МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное автономное

образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

«Основы работы с SQLite3»

Отчет по лабораторной работе № 2.20

по дисциплине «Основы программной инженерии»

Выполнил студент группы ПИЖ-б-о-21-1

Кучеренко С. Ю. _____« » 2023г.

Подпись студента_____

Работа защищена « »_____2023г.

Проверил Воронкин Р.А. _____

(подпись)

Ставрополь 2023

Цель работы: исследовать базовые возможности системы управления базами данных SQLite3.

Ход работы:

1. Изучить теоретический материал работы.

2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия

MIT и выбранный Вами язык программирования (выбор языка программирования будет доступен после установки флажка Add .gitignore).

3. Выполните клонирование созданного репозитория на рабочий компьютер.

4. Дополните файл .gitignore необходимыми правилами для выбранного языка программирования и интегрированной среды разработки.

5. Добавьте в файл README.md информацию о группе и ФИО студента, выполняющего лабораторную работу.

6. Добавьте файл README и зафиксируйте сделанные изменения.

7. Решите задачу: выполните в песочнице команды:

```
create table customer(name);

select *
from customer;

.schema customer
```

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

Рисунок 1 – Решение

8. Решите задачу: с помощью команды .help найдите в песочнице команду, которая отвечает за вывод времени выполнения запроса. Если ее включить, в результатах запроса добавится строчка:

```
Run Time: real XXX user XXX sys XXX
```

```
sqlite> .timer ON
sqlite> select count(*) from customer;
0
Run Time: real 0.000 user 0.000185 sys 0.000000
sqlite>
```

Рисунок 2 – Решение

9. Решите задачу: загрузите файл city.csv в песочнице:

```
.import --csv city.csv city
```

Затем выполните такой запрос:

```
select max(length(city)) from city;
```

Какое число он вернул?

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

Рисунок 3 – Решение

10. Решите задачу: загрузите файл city.csv в песочнице 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. Вам поможет команда .help import . Всего должно получиться две команды:

```
do_something
.import city.csv city
```

Какая команда должна быть вместо `do_something`?

```
sqlite> .mode csv
sqlite> .import city.csv city;
sqlite> select count(*) from city;
1117
sqlite>
```

Рисунок 4 – Решение

11. Решите задачу: напишите в песочнице запрос, который посчитает количество городов для каждого часового пояса в Сибирском и Приволжском федеральных округах. Выведите столбцы timezone и city_count , отсортируйте по значению часового пояса:

```
┌──────────┬────────────┐
| timezone | city_count |
├──────────┼────────────┤
| UTC+3    | xxx        |
| UTC+4    | xx         |
| UTC+5    | xx         |
| UTC+6    | x          |
| UTC+7    | xx         |
| UTC+8    | xx         |
└──────────┴────────────┘
```

```
sqlite> .mode box
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;
┌──────────┬────────────┐
│ timezone │ city_count │
├──────────┼────────────┤
│ UTC+3    │ 101        │
│ UTC+4    │ 41         │
│ UTC+5    │ 58         │
│ UTC+6    │ 6          │
│ UTC+7    │ 86         │
│ UTC+8    │ 22         │
└──────────┴────────────┘
sqlite>
```

Рисунок 5 – Решение

12. Решите задачу: напишите в песочнице запрос, который найдет три ближайших к Самаре города, не считая саму Самару. Укажите в ответе названия этих трех городов через запятую в порядке удаления от Самары. Например:

```
Нижний Новгород, Москва, Владивосток
```

> Чтобы посчитать расстояние между двумя городами, используйте формулу из школьного курса геометрии:
>
> $$distance^2 = (lat_1 - lat_2)^2 + (lon_1 - lon_2)^2 \quad (1)$$
>
> Где (lat_1, lon_1) — координаты первого города, а (lat_2, lon_2) — координаты второго.

```
sqlite> .import --csv city.csv city
sqlite> SELECT c2.city,
   ...> SQRT(POW((CAST(c2.geo_lat AS REAL) - c1.geo_lat),2) +
POW((CAST(c2.geo_lon AS REAL) - c1.geo_lon),2)) AS distance
   ...> FROM city AS c1
   ...> JOIN city AS c2 ON c1.city = 'Самара' AND c2.city != '
Самара'
   ...> ORDER BY distance
   ...> LIMIT 3;
Новокуйбышевск|0.18569700863441
Чапаевск|0.358068603404667
Кинель|0.528066220190501
sqlite>
```

