

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7
дисциплины «Анализ данных»

Выполнил:
Гайчук Дарья Дмитриевна
2 курс, группа ИВТ-6-о-22-1,
09.03.01 «Информатика
и вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р.А.-доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: «Взаимодействие с базами данных SQLite3 с помощью языка программирования Python»

Цель работы: исследовать базовые возможности системы управления базами данных SQLite3

Порядок выполнения работы:

1. Создала новый репозиторий и клонировала его на свой компьютер.

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner * / Repository name * LR_2.21
✔ LR_2.21 is available.

Great repository names are short and memorable. Need inspiration? How about [upgraded-fortnight](#) ?

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:

☒ Add a README file
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore
.gitignore template: Python
Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license
License: MIT License
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set `main` as the default branch. Change the default name in your [settings](#).

① You are creating a public repository in your personal account.

Рисунок 1. Создан новый репозиторий

2. Клонировала репозиторий на свой компьютер. В ходе данной лабораторной работы работала с моделью ветвления git-flow.

```
● @Ichizuchi →/workspaces/LR_2.21 (main) $ git clone https://github.com/Ichizuchi/LR_2.21.git
Cloning into 'LR_2.21'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
● @Ichizuchi →/workspaces/LR_2.21 (main) $ git branch develop
● @Ichizuchi →/workspaces/LR_2.21 (main) $ git checkout develop
Switched to branch 'develop'
○ @Ichizuchi →/workspaces/LR_2.21 (develop) $
```

Рисунок 2. Клонирование и ветка develop

Создала виртуальное окружение Anaconda с именем репозитория.

```

@Ichizuchi →/workspaces/LR_2.21 (develop) $ conda create -n myenv python=3.10
Retrieving notices: ...working... done
Channels:
- defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

environment location: /opt/conda/envs/myenv

added / updated specs:
- python=3.10

The following packages will be downloaded:

package | build
-----|-----
_libgcc_mutex-0.1 | main 3 KB
_openmp_mutex-5.1 | 1_gnu 21 KB
bzip2-1.0.8 | h5eee18b_5 262 KB
ca-certificates-2024.3.11 | h06a4308_0 127 KB
ld_impl_linux-64-2.38 | h1181459_1 654 KB
libffi-3.4.4 | h6a678d5_0 142 KB
libgcc-ng-11.2.0 | h1234567_1 5.3 MB
libgomp-11.2.0 | h1234567_1 474 KB
libstdc++-11.2.0 | h1234567_1 4.7 MB

```

Рисунок 3. Создание виртуального окружения

3. Работа над примером №1

The screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project structure for 'LR_2.21' with files like 'employees.db', 'main.py', 'mydatabase.db', 'primer1.py', 'README.md', and 'workers.db'. The code editor displays the content of 'primer1.py', which includes a function 'add_worker' and a function 'select_all'. The terminal shows the command 'python primer1.py display' being executed, resulting in a table of worker data.

```

76 def add_worker(database_path: Path, name: str, post: str, year: int) -> None:
77     """
78     INSERT INTO workers (worker_name, post_id, worker_year)
79     VALUES (?, ?, ?)
80     """
81     (name, post_id, year),
82 )
83 conn.commit()
84 conn.close()
85
86 def select_all(database_path: Path) -> t.List[t.Dict[str, t.Any]]:
87     """
88     Выбрать всех работников.
89     """
90     conn = sqlite3.connect(database_path)
91     cursor = conn.cursor()
92
93     cursor.execute(
94         """
95         SELECT workers.worker_name, posts.post_title, workers.worker_year
96         FROM workers
97         INNER JOIN posts ON posts.post_id = workers.post_id
98         """
99     )
100     rows = cursor.fetchall()
101
102     conn.close()
103     return [
104         {
105             "name": row[0],
106             "post": row[1],
107             "year": row[2],
108         }
109         for row in rows
110     ]

