

**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

Трушева В. О. .« » 2023г.

Подпись студента \_\_\_\_\_

Работа защищена « \_\_\_\_\_ 20\_\_ г.

Проверила Воронкин Р.А. \_\_\_\_\_

(подпись)

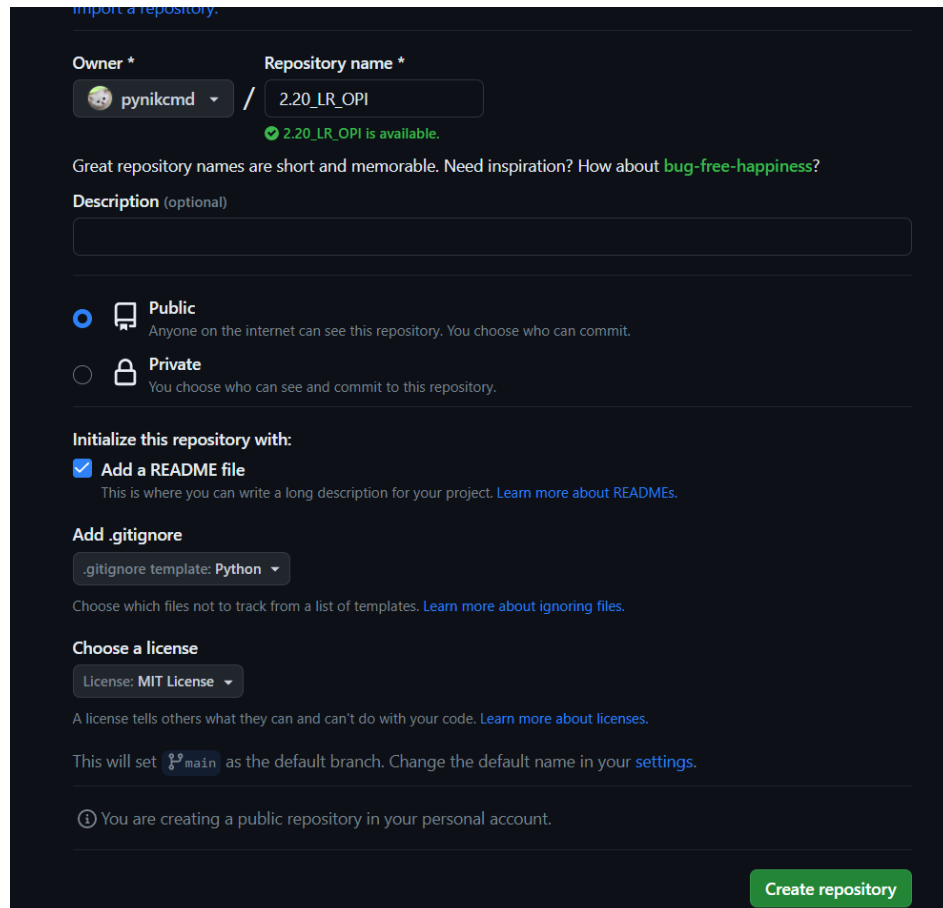
Ставрополь 2023

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

## Методика и порядок выполнения 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).



import a repository.

Owner \* / Repository name \*

pynikcmd / 2.20\_LR\_OPI

2.20\_LR\_OPI is available.

Great repository names are short and memorable. Need inspiration? How about [bug-free-happiness?](#)

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](#).

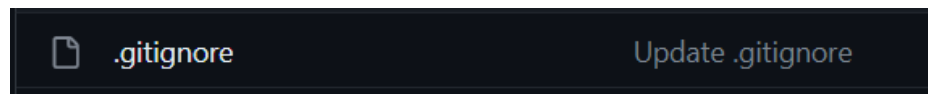
You are creating a public repository in your personal account.

Create repository

3. Выполните клонирование созданного репозитория на рабочий компьютер.

```
D:\fgit>git clone https://github.com/pynikcmd/2.20_LR_OPI.git
Cloning into '2.20_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 | 2.56 MiB/s, done.
```

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
```

Вот что здесь происходит:

- Первая команда ( create table ) создает таблицу customer с единственным столбцом name .
- Вторая команда ( select ) показывает содержимое таблицы customer (она пустая).
- 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. Специальные команды всегда начинаются с точки, занимают ровно одну строку, а точку запятой в конце ставить не надо.

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

Запрос и результат запроса

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 строка:

```
Run Time: real xxx user xxx sys xxx
```

Например:

```
sqlite> .SOMETHING on
sqlite> select count(*) from city;
1117
Run Time: real 0.000 user 0.000106 sys 0.000069
```

Какая команда должна быть вместо SOMETHING?

Команда, которая отвечает за вывод времени выполнения запроса в SQLite, называется `".timer"`.