Рисунок 6 – Решение

13. Решите задачу: напишите в песочнице запрос, который посчитает количество городов в каждом часовом поясе. Отсортируйте по количеству городов по убыванию. Получится примерно так:

```
┌──────────┬────────────┐
│ timezone │ city_count │
├──────────┼────────────┤
│ UTC+3    │ xxx        │
│ UTC+5    │ xxx        │
│ UTC+7    │ xxx        │
│ UTC+4    │ xxx        │
│ ...      │            │
└──────────┴────────────┘
```

А теперь выполните этот же запрос, но так, чтобы результат был

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

Как выглядит четвертая строка результата?

```
sqlite> .mode box
sqlite> SELECT timezone, COUNT(city) 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 7 – Решение

```
sqlite> .headers on
sqlite> .mode csv
sqlite> .separator '|'
sqlite> SELECT timezone, COUNT(city) 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 8 – Решение

14. Выполните индивидуальное задание. Каждый запрос к базе данных сохраните в файл с расширением sql. Зафиксируйте изменения.

Описание данных в таблице:

1. Индекс
2. Name - Широко известное имя игрока

3. Full_name - Полное имя игрока
4. Age - Возраст игрока в годах
5. Height - Рост игрока в метрах
6. Nationality - Национальность игрока
7. Place_of_birth - Место, где родился игрок
8. Price - Текущая цена в миллионах
9. Max_price - Максимальная цена, которую когда-либо получал игрок, в миллионах
10. Position - Положение игрока на поле

Запросы:

1. Вывести имена немецких игроков, чья максимальная цена превышает 5 миллионов

```
sqlite> .once z1.csv
sqlite> SELECT name, nationality, max_price
   ...> FROM ind
   ...> WHERE "max_price" > 50 AND nationality == "Germany"
   ...> ORDER BY Price DESC;
sqlite> .once z1.json
sqlite> SELECT name, nationality, max_price
   ...> WHERE max_price > 50 AND nationality == 'Germany'
   ...> ;
Parse error: no such column: name
  SELECT name, nationality, max_price WHERE max_price > 50 AND nationality == 'G
         ^--- error here
sqlite> .once z1.json
sqlite> SELECT name, nationality, max_price
   ...> FROM ind
   ...> WHERE "max_price" > 50 AND nationality == "Germany"
   ...> ORDER BY Price DESC;
```

2. Вывести всех вратарей, чей возраст 20 лет

```
sqlite> .once z3.csv
sqlite> SELECT name, position, age
   ...> FROM ind
   ...> WHERE position == 'Goalkeeper' AND age == 20;
sqlite> .once z3.json
sqlite> SELECT name, position, age
   ...> FROM ind
   ...> WHERE position == 'Goalkeeper' AND age == 20;
sqlite>
```

3. Вывести количество игроков, родившихся в Париже

```
sqlite> .once z3.csv
sqlite> SELECT place_of_birth, count(*)
   ...> FROM ind
   ...> WHERE place_of_birth == 'Paris';
sqlite> .once z3.json
sqlite> SELECT place_of_birth, count(*)
   ...> FROM ind
   ...> WHERE place_of_birth == 'Paris';
sqlite>
```

4. Средний возраст игроков из Венгрии

```
sqlite> .once z4.csv
sqlite> SELECT nationality,  AVG(age) as "Средний возраст футболиста"
   ...> FROM ind
   ...> ORDER BY "Средний возраст футболиста" DESC;
sqlite> .once z4.csv
sqlite> SELECT nationality, AVG(age) as "Средний возраст игрока"
   ...> FROM ind
   ...> WHERE nationality == "Hungary";
sqlite> .once z4.json
sqlite> SELECT nationality, AVG(age) as "Средний возраст игрока"
   ...> FROM ind
   ...> WHERE nationality == "Hungary";
sqlite>
```

