

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций
«Взаимодействие с базами данных SQLite3 с помощью языка
программирования Python»**

**Отчет по лабораторной работе № 2.21
по дисциплине «Основы программной инженерии»**

Выполнил студент группы

ПИЖ-б-о-21-1

Трушева В. О. _____. « » 2023г.

Подпись студента _____

Работа защищена « _____ 20 __ г.

Проверила Воронкин Р.А. _____

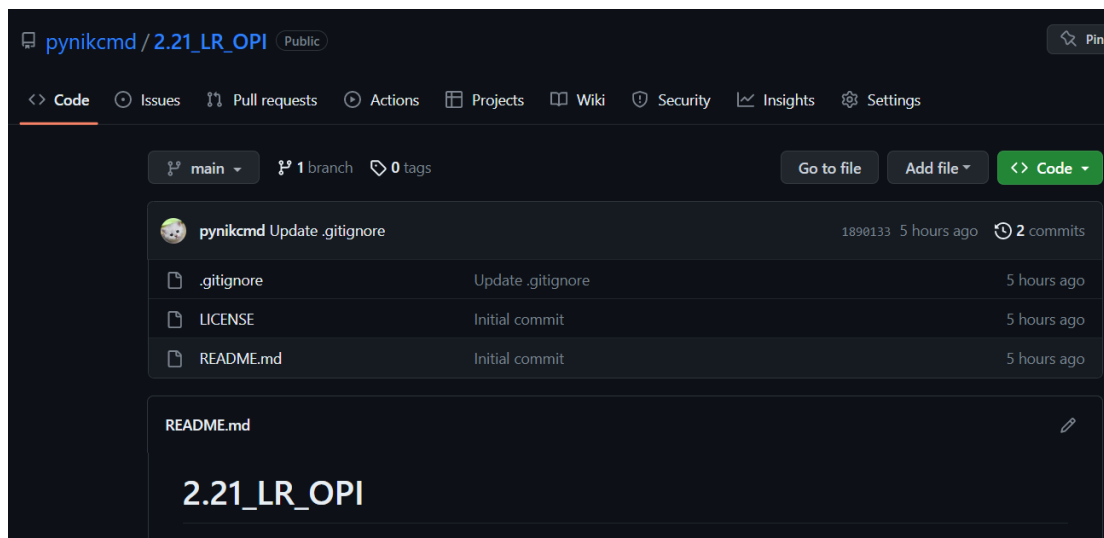
(подпись)

Ставрополь 2023

Цель работы: исследовать взаимодействие с базами данных SQLite3 с помощью языка программирования Python.

Методика и порядок выполнения работы

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



3. Выполните клонирование созданного репозитория.

```
D:\fgit>git clone https://github.com/pynikcmd/2.21_LR_OPI.git
Cloning into '2.21_LR_OPI'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), 5.12 KiB | 748.00 KiB/s, done.
```

4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.



