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 №2.20
дисциплины «Анализ данных»

Выполнил:
Говоров Егор Юрьевич
2 курс, группа ИВТ-б-о-22-1,
15.05.2024«Информатика и вычислительная техника»,
направленность (профиль) «Информатика и вычислительная техника», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р. А., 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:

Задание 1. Решить задачу: выполнить в песочнице команды:

```
create table customer(name);

select *
from customer;

.schema customer
```

Рисунок 1. Команды для выполнения

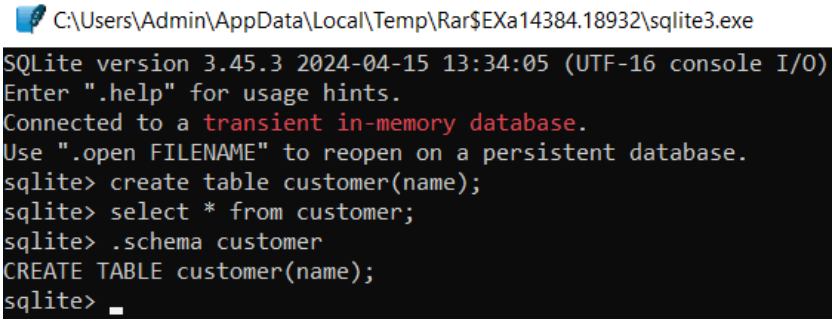

Рисунок 2. Выполнение первого общего задания

Вот что здесь происходит:

1. Первая команда (create table) создает таблицу customer с единственным столбцом name .

2. Вторая команда (select) показывает содержимое таблицы customer (она пустая).

3. Третья команда (.schema) показывает список и структуру всех таблиц в базе.

create и select — это SQL-запросы, часть стандарта SQL. Запрос может занимать несколько строк, а в конце всегда ставится точка с запятой.

.schema — это специальная команда SQLite, не часть стандарта SQL.

Задание 2. Решить задачу: с помощью команды .help найти в песочнице команду, которая отвечает за вывод времени выполнения запроса.

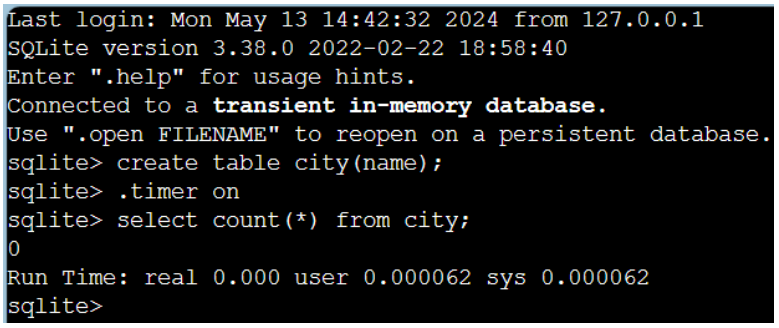

Рисунок 3. Выполнение второго общего задания

Задание 3. Решить задачу: загрузить файл city.csv в песочнице, затем выполнить запрос, который вернет число, получить число после выполнения запроса.

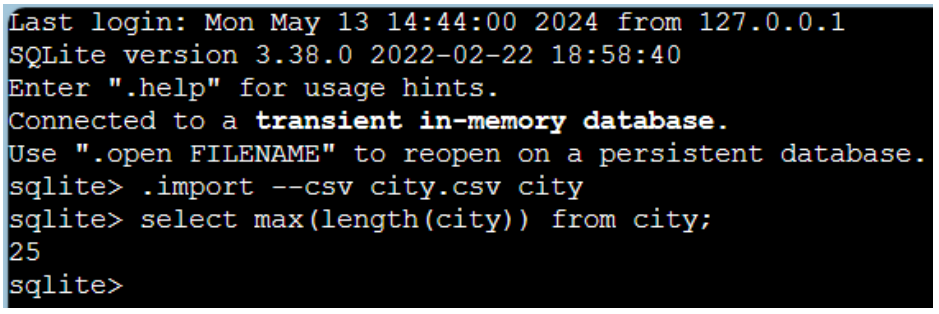

Рисунок 4. Выполнение третьего общего задания

Задание 4. Решить задачу: загрузить файл city.csv в песочницу с помощью команды .import , но без использования опции --csv . Эта опция появилась только в недавней версии SQLite (3.32, май 2020), так что полезно знать способ, подходящий для старых версий.

