

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 «Программирование на языке Python»

Вариант_15_

Выполнила:
Маньшина Дарья Алексеевна
2 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., канд. тех. наук,
доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

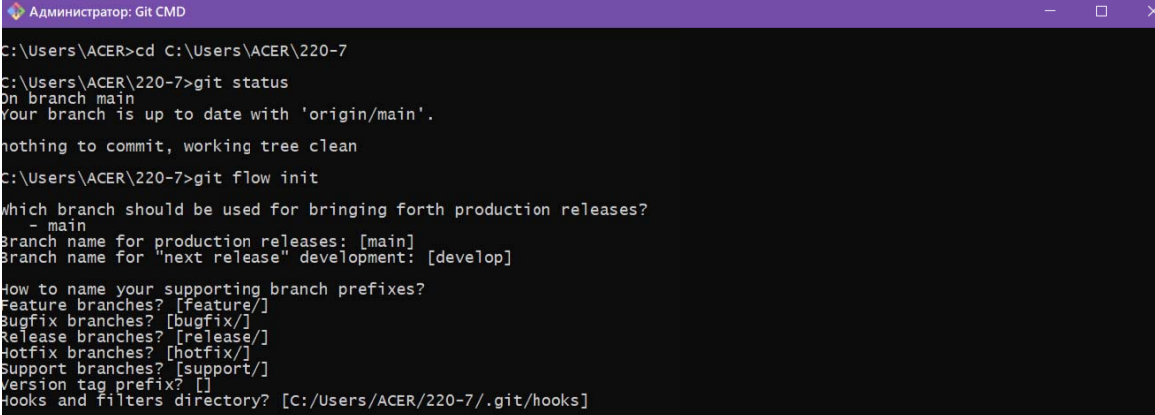
Ставрополь, 2023 г.

Тема: основы работы с SQLite3

Цель: исследовать базовые возможности системы управления базами данных SQLite3.

Ход работы:

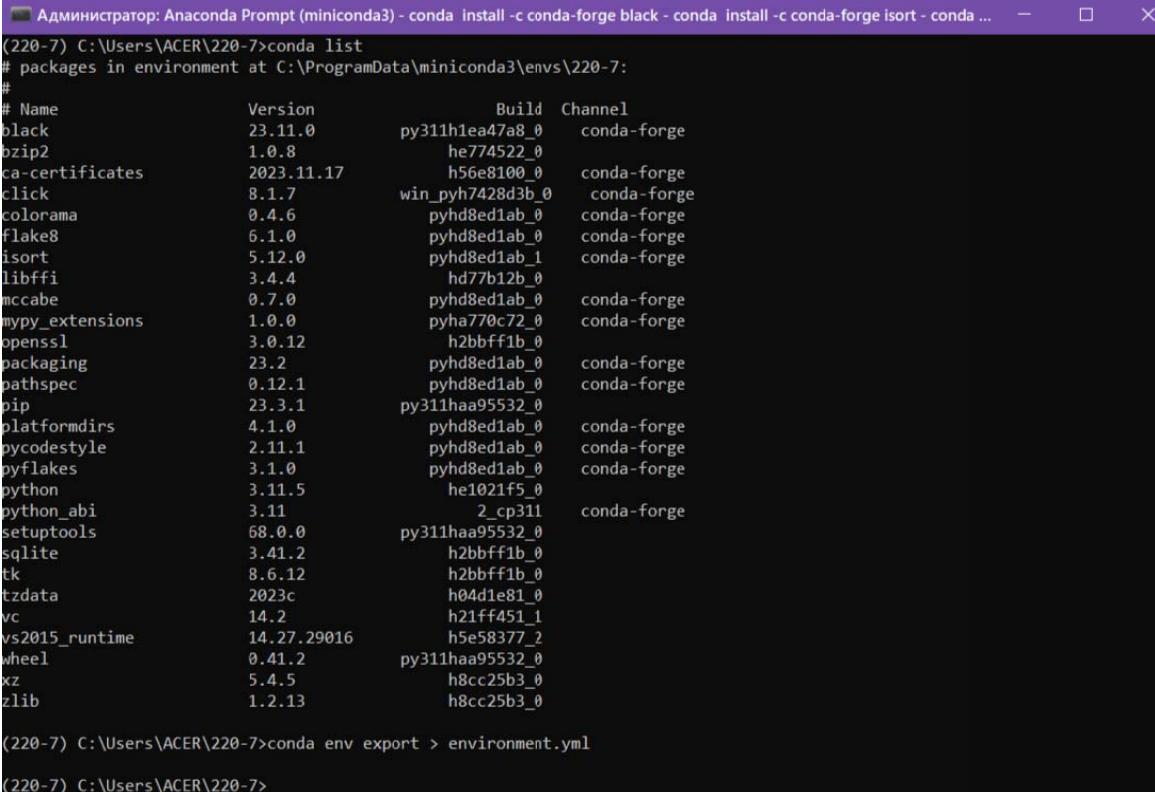
1. Создадим новый репозиторий. Клонировем его и сделаем способ ветвления git-flow. Также установим пакеты: black, isort, flake8



```
Администратор: Git CMD
C:\Users\ACER>cd C:\Users\ACER\220-7
C:\Users\ACER\220-7>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
C:\Users\ACER\220-7>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? [C:/Users/ACER/220-7/.git/hooks]
```

