**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Базы данных»

Отчет по лабораторной работе №7

«Использование триггеров и хранимых процедур PostgreSQL»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-45Б |  | преподаватель каф. ИУ5 |
| Коновалов И.Н. |  | Силантьева Е.Ю. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2023 г.

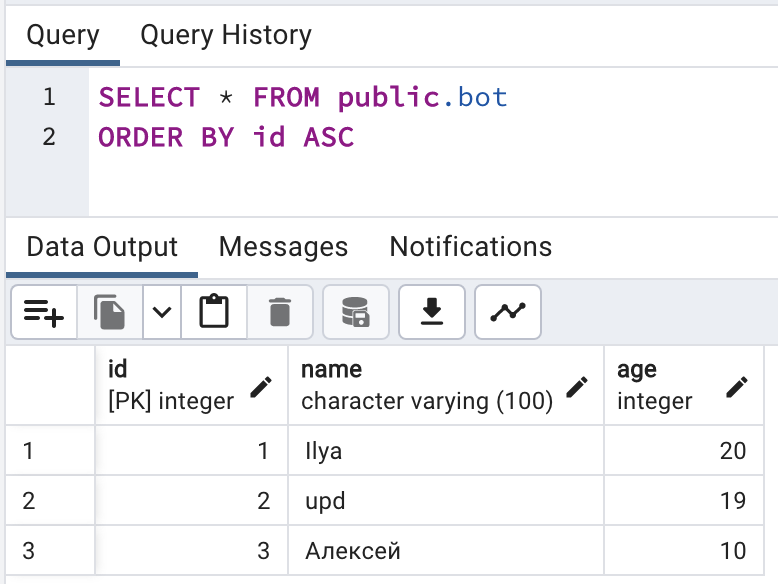
Цель лабораторной работы:

Реализовать CRUD операции чтобы можно было вызывать их через интерфейс телеграмм бота.

