**Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации**

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

**Московский технический университет связи и информатики**

Кафедра «Информатики»

Практическая работа №8 по дисциплине ВвИТ

«Бот UI»

Выполнил студент группы БИН2003 Беляков А.Н.

Проверил: Егор Аршинов

Москва 2021

**1. Цель работы:** создать свое приложение для удобной работы с sql таблицами

**2.** **Выполнение работы:**

import sys  
import psycopg2  
from PyQt5.QtWidgets import (QApplication, QWidget, QTabWidget, QAbstractScrollArea, QVBoxLayout, QHBoxLayout,  
 QTableWidget, QGroupBox, QTableWidgetItem, QPushButton, QMessageBox)  
  
  
class Window(QWidget):  
 def \_\_init\_\_(self):  
 super(Window, self).\_\_init\_\_()  
  
 self.setWindowTitle('Shedule')  
 self.\_connect\_to\_db() ## подключение к базе  
  
 self.vbox = QHBoxLayout(self) ## лейаут для выравнивания  
  
 self.tabs = QTabWidget(self) ## добавили виджеты  
 self.vbox.addWidget(self.tabs)  
  
 self.update\_shedule\_button = QPushButton('Update')  
 self.vbox.addWidget(self.update\_shedule\_button)  
  
 self.update\_shedule\_button.clicked.connect(self.\_update\_shedule)  
  
 self.\_create\_teacher\_tab()  
 self.\_create\_subject\_tab()  
 self.\_create\_shedule\_tab()  
 self.\_create\_shedule\_tab1()  
 self.\_create\_shedule\_tab2()  
 self.\_create\_shedule\_tab3()  
 self.\_create\_shedule\_tab4()  
  
 def \_connect\_to\_db(self):  
 self.conn = psycopg2.connect(database="timetable",  
 user="postgres",  
 password="1488",  
 host="localhost",  
 port="5432")  
 self.cursor = self.conn.cursor()  
  
 def \_create\_teacher\_tab(self):  
 self.t\_tab = QWidget()  
 self.tabs.addTab(self.t\_tab, 'Teachers')  
  
 self.t\_gbox = QGroupBox('Teachers')  
  
 self.svbox = QVBoxLayout()  
 self.shbox1 = QVBoxLayout() ## выравнивание по осям  
 self.shbox2 = QHBoxLayout()  
  
  
  
 self.shbox1.addWidget(self.t\_gbox)  
  
 self.\_create\_t\_table()  
  
 self.t\_tab.setLayout(self.svbox)  
  
 def \_create\_subject\_tab(self):  
 self.sub\_tab = QWidget()  
 self.tabs.addTab(self.sub\_tab, 'Subjects')  
  
 self.sub\_gbox = QGroupBox('Subjects')  
  
 self.svbox = QVBoxLayout()  
 self.shbox1 = QVBoxLayout() ## выравнивание по осям  
 self.shbox2 = QHBoxLayout()  
  
  
  
 self.shbox1.addWidget(self.sub\_gbox)  
  
 self.\_create\_sub\_table()  
  
 self.sub\_tab.setLayout(self.svbox)  
  
 def \_create\_shedule\_tab(self):  
 self.shedule\_tab = QWidget()  
 self.tabs.addTab(self.shedule\_tab, 'Shedule')  
  
 self.monday\_gbox = QGroupBox('Monday') ## рамка для таблицы  
  
 self.svbox = QVBoxLayout()  
 self.shbox1 = QVBoxLayout() ## выравнивание по осям  
 self.shbox2 = QHBoxLayout()  
  
 self.svbox.addLayout(self.shbox1) ## добавление 2х в 1  
 self.svbox.addLayout(self.shbox2)  
  
 self.shbox1.addWidget(self.monday\_gbox)  
  
 self.\_create\_monday\_table()  
  
 self.shedule\_tab.setLayout(self.svbox)  
  
 def \_create\_shedule\_tab1(self):  
 self.tuesday\_gbox = QGroupBox('Tuesday') ## рамка для таблицы  
  
 self.shbox1.addWidget(self.tuesday\_gbox)  
  
 self.\_create\_tuesday\_table()  
  