Рисунок 1. Клонирование репозитория



```
Администратор: Anaconda Prompt (miniconda3) - conda install -c conda-forge black - conda install -c conda-forge isort - conda ...
(220-7) C:\Users\ACER\220-7>conda list
# packages in environment at C:\ProgramData\miniconda3\envs\220-7:
#
# Name                                Version                                Build                                Channel
black                                 23.11.0                                py311h1ea47a8_0                      conda-forge
bzip2                                 1.0.8                                  he774522_0                          conda-forge
ca-certificates                      2023.11.17                             h56e8100_0                          conda-forge
click                                 8.1.7                                  win_pyh7428d3b_0                    conda-forge
colorama                              0.4.6                                  pyhd8ed1ab_0                        conda-forge
flake8                                6.1.0                                  pyhd8ed1ab_0                        conda-forge
isort                                 5.12.0                                  pyhd8ed1ab_1                        conda-forge
libffi                                3.4.4                                  hd77b12b_0                          conda-forge
mccabe                                0.7.0                                  pyhd8ed1ab_0                        conda-forge
mypy_extensions                      1.0.0                                  pyha770c72_0                        conda-forge
openssl                               3.0.12                                 h2bfff1b_0                          conda-forge
packaging                             23.2                                   pyhd8ed1ab_0                        conda-forge
pathspec                              0.12.1                                 pyhd8ed1ab_0                        conda-forge
pip                                   23.3.1                                 py311h9a95532_0                     conda-forge
platformdirs                         4.1.0                                  pyhd8ed1ab_0                        conda-forge
pycodestyle                           2.11.1                                 pyhd8ed1ab_0                        conda-forge
pyflakes                              3.1.0                                  pyhd8ed1ab_0                        conda-forge
python                                3.11.5                                 he1021f5_0                          conda-forge
python_abi                            3.11                                   2_cp311                             conda-forge
setuptools                            68.0.0                                 py311h9a95532_0                     conda-forge
sqlite                                 3.41.2                                 h2bfff1b_0                          conda-forge
tk                                     8.6.12                                 h2bfff1b_0                          conda-forge
tzdata                                2023c                                  h04d1e81_0                          conda-forge
vc                                     14.2                                   h21ffa51_1                          conda-forge
vs2015_runtime                       14.27.29016                           h5e58377_2                          conda-forge
wheel                                 0.41.2                                 py311h9a95532_0                     conda-forge
xz                                     5.4.5                                  h8cc25b3_0                          conda-forge
zlib                                   1.2.13                                 h8cc25b3_0                          conda-forge
(220-7) C:\Users\ACER\220-7>conda env export > environment.yml
(220-7) C:\Users\ACER\220-7>
```