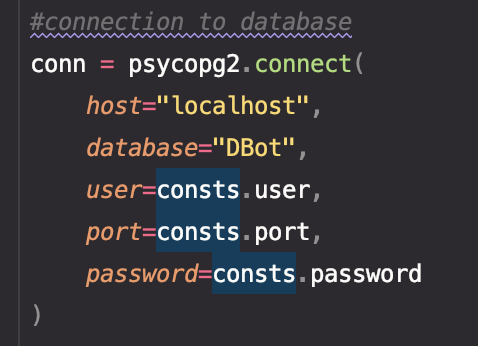
1. Импортируем библиотеки



1. Создаем таблицу, для которой будем реализовывать CRUD



1. Подключаемся к БД



1. Пишем CRUD

Файл db\_processor.py

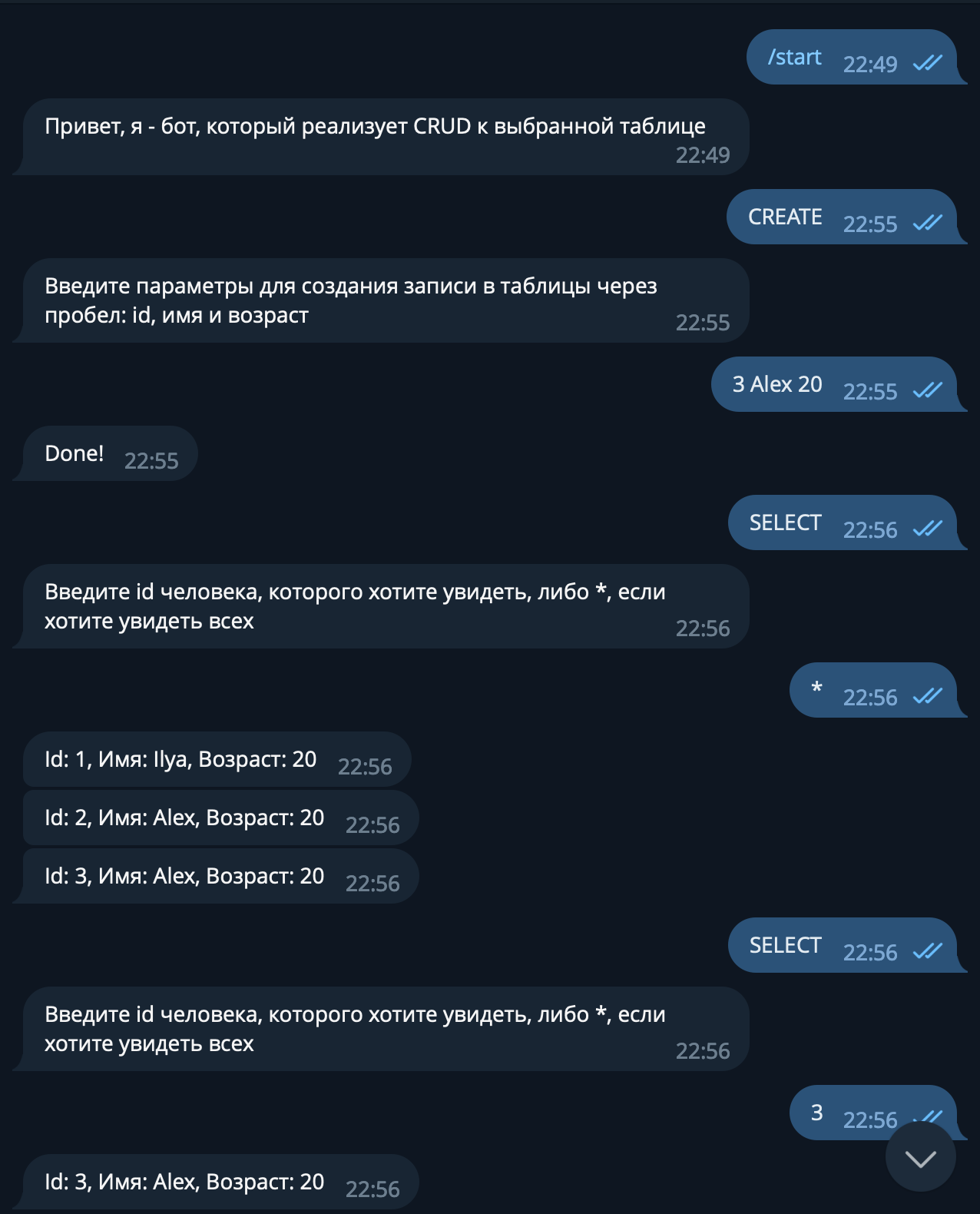
*import* psycopg2  
  
*def* insert(*id*: *int*, *name*: *str*, *age*: *int*, *conn*):  
 cur = *conn*.cursor()  
 cur.execute("INSERT INTO bot (id, name, age) VALUES (*%s*, *%s*, *%s*)", (*id*, *name*, *age*))  
 *conn*.commit()  
 cur.close()  
  
*def* select(*id*: *int*, *conn*):  
 cur = *conn*.cursor()  
 cur.execute("SELECT \* FROM bot WHERE id = (*%s*)", (*id*,))  
 answer = cur.fetchall()  
 *return* answer  
 *for* elem *in* answer:  
 *print*("Id = ", elem[0])  
 *print*("Имя = ", elem[1])  
 *print*("Возраст = ", elem[2], "\n")  
 cur.close()  
  
*def* select\_all(*conn*):  
 cur = *conn*.cursor()  
 cur.execute("SELECT \* FROM bot ORDER BY id")  
 answer = cur.fetchall()  
 *return* answer  
 *for* elem *in* answer:  
 *print*("Id = ", elem[0])  
 *print*("Имя = ", elem[1])  
 *print*("Возраст = ", elem[2], "\n")  
 cur.close()  
  
*def* update(*id*: *int*, *name*: *str*, *age*: *int*, *conn*):  
 cur = *conn*.cursor()  
 cur.execute("UPDATE bot SET name = (*%s*), age = (*%s*) WHERE id = (*%s*)", (*name*, *age*, *id*))  
 *conn*.commit()  
 cur.close()  
  
*def* delete(*id*: *int*, *conn*):  
 cur = *conn*.cursor()  
 cur.execute("DELETE FROM bot WHERE id = (*%s*)", (*id*,))  
 *conn*.commit()  
 cur.close()

