
def dodaj_rekord():
    imie = entry_imie.get()
    nazwisko = entry_nazwisko.get()
    wiek = entry_wiek.get()

    # Dodawanie rekordu do bazy danych, jeśli wiek >= 18
    if int(wiek) >= 18:
        cursor.execute("INSERT INTO moja_tabela (imie, nazwisko, wiek)
VALUES (?, ?, ?)", (imie, nazwisko, wiek))
        connection.commit()
        aktualizuj_liste()

# Funkcja do usuwania rekordów z bazy danych
def usun_rekord():
    selected_item = treeview.focus()
    if selected_item:
        rekord_name = treeview.item(selected_item)['values']
        rekord_id = rekord_name[0]

        cursor.execute("DELETE FROM moja_tabela WHERE id=?",
(rekord_id,))
        connection.commit()
        aktualizuj_liste()

# Funkcja do edytowania rekordów w bazie danych
def edytuj_rekord():
    selected_item = treeview.focus()
    if selected_item:
        rekord_name = treeview.item(selected_item)['values']
        rekord_id = rekord_name[0]
        imie = entry_imie.get()
        nazwisko = entry_nazwisko.get()
        wiek = entry_wiek.get()

        cursor.execute("UPDATE moja_tabela SET imie=?, nazwisko=?, wiek=?
WHERE id=?", (imie, nazwisko, wiek, rekord_id))
        connection.commit()
        aktualizuj_liste()

# Funkcja do wybierania rekordów z listy
def wybierz_rekord(event):
    selected_item = treeview.focus()
    if selected_item:
        rekord_name = treeview.item(selected_item)['values']
        rekord_id = rekord_name[0]

        # Pobieranie i wyświetlanie rekordu z bazy danych
```

```
        cursor.execute("SELECT * FROM moja_tabela WHERE id=?",
(rekord_id,))
        rekord = cursor.fetchone()
        if rekord:
            entry_imie.delete(0, tk.END)
            entry_nazwisko.delete(0, tk.END)
            entry_wiek.delete(0, tk.END)
            entry_imie.insert(0, rekord[1])
            entry_nazwisko.insert(0, rekord[2])
            entry_wiek.insert(0, rekord[3])

# Funkcja do wyszukiwania rekordów w bazie danych
def wyszukaj_rekordy():
    imie = entry_wyszukaj_imie.get()
    nazwisko = entry_wyszukaj_nazwisko.get()
    cursor.execute("SELECT * FROM moja_tabela WHERE imie LIKE ? AND
nazwisko LIKE ?", ('%' + imie + '%', '%' + nazwisko + '%'))
    wyniki = cursor.fetchall()
    aktualizuj_liste(wyniki)

# Funkcja do sortowania rekordów
def sortuj_rekordy():
    kolumna = combo_sortowanie.get()
    cursor.execute(f"SELECT * FROM moja_tabela ORDER BY {kolumna}")
    wyniki = cursor.fetchall()
    aktualizuj_liste(wyniki)

# Funkcja do filtrowania rekordów po wieku
def filtruj_rekordy():
    wiek_min = entry_filtr_wiek_min.get()
    wiek_max = entry_filtr_wiek_max.get()
    cursor.execute("SELECT * FROM moja_tabela WHERE wiek BETWEEN ? AND
?", (wiek_min, wiek_max))
    wyniki = cursor.fetchall()
    aktualizuj_liste(wyniki)

# Funkcja do aktualizacji listy rekordów w widoku
def aktualizuj_liste(wyniki=None):
    if not wyniki:
        cursor.execute("SELECT * FROM moja_tabela")
        wyniki = cursor.fetchall()
    treeview.delete(*treeview.get_children())

    for rekord in wyniki:
        treeview.insert('', 'end', values=rekord)

