

## Практическое занятие №16

**Тема:** Разработка многооконного приложения для работы с однотабличной БД в IDE PyCharm Community

**Цель:** закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, работы с БД в IDE PyCharm Community

**Постановка задачи**

1.

Приложение СТРАХОВАЯ КОМПАНИЯ для некоторой организации. БД должна содержать таблицу Договор со следующей структурой записи: дата заключения, страховая сумма, вид страхования, тарифная ставка и филиал, в котором заключался договор. БД должна обеспечивать получение информации о договорах по страховой сумме.

**Текст программы:**

1.

```
# Приложение СТРАХОВАЯ КОМПАНИЯ для некоторой организации. БД должна содержать
таблицу Договор со следующей структурой
# записи: дата заключения, страховая сумма, вид страхования, тарифная ставка и
филиал, в котором заключался договор.
# БД должна обеспечивать получение информации о договорах по страховой сумме.
```

```
import tkinter as tk
from tkinter import ttk
import sqlite3 as sq
from tkcalendar import DateEntry

class Main(tk.Frame):

    """Класс для главного окна"""

    def __init__(self, root):
        super().__init__(root)
        self.init_main()
        self.db = db
        self.view_records()

    def init_main(self):
        toolbar = tk.Frame(bg='#a0dea0', bd=4)
        toolbar.pack(side=tk.TOP, fill=tk.X)

        self.add_img = tk.PhotoImage(file="pres.gif")
        self.btn_open_dialog = tk.Button(toolbar, text='Добавить договор',
command=self.open_dialog, bg='#5da130', bd=0,
compound=tk.TOP, image=self.add_img)
        self.btn_open_dialog.pack(side=tk.LEFT)

        self.update_img = tk.PhotoImage(file="red.gif")
        btn_edit_dialog = tk.Button(toolbar, text="Редактировать",
command=self.open_update_dialog, bg='#5da130',
bd=0, compound=tk.TOP,
image=self.update_img)
        btn_edit_dialog.pack(side=tk.LEFT)

        self.delete_img = tk.PhotoImage(file="bin.gif")
        btn_delete = tk.Button(toolbar, text="Удалить договор",
command=self.delete_records, bg='#5da130',
bd=0, compound=tk.TOP,
image=self.delete_img)
        btn_delete.pack(side=tk.LEFT)
```

```

        self.search_img = tk.PhotoImage(file="sea.gif")
        btn_search = tk.Button(toolbar, text="Поиск договора",
command=self.open_search_dialog, bg='#5da130',
                                bd=0, compound=tk.TOP, image=self.search_img)
        btn_search.pack(side=tk.LEFT)

        self.refresh_img = tk.PhotoImage(file="upd.gif")
        btn_refresh = tk.Button(toolbar, text="Обновить экран",
command=self.view_records, bg='#5da130',
                                bd=0, compound=tk.TOP, image=self.refresh_img)
        btn_refresh.pack(side=tk.LEFT)

        self.tree = ttk.Treeview(self, columns=('id', 'date', 'sum', 'insurance',
'bid', 'filial'), height=15, show='headings')

        self.tree.column('id', width=170, anchor=tk.CENTER)
        self.tree.column('date', width=180, anchor=tk.CENTER)
        self.tree.column('sum', width=180, anchor=tk.CENTER)
        self.tree.column('insurance', width=250, anchor=tk.CENTER)
        self.tree.column('bid', width=140, anchor=tk.CENTER)
        self.tree.column('filial', width=140, anchor=tk.CENTER)

        self.tree.heading('id', text='ID')
        self.tree.heading('date', text='Дата заключения')
        self.tree.heading('sum', text='Страховая сумма')
        self.tree.heading('insurance', text='Вид страхования')
        self.tree.heading('bid', text='Тарифная ставка')
        self.tree.heading('filial', text='Филиал')

        self.tree.pack()

    def records(self, id, date, sum, insurance, bid, filial):
        self.db.insert_data( id, date, sum, insurance, bid, filial)
        self.view_records()

    def update_record(self, id, date, sum, insurance, bid, filial):
        self.db.cur.execute("""UPDATE users SET id=?, date=?, sum=?,
insurance=?, bid=? WHERE id=?""",
                                (id, date, sum, insurance, bid, filial,
self.tree.set(self.tree.selection()[0], '#1'))))
        self.db.con.commit()
        self.view_records()

    def view_records(self):
        self.db.cur.execute("""SELECT * FROM users""")
        [self.tree.delete(i) for i in self.tree.get_children()]
        [self.tree.insert('', 'end', values=row) for row in
self.db.cur.fetchall()]

    def delete_records(self):
        for selection_item in self.tree.selection():
            self.db.cur.execute("""DELETE FROM users WHERE id=?""",
(self.tree.set(selection_item, '#1'),))
            self.db.con.commit()
            self.view_records()

    # def search_records(self, user_id):
    #     user_id = ("% " + user_id + "%",)
    #     self.db.cur.execute("""SELECT * FROM users WHERE name LIKE ?""",
user_id)
    #     [self.tree.delete(i) for i in self.tree.get_children()]
    #     [self.tree.insert('', 'end', values=row) for row in
self.db.cur.fetchall()]

    def search_records(self, bid):
        bid = (bid,)
        self.db.cur.execute("""SELECT * FROM users WHERE bid >= ?""", bid)