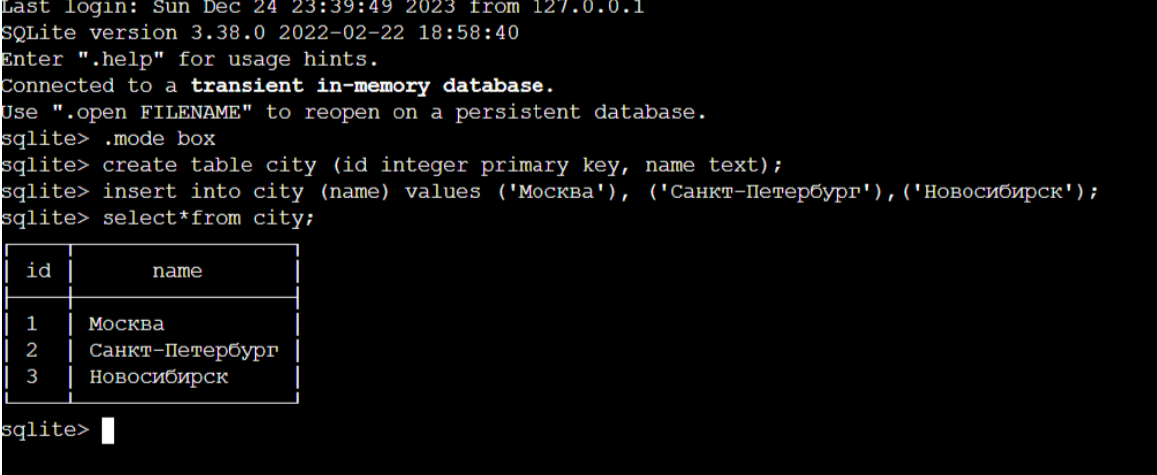
Рисунок 2. Установка пакетов

Выполним примеры, предложенные в методичке.

Программа:

```
sqlite> .mode box
sqlite> create table city (id integer primary key, name text);
sqlite> insert into city (name) values ('Москва'), ('Санкт-Петербург'),('Новосибирск');
sqlite> select*from city;
```

Результат:



```
Last login: Sun Dec 24 23:39:49 2023 from 127.0.0.1
SQLite version 3.38.0 2022-02-22 18:58:40
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .mode box
sqlite> create table city (id integer primary key, name text);
sqlite> insert into city (name) values ('Москва'), ('Санкт-Петербург'),('Новосибирск');
sqlite> select*from city;
```

id	name
1	Москва
2	Санкт-Петербург
3	Новосибирск

```
sqlite> 
```

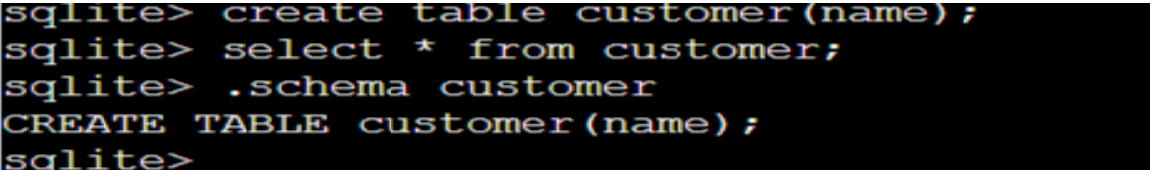
Рисунок 3. Проверка работы песочницы

Далее решим предложенные задания.

Программа задания №1:

```
sqlite> create table customer(name);
sqlite> select * from customer;
sqlite> .schema customer
```

Результат:



```
sqlite> create table customer(name);
sqlite> select * from customer;
sqlite> .schema customer
CREATE TABLE customer(name);
sqlite>
```

Рисунок 4. Выполнение задания №1

Условие задания №2: с помощью команды .help найти в песочнице команду, которая отвечает за вывод времени выполнения запроса.

Программа:

```
create table city(name);
.timer on
select count(*) from city;
```

Результат:

```
sqlite> create table city(name);
sqlite> .timer on
sqlite> select count(*) from city;
0
Run Time: real 0.000 user 0.000106 sys 0.000053
sqlite>
```

Рисунок 5. Выполнение задания №2

Условие задания №3: загрузить файл city.csv в песочнице, затем выполнить запрос, который вернет число, получить число после выполнения запроса.

Программа примера №3:

```
.import --csv city.csv city
select max(length(city)) from city;
```

Результат:

```
sqlite> .import --csv city.csv city
sqlite> select max(length(city)) from city;
25
sqlite>
```

Рисунок 6. Выполнение задания №3

Условие задания №4: решить задачу: загрузить файл city.csv в песочницу с помощью команды .import , но без использования опции --csv .