5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

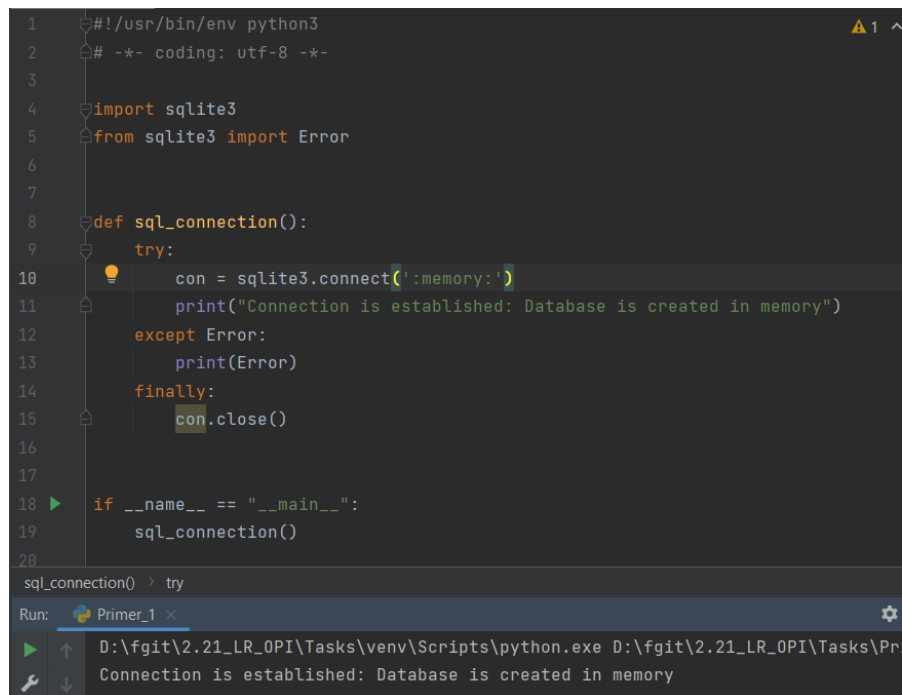
```
D:\fgit\2.21_LR_OPI>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/fgit/2.21_LR_OPI/.git/hooks]
```

6. Создайте проект PyCharm в папке репозитория.

7. Проработайте примеры лабораторной работы. Создайте для них отдельные модули языка Python. Зафиксируйте изменения в репозитории.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sqlite3
5  from sqlite3 import Error
6
7
8  def sql_connection():
9      try:
10         con = sqlite3.connect(':memory:')
11         print("Connection is established: Database is created in memory")
12     except Error:
13         print(Error)
14     finally:
15         con.close()
16
17
18  if __name__ == "__main__":
19     sql_connection()
20
```

sql_connection() > try

Run: Primer_1

D:\fgit\2.21_LR_OPI\Tasks\venv\Scripts\python.exe D:\fgit\2.21_LR_OPI\Tasks\Pr
Connection is established: Database is created in memory

Пример 1

```
8 def sql_connection():
9     try:
10         con = sqlite3.connect('mydatabase.db')
11         return con
12
13     except Error:
14         print(Error)
15
16     return None
17
18
19 def sql_table(con):
20     cursor_obj = con.cursor()
21     cursor_obj.execute(
22         """
23         CREATE TABLE employees (
24             id integer PRIMARY KEY,
25             name text,
26             salary real,
27             department text,
28             position text,
29             hireDate text)
30         """
31     )
32     con.commit()
33
34
35 if __name__ == "__main__":
36     con = sql_connection()
37     sql_table(con)
38
```

Пример 2

id	name	salary	department	position	hireDate
Фи...	Фил...	Фил...	Фильтр	Фильтр	Фильтр

Результат примера 2

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  import sqlite3
6
7  con = sqlite3.connect('mydatabase.db')
8
9
10 def sql_insert(con, entities):
11     cursor_obj = con.cursor()
12     cursor_obj.execute(
13         """
14         INSERT INTO employees(id, name, salary, department, position, hireDate)
15         VALUES(?, ?, ?, ?, ?, ?)
16         """,
17         entities
18     )
19     con.commit()
20
21
22 if __name__ == "__main__":
23     entities = (2, 'Andrew', 800, 'IT', 'Tech', '2018-02-06')
24     sql_insert(con, entities)

```

Пример 3

	id	name	salary	department	position	hireDate
	Фи...	Фильтр	Фил...	Фильтр	Фильтр	Фильтр
1	2	Andrew	800.0	IT	Tech	2018-02-06