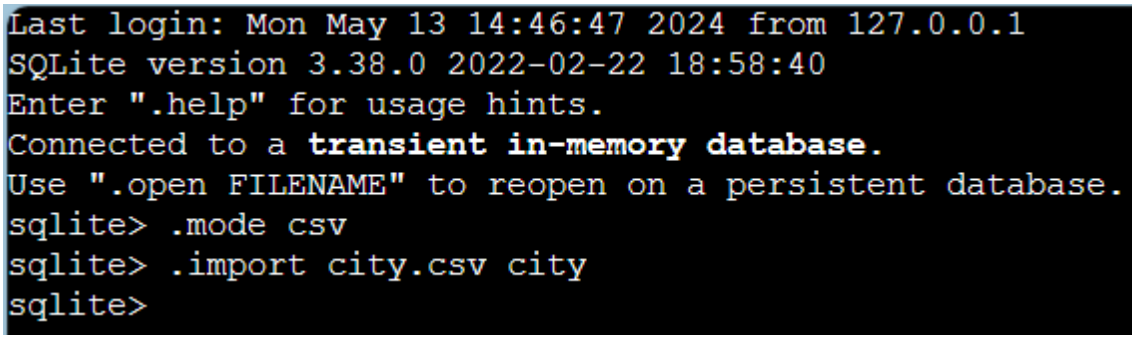

Рисунок 5. Выполнение четвертого общего задания

Задание 5. Решить задачу: 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:

```
Last login: Mon May 13 14:56:07 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 6. Выполнение пятого общего задания

Задание 6. Решить задачу: 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.

```
Last login: Mon May 13 14:57:14 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> CREATE TABLE cities (
   ...> city_name TEXT,
   ...> latitude REAL,
   ...> longitude REAL
   ...> );
sqlite> INSERT INTO cities (city_name, latitude, longitude)
   ...> VALUES
   ...> ('Самара', 53.195873, 50.100193),
   ...> ('Ульяновск', 54.308067, 48.374867),
   ...> ('Пенза', 53.200066, 45.004944),
   ...> ('Саранск', 54.180760, 45.186226);
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;
Ульяновск
Пенза
Саранск
sqlite>
```

Рисунок 7. Выполнение шестого общего задания

Задание 7. Решить задачу: написать в песочнице запрос, который посчитает количество городов в каждом часовом поясе. Отсортировать по количеству городов по убыванию.

```
Last login: Mon May 13 15:07:35 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
   ...> 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
sqlite>
```

Рисунок 8. Выполнение седьмого общего задания

Выполнить этот же запрос, но так, чтобы результат был

- в формате CSV
- с заголовками
- с разделителем «pipe» |

```
Last login: Mon May 13 15:11:31 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> .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
sqlite>
```

Рисунок 9. Выполнение седьмого общего с изменениями

Индивидуальное задание

Условие задания: Загрузить в SQLite выбранный датасет в формате CSV (датасет можно найти на сайте Kaggle). Сформировать более пяти запросов к таблицам БД. Выгрузить результат выполнения запросов в форматы CSV и JSON.

Скачал датасет с сайта Kaggle под названием Pokemon, который содержит информацию о покемонах.

Создадим запросы к таблицам БЗ.

```
sqlite> .mode csv
sqlite> .import Pokemon.csv Pokemon
```

Рисунок 10. Импортирование датасета в таблицу

```
sqlite> CREATE TABLE IF NOT EXISTS Pokemon (
(x1...>     number INTEGER,
(x1...>     name TEXT,
(x1...>     type1 TEXT,
(x1...>     type2 TEXT,
(x1...>     total INTEGER,
(x1...>     hp INTEGER,
(x1...>     attack INTEGER,
(x1...>     defense INTEGER,
(x1...>     sp_attack INTEGER,
(x1...>     sp_defense INTEGER,
(x1...>     speed INTEGER,
(x1...>     generation INTEGER,
(x1...>     legendary TEXT
(x1...> );
sqlite> .mode csv
sqlite> .import Pokemon.csv Pokemon
sqlite> SELECT * FROM Pokemon LIMIT 5;
number,name,type1,type2,total,hp,attack,defense,sp_attack,sp_defense,speed,generation,legendary
1,Bulbasaur,Grass,Poison,318,45,49,49,65,65,45,1,FALSE
2,Ivysaur,Grass,Poison,405,60,62,63,80,80,60,1,FALSE
3,Venusaur,Grass,Poison,525,80,82,83,100,100,80,1,FALSE
3,"Mega Venusaur",Grass,Poison,625,80,100,123,122,120,80,1,FALSE
sqlite> .header on
sqlite> .output Pokemon_first_five.csv
sqlite> SELECT * FROM Pokemon LIMIT 5;
sqlite> .output Pokemon_first_five.json
sqlite> SELECT * FROM Pokemon LIMIT 5;
sqlite> .output stdout
sqlite>
```

Рисунок 11. Запрос на вывод первых пяти покемонов