Программа примера №4:

```
.mode csv
.import city.csv city
```

Результат:

```
sqlite> .mode csv
sqlite> .import city.csv city
sqlite>
```

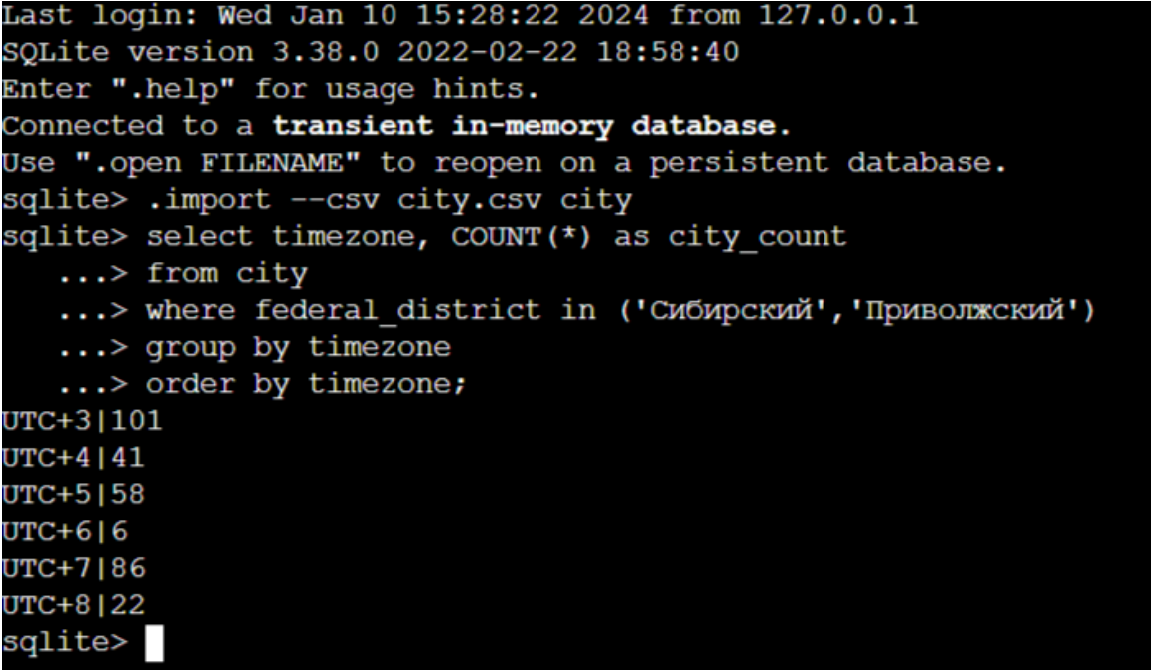
Рисунок 7. Выполнение задания №4

Условие задания №5: написать в песочнице запрос, который посчитает количество городов для каждого часового пояса в Сибирском и Приволжском федеральных округах. Вывести столбцы `timezone` и `city_count`, отсортируйте по значению часового пояса.

Программа:

```
.import --csv city.csv city
select timezone, COUNT(*) as city_count
from city
where federal_district in ('Сибирский','Приволжский')
group by timezone
order by timezone;
```

Результат:



```
Last login: Wed Jan 10 15:28:22 2024 from 127.0.0.1
SQLite version 3.38.0 2022-02-22 18:58:40
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .import --csv city.csv city
sqlite> select timezone, COUNT(*) as city_count
...> from city
...> where federal_district in ('Сибирский','Приволжский')
...> group by timezone
...> order by timezone;
UTC+3|101
UTC+4|41
UTC+5|58
UTC+6|6
UTC+7|86
UTC+8|22
sqlite> 
```