Результат примера 3

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sqlite3
5
6  con = sqlite3.connect('mydatabase.db')
7
8
9  def sql_update(con):
10     cursor_obj = con.cursor()
11     cursor_obj.execute(
12         "UPDATE employees SET name = 'Rogers' where id = 2"
13     )
14     con.commit()
15
16
17 sql_update(con)
18

```

Пример 4

	id	name	salary	department	position	hireDate
	Фи...	Фильтр	Фил...	Фильтр	Фильтр	Фильтр
1	2	Rogers	800.0	IT	Tech	2018-02-06

Результат примера 4

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sqlite3
5
6  con = sqlite3.connect('mydatabase.db')
7
8
9  def sql_fetch(con):
10     cursor_obj = con.cursor()
11     cursor_obj.execute("SELECT * FROM employees")
12     rows = cursor_obj.fetchall()
13     for row in rows:
14         print(row)
15
16
17  if __name__ == "__main__":
18     sql_fetch(con)

```

Run: Primer_5 x

D:\fgit\2.21_LR_OPI\Tasks\venv\Scripts\python.exe D:\fgit\2.21_LR_OPI\Tasks\venv\Scripts\python.exe D:\fgit\2.21_LR_OPI\Tasks\venv\Scripts\python.exe (2, 'Rogers', 800.0, 'IT', 'Tech', '2018-02-06')

Пример 5

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sqlite3
5
6  con = sqlite3.connect('mydatabase.db')
7
8
9  def sql_fetch(con):
10     cursor_obj = con.cursor()
11     cursor_obj.execute(
12         "SELECT id, name FROM employees WHERE salary >= 800.0"
13     )
14     rows = cursor_obj.fetchall()
15     for row in rows:
16         print(row)
17
18
19  if __name__ == "__main__":
20     sql_fetch(con)
21

```

sql_fetch()

Run: Primer_6 x

D:\fgit\2.21_LR_OPI\Tasks\venv\Scripts\python.exe D:\fgit\2.21_LR_OPI\Tasks\venv\Scripts\python.exe (2, 'Rogers')

Пример 6

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sqlite3
5
6  con = sqlite3.connect('mydatabase.db')
7
8
9  def sql_fetch(con):
10     cursor_obj = con.cursor()
11     cursor_obj.execute(
12         "SELECT name from sqlite_master where type='table'"
13     )
14     print(cursor_obj.fetchall())
15
16
17  if __name__ == "__main__":
18     sql_fetch(con)
```

Run: Primer_7 x

D:\fgit\2.21_LR_OPI\Tasks\venv\Scripts\python.exe D:\fgit\2.21_L
[('employees',)]

Пример 7

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sqlite3
5
6  con = sqlite3.connect('mydatabase.db')
7
8
9  def sql_fetch(con):
10     con = sqlite3.connect('mydatabase.db')
11
12     cursor_obj = con.cursor()
13     cursor_obj.execute(
14         "CREATE TABLE IF NOT EXISTS projects(id INTEGER, name TEXT)"
15     )
16     data = [
17         (1, "Ridesharing"),
18         (2, "Water Purifying"),
19         (3, "Forensics"),
20         (4, "Botany")
21     ]
22     cursor_obj.executemany("INSERT INTO projects VALUES (?, ?)", data)
23     con.commit()
24
25
26  if __name__ == "__main__":
27     sql_fetch(con)
28  |
```

Пример 8

	id	name
Фи...	Фильтр	
1	1	Ridesharing
2	2	Water Purifying
3	3	Forensics
4	4	Botany
5	1	Ridesharing
6	2	Water Purifying
7	3	Forensics
8	4	Botany

Результат 8 примера

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  import sqlite3
6  import datetime
7
8  con = sqlite3.connect('mydatabase.db')
9
10
11 def sql_fetch(con):
12     con = sqlite3.connect('mydatabase.db')
13
14     cursor_obj = con.cursor()
15     cursor_obj.execute(
16         """
17         CREATE TABLE IF NOT EXISTS assignments(
18             id INTEGER, name TEXT, date DATE
19         )
20         """
21     )
22     data = [
23         (1, "Ridesharing", datetime.date(2017, 1, 2)),
24         (2, "Water Purifying", datetime.date(2018, 3, 4))
25     ]
26     cursor_obj.executemany("INSERT INTO assignments VALUES(?, ?, ?)", data)
27     con.commit()
28
29
30 if __name__ == "__main__":
31     sql_fetch(con)
32