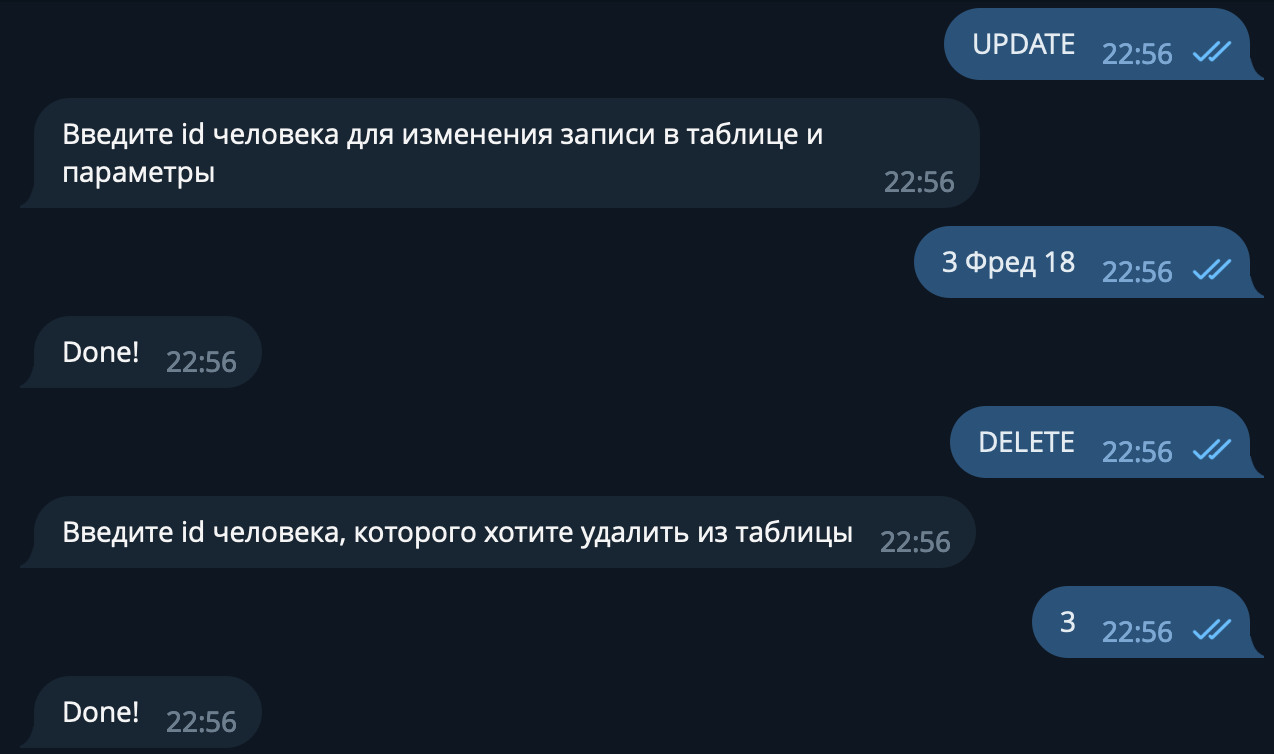
1. Реализовываем работу бота

Файл main.py

*from* aiogram *import* Bot, Dispatcher, executor, types  
*from* aiogram.contrib.fsm\_storage.memory *import* MemoryStorage  
*from* aiogram.dispatcher *import* FSMContext  
*from* aiogram.dispatcher.filters.state *import* State, StatesGroup  
  
*import* psycopg2  
  
*import* db\_processor  
  
*import* consts  
  
tgToken = consts.tgToken  
bot = Bot(*token*=tgToken)  
dp = Dispatcher(bot, *storage*= MemoryStorage())  
  
*#connection to database*conn = psycopg2.connect(  
 *host*="localhost",  
 *database*="DBot",  
 *user*=consts.user,  
 *port*=consts.port,  
 *password*=consts.password  
)  
  