5. Максимальная стоимость игрока каждого клуба

```
sqlite> .once z5.csv
sqlite> SELECT club, MAX(max_price) as "Максимальная стоимость"
   ...> FROM ind
   ...> GROUP BY club
   ...> ORDER BY "Максимальная стоимость" DESC;
sqlite> .once z5.json
sqlite> SELECT club, MAX(max_price) as "Максимальная стоимость"
   ...> FROM ind
   ...> GROUP BY club
   ...> ORDER BY "Максимальная стоимость" DESC;
sqlite>
```

15. Добавьте отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксируйте изменения.

16. Отправьте изменения в локальном репозитории в удаленный репозиторий GitHub.

17. Проконтролируйте изменения, произошедшие в репозитории GitHub.

18. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Контрольные вопросы

1. Каково назначение реляционных баз данных и СУБД?

Теперь вернемся к вопросу о том, что такое реляционная базы данных (РБД). Слово "реляция" происходит от "relation", то есть "отношение". Это означает, что в РБД существуют механизмы установления связей между таблицами. Делается это с помощью так называемых первичных и внешних ключей.

2. Каково назначение языка SQL?

SQL – это язык программирования декларативного типа. В отличие от привычных нам процедурных языков, в которых есть условия, циклы и функции, в декларативных языках подобных алгоритмических конструкций почти нет. Декларативные выражения представляют собой скорее запросы, описание того, что хочет получить человек.

Язык SQL предназначен для создания и изменения реляционных баз данных, а также извлечения из них данных. Другими словами, SQL – это инструмент, с помощью которого человек управляет базой данных. При этом ключевыми операциями являются создание таблиц, добавление записей в таблицы, изменение и удаление записей, выборка записей из таблиц, изменение структуры таблиц.

3. Из чего состоит язык SQL?

Сам язык SQL состоит из операторов, инструкций и вычисляемых функций. Зарезервированные слова, которыми обычно выступают операторы, принято писать заглавными буквами. Однако

написание их не прописными, а строчными буквами к ошибке не приводит.

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

SQLite – это система управления базами данных, отличительной особенностью которой является ее встраиваемость в приложения. Это значит, что большинство СУБД являются самостоятельными приложениями, взаимодействие с которыми организовано по принципу клиент-сервер. Программа-клиент посылает запрос на языке SQL, СУБД, которая в том числе может находиться на удаленном компьютере, возвращает результат запроса.

В свою очередь SQLite является написанной на языке C библиотекой, которую динамически или статически подключают к программе. Для большинства языков программирования есть свои привязки (API) для библиотеки SQLite. Так в Python СУБД SQLite импортируют командой import sqlite3 . Причем модуль sqlite3 входит в стандартную библиотеку языка и не требует отдельной установки.

5. Как установить SQLite в Windows и Linux?

Для операционной системы Windows скачивают свой архив (sqlite-toolswin32-*.zip) и распаковывают. Далее настраивают путь к каталогу, добавляя адрес каталога к переменной PATH (подобное можно сделать и в Linux). Возможно, как и в Linux работает вызов утилиты по ее адресу. Android же имеет уже встроенную библиотеку SQLite.

6. Как создать базу данных SQLite?

С помощью sqlite3 создать или открыть существующую базу данных можно двумя способами. Во-первых, при вызове утилиты sqlite3 в качестве аргумента можно указать имя базы данных. Если БД существует, она будет открыта. Если ее нет, она будет создана и открыта.

$ sqlite3 your.db

Во вторых, работая в самой программе, можно выполнить команду .open your.db

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

Выяснить, какая база данных является текущей, можно с помощью команды .databases утилиты sqlite3. Если вы работаете с одной БД, а потом открываете другую, то текущей становится вторая БД.

8. Как создать и удалить таблицу в SQLite?

Таблицы базы данных создаются с помощью директивы CREATE TABLE языка SQL. После CREATE TABLE идет имя таблицы, после которого в скобках перечисляются имена столбцов и их тип. Для удаления целой таблицы из базы данных используется директива DROP TABLE, после которой идет имя удаляемой таблицы.