```

Пример 9

	id	name	date
Фи...	Фильтр		Фильтр
1	1	Ridesharing	2017-01-02
2	2	Water Purifying	2018-03-04

Результат 9 примера


```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  import argparse
6      from pathlib import Path
7      import sqlite3
8  import typing as t
9
10
11 def display_workers(staff: t.List[t.Dict[str, t.Any]]) -> None:
12     """
13     Отобразить список работников.
14     """
15     # Проверить, что список работников не пуст.
16     if staff:
17         # Заголовок таблицы.
18         line = '+-{}-+-{}-+-{}-+-{}-+'.format(
19             '-' * 4,
20             '-' * 30,
21             '-' * 20,
22             '-' * 8
23         )
24         print(line)
25         print(
26             '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
27                 "No",
28                 "Имя",

```

Пример 10

Таблица: workers				
	worker_id	worker_name	post_id	worker_year
	Фильтр	Фильтр	Фильтр	Фильтр
1	1	Name	1	2019
2	2	Nika	2	2014

Результат 10 примера

Таблица: sqlite_sequence		
	name	seq
	Фильтр	Фи...
1	posts	2
2	workers	2

Результат 10 примера

Таблица: posts

	post_id	post_title
	Фильтр	Фильтр
1	1	Directer
2	2	Post

Результат 10 примера

8. Приведите в отчете скриншоты результатов выполнения примера при различных исходных данных вводимых с клавиатуры.

```
D:\fgit\2.21_LR_OPI\Tasks>python Primer_10.py add --db="workers.db" --name="Name" --post="Directer" --year="2019"
D:\fgit\2.21_LR_OPI\Tasks>python Primer_10.py add --db="workers.db" --name="Nika" --post="Post" --year=2014
D:\fgit\2.21_LR_OPI\Tasks>python Primer_10.py select --db="workers.db" --period=5
+-----+-----+-----+-----+
| No |          Ф.И.О.          |      Должность      |      Год      |
+-----+-----+-----+-----+
| 1 | Nika                    | Post                | 2014          |
+-----+-----+-----+-----+
D:\fgit\2.21_LR_OPI\Tasks>python Primer_10.py display --db="workers.db"
+-----+-----+-----+-----+
| No |          Ф.И.О.          |      Должность      |      Год      |
+-----+-----+-----+-----+
| 1 | Name                    | Directer            | 2019          |
+-----+-----+-----+-----+
| 2 | Nika                    | Post                | 2014          |
+-----+-----+-----+-----+
```

Результат 10 примера

9. Зафиксируйте сделанные изменения в репозитории.

10. Приведите в отчете скриншоты работы программ решения индивидуальных заданий.

База данных содержит: фамилия, имя; номер телефона; дата рождения. В функции `find_nomer()` осуществляется вывод на экран информации о человеке, номер телефона которого введен с клавиатуры; если такого нет, выдать на дисплей соответствующее сообщение.

```

D:\fgit\2.21_LR_OPI\Tasks>python Ind.py add --name="Name 1" --phone=1234222345 --birth=01.04.2022
D:\fgit\2.21_LR_OPI\Tasks>python Ind.py add --name="Name 2" --phone=1874392245 --birth=01.09.2000
D:\fgit\2.21_LR_OPI\Tasks>python Ind.py display
+-----+-----+-----+-----+
| № | Ф.И.О. | Дата рождения | Номер |
+-----+-----+-----+-----+
| 1 | Name 1 | 01.04.2022 | 1234222345 |
+-----+-----+-----+-----+
| 2 | Name 2 | 01.09.2000 | 1874392245 |
+-----+-----+-----+-----+

D:\fgit\2.21_LR_OPI\Tasks>python Ind.py find -s=1234222345
+-----+-----+-----+-----+
| № | Ф.И.О. | Дата рождения | Номер |
+-----+-----+-----+-----+
| 1 | Name 1 | 01.04.2022 | 1234222345 |
+-----+-----+-----+-----+