Рисунок 8. Выполнение задания №5

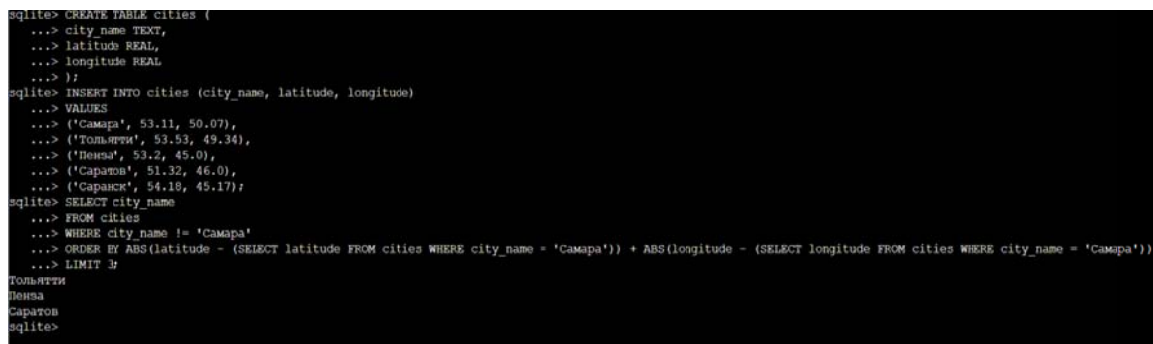
Условие задания №6: написать в песочнице запрос, который найдет три ближайших к Самаре города, не считая саму Самару. Указать в ответе названия этих трех городов через запятую в порядке удаления от Самары.

Программа примера №6:

```
CREATE TABLE cities (
city_name TEXT,
latitude REAL,
longitude REAL
);
INSERT INTO cities (city_name, latitude, longitude)
```

```
VALUES
('Самара', 53.11, 50.07),
('Тольятти', 53.53, 49.34),
('Пенза', 53.2, 45.0),
('Саратов', 51.32, 46.0),
('Саранск', 54.18, 45.17);
SELECT city_name
FROM cities
WHERE city_name != 'Самара'
ORDER BY ABS(latitude - (SELECT latitude FROM cities WHERE city_name = 'Самара')) +
ABS(longitude - (SELECT longitude FROM cities WHERE city_name = 'Самара'))
LIMIT 3;
```

Результат:



```
sqlite> CREATE TABLE cities (
...> city_name TEXT,
...> latitude REAL,
...> longitude REAL
...> );
sqlite> INSERT INTO cities (city_name, latitude, longitude)
...> VALUES
...> ('Самара', 53.11, 50.07),
...> ('Тольятти', 53.53, 49.34),
...> ('Пенза', 53.2, 45.0),
...> ('Саратов', 51.32, 46.0),
...> ('Саранск', 54.18, 45.17);
sqlite> SELECT city_name
...> FROM cities
...> WHERE city_name != 'Самара'
...> ORDER BY ABS(latitude - (SELECT latitude FROM cities WHERE city_name = 'Самара')) + ABS(longitude - (SELECT longitude FROM cities WHERE city_name = 'Самара'))
...> LIMIT 3;
Тольятти
Пенза
Саратов
sqlite>
```

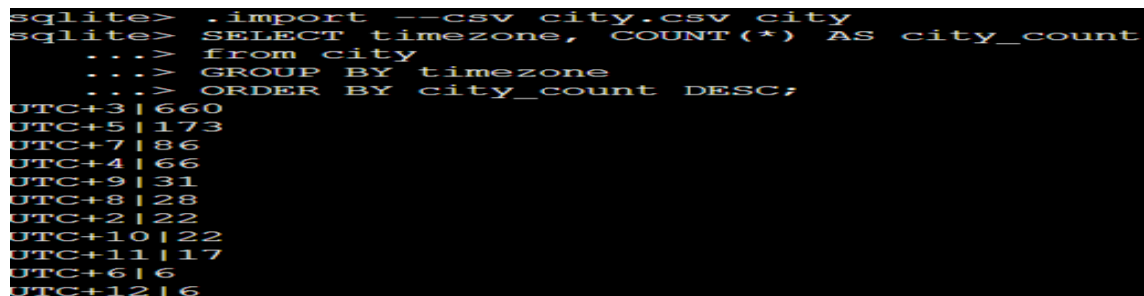
Рисунок 9. Выполнение задания №6

Условие задания №7: написать в песочнице запрос, который посчитает количество городов в каждом часовом поясе. Отсортировать по количеству городов по убыванию.