 def \_create\_shedule\_tab2(self):  
 self.wednesday\_gbox = QGroupBox('Wednesday') ## рамка для таблицы  
  
 self.shbox1.addWidget(self.wednesday\_gbox)  
  
 self.\_create\_wednesday\_table()  
  
 def \_create\_shedule\_tab3(self):  
 self.thursday\_gbox = QGroupBox('Thursday') ## рамка для таблицы  
  
 self.shbox1.addWidget(self.thursday\_gbox)  
  
 self.\_create\_thursday\_table()  
  
 def \_create\_shedule\_tab4(self):  
 self.friday\_gbox = QGroupBox('Friday') ## рамка для таблицы  
  
 self.shbox1.addWidget(self.friday\_gbox)  
  
 self.\_create\_friday\_table()  
  
 self.shedule\_tab.setLayout(self.svbox)  
  
 def \_create\_t\_table(self):  
 self.t\_table = QTableWidget()  
 self.t\_table.setSizeAdjustPolicy(QAbstractScrollArea.AdjustToContents)  
  
 self.t\_table.setColumnCount(5)  
 self.t\_table.setHorizontalHeaderLabels(['id', 'Full name', 'Subject', '', ''])  
  
 self.\_update\_t\_table()  
  
 self.svbox = QVBoxLayout()  
 self.svbox.addWidget(self.t\_table)  
 self.t\_gbox.setLayout(self.svbox)  
  
 def \_create\_sub\_table(self):  
 self.sub\_table = QTableWidget()  
 self.sub\_table.setSizeAdjustPolicy(QAbstractScrollArea.AdjustToContents)  
  
 self.sub\_table.setColumnCount(4)  
 self.sub\_table.setHorizontalHeaderLabels(['id', 'Subject', '', ''])  
  
 self.\_update\_sub\_table()  
  
 self.svbox = QVBoxLayout()  
 self.svbox.addWidget(self.sub\_table)  
 self.sub\_gbox.setLayout(self.svbox)  
  
 def \_create\_monday\_table(self):  
 self.monday\_table = QTableWidget()  
 self.monday\_table.setSizeAdjustPolicy(QAbstractScrollArea.AdjustToContents)  
  
 self.monday\_table.setColumnCount(6)  
 self.monday\_table.setHorizontalHeaderLabels(['id', 'Subject', 'Room', 'Time', '', ''])  
  
 self.\_update\_monday\_table()  
  
 self.mvbox = QVBoxLayout()  
 self.mvbox.addWidget(self.monday\_table)  
 self.monday\_gbox.setLayout(self.mvbox)  
  
 def \_create\_tuesday\_table(self):  
 self.tuesday\_table = QTableWidget()  
 self.tuesday\_table.setSizeAdjustPolicy(QAbstractScrollArea.AdjustToContents)  
  
 self.tuesday\_table.setColumnCount(6)  
 self.tuesday\_table.setHorizontalHeaderLabels(['id', 'Subject', 'Room', 'Time', '', ''])  
  
 self.\_update\_tuesday\_table()  
  
 self.tvbox = QVBoxLayout()  
 self.tvbox.addWidget(self.tuesday\_table)  
 self.tuesday\_gbox.setLayout(self.tvbox)  
  
 def \_create\_wednesday\_table(self):  
 self.wednesday\_table = QTableWidget()  
 self.wednesday\_table.setSizeAdjustPolicy(QAbstractScrollArea.AdjustToContents)  
  
 self.wednesday\_table.setColumnCount(6)  
 self.wednesday\_table.setHorizontalHeaderLabels(['id', 'Subject', 'Room', 'Time', '', ''])  
  
 self.\_update\_wednesday\_table()  
  
 self.wvbox = QVBoxLayout()  
 self.wvbox.addWidget(self.wednesday\_table)  
 self.wednesday\_gbox.setLayout(self.wvbox)  
  
 def \_create\_thursday\_table(self):  
 self.thursday\_table = QTableWidget()  
 self.thursday\_table.setSizeAdjustPolicy(QAbstractScrollArea.AdjustToContents)  
  
 self.thursday\_table.setColumnCount(6)  
 self.thursday\_table.setHorizontalHeaderLabels(['id', 'Subject', 'Room', 'Time', '', ''])  
  