```

sqlite> .help
.archive ...           Manage SQL archives
.auth ON|OFF           Show authorizer callbacks
.backup ?DB? FILE      Backup DB (default "main") to FILE
.bail on|off           Stop after hitting an error.  Default OFF
.binary on|off         Turn binary output on or off.  Default OFF
.cd DIRECTORY          Change the working directory to DIRECTORY
.changes on|off        Show number of rows changed by SQL
.check GLOB            Fail if output since .testcase does not match
.clone NEWDB           Clone data into NEWDB from the existing database
.connection [close] [#] Open or close an auxiliary database connection
.databases             List names and files of attached databases
.dbconfig ?op? ?val?   List or change sqlite3_db_config() options
.dbinfo ?DB?          Show status information about the database
.dump ?OBJECTS?        Render database content as SQL
.echo on|off           Turn command echo on or off
.eqp on|off|full|...   Enable or disable automatic EXPLAIN QUERY PLAN
.excel                Display the output of next command in spreadsheet
.exit ?CODE?           Exit this program with return-code CODE
.expert              EXPERIMENTAL. Suggest indexes for queries
.explain ?on|off|auto? Change the EXPLAIN formatting mode.  Default: auto
.filectrl CMD ...      Run various sqlite3_file_control() operations
.fullschema ?--indent? Show schema and the content of sqlite_stat tables
.headers on|off        Turn display of headers on or off
.help ?-all? ?PATTERN? Show help text for PATTERN
.import FILE TABLE    Import data from FILE into TABLE
.imposter INDEX TABLE Create imposter table TABLE on index INDEX
.indexes ?TABLE?       Show names of indexes
.limit ?LIMIT? ?VAL?   Display or change the value of an SQLITE_LIMIT
.lint OPTIONS          Report potential schema issues.
.load FILE ?ENTRY?     Load an extension library

```

Вывод команды .help

```

.timer on|off          Turn SQL timer on or off

```

Описание команды .timer

```

sqlite> .timer ON
sqlite> SELECT COUNT(*) FROM customer;
0
Run Time: real 0.000 user 0.000094 sys 0.000058

```

Запрос и результат запроса

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? 25

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

max(length(city))
25

Запрос и результат запроса

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`? `.mode csv`

```
sqlite> .mode csv
sqlite> .import city.csv city;
sqlite> select count(*) from city;
1120
```

Запрос и результат запроса

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

Укажите в ответе значение city\_count для timezone = UTC+5. Ответ:

58.

```
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

Запрос и результат запроса

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> 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;
```

city	distance
Новокуйбышевск	0.18569700863441
Чапаевск	0.358068603404667
Кинель	0.528066220190501

Запрос и результат запроса

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
...	

```
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

Запрос и результат запроса

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> .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
```

Запрос и результат запроса

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

Описание данных в таблице:

1. Car\_Make: Марка спортивного автомобиля, представляющая марку или компанию, которая произвела автомобиль. Примеры марок автомобилей в этом наборе данных включают Porsche, Lamborghini, Ferrari, Audi и McLaren.

2. Car\_Model: модель спортивного автомобиля, представляющая конкретную версию или вариант автомобиля, выпускаемого производителем. Примеры моделей автомобилей в этом наборе данных включают 911, Huracan, 488 GTB, R8, 720S, M8, AMG GT, Corvette, Mustang Shelby GT500 и GT-R Nismo.

3. Год: год производства спортивного автомобиля, который указывает модельный год, когда автомобиль был впервые представлен или стал доступен для покупки.

4. Engine\_Size: объем двигателя спортивного автомобиля в литрах, который представляет собой объем цилиндров двигателя. Большой объем двигателя обычно указывает на более высокую мощность и производительность. Объем двигателя в этом наборе данных варьируется от 2,0 л до 8,0 л, при этом у некоторых автомобилей вместо этого есть электродвигатели.

5. Мощность в лошадиных силах: мощность спортивного автомобиля, которая представляет собой выходную мощность двигателя автомобиля. Более высокая мощность обычно указывает на более быстрое ускорение и более высокую максимальную скорость. Значения лошадиных сил в этом наборе данных варьируются от 300 до 1479.

6. Крутящий момент: Крутящий момент спортивного автомобиля в фунто-футах, который представляет вращающую силу, создаваемую двигателем. Более высокие значения крутящего момента обычно указывают на более сильное ускорение и лучшую управляемость. Значения крутящего момента в этом наборе данных варьируются от 270 до 1180.

7. Время: время, необходимое спортивному автомобилю для разгона от 0 до 60 миль в час, что является общепринятой мерой ускорения и производительности. Меньшее время разгона от 0 до 60 миль в час обычно указывает на более быстрое ускорение и лучшую производительность. Время разгона от 0 до 60 миль в час в этом наборе данных составляет от 1,85 до 5,3 секунды.

8. Цена: цена спортивного автомобиля в долларах США, которая представляет собой стоимость покупки автомобиля. Цены в этом наборе данных варьируются от 25 000 до 3 000 000 долларов.

Запросы.

Вывести марки автомобилей, которые производятся после 2021 года и имеют мощность более 300 л.с., отсортированных по убыванию цены:

```
sqlite> SELECT Car_Make, Horsepower, Price, Year  
...> FROM data  
...> WHERE Year > 2021 AND Horsepower > 300  
...> ORDER BY Price DESC;
```

Запрос

Car_Make	Horsepower	Price	Year
Mercedes-Benz	429	96,25	2022
Mercedes-Benz	429	96	2022
Lexus	471	93,05	2022
Lexus	471	93,05	2022
Lexus	471	93,05	2022
Lexus	471	92,95	2022
Lexus	471	92,95	2022
Lexus	471	92,95	2022
Lexus	471	92,95	2022

Результат запроса

Вывести марки автомобилей и среднюю стоимость по маркам, у которых время разгона от 0 до 60 миль в час составляет менее 4 секунд:

```
sqlite> SELECT Car_Make, AVG("Price") as "Средняя стоимость"  
...> FROM data  
...> WHERE "Time" < 4  
...> GROUP BY Car_Make;
```

Запрос

Car_Make	Средняя стоимость
Acura	157.375
Alfa Romeo	77.6
Ariel	75.0
Aston Martin	215.061224489796
Audi	94.8840579710145
BMW	89.64
Bentley	215.24
Bugatti	3.04347826086957
Chevrolet	63.5263157894737

Результат запроса

Вывести модели автомобилей, у которых мощность двигателя выше 400 л.с. и время разгона до 60 миль в час меньше 4 секунд, с сортировкой по убыванию мощности двигателя:

```
sqlite> SELECT Car_Model, Horsepower, Time
...> FROM data
...> WHERE Horsepower > 400 AND Time < 4
...> ORDER BY Horsepower DESC;
```

Запрос

Car_Model	Horsepower	Time
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5
SF90 Stradale	986	2.5

Результат запроса

Вывести количество автомобилей каждой марки, производимых до 2022 года и имеющих стоимость более 50 000 долларов США:

```
sqlite> SELECT Car_Make, COUNT(*) as "Количество автомобилей"
...> FROM data
...> WHERE Year < 2022 AND "Price" > 50000
...> GROUP BY Car_Make;
```

Запрос

Car_Make	Количество автомобилей
Alfa Romeo	12
Alpine	1
Ariel	1
Audi	10
BMW	9
Chevrolet	32
Dodge	22
Ferrari	16
Ford	6

Результат запроса

Вывести производителей машин, у которых в среднем объем двигателя больше 4 литров, и отсортировать их по убыванию стоимости:

```
sqlite> SELECT Car_Make, AVG(Engine_Size) as "Средний объем двигателя (в литрах)", MAX(Price) as "Максимальная стоимость"
...> FROM data
...> GROUP BY Car_Make
...> HAVING AVG(Engine_Size) > 4
...> ORDER BY "Максимальная стоимость" DESC;
```

Запрос

Car_Make	Средний объем двигателя (в литрах)	Максимальная стоимость
Mercedes-Benz	4.62407407407407	96,25
Lexus	5.0	93,05
Chevrolet	6.18833333333333	85
Dodge	6.29024390243902	82,19
Mercedes-AMG	4.2	81,55
Ferrari	4.03636363636364	625
Lamborghini	5.7030303030303	573,966
Bugatti	8.0	5,200,000
Rolls-Royce	6.73	346,3

### Результат запроса

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. Резервированные 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-tools-win32-*.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 заданного поля.

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 по имени 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`