9. Что является первичным ключом в таблице?

Чтобы исключить возможность ввода одинаковых идентификаторов, столбец ID назначают первичным ключом.

10. Как сделать первичный ключ таблицы автоинкрементным?

Если нам не важно, какие конкретно идентификаторы будут записываться в поле _id, а важна только уникальность поля, следует назначить полю еще один ограничитель – автоинкремент – AUTOINCREMENT.

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

Ограничитель NOT NULL используют, чтобы запретить оставление поля пустым. По умолчанию, если поле не является первичным ключом, в него можно не помещать данные. В этом случае полю будет присвоено значение NULL. В случае NOT NULL вы не сможете добавить запись, не указав значения соответствующего поля.

Однако, добавив ограничитель DEFAULT, вы сможете не указывать значение. DEFAULT задает значение по умолчанию. В результате, когда данные в поле не передаются при добавлении записи, поле заполняется тем, что было указано по умолчанию.

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

С помощью внешнего ключа устанавливается связь между записями разных таблиц. Внешний ключ в одной таблице для другой является первичным. Внешние ключи не обязаны быть уникальными. В одной таблице может быть несколько внешних ключей, при этом каждый будет устанавливать связь со своей таблицей, где он является первичным.

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

С помощью оператора INSERT языка SQL выполняется вставка данных в таблицу.

14. Как выбрать данные из таблицы SQLite?

С помощью оператора SELECT.

15. Как ограничить выборку данных с помощью условия WHERE?

Такая команда отображает значения всех столбцов и строк заданной таблицы. На выборку всех столбцов указывает звездочка после слова SELECT.

А все строки будут выбраны потому, что после имени таблицы нет оператора WHERE языка SQL.WHERE позволяет задавать условие, согласно которому отображаются только удовлетворяющие ему строки.

16. Как упорядочить выбранные данные?

При выводе данных их можно не только фильтровать с помощью WHERE, но и сортировать по возрастанию или убыванию с помощью оператора ORDER BY.

17. Как выполнить обновление записей в таблице SQLite?

UPDATE ... SET – обновление полей записи

18. Как удалить записи из таблицы SQLite?

DELETE FROM – удаление записей таблицы

19. Как сгруппировать данные из выборке из таблицы SQLite?

В SQL кроме функций агрегирования есть оператор GROUP BY, который выполняет группировку записей по вариациям заданногополя.

20. Как получить значение агрегатной функции (например:

минимум, максимум, количество записей и т. д.) в выборке из таблицы SQLite?

Для этих целей в языке SQL предусмотрены различные функции агрегирования данных. Наиболее используемые – count(), sum(), avr(), min(), max().

21. Как выполнить объединение нескольких таблиц в операторе SELECT?

После FROM указываются обе сводимые таблицы через JOIN. В данном случае неважно, какую указывать до JOIN, какую после. После ключевого слова ON записывается условие сведения. Условие сообщает, как соединять строки разных таблиц.

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

Шаблоны реализуют поиск по таблице, если неизвестно полное название данных в строке. Подзапросы помогают уменьшить работу путём создания дополнительного запроса внутри основного

23. Каково назначение представлений VIEW в SQLite?

Бывает удобно сохранить результат выборки для дальнейшего использования. Для этих целей в языке SQL используется оператор CREATE VIEW, который создает представление – виртуальную таблицу.

В эту виртуальную таблицу как бы сохраняется результат запроса.

24. Какие существуют средства для импорта данных в SQLite?

.import --csv city.csv city

25. Каково назначение команды .schema?

Показывает какие столбцы есть в таблице, тип их данных и прочие свойства.

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

select federal_district as district,count(*) as city_count
from city group by 1 order by 2 desc;

27. Каково назначение "табличных выражений" в SQLite?

Выражение with history as (...) создает именованный запрос.

Название — history , а содержание — селект в скобках (век основания для каждого города).

К history можно обращаться поимени в остальном запросе, что мы и делаем.

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

.mode csv

29. Какие еще форматы для экспорта данных Вам известны?

.mode list , .mode json