 self.\_update\_thursday\_table()  
  
 self.thvbox = QVBoxLayout()  
 self.thvbox.addWidget(self.thursday\_table)  
 self.thursday\_gbox.setLayout(self.thvbox)  
  
 def \_create\_friday\_table(self):  
 self.friday\_table = QTableWidget()  
 self.friday\_table.setSizeAdjustPolicy(QAbstractScrollArea.AdjustToContents)  
  
 self.friday\_table.setColumnCount(6)  
 self.friday\_table.setHorizontalHeaderLabels(['id', 'Subject', 'Room', 'Time', '', ''])  
  
 self.\_update\_friday\_table()  
  
 self.fvbox = QVBoxLayout()  
 self.fvbox.addWidget(self.friday\_table)  
 self.friday\_gbox.setLayout(self.fvbox)  
  
 def \_update\_t\_table(self):  
 self.cursor.execute("SELECT id, full\_name, subject FROM list.teacher")  
 records = list(self.cursor.fetchall())  
 self.t\_table.setRowCount(len(records) + 1)  
 try:  
 for i, r in enumerate(records):  
 r = list(r)  
 self.t\_table.setItem(i, 0, QTableWidgetItem(str(r[0])))  
 self.t\_table.setItem(i, 1, QTableWidgetItem(str(r[1])))  
 self.t\_table.setItem(i, 2, QTableWidgetItem(str(r[2])))  
 joinButton = QPushButton("Join")  
 self.t\_table.setCellWidget(i, 3, joinButton)  
  
 joinButton.clicked.connect(lambda ch, num=i: self.\_change\_data\_from\_t(num))  
  
 deleteButton = QPushButton("Delete")  
 self.t\_table.setCellWidget(i, 4, deleteButton)  
  
 deleteButton.clicked.connect(lambda ch, num=i: self.\_delete\_data\_from\_t(num))  
  
 addButton = QPushButton("Add")  
 self.t\_table.setCellWidget(i+1, 3, addButton)  
 addButton.clicked.connect(lambda ch, num=i+1: self.\_add\_t(num))  
 self.t\_table.resizeRowsToContents()  
 except:  
 addButton = QPushButton("Add")  
 self.t\_table.setCellWidget(0, 3, addButton)  
 addButton.clicked.connect(lambda ch, num=0: self.\_add\_t(num))  
 self.t\_table.resizeRowsToContents()  
  
 def \_update\_sub\_table(self):  
 self.cursor.execute("SELECT id, name FROM list.subject")  
 records = list(self.cursor.fetchall())  
 self.sub\_table.setRowCount(len(records) + 1)  
 try:  
 for i, r in enumerate(records):  
 r = list(r)  
 self.sub\_table.setItem(i, 0, QTableWidgetItem(str(r[0])))  
 self.sub\_table.setItem(i, 1, QTableWidgetItem(str(r[1])))  
 joinButton = QPushButton("Join")  
 self.sub\_table.setCellWidget(i, 2, joinButton)  
  
 joinButton.clicked.connect(lambda ch, num=i: self.\_change\_data\_from\_sub(num))  
  
 deleteButton = QPushButton("Delete")  
 self.sub\_table.setCellWidget(i, 3, deleteButton)  
  
 deleteButton.clicked.connect(lambda ch, num=i: self.\_delete\_data\_from\_sub(num))  
  
 addButton = QPushButton("Add")  
 self.sub\_table.setCellWidget(i+1, 2, addButton)  
 addButton.clicked.connect(lambda ch, num=i+1: self.\_add\_sub(num))  
 self.sub\_table.resizeRowsToContents()  
 except:  
 addButton = QPushButton("Add")  
 self.sub\_table.setCellWidget(0, 2, addButton)  
 addButton.clicked.connect(lambda ch, num=0: self.\_add\_sub(num))  
 self.sub\_table.resizeRowsToContents()  
  
  
 def \_update\_monday\_table(self):  
 self.cursor.execute("SELECT id, subject, room\_numb, start\_time FROM list.timetable WHERE day = 'monday'")  
 records = list(self.cursor.fetchall())  
  