Программа:

```
import --csv city.csv city
SELECT timezone, COUNT(*) AS city_count
from city
GROUP BY timezone
ORDER BY city_count DESC;
```

Результат:



```
sqlite> .import --csv city.csv city
sqlite> SELECT timezone, COUNT(*) AS city_count
...> from city
...> GROUP BY timezone
...> ORDER BY city_count DESC;
UTC+3|660
UTC+5|173
UTC+7|86
UTC+4|66
UTC+9|31
UTC+8|28
UTC+2|22
UTC+10|22
UTC+11|17
UTC+6|6
UTC+12|6
```

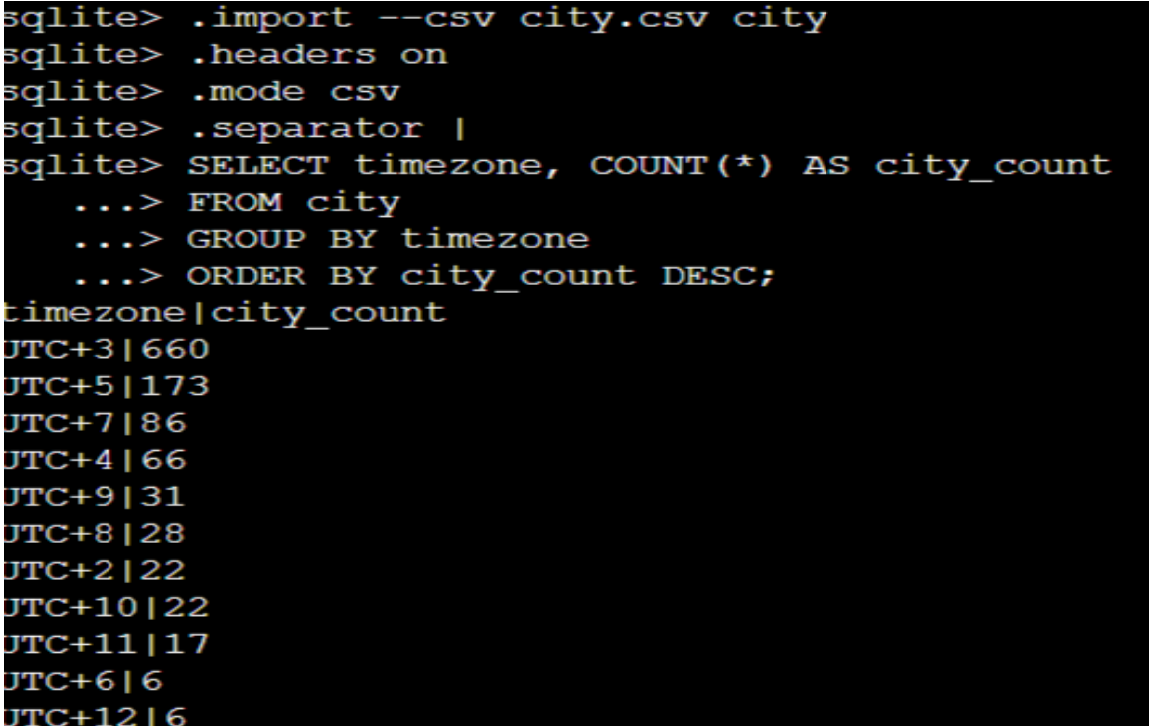
Рисунок 10. Выполнение задания №7

Условие задания №8: написать в песочнице запрос, который посчитает количество городов в каждом часовом поясе. Отсортировать по количеству городов по убыванию. Результат должен быть: в формате CSV, с заголовками

Программа:

```
.import --csv city.csv city
.headers on
.mode csv
.separator |
SELECT timezone, COUNT(*) AS city_count
FROM city
GROUP BY timezone
ORDER BY city_count DESC;
```

Результат:



```
sqlite> .import --csv city.csv city
sqlite> .headers on
sqlite> .mode csv
sqlite> .separator |
sqlite> SELECT timezone, COUNT(*) AS city_count
...> FROM city
...> GROUP BY timezone
...> ORDER BY city_count DESC;
timezone|city_count
UTC+3|660
UTC+5|173
UTC+7|86
UTC+4|66
UTC+9|31
UTC+8|28
UTC+2|22
UTC+10|22
UTC+11|17
UTC+6|6
UTC+12|6
```

Рисунок №11. Выполнение задания №8

Выполним индивидуальное задание. Его условие: загрузить в SQLite выбранный датасет в формате CSV (датасет можно найти на сайте Kaggle). Сформировать более пяти запросов к таблицам БД.