```

```
[self.tree.delete(i) for i in self.tree.get_children()]
[self.tree.insert('', 'end', values=row) for row in
self.db.cur.fetchall()]

def open_dialog(self):
    Child(root, app)

def open_update_dialog(self):
    Update()

def open_search_dialog(self):
    Search()

class Child(tk.Toplevel):

    """Класс для дочернего окна"""

    def __init__(self, root, app):
        super().__init__(root)
        self.init_child()
        self.view = app

    def init_child(self):
        self.title('Добавить договор')
        self.geometry('600x320+400+300')
        self.resizable(False, False)

        label_description_id = tk.Label(self, text='ID')
        label_description_id.place(x=50, y=15)
        self.entry_description_id = ttk.Entry(self)
        self.entry_description_id.place(x=210, y=15)

        label_description = tk.Label(self, text='Дата заключения')
        label_description.place(x=50, y=40)
        self.entry_description = DateEntry(self, width = 12)
        self.entry_description.place(x=210, y=40)

        label_name = tk.Label(self, text='Страховая сумма')
        label_name.place(x=50, y=75)
        self.entry_name = ttk.Entry(self)
        self.entry_name.place(x=210, y=75)

        label_sex = tk.Label(self, text='Вид страхования')
        label_sex.place(x=50, y=100)
        self.combobox = ttk.Combobox(self, values=[u'Страхование
предпринимательских рисков', u'Имущественное страхование', u'Личное
страхование', u'Страхование ответственности'], width=40)
        self.combobox.current(0)
        self.combobox.place(x=210, y=100)

        label_old = tk.Label(self, text='Тарифная ставка')
        label_old.place(x=50, y=125)
        self.entry_old = ttk.Entry(self)
        self.entry_old.place(x=210, y=125)

        label_score = tk.Label(self, text='Филиал')
        label_score.place(x=50, y=150)
        self.entry_score = ttk.Entry(self)
        self.entry_score.place(x=210, y=150)

        btn_cancel = ttk.Button(self, text='Закрыть', command=self.destroy)
        btn_cancel.place(x=300, y=175)

        self.btn_ok = ttk.Button(self, text='Добавить')
        self.btn_ok.place(x=220, y=175)
        self.btn_ok.bind('<Button-1>', lambda event:
```

```
self.view.records(self.entry_description_id.get(),
self.entry_description.get(),
self.entry_name.get(),
self.combobox.get(),
self.entry_old.get(),
self.entry_score.get()))

        self.grab_set()
        self.focus_set()

class Update(Child):
    def __init__(self):
        super().__init__(root, app)
        self.init_edit()
        self.view = app

    def init_edit(self):
        self.title("Редактировать запись")
        btn_edit = ttk.Button(self, text="Редактировать")
        btn_edit.place(x=205, y=170)
        btn_edit.bind('<Button-1>', lambda event:
self.view.update_record(self.entry_description_id.get(),
self.entry_description.get(),
self.entry_name.get(),
self.combobox.get(),
self.entry_old.get(),
self.entry_score.get()))
        self.btn_ok.destroy()

class Search(tk.Toplevel):
    def __init__(self):
        super().__init__()
        self.init_search()
        self.view = app

    def init_search(self):
        self.title("Поиск по тарифной ставке")
        self.geometry("400x100+400+300")
        self.resizable(False, False)

        label_search = tk.Label(self, text="Поиск")
        label_search.place(x=50, y=20)

        self.entry_search = ttk.Entry(self)
        self.entry_search.place(x=105, y=20, width=150)

        btn_cancel = ttk.Button(self, text="Заккрыть", command=self.destroy)
        btn_cancel.place(x=185, y=50)

        btn_search = ttk.Button(self, text="Поиск")
        btn_search.place(x=105, y=50)
        btn_search.bind('<Button-1>', lambda event:
self.view.search_records(self.entry_search.get()))
        btn_search.bind('<Button-1>', lambda event: self.destroy(), add='+')

class DB:
    def __init__(self):
```

```
with sq.connect('company.db') as self.con:
    self.cur = self.con.cursor()
    self.cur.execute("""CREATE TABLE IF NOT EXISTS users (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        date DATETIME,
        sum INTEGER NOT NULL,
        insurance TEXT NOT NULL,
        bid INTEGER,
        filial TEXT
    ) """)

def insert_data(self, id, date, sum, insurance, bid, filial):
    self.cur.execute("""INSERT INTO users(id, date,sum, insurance, bid,
filial) VALUES (?, ?, ?, ?, ?, ?)""",
                        (id, date, sum, insurance, bid, filial))
    self.con.commit()

if __name__ == "__main__":
    root = tk.Tk()
    db = DB()
    app = Main(root)
    app.pack()
    root.title("Работа с базой данных страховой компании")
    root.geometry("950x450+300+200")
    root.resizable(False, False)
    root.mainloop()
```