 self.monday\_table.setRowCount(len(records)+1)  
 try:  
 for i, r in enumerate(records):  
 r = list(r)  
 self.monday\_table.setItem(i, 0, QTableWidgetItem(str(r[0])))  
 self.monday\_table.setItem(i, 1, QTableWidgetItem(str(r[1])))  
 self.monday\_table.setItem(i, 2, QTableWidgetItem(str(r[2])))  
 self.monday\_table.setItem(i, 3, QTableWidgetItem(str(r[3])))  
 joinButton = QPushButton("Join")  
 self.monday\_table.setCellWidget(i, 4, joinButton)  
  
 joinButton.clicked.connect(lambda ch, num=i: self.\_change\_data\_from\_table(num))  
  
 deleteButton = QPushButton("Delete")  
 self.monday\_table.setCellWidget(i, 5, deleteButton)  
  
 deleteButton.clicked.connect(lambda ch, num=i: self.\_delete\_data\_from\_table(num))  
  
 addButton = QPushButton("Add")  
 self.monday\_table.setCellWidget(i+1, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=i+1: self.\_add\_data(num))  
 self.monday\_table.resizeRowsToContents()  
 except:  
 addButton = QPushButton("Add")  
 self.monday\_table.setCellWidget(0, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=0: self.\_add\_data(num))  
 self.monday\_table.resizeRowsToContents()  
  
 def \_update\_tuesday\_table(self):  
 self.cursor.execute("SELECT id, subject, room\_numb, start\_time FROM list.timetable WHERE day = 'tuesday'")  
 records = list(self.cursor.fetchall())  
  
 self.tuesday\_table.setRowCount(len(records)+1)  
 try:  
 for i, r in enumerate(records):  
 r = list(r)  
 self.tuesday\_table.setItem(i, 0, QTableWidgetItem(str(r[0])))  
 self.tuesday\_table.setItem(i, 1, QTableWidgetItem(str(r[1])))  
 self.tuesday\_table.setItem(i, 2, QTableWidgetItem(str(r[2])))  
 self.tuesday\_table.setItem(i, 3, QTableWidgetItem(str(r[3])))  
 joinButton = QPushButton("Join")  
 self.tuesday\_table.setCellWidget(i, 4, joinButton)  
  
 joinButton.clicked.connect(lambda ch, num=i: self.\_change\_data\_from\_table1(num))  
  
 deleteButton = QPushButton("Delete")  
 self.tuesday\_table.setCellWidget(i, 5, deleteButton)  
  
 deleteButton.clicked.connect(lambda ch, num=i: self.\_delete\_data\_from\_table1(num))  
  
 addButton = QPushButton("Add")  
 self.tuesday\_table.setCellWidget(i + 1, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=i + 1: self.\_add\_data1(num))  
 self.tuesday\_table.resizeRowsToContents()  
 except:  
 addButton = QPushButton("Add")  
 self.tuesday\_table.setCellWidget(0, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=0: self.\_add\_data1(num))  
 self.tuesday\_table.resizeRowsToContents()  
  
 def \_update\_wednesday\_table(self):  
 self.cursor.execute("SELECT id, subject, room\_numb, start\_time FROM list.timetable WHERE day = 'wednesday'")  
 records = list(self.cursor.fetchall())  
  
 self.wednesday\_table.setRowCount(len(records)+1)  
 try:  
 for i, r in enumerate(records):  
 r = list(r)  
 self.wednesday\_table.setItem(i, 0, QTableWidgetItem(str(r[0])))  
 self.wednesday\_table.setItem(i, 1, QTableWidgetItem(str(r[1])))  
 self.wednesday\_table.setItem(i, 2, QTableWidgetItem(str(r[2])))  
 self.wednesday\_table.setItem(i, 3, QTableWidgetItem(str(r[3])))  
 joinButton = QPushButton("Join")  
 self.wednesday\_table.setCellWidget(i, 4, joinButton)  
  
 joinButton.clicked.connect(lambda ch, num=i: self.\_change\_data\_from\_table2(num))  
  
 deleteButton = QPushButton("Delete")  
 self.wednesday\_table.setCellWidget(i, 5, deleteButton)  
  
 deleteButton.clicked.connect(lambda ch, num=i: self.\_delete\_data\_from\_table2(num))  
  