Для добавления файла json используем команду:

```
sqlite> .mode json
sqlite> .output bim.json
```

Рисунок №12. Добавления файла json

Позже выполним несколько запросов. Нам необходимо сделать от пяти и более.

```
sqlite> .mode csv
sqlite> .headers on
sqlite> .output billionaires_country.csv
sqlite> SELECT country, COUNT(*) as count
...> FROM billionaires
...> GROUP BY country;
sqlite>
```

Рисунок №13. Первый запрос

```
sqlite> .mode csv
sqlite> .headers on
sqlite> .output billionaires_50.csv
sqlite> SELECT name, net_worth
...> FROM billionaires
...> WHERE net_worth > 50.0;
sqlite> _
```

Рисунок №14. Второй запрос

```
sqlite> .mode csv
sqlite> .headers on
sqlite> .output top10_billionaires.csv
sqlite> SELECT name, net_worth
...> FROM billionaires
...> ORDER BY net_worth DESC
...> LIMIT 10;
sqlite>
```

Рисунок №15. Третий запрос

```
sqlite> .mode csv
sqlite> .headers on
sqlite> .output billioners_strani.csv
sqlite> SELECT country, AVG(net_worth) as average_net_worth
...> FROM billionaires
...> GROUP BY country;
sqlite>
```

Рисунок №16. Четвертый запрос

```
sqlite> .mode csv
sqlite> .headers on
sqlite> .output billioners_min.csv
sqlite> SELECT name, net_worth
...> FROM billionaires
...> WHERE net_worth = (SELECT MIN(net_worth) FROM billionaires);
sqlite> _
```

Рисунок №17. Пятый запрос

Вывод: в ходе лабораторной работы исследовала базовые возможности системы управления базами данных SQLite3.

Контрольные вопросы

1. Каково назначение реляционных баз данных и СУБД?

Реляционные базы данных и системы управления базами данных (СУБД) служат для хранения и извлечения данных. СУБД предоставляет инструменты для создания и управления базами данных, а также для работы с данными, хранящимися в них.

2. Каково назначение языка SQL?

Язык структурированных запросов (SQL) используется для взаимодействия с реляционными базами данных. Он позволяет выполнять операции над данными (например, выборку, добавление, изменение и удаление данных), а также управлять структурой базы данных (создавать и изменять таблицы, индексы и т. д.).

3. Из чего состоит язык SQL?

SQL состоит из набора операторов, которые используются для выполнения различных операций над данными. Основные операторы включают SELECT для выборки данных, INSERT для добавления новых записей, UPDATE для изменения существующих записей и DELETE для удаления записей.

4. В чем отличие СУБД SQLite от клиент-серверных СУБД?

SQLite - это компактная, быстрая и простая для использования СУБД, которая хранит данные в одном файле. Она не требует сервера для своей работы и может использоваться локально на компьютере пользователя. Отличие SQLite от клиент-серверных СУБД, таких как MySQL или

PostgreSQL, заключается в том, что SQLite не требует отдельного сервера и может быть использован на любом компьютере.

5. Как установить SQLite в Windows и Linux?

SQLite доступен для Windows, macOS и Linux, и его можно установить с помощью пакетов программного обеспечения для этих операционных систем. Для установки SQLite на Windows можно использовать менеджер пакетов, например, Chocolatey, или установить его вручную, скачав архив с официального сайта. Для установки на Linux можно использовать менеджер пакетов вашего дистрибутива или установить SQLite вручную, скачав исходный код или бинарный пакет с официального сайта.

6. Как создать базу данных SQLite?

Создание базы данных SQLite можно выполнить с помощью утилиты командной строки `sqlite3` или через графический интерфейс, например, DB Browser for SQLite. Сначала нужно создать новый файл базы данных с расширением `.db`, а затем использовать утилиту `sqlite3` для создания структуры базы данных.

7. Как выяснить в SQLite какая база данных является текущей?

Чтобы выяснить, какая база данных является текущей, можно использовать команду `.database` в утилите `sqlite3`. Эта команда переключает контекст на указанную базу данных. Если база данных не указана, команда выводит текущую базу данных.