```

```

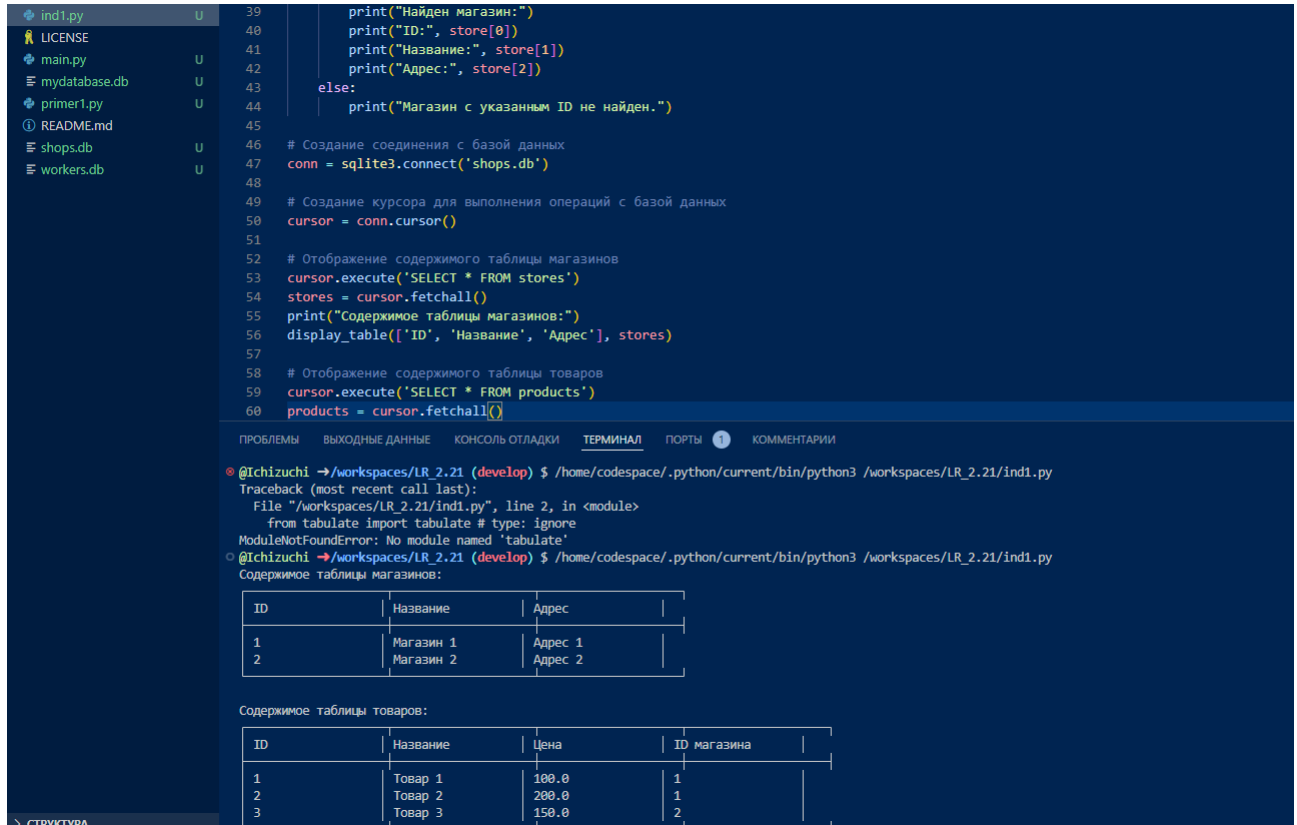
@Ichizuchi →/workspaces/LR_2.21 (develop) $ python primer1.py display
+-----+-----+-----+-----+
| No | Ф.И.О. | Должность | Год |
+-----+-----+-----+-----+
| 1 | Иванов Иван Иванович | Инженер | 2020 |
+-----+-----+-----+-----+

```

Рисунок 4. Результат выполнения

Выполнение индивидуального задания

С помощью команд sqlite3 создадим базу данных, а затем в ней создадим таблицы. Добавим магазины и товары в них с помощью команд консоли, а затем отобразим содержимое таблицы.



```
39     print("Найден магазин:")
40     print("ID:", store[0])
41     print("Название:", store[1])
42     print("Адрес:", store[2])
43 else:
44     print("Магазин с указанным ID не найден.")
45
46 # Создание соединения с базой данных
47 conn = sqlite3.connect('shops.db')
48
49 # Создание курсора для выполнения операций с базой данных
50 cursor = conn.cursor()
51
52 # Отображение содержимого таблицы магазинов
53 cursor.execute('SELECT * FROM stores')
54 stores = cursor.fetchall()
55 print("Содержимое таблицы магазинов:")
56 display_table(['ID', 'Название', 'Адрес'], stores)
57
58 # Отображение содержимого таблицы товаров
59 cursor.execute('SELECT * FROM products')
60 products = cursor.fetchall()
```

PROBLEMY Выходные данные КОНСОЛЬ ОТЛАДКИ **ТЕРМИНАЛ** ПОРТЫ 1 КОММЕНТАРИИ

@Ichizuchi →/workspaces/LR_2.21 (develop) \$ /home/codespace/.python/current/bin/python3 /workspaces/LR_2.21/ind1.py
Traceback (most recent call last):
 File "/workspaces/LR_2.21/ind1.py", line 2, in <module>
 from tabulate import tabulate # type: ignore
ModuleNotFoundError: No module named 'tabulate'
@Ichizuchi →/workspaces/LR_2.21 (develop) \$ /home/codespace/.python/current/bin/python3 /workspaces/LR_2.21/ind1.py
Содержимое таблицы магазинов:

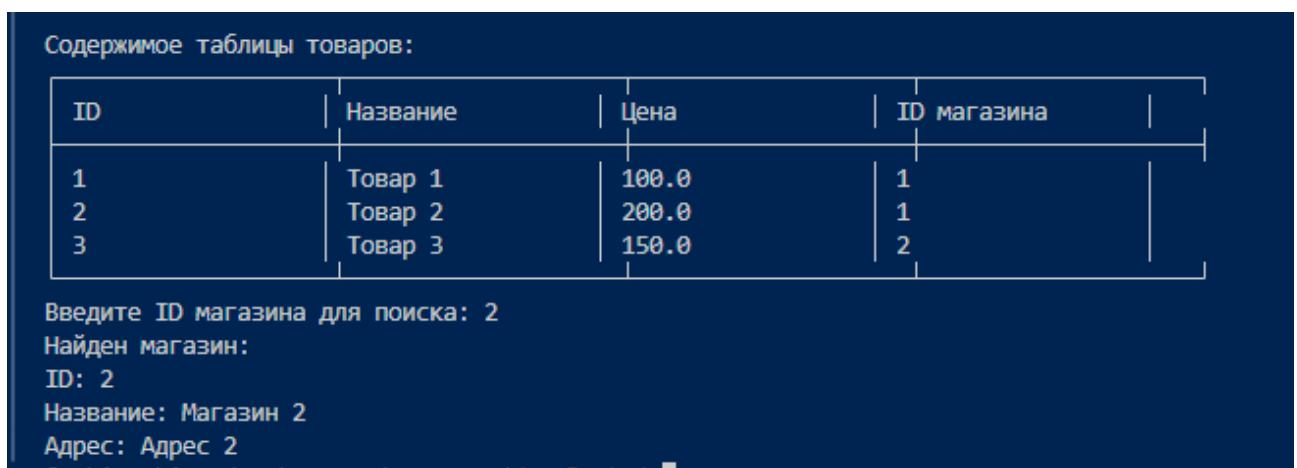
ID	Название	Адрес
1	Магазин 1	Адрес 1
2	Магазин 2	Адрес 2

Содержимое таблицы товаров:

ID	Название	Цена	ID магазина
1	Товар 1	100.0	1
2	Товар 2	200.0	1
3	Товар 3	150.0	2

Рисунок 5. Содержимое таблицы с магазинами

При помощи команды магазина найдем нужный магазин



Содержимое таблицы товаров:

ID	Название	Цена	ID магазина
1	Товар 1	100.0	1
2	Товар 2	200.0	1
3	Товар 3	150.0	2

Введите ID магазина для поиска: 2
Найден магазин:
ID: 2
Название: Магазин 2
Адрес: Адрес 2

Рисунок 6. Нужный магазин найден

1. Сформировала файлы environment.yml и requirements.txt

```

● @Ichizuchi →/workspaces/LR_2.21 (develop) $ conda env export > environment.yml
● @Ichizuchi →/workspaces/LR_2.21 (develop) $ pip freeze > requirements.txt
● @Ichizuchi →/workspaces/LR_2.21 (develop) $ conda init
no change      /opt/conda/condabin/conda
no change      /opt/conda/bin/conda
no change      /opt/conda/bin/conda-env
no change      /opt/conda/bin/activate
no change      /opt/conda/bin/deactivate
no change      /opt/conda/etc/profile.d/conda.sh
no change      /opt/conda/etc/fish/conf.d/conda.fish
no change      /opt/conda/shell/condabin/Conda.psm1
no change      /opt/conda/shell/condabin/conda-hook.ps1
no change      /opt/conda/lib/python3.12/site-packages/xontrib/conda.xsh
no change      /opt/conda/etc/profile.d/conda.csh
modified       /home/codespace/.bashrc

==> For changes to take effect, close and re-open your current shell. <==

```

Рисунок 7. Файлы environment.yml и requirements.txt

Ответы на контрольные вопросы

1. Непосредственно модуль sqlite3 – это API к СУБД SQLite. Своего рода адаптер, который переводит команды, написанные на Питоне, в команды, которые понимает SQLite. Как и наоборот, доставляет ответы от SQLite в python-программу.

2. Для взаимодействия с базой данных SQLite3 в Python необходимо создать объект cursor. Вы можете создать его с помощью метода cursor(). Курсор SQLite3 – это метод объекта соединения. Для выполнения инструкций SQLite3 сначала устанавливается соединение, а затем создается объект курсора с использованием объекта соединения

3. При создании соединения с SQLite3 автоматически создается файл базы данных, если он еще не существует. Этот файл базы данных создается на диске, мы также можем создать базу данных в оперативной памяти с помощью функции :memory: with the connect. Такая база данных называется базой данных в памяти.

4. С помощью команды закрытия close().

5. Чтобы вставить данные в таблицу, используется оператор INSERT INTO.

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

7. Оператор SELECT используется для выбора данных из определенной таблицы. Если вы хотите выбрать все столбцы данных из таблицы, вы можете использовать звездочку (*).

8. SQLite3 rowcount используется для возврата количества строк, которые были затронуты или выбраны последним выполненным SQLзапросом.

9. 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. При создании таблицы мы должны убедиться, что она еще не существует. Аналогично, при удалении/удалении таблицы она должна существовать. Чтобы проверить, не существует ли таблица уже, мы используем IF NOT EXISTS с оператором CREATE TABLE следующим образом.

11. Метод executemany можно использовать для вставки нескольких строк одновременно.

12. В базе данных Python SQLite3 мы можем легко хранить дату или время, импортируя модуль datetime . Следующие форматы являются наиболее часто используемыми форматами для datetime:

Вывод: в результате выполнения работы были приобретены навыки по работе с базовыми возможностями системы управления базами данных SQLite3.