 addButton = QPushButton("Add")  
 self.wednesday\_table.setCellWidget(i + 1, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=i + 1: self.\_add\_data2(num))  
 self.wednesday\_table.resizeRowsToContents()  
 except:  
 addButton = QPushButton("Add")  
 self.wednesday\_table.setCellWidget(0, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=0: self.\_add\_data2(num))  
 self.wednesday\_table.resizeRowsToContents()  
  
 def \_update\_thursday\_table(self):  
 self.cursor.execute("SELECT id, subject, room\_numb, start\_time FROM list.timetable WHERE day = 'thursday'")  
 records = list(self.cursor.fetchall())  
  
 self.thursday\_table.setRowCount(len(records)+1)  
 try:  
 for i, r in enumerate(records):  
 r = list(r)  
 self.thursday\_table.setItem(i, 0, QTableWidgetItem(str(r[0])))  
 self.thursday\_table.setItem(i, 1, QTableWidgetItem(str(r[1])))  
 self.thursday\_table.setItem(i, 2, QTableWidgetItem(str(r[2])))  
 self.thursday\_table.setItem(i, 3, QTableWidgetItem(str(r[3])))  
 joinButton = QPushButton("Join")  
 self.thursday\_table.setCellWidget(i, 4, joinButton)  
  
 joinButton.clicked.connect(lambda ch, num=i: self.\_change\_data\_from\_table3(num))  
  
 deleteButton = QPushButton("Delete")  
 self.thursday\_table.setCellWidget(i, 5, deleteButton)  
  
 deleteButton.clicked.connect(lambda ch, num=i: self.\_delete\_data\_from\_table3(num))  
  
 addButton = QPushButton("Add")  
 self.thursday\_table.setCellWidget(i + 1, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=i + 1: self.\_add\_data3(num))  
 self.thursday\_table.resizeRowsToContents()  
 except:  
 addButton = QPushButton("Add")  
 self.thursday\_table.setCellWidget(0, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=0: self.\_add\_data3(num))  
 self.thursday\_table.resizeRowsToContents()  
  
 def \_update\_friday\_table(self):  
 self.cursor.execute("SELECT id, subject, room\_numb, start\_time FROM list.timetable WHERE day = 'friday'")  
 records = list(self.cursor.fetchall())  
  
 self.friday\_table.setRowCount(len(records)+1)  
 try:  
 for i, r in enumerate(records):  
 r = list(r)  
 self.friday\_table.setItem(i, 0, QTableWidgetItem(str(r[0])))  
 self.friday\_table.setItem(i, 1, QTableWidgetItem(str(r[1])))  
 self.friday\_table.setItem(i, 2, QTableWidgetItem(str(r[2])))  
 self.friday\_table.setItem(i, 3, QTableWidgetItem(str(r[3])))  
 joinButton = QPushButton("Join")  
 self.friday\_table.setCellWidget(i, 4, joinButton)  
  
 joinButton.clicked.connect(lambda ch, num=i: self.\_change\_data\_from\_table4(num))  
  
 deleteButton = QPushButton("Delete")  
 self.friday\_table.setCellWidget(i, 5, deleteButton)  
  
 deleteButton.clicked.connect(lambda ch, num=i: self.\_delete\_data\_from\_table4(num))  
  
 addButton = QPushButton("Add")  
 self.friday\_table.setCellWidget(i + 1, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=i + 1: self.\_add\_data4(num))  
 self.friday\_table.resizeRowsToContents()  
 except:  
 addButton = QPushButton("Add")  
 self.friday\_table.setCellWidget(0, 4, addButton)  
 addButton.clicked.connect(lambda ch, num=0: self.\_add\_data4(num))  
 self.friday\_table.resizeRowsToContents()  
  
 def \_change\_data\_from\_t(self, rowNumb):  
 row = list()  
 for i in range(self.t\_table.columnCount()):  
 try:  
 row.append(self.t\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row)  
 try:  
 self.cursor.execute("UPDATE list.teacher set full\_name =%s, subject =%s WHERE id =%s",  
 (str(row[1]), (str(row[2]), str(row[0]))))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
  