# Inicjalizacja głównego okna
root = tk.Tk()
root.title("Aplikacja do zarządzania bazą danych")

# Inicjalizacja bazy danych
connection = sqlite3.connect('moja_baza_danych.db')
cursor = connection.cursor()
```

```
# Tworzenie tabeli, jeśli nie istnieje
cursor.execute('''CREATE TABLE IF NOT EXISTS moja_tabela
                  (id INTEGER PRIMARY KEY, imie TEXT, nazwisko TEXT, wiek
INTEGER)''')
connection.commit()

# Tworzenie widżetów interfejsu użytkownika

# ... (reszta kodu)

root.mainloop()
```

7. Testy jednostkowe wybranych fragmentów kodu

```
import unittest

class TestMojaAplikacja(unittest.TestCase):
    def test_dodaj_rekord(self):
        connection = sqlite3.connect(':memory:')  # Tworzenie tymczasowej
bazy danych w pamięci
        cursor = connection.cursor()

        # Tworzenie tabeli
        cursor.execute('''CREATE TABLE IF NOT EXISTS moja_tabela
                          (id INTEGER PRIMARY KEY, imie TEXT, nazwisko
TEXT, wiek INTEGER)''')

        # Dodawanie rekordu
        dodaj_rekord()

        # Sprawdzenie czy rekord został dodany
        cursor.execute("SELECT COUNT(*) FROM tabela")
        count = cursor.fetchone()[0]
        self.assertEqual(count, 1)

        connection.close()

    def test_wyszukaj_rekordy(self):
        connection = sqlite3.connect(':memory:')  # Tworzenie tymczasowej
bazy danych w pamięci
        cursor = connection.cursor()

        # Tworzenie tabeli
        cursor.execute('''CREATE TABLE IF NOT EXISTS tabela
                          (id INTEGER PRIMARY KEY, imie TEXT, nazwisko
TEXT, wiek INTEGER)''')

        # Dodawanie rekordów
        cursor.execute("INSERT INTO moja_tabela (imie, nazwisko, wiek)
VALUES (?, ?, ?)", ('Jan', 'Kowalski', 30))
        cursor.execute("INSERT INTO moja_tabela (imie, nazwisko, wiek)
VALUES (?, ?, ?)", ('Anna', 'Nowak', 25))

        # Wyszukiwanie rekordów
        entry_wyszukaj_imie.insert(0, 'Jan')
        entry_wyszukaj_nazwisko.insert(0, 'Kowalski')
        wyszukaj_rekordy()

        # Sprawdzenie czy wyniki są poprawne
        cursor.execute("SELECT COUNT(*) FROM tabela")
        count = cursor.fetchone()[0]
        self.assertEqual(count, 1)

        connection.close()

if __name__ == '__main__':
    unittest.main()
```

8. Instrukcja instalacji i uruchomienia

Aby uruchomić aplikację, wykonaj następujące kroki:

Zainstaluj bibliotekę Tkinter, sqlite3 oraz random

Uruchom plik Main.py w interpreterze Pythona

9. Instrukcje użytkowania

Po uruchomieniu aplikacji:

Możesz dodawać nowe rekordy (osoby), wpisując dane do pól "Imie", "Nazwisko" i "Wiek", a następnie klikając przycisk "Dodaj".

Możesz usunąć wybrany rekord, zaznaczając go na liście i klikając przycisk "Usuń".

Możesz edytować wybrany rekord, zaznaczając go na liście, wprowadzając zmiany w polach "Imie", "Nazwisko" i "Wiek", a następnie klikając przycisk "Edytuj".

Możesz wyszukiwać rekordy po imieniu i nazwisku, wpisując odpowiednie dane do pól "Wyszukaj Imię" i "Wyszukaj Nazwisko", a następnie klikając przycisk "Wyszukaj".

Możesz sortować rekordy według wybranej kolumny, wybierając kolumnę z rozwijanej listy "Sortuj" i klikając przycisk "Sortuj".

Możesz filtrować rekordy po wieku, wpisując minimalny i maksymalny wiek do pól "Filtr wieku" i klikając przycisk "Filtruj".