```

Результат индивидуального задания

Таблица: dates

	date_id	birth_year	people_id	phone_number
	Фильтр	Фильтр	Фильтр	Фильтр
1	1	01.04.2022	1	1234222345
2	2	01.09.2000	2	1874392245
3	3	01.09.2005	3	1874392245

Результат индивидуального задания

Таблица: people

	people_id	people_name	phone_number
	Фильтр	Фильтр	Фильтр
1	1	Name 1	1234222345
2	2	Name 2	1874392245
3	3	Name 3	1874392245

Результат индивидуального задания

Таблица: sqlite_sequence

	name	seq
	Фил...	Фи...
1	people	3
2	dates	3

Результат индивидуального задания

11. Зафиксируйте сделанные изменения в репозитории.
12. Добавьте отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксируйте изменения.
13. Выполните слияние ветки для разработки с веткой master/main.
14. Отправьте сделанные изменения на сервер GitHub.
15. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Контрольные вопросы

1. Каково назначение модуля sqlite3?

Модуль sqlite3 позволяет создавать, подключаться к базе данных SQLite, выполнять запросы на выборку, добавление, обновление и удаление данных из таблиц базы данных.

2. Как выполняется соединение с базой данных SQLite3? Что такое курсор базы данных?

Чтобы использовать SQLite3 в Python, прежде всего, вам нужно будет импортировать модуль sqlite3, а затем создать объект соединения, который соединит нас с базой данных и позволит нам выполнять операторы SQL. Объект соединения создается с помощью функции connect(). Курсор SQLite3 – это метод объекта соединения. Для выполнения инструкций SQLite3 сначала устанавливается соединение, а

затем создается объект курсора с использованием объекта соединения следующим образом:

```
conn = sqlite3.connect(database_path)
cursor = conn.cursor()
```

3. Как подключиться к базе данных SQLite3, находящейся в оперативной памяти компьютера?

При создании соединения с SQLite3 автоматически создается файл базы данных, если он еще не существует. Этот файл базы данных создается на диске, мытакже можем создать базу данных в оперативной памяти с помощью функции

:memory: with the connect

Такая база данных называется базой данных в памяти.

4. Как корректно завершить работу с базой данных SQLite3?

После этого вне зависимости от того возникло или нет исключение по работе с базой данных, выполняются операторы блока finally, в котором соединение закрывается. Закрытие соединения необязательно, но это хорошая практика программирования, поэтому вы освобождаете память от любых неиспользуемых ресурсов.

5. Как осуществляется вставка данных в таблицу базы данных SQLite3?

Чтобы вставить данные в таблицу, используется оператор INSERT INTO.

6. Как осуществляется обновление данных таблицы базы данных SQLite3?

Чтобы обновить данные в таблице, просто создайте соединение, затем создайте объект курсора с помощью соединения и, наконец, используйте оператор UPDATE в методе execute ().

7. Как осуществляется выборка данных из базы данных SQLite3?

Оператор SELECT используется для выбора данных из определенной таблицы.

8. Каково назначение метода rowcount?

SQLite3 rowcount используется для возврата количества строк, которые были затронуты или выбраны последним выполненным SQL-запросом.

9. Как получить список всех таблиц базы данных SQLite3?

Чтобы перечислить все таблицы в базе данных SQLite3, вы должны запросить данные из таблицы sqlite_master, а затем использовать fetchall() для получения результатов из инструкции SELECT.

10. Как выполнить проверку существования таблицы как при ее добавлении, так и при ее удалении?

Чтобы проверить, не существует ли таблица уже, мы используем IF NOT EXISTS с оператором CREATE TABLE

11. Как выполнить массовую вставку данных в базу данных SQLite3?

Метод executemany можно использовать для вставки нескольких строк одновременно.

12. Как осуществляется работа с датой и временем при работе с базами данных SQLite3?

В базе данных Python SQLite3 мы можем легко хранить дату или время, импортируя модуль `datetime`.