 def \_change\_data\_from\_sub(self, rowNumb):  
 row = list()  
 for i in range(self.sub\_table.columnCount()):  
 try:  
 row.append(self.sub\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row)  
 try:  
 self.cursor.execute("UPDATE list.subject set name =%s WHERE id =%s",  
 (str(row[1]), str(row[0])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
  
 def \_change\_data\_from\_table(self, rowNumb):  
 row = list()  
 for i in range(self.monday\_table.columnCount()):  
 try:  
 row.append(self.monday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 try:  
 self.cursor.execute("UPDATE list.timetable set subject =%s, room\_numb =%s, start\_time =%s WHERE id =%s",  
 (str(row[1]), str(row[2]), str(row[3]), str(row[0])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
  
 def \_change\_data\_from\_table1(self, rowNumb):  
 row = list()  
 for i in range(self.tuesday\_table.columnCount()):  
 try:  
 row.append(self.tuesday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 try:  
 self.cursor.execute("UPDATE list.timetable set subject =%s, room\_numb =%s, start\_time =%s WHERE id =%s",  
 (str(row[1]), str(row[2]), str(row[3]), str(row[0])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
  
 def \_change\_data\_from\_table2(self, rowNumb):  
 row = list()  
 for i in range(self.wednesday\_table.columnCount()):  
 try:  
 row.append(self.wednesday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 try:  
 self.cursor.execute("UPDATE list.timetable set subject =%s, room\_numb =%s, start\_time =%s WHERE id =%s",  
 (str(row[1]), str(row[2]), str(row[3]), str(row[0])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
  
 def \_change\_data\_from\_table3(self, rowNumb):  
 row = list()  
 for i in range(self.thursday\_table.columnCount()):  
 try:  
 row.append(self.thursday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 try:  
 self.cursor.execute("UPDATE list.timetable set subject =%s, room\_numb =%s, start\_time =%s WHERE id =%s",  
 (str(row[1]), str(row[2]), str(row[3]), str(row[0])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
  
 def \_change\_data\_from\_table4(self, rowNumb):  
 row = list()  
 for i in range(self.friday\_table.columnCount()):  
 try:  
 row.append(self.friday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 try:  
 self.cursor.execute("UPDATE list.timetable set subject =%s, room\_numb =%s, start\_time =%s WHERE id =%s",  
 (str(row[1]), str(row[2]), str(row[3]), str(row[0])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
  
 def \_delete\_data\_from\_t(self, rowNumb):  
 row = self.t\_table.item(rowNumb, 0).text()  
 print(row)  
 self.cursor.execute(f"DELETE FROM list.teacher WHERE id = {row}")  
 self.conn.commit()  
 self.t\_table.setRowCount(0)  
 self.\_update\_t\_table()  
  
 def \_delete\_data\_from\_sub(self, rowNumb):  
 row = self.sub\_table.item(rowNumb, 0).text()  
 print(row)  
 self.cursor.execute(f"DELETE FROM list.subject WHERE id = {row}")  
 self.conn.commit()  
 self.sub\_table.setRowCount(0)  
 self.\_update\_sub\_table()  
  
 def \_delete\_data\_from\_table(self, rowNumb):  
 row = self.monday\_table.item(rowNumb, 0).text()  
 print(row)  
 self.cursor.execute(f"DELETE FROM list.timetable WHERE id = {row}")  
 self.conn.commit()  
 self.monday\_table.setRowCount(0)  
 self.\_update\_monday\_table()  
  
 def \_delete\_data\_from\_table1(self, rowNumb):  
 row = self.tuesday\_table.item(rowNumb, 0).text()  
 print(row)  
 self.cursor.execute(f"DELETE FROM list.timetable WHERE id = {row}")  
 self.conn.commit()  
 self.tuesday\_table.setRowCount(0)  
 self.\_update\_tuesday\_table()  
  
 def \_delete\_data\_from\_table2(self, rowNumb):  
 row = self.wednesday\_table.item(rowNumb, 0).text()  
 print(row)  
 self.cursor.execute(f"DELETE FROM list.timetable WHERE id = {row}")  
 self.conn.commit()  
 self.wednesday\_table.setRowCount(0)  
 self.\_update\_wednesday\_table()  
  