@dp.message\_handler(*commands*=['start'])  
*async def* processStartCommand(*message*: types.Message):  
 kb = [  
 [types.KeyboardButton(*text*="CREATE")],  
 [types.KeyboardButton(*text*="SELECT")],  
 [types.KeyboardButton(*text*="UPDATE")],  
 [types.KeyboardButton(*text*="DELETE")],  
 ]  
 keyboard = types.ReplyKeyboardMarkup(  
 *keyboard*=kb,  
 *resize\_keyboard*=*True*,  
 *input\_field\_placeholder*="Введите желаемое действие"  
 )  
  
 *await message*.answer(  
 'Привет, я - бот, который реализует CRUD к выбранной таблице',  
 *reply\_markup*=keyboard)  
  
*class* Form(StatesGroup):  
 cr = State()  
 sel = State()  
 upd = State()  
 dl = State()  
  
@dp.message\_handler(*state* = Form.cr)  
*async def* creator(*message*: types.Message, *state*: FSMContext):  
 *await state*.finish()  
 data = *message*.text  
 *try*:  
 data = data.split(" ")  
 id = *int*(data[0])  
 age = *int*(data[2])  
 db\_processor.insert(id, data[1], age, conn)  
 *await message*.answer('Done!')  
 *except Exception as* e:  
 *await message*.answer(e)  
  
@dp.message\_handler(*state* = Form.sel)  
*async def* reader(*message*: types.Message, *state*: FSMContext):  
 *await state*.finish()  
 data = *message*.text  
 *try*:  
 *if* data == '\*':  
 temp = db\_processor.select\_all(conn)  
 *for* i *in range*(*len*(temp)):  
 *await message*.answer(f'Id: {temp[i][0]}, Имя: {temp[i][1]}, Возраст: {temp[i][2]}')  
 *else*:  
 id = *int*(data)  
 temp = db\_processor.select(id, conn)  
 *await message*.answer(f'Id: {temp[0][0]}, Имя: {temp[0][1]}, Возраст: {temp[0][2]}')  
 *except Exception as* e:  
 *await message*.answer(e)  
 *await message*.reply('Произошла ошибка, попробуйте еще раз')  
  
@dp.message\_handler(*state* = Form.upd)  
*async def* updater(*message*: types.Message, *state*: FSMContext):  
 *await state*.finish()  
 data = *message*.text  
 *try*:  
 data = data.split(" ")  
 id = *int*(data[0])  
 age = *int*(data[2])  
 db\_processor.update(id, data[1], age, conn)  
 *await message*.answer('Done!')  
 *except Exception as* e:  
 *await message*.answer(e)  
 *await message*.reply('Произошла ошибка, попробуйте еще раз')  
  
@dp.message\_handler(*state* = Form.dl)  
*async def* deleter(*message*: types.Message, *state*: FSMContext):  
 *await state*.finish()  
 data = *message*.text  
 *try*:  
 id = *int*(data)  
 db\_processor.delete(id, conn)  
 *await message*.answer('Done!')  
 *except Exception as* e:  
 *await message*.answer(e)  
 *await message*.reply('Произошла ошибка, попробуйте еще раз')  
  
@dp.message\_handler(*lambda message*: *message*.text =='CREATE')  
*async def* createFunc(*message*: types.Message):  
 *await message*.answer('Введите параметры для создания записи в таблицы через пробел: id, имя и возраст')  
 *await* Form.cr.set()  
  
@dp.message\_handler(*lambda message*: *message*.text =='SELECT')  
*async def* selectFunc(*message*: types.Message):  
 *await message*.answer('Введите id человека, которого хотите увидеть, либо \*, если хотите увидеть всех')  
 *await* Form.sel.set()  
  
@dp.message\_handler(*lambda message*: *message*.text =='UPDATE')  
*async def* updateFunc(*message*: types.Message):  
 *await message*.answer('Введите id человека для изменения записи в таблице и параметры')  
 *await* Form.upd.set()  
  
@dp.message\_handler(*lambda message*: *message*.text =='DELETE')  
*async def* deleteFunc(*message*: types.Message):  
 *await message*.answer('Введите id человека, которого хотите удалить из таблицы')  
 *await* Form.dl.set()  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 executor.start\_polling(dp)

1. Пример работы бота





Выводы:

Научились писать собственного телеграмм-бота, реализацию CRUD к таблице при помощи языка Python3.