МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Вятский государственный университет»**

**(ФГБОУ ВО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

**«Базы данных»**

Связывание приложения на Python с базой данных под управлением PostgreSQL

Выполнил студент группы ИВТб-31 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Кудяшев Я.Ю./

(подпись)

Проверил преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Клюкин В.Л/

(подпись)

Киров 2022

# Задание и цель лабораторной работы

Цель работы:

* Познакомиться со стандартным API в python для связывания приложения с БД
* Изучить некоторые шаблоны проектирования, связанные с работой с БД
* Освоить на практике основы взаимодействия с БД под управлением PostgreSQL в приложении на python

Задание на лабораторную работу:

Создать приложение с графическим приложением на языке python. Приложение должно использовать БД, разработанную в предыдущих лабораторных работах.

Для любой одной таблицы, которая содержит внешний ключ на другую таблицу, приложение должно выполнять следующее:

* Выводить строки таблицы
* Предоставлять любой фильтр по значениям строк. (Например, «Дата с … по …» или «Имя содержит …»)
* Предоставлять возможность добавления новых строк
* Предоставлять возможность удаления строки

Требования к реализации:

* Заголовки должны быть осмысленными. Например, вместо «name» в таблице должен быть заголовок «Имя»
* При добавлении новой строки внешний ключ выбирается из списка
* Сохранение или удаление строки должно быть реализовано с помощью функции PL/pgSQL
* Фильтрация значений при поиске должна производиться через запрос, а не в полученной коллекции
* Разрешается использование любого фреймворка
* При разработке нужно использовать шаблоны проектирования, связанные с работой с БД

# Экранные формы

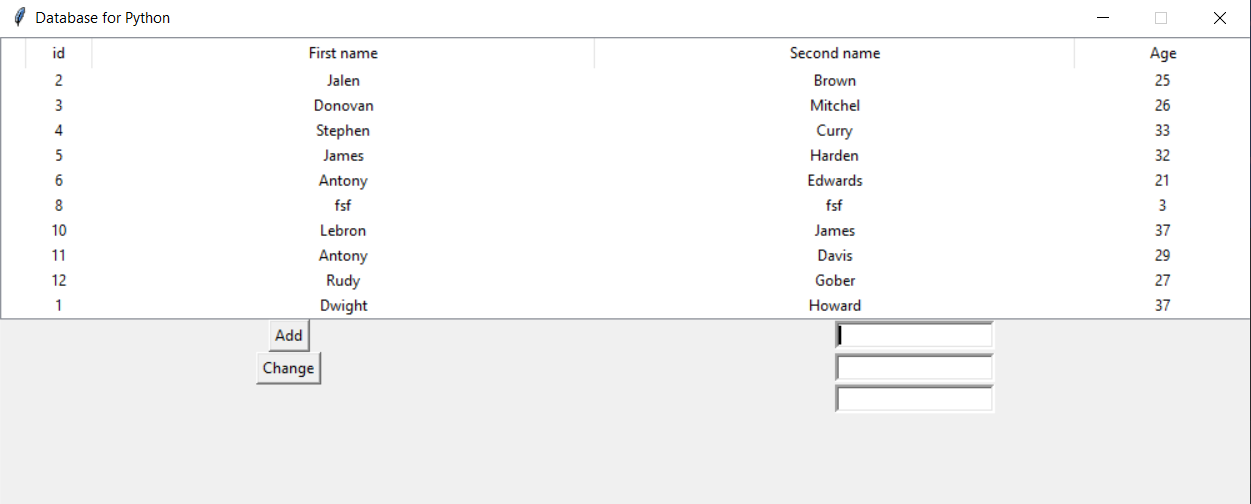


Рисунок 1 – Пример отображения базы данных в приложении

# Код программы

import another

from tkinter import \*

from tkinter import ttk

lst = [("","","",""),

("","","",""),

("","","",""),

("","","",""),

("","","",""),

("","","",""),

("","","",""),

("","","",""),

("","","",""),

("","","",""),

("","","","")]