 def \_delete\_data\_from\_table3(self, rowNumb):  
 row = self.thursday\_table.item(rowNumb, 0).text()  
 print(row)  
 self.cursor.execute(f"DELETE FROM list.timetable WHERE id = {row}")  
 self.conn.commit()  
 self.thursday\_table.setRowCount(0)  
 self.\_update\_thursday\_table()  
  
 def \_delete\_data\_from\_table4(self, rowNumb):  
 row = self.friday\_table.item(rowNumb, 0).text()  
 print(row)  
 self.cursor.execute(f"DELETE FROM list.timetable WHERE id = {row}")  
 self.conn.commit()  
 self.friday\_table.setRowCount(0)  
 self.\_update\_friday\_table()  
  
 def \_add\_t(self, rowNumb):  
 row = list()  
 for i in range(self.t\_table.columnCount()):  
 try:  
 row.append(self.t\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row[1])  
 try:  
 self.cursor.execute("INSERT INTO list.teacher (full\_name, subject) values (%s, %s)", (str(row[1]), str(row[2])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
 self.\_update\_t\_table()  
  
 def \_add\_sub(self, rowNumb):  
 row = list()  
 for i in range(self.sub\_table.columnCount()):  
 try:  
 row.append(self.sub\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row[1])  
 try:  
 self.cursor.execute("INSERT INTO list.subject (name) values (%s)", (str(row[1])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
 self.\_update\_sub\_table()  
  
 def \_add\_data(self, rowNumb):  
 row = list()  
 for i in range(self.monday\_table.columnCount()):  
 try:  
 row.append(self.monday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row)  
 try:  
 self.cursor.execute("INSERT INTO list.timetable (day, subject, room\_numb, start\_time) values ('monday', %s, %s, %s)",  
 (str(row[1]), str(row[2]), str(row[3])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
 self.\_update\_monday\_table()  
  
 def \_add\_data1(self, rowNumb):  
 row = list()  
 for i in range(self.tuesday\_table.columnCount()):  
 try:  
 row.append(self.tuesday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row)  
 try:  
 self.cursor.execute("INSERT INTO list.timetable (day, subject, room\_numb, start\_time) values ('tuesday', %s, %s, %s)",  
 (str(row[1]), str(row[2]), str(row[3])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
 self.\_update\_tuesday\_table()  
  
 def \_add\_data2(self, rowNumb):  
 row = list()  
 for i in range(self.wednesday\_table.columnCount()):  
 try:  
 row.append(self.wednesday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row)  
 try:  
 self.cursor.execute("INSERT INTO list.timetable (day, subject, room\_numb, start\_time) values ('wednesday', %s, %s, %s)",  
 (str(row[1]), str(row[2]), str(row[3])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
 self.\_update\_wednesday\_table()  
  
 def \_add\_data3(self, rowNumb):  
 row = list()  
 for i in range(self.thursday\_table.columnCount()):  
 try:  
 row.append(self.thursday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row)  
 try:  
 self.cursor.execute("INSERT INTO list.timetable (day, subject, room\_numb, start\_time) values ('thursday', %s, %s, %s)",  
 (str(row[1]), str(row[2]), str(row[3])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
 self.\_update\_thursday\_table()  
  
 def \_add\_data4(self, rowNumb):  
 row = list()  
 for i in range(self.friday\_table.columnCount()):  
 try:  
 row.append(self.friday\_table.item(rowNumb, i).text())  
 except:  
 row.append(None)  
 print(row)  
 try:  
 self.cursor.execute("INSERT INTO list.timetable (day, subject, room\_numb, start\_time) values ('friday', %s, %s, %s)",  
 (str(row[1]), str(row[2]), str(row[3])))  
 self.conn.commit()  
 except:  
 QMessageBox.about(self, "Error", "Enter all fields")  
 self.\_update\_friday\_table()  
  
  
 def \_update\_shedule(self):  
 self.\_update\_t\_table()  
 self.\_update\_sub\_table()  
 self.\_update\_monday\_table()  
 self.\_update\_tuesday\_table()  
 self.\_update\_wednesday\_table()  
 self.\_update\_thursday\_table()  
 self.\_update\_friday\_table()  
  
  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app = QApplication(sys.argv)  
 win = Window()  
 win.show()  
 sys.exit(app.exec())

**3. Результат работы:**