```
sqlite> CREATE TABLE IF NOT EXISTS Pokemon (
(x1...>     number INTEGER,
(x1...>     name TEXT,
(x1...>     type1 TEXT,
(x1...>     type2 TEXT,
(x1...>     total INTEGER,
(x1...>     hp INTEGER,
(x1...>     attack INTEGER,
(x1...>     defense INTEGER,
(x1...>     sp_attack INTEGER,
(x1...>     sp_defense INTEGER,
(x1...>     speed INTEGER,
(x1...>     generation INTEGER,
(x1...>     legendary TEXT
(x1...> );
sqlite> .mode csv
sqlite> .import Pokemon.csv Pokemon
sqlite> .output Pokemon_gen1.csv
sqlite> SELECT * FROM Pokemon WHERE generation = 1;
sqlite> .output Pokemon_gen1.json
sqlite> SELECT * FROM Pokemon WHERE generation = 1;
sqlite> .output stdout
sqlite>
```

Рисунок 12. Запрос на вывод первого поколения

```
SQLite version 3.45.3 2024-04-15 13:34:05 (UTF-16 console I/O)
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> CREATE TABLE IF NOT EXISTS Pokemon (
(x1...>     number INTEGER,
(x1...>     name TEXT,
(x1...>     type1 TEXT,
(x1...>     type2 TEXT,
(x1...>     total INTEGER,
(x1...>     hp INTEGER,
(x1...>     attack INTEGER,
(x1...>     defense INTEGER,
(x1...>     sp_attack INTEGER,
(x1...>     sp_defense INTEGER,
(x1...>     speed INTEGER,
(x1...>     generation INTEGER,
(x1...>     legendary TEXT
(x1...> );
sqlite> .mode csv
sqlite> .import Pokemon.csv Pokemon
sqlite> .header on
sqlite> .output Pokemon_total_gt_500.csv
sqlite> SELECT * FROM Pokemon WHERE total > 500;
sqlite> .output Pokemon_total_gt_500.json
sqlite> SELECT * FROM Pokemon WHERE total > 500;
sqlite> .output stdout
sqlite>
```

Рисунок 13. Урон > 500

```
SQLite version 3.45.3 2024-04-15 13:34:05 (UTF-16 console I/O)
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> CREATE TABLE IF NOT EXISTS Pokemon (
(x1...>     number INTEGER,
(x1...>     name TEXT,
(x1...>     type1 TEXT,
(x1...>     type2 TEXT,
(x1...>     total INTEGER,
(x1...>     hp INTEGER,
(x1...>     attack INTEGER,
(x1...>     defense INTEGER,
(x1...>     sp_attack INTEGER,
(x1...>     sp_defense INTEGER,
(x1...>     speed INTEGER,
(x1...>     generation INTEGER,
(x1...>     legendary TEXT
(x1...> );
sqlite> .mode csv
sqlite> .import Pokemon.csv Pokemon
sqlite> .header on
sqlite> .output Pokemon_fire_water.csv
sqlite> SELECT * FROM Pokemon WHERE type1 IN ('Fire', 'Water');
sqlite> .output Pokemon_fire_water.json
sqlite> SELECT * FROM Pokemon WHERE type1 IN ('Fire', 'Water');
sqlite> .output stdout
sqlite>
```

Рисунок 14. Огненный и водный тип

Ответы на контрольные вопросы:

1. **Каково назначение реляционных баз данных и СУБД?**

- Реляционные базы данных предназначены для организации и хранения данных в виде таблиц с отношениями между ними.
- Системы управления базами данных (СУБД) предоставляют средства для создания, управления и обращения с базами данных.

2. **Каково назначение языка SQL?**

SQL (Structured Query Language) используется для взаимодействия с реляционными базами данных. Он предоставляет стандартизированный способ создания, изменения, управления и запросов к данным в базе данных.

3. **Из чего состоит язык SQL?**

SQL состоит из нескольких подмножеств:

- DDL (Data Definition Language): Определение структуры базы данных (CREATE, ALTER, DROP).
- DML (Data Manipulation Language): Манипуляция данными (SELECT, INSERT, UPDATE, DELETE).
- DCL (Data Control Language): Управление доступом и правами (GRANT, REVOKE).

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

- SQLite является встроенной базой данных и хранится в виде одного файла.
- Клиент-серверные СУБД (например, MySQL, PostgreSQL) имеют отдельные серверы, к которым подключаются клиенты для доступа к данным.

5. **Как установить SQLite в Windows и Linux?**

- Windows: Можно загрузить исполняемый файл SQLite с официального сайта и выполнить установку.
- Linux: В большинстве дистрибутивов Linux SQLite уже установлен. Для установки можно воспользоваться менеджером пакетов (например, sudo apt-get install sqlite в Ubuntu).

6. **Как создать базу данных SQLite?**

- В командной строке SQLite: sqlite3 имя_базы_данных.db.

- Внутри SQLite: CREATE DATABASE имя_базы_данных;.

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

- В командной строке SQLite: .database или .dbinfo.
- Внутри SQLite: PRAGMA database_list;

8. Как создать и удалить таблицу в SQLite?

Создание таблицы:

```
CREATE TABLE название (
поле1 тип1,
 ...
);
```

Удаление таблицы:

DROP TABLE название;

9. Что является первичным ключом в таблице?

Первичный ключ (Primary Key) в таблице — это уникальный идентификатор каждой записи. Он обеспечивает уникальность идентификации записей в таблице.

10. Как сделать первичный ключ таблицы автоинкрементным?

При создании таблицы в SQLite можно сделать поле первичного ключа автоинкрементным, используя ключевое слово AUTOINCREMENT

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

- NOT NULL: Гарантирует, что значение в столбце не может быть NULL (пустым).
- DEFAULT value: Устанавливает значение по умолчанию для столбца, если вставляемые данные не предоставляют значение для этого столбца.

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

Внешний ключ (Foreign Key) используется для связи двух таблиц по значениям в столбцах. Он обеспечивает целостность ссылочной целостности данных.

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

Используйте оператор INSERT INTO.

14. Как выбрать данные из таблицы SQLite?

Используйте оператор SELECT.

15. Как ограничить выборку данных с помощью условия WHERE?

Используйте WHERE для установки условий:

```
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

16. Как упорядочить выбранные данные?

Используйте ORDER BY.

17. Как выполнить обновление записей в таблице SQLite?

Используйте UPDATE.

18. Как удалить записи из таблицы SQLite?

Используйте DELETE.

19. Как сгруппировать данные из выборке из таблицы SQLite?

Используйте GROUP BY.

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

Используйте агрегатные функции, такие как MIN, MAX, SUM, AVG, и т. д.:

21. Как выполнить объединение нескольких таблиц в операторе SELECT?

Используйте JOIN.

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

Подзапросы используются для вложенных запросов, а шаблоны предоставляют средства создания более обобщенных запросов.

23. Каково назначение представлений VIEW в SQLite?

Представления позволяют создавать виртуальные таблицы на основе результатов запросов, что облегчает повторное использование и улучшает структуру запросов.

24. Какие существуют средства для импорта данных в SQLite?

QLite предоставляет команды .import и .read для импорта данных из внешних источников, а также можно использовать SQL-запросы с оператором INSERT.

25. Каково назначение команды .schema ?

Команда .schema используется для вывода SQL-кода, описывающего структуру базы данных, включая определение таблиц, индексов и других объектов

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

Используйте GROUP BY для объединения строк по значениям в одном или нескольких столбцах.

27. Каково назначение "табличных выражений" в SQLite?

Табличные выражения (Common Table Expressions или CTE) позволяют создавать временные результаты запросов, которые можно использовать внутри других запросов. Это обеспечивает более чистый и структурированный код SQL.

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

Экспорт в CSV: Используйте команду .mode csv перед выполнением запроса, а затем .output filename.csv для указания файла вывода.

29. Какие еще форматы для экспорта данных Вам известны?

Дополнительно к CSV и JSON, SQLite поддерживает экспорт в форматах XML, HTML, и SQL INSERT. Для экспорта в эти форматы также можно использовать соответствующие команды .mode и .output:

Вывод: исследовал базовые возможности системы управления базами данных SQLite3.