another.openСonnection()

window = Tk()

window.title("Database for Python")

window.geometry('1000x500')

window.resizable(width=False, height=False) #Unchanchable window

window.update\_idletasks() #Window in center

w, h = window.winfo\_width(), window.winfo\_height()

window.geometry(f"+{(window.winfo\_screenwidth()-w)//2}+{(window.winfo\_screenheight()-h)//2}")

frame\_list = Frame(window, bg='blue')

frame\_list.grid(column=0,row=0,columnspan=2,sticky="we")

first\_text = StringVar(window)

second\_text = StringVar(window)

third\_text = StringVar(window)

heads = ['id', 'First name', 'Second name', 'Age']

table = ttk.Treeview(frame\_list)

table['columns'] = heads

for header in heads:

table.heading(header, text=header, anchor='center')

table.column(header, anchor='center')

for row in lst:

table.insert('', END, values=row)

btn3 = Entry(window, text="0", width=20, bd=3,textvariable=third\_text)

btn3.grid(column=1, row=4, columnspan=1)

btn6 = Button(window, text="Add")

# btn6.pack()

# btn6.place(x=300, y=440)

btn6.grid(column=0, row=2, columnspan=1)

btn6.bind('<Button-1>', another.add)

btn7 = Button(window, text="Change")

# btn7.pack()

# btn7.place(x=300, y=440)

btn7.grid(column=0, row=3, columnspan=1)

btn7.bind('<Button-1>', another.change)

btn5 = Entry(window, text="0", width=20, bd=3,textvariable=first\_text)

btn5.grid(column=1, row=2, columnspan=1)

btn4 = Entry(window, text="0", width=20, bd=3,textvariable=second\_text)

btn4.grid(column=1, row=3, columnspan=1)

table.pack(expand=YES, fill=BOTH)

#Components

#подключение к базе данных

window.mainloop()

#another.closeConnection()

import psycopg2

from psycopg2 import Error #для обработки исключений

from psycopg2.extensions import ISOLATION\_LEVEL\_AUTOCOMMIT

def openСonnection():

try:

connection = psycopg2.connect(user="postgres",

password="123123",

host="localhost",

port="5432",

database="lab")

# connection.set\_isolation\_level(ISOLATION\_LEVEL\_AUTOCOMMIT)

# Курсор для выполнения операций с базой данных

cursor = connection.cursor()

cursor.execute("SELECT id,first\_name,second\_name,age FROM public.players")

import database

database.lst = cursor.fetchall()

# Распечатать сведения о PostgreSQL

print("Информация о сервере PostgreSQL")

print(connection.get\_dsn\_parameters(), "\n")

except (Exception, Error) as error:

print("Ошибка при работе с PostgreSQL", error)

def add(self):

connection = psycopg2.connect(user="postgres",

password="galinka220",

host="localhost",

port="5432",

database="lab")

# connection.set\_isolation\_level(ISOLATION\_LEVEL\_AUTOCOMMIT)

# Курсор для выполнения операций с базой данных

cursor = connection.cursor()

import database

record\_to\_insert = (str(database.btn5.get()),str(database.btn4.get()),int(database.btn3.get()))

cursor.execute("INSERT INTO players(first\_name, second\_name,id\_teams,age) VALUES(%s, %s, 1,%s)",record\_to\_insert)

cursor.execute("SELECT id,first\_name,second\_name,age FROM public.players")

database.lst = cursor.fetchall()

def insertGroup(self, name, number\_of\_students):

self.\_connection.query("INSERT INTO group(name, number\_of\_students) VALUES(%s, %s)" % (name,number\_of\_students))

def change():

print()

1. **Вывод**

В ходе выполнения лабораторной работы были изучены стандартные API в python для связывания приложения с БД**,** изучены некоторые шаблоны проектирования, связанные с работой с БД и освоены на практике основы взаимодействия с БД под управлением PostgreSQL в приложении на python.