8. Как создать и удалить таблицу в SQLite?

Создание таблицы SQLite выполняется с помощью команды `CREATE TABLE`, удаление — `DROP TABLE`.

9. Что является первичным ключом в таблице?

Первичным ключом является уникальный идентификатор записи в таблице, обычно это поле с автоинкрементом.

10. Как сделать первичный ключ таблицы автоинкрементным?

Сделать первичный ключ автоинкрементным можно с помощью ключевого слова `AUTOINCREMENT` в команде `CREATE TABLE`.

11. Каково назначение инструкций `NOT` и `DEFAULT` при создании таблиц?

Инструкция `NOT` используется при создании индексов для указания того, что индекс не уникальный. `DEFAULT` используется при определении столбцов таблицы для указания значения по умолчанию.

12. Каково назначение внешних ключей в таблице? Как создать внешний ключ в таблице?

Внешние ключи используются для обеспечения ссылочной целостности между таблицами. Они служат для связывания двух таблиц таким образом, что изменение данных в одной таблице может привести к изменению данных в другой таблице. Создание внешнего ключа осуществляется с помощью ключевого слова `REFERENCES` в команде создания таблицы.

13. Как выполнить вставку строки в таблицу базы данных SQLite?

Вставка строки в таблицу SQLite осуществляется с помощью команды `INSERT`.

14. Как выбрать данные из таблицы SQLite?

Выборка данных из таблицы SQLite осуществляется с помощью оператора `SELECT`.

15. Как ограничить выборку данных с помощью условия WHERE?

Условие WHERE позволяет ограничить выборку данными, удовлетворяющими определенному условию.

16. Как упорядочить выбранные данные?

Упорядочить выборку можно с помощью предложения ORDER BY.

17. Как выполнить обновление записей в таблице SQLite?

Обновление записей в таблице осуществляется с помощью оператора UPDATE.

18. Как удалить записи из таблицы SQLite?

Удаление записей из таблицы осуществляется с помощью оператора DELETE.

19. Как сгруппировать данные из выборки из таблицы SQLite?

Группировка данных осуществляется с помощью агрегатных функций, таких как MIN, MAX, COUNT и т.д.

20. Как получить значение агрегатной функции (например: минимум, максимум, количество записей и т. д.) в выборке из таблицы SQLite?

Агрегатные функции можно использовать вместе с оператором SELECT.

21. Как выполнить объединение нескольких таблиц в операторе SELECT?

Объединение нескольких таблиц в одном запросе SELECT можно осуществить с помощью конструкции JOIN. Существуют различные типы соединений: INNER JOIN, LEFT JOIN, RIGHT JOIN и др.

22. Каково назначение подзапросов и шаблонов при работе с таблицами SQLite?

Подзапросы используются для определения данных, которые будут использоваться в основном запросе. Шаблоны используются для упрощения написания сложных запросов.

23. Каково назначение представлений VIEW в SQLite?

Представления VIEW используются для создания виртуальных таблиц, которые могут быть использованы в запросах точно так же, как и реальные таблицы.

24. Какие существуют средства для импорта данных в SQLite?

Средствами для импорта данных могут служить различные форматы файлов, такие как CSV, JSON, XML и др. Также можно использовать инструменты для работы с базами данных, такие как phpMyAdmin или MySQL Workbench.

25. Каково назначение команды .schema ?

Команда .schema выводит структуру текущей базы данных. Она может быть использована для получения информации о таблицах, полях и индексах.

26. Как выполняется группировка и сортировка данных в запросах SQLite?

Группировка и сортировка выполняются с помощью ключевых слов GROUP BY и ORDER BY соответственно.

27. Каково назначение "табличных выражений" в SQLite?

Табличные выражения используются для создания временных таблиц на основе результатов других запросов.

28. Как осуществляется экспорт данных из SQLite в форматы CSV и JSON?

Экспорт данных в форматы CSV и JSON можно выполнить с помощью инструментов для работы с базами данных или с использованием встроенных функций SQLite.

29. Какие еще форматы для экспорта данных Вам известны?

Кроме форматов CSV и JSON, SQLite также поддерживает экспорт данных в формат XML.

#