## Протокол работы программы:

1.

Работа с базой данных страховой компании

ID	Дата заключения	Страховая сумма	Вид страхования	Тарифная ставка
1	01/02/2019	33	Страхование предпринимательских рисков	4
2	02/03/2020	4352	Имущественное страхование	45

Работа с базой данных страховой компании

+

Добавить договор

Редактировать

Удалить договор

Поиск договора

Обновить экран

ID	Дата заключения	Страховая сумма	Вид страхования	Тарифная ставка	
1	01/02/2019	33	Страхование предпринимательских рисков	4	
2	02/03/2020	4352	Имущественное страхование	45	

Поиск по тарифной ставке

Поиск

45

Поиск

Заккрыть

Работа с базой данных страховой компании

+

Добавить договор

Редактировать

Удалить договор

Поиск договора

Обновить экран

ID	Дата заключения	Страховая сумма	Вид страхования	Тарифная ставка	Филиал
4	5/4/22	45	Страхование предпринимательских рисков	567	fssfa
5	3/9/22	3456	Личное страхование	433	eflasd

Редактировать запись

ID

6

Дата заключения

5/4/22

Страховая сумма

567

Вид страхования

Страхование предпринимательских рисков

Тарифная ставка

456

Филиал

dgs

Редактировать

Заккрыть

Работа с базой данных страховой компании

+

Добавить договор

Редактировать

Удалить договор

Поиск договора

Обновить экран

ID	Дата заключения	Страховая сумма	Вид страхования	Тарифная ставка	Филиал
4	5/4/22	45	Страхование предпринимательских рисков	567	fssfa
6	5/4/22	567	Страхование предпринимательских рисков	456	dgs

**Вывод:** в процессе выполнения практического занятия выработала навыки составления программ с применением ткинтера, БД и знаний ООП в IDE PyCharm Community. Выполнены разработка кода, отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHub.