

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ-
СКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Анализ данных

Отчет по лабораторной работе №2.20

Тема: «Основы работы
с SQLite3»

Выполнил студент группы

ИВТ-б-о-21-1

Стригалов Дмитрий. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, старший
преподаватель Воронкин Р.А.

(подпись)

Ставрополь 2023

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

Ход работы:

Задание №1. Выполнение команд. Вот что здесь происходит:

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

Рисунок 1 – Создание таблицы customer со столбцом (name)

Что вернула команда .schema?

Данная команда показала какие столбцы есть в таблице.

Задание №2. 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:

Например: Run Time: real XXX user XXX sys XXX

```
.timer on|off          Turn SQL timer on or off
```

Рисунок 2 – С помощью команды .help нашёл команду, которая отвечает за время выполнения запроса

```
sqlite> .timer on
```

Рисунок 3 – Включаем таймер (чтобы увидеть время выполнения запросов)

```
sqlite> select count(*) from city;  


|           |
|-----------|
| count (*) |
| 1117      |

  
Run Time: real 0.000 user 0.000255 sys 0.000000  
sqlite> 
```

Рисунок 4 – Вводим необходимый запрос и получаем время его выполнения

Задание №3. Решите задачу: загрузите файл city.csv в песочнице. Затем выполните такой запрос: select max(length(city)) from city;.

```
sqlite> .import --csv city.csv city
sqlite> select max(length(city)) from city;
25
Run Time: real 0.002 user 0.001255 sys 0.000193
```

Рисунок 5 – Вывод запроса

Какое число он вернул?

Ответ: 25

Задание №4. 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.

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

Рисунок 6 – Добавление данных без использования опции `--csv`

Задание №5. 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.

Запрос:

Select timezone, count(city) as city_count from city where federal_district = 'Приволжский' or federal_district = 'Сибирский' group by timezone order by timezone ASC;

```
UTC+3,202
UTC+4,82
UTC+5,116
UTC+6,12
UTC+7,172
UTC+8,44
```

Рисунок 7 – результат выполнения запроса

Задание №6. 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.

Запрос:

with geo_las as (select geo_lat as geo_las from city where city = 'Самапа'),
 geo_los as (select geo_lon as geo_los from city where city = 'Самапа'),
 geo_lam as (select geo_lat as geo_lam, city from city), geo_lou as (select
 geo_lon as geo_lou from city)

Select sqrt((power((geo_las - geo_lam),2) + power((geo_los - geo_lou),2)))

As distance, city from (geo_las ,geo_los ,geo_lam, geo_lou)

Where city != 'Самапа' ORDER by distance ASC limit 3;

```
sqlite> with geo_las as (select geo_lat as geo_las from city where city = 'Самапа'),
...> geo_los as (select geo_lon as geo_los from city where city = 'Самапа'),
...> geo_lam as (select geo_lat as geo_lam, city from city),
...> geo_lou as (select geo_lon as geo_lou from city)
...> select sqrt((power((geo_las - geo_lam),2) + power((geo_los - geo_lou),2)))
...> as distance, city from (geo_las, geo_los, geo_lam, geo_lou)
...> where city != 'Самапа'
...> order by distance asc limit 3;
0.001052999999999886|Заречный
0.0094843000000004|Каменка
0.01199310000000051|Елизово
```

Рисунок 8 – Написанный запрос и результат выполнения

Задание №7. Решите задачу: напишите в песочнице запрос, который посчитает количество городов в каждом часовом поясе. Отсортируйте по количеству городов по убыванию.

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

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

Как выглядит четвертая строка результата?

```
sqlite> select timezone,
...> count (*) city_count
...> from city
...> group by 1
...> order by 2 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 9 – Результат выполнения запроса в формате csv с заголовками

```
sqlite> .separator |
sqlite> select timezone,
...> count (*) city_count
...> from city
...> group by 1
...> order by 2 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 10 – Результат выполнения запроса с «pipe» разделителем

Индивидуальное задание. Загрузите в SQLite выбранный Вами датасет в формате CSV (датасет можно найти на сайте Kaggle). Сформируйте более пяти запросов к таблицам БД. Выгрузите результат выполнения запросов в форматы CSV и JSON. Выбранный датасет:

Covid-19 Weekly Trends In Europe - Latest Data

Data Card Code (3) Discussion (1)

61

New Notebook

covid_europe_weekly_trend.csv (2.26 kB)

Download Icon Full Screen Icon Arrow Icon

Detail Compact Column

10 of 10 columns

Country/Other	# Cases in the last ...	# Cases in the prec...	# Weekly Case % C...	# Cases in the last ...	# Deaths
Names of countries, territories and islands in Europe	Cases in the last 7 days	Cases in the preceding 7 days	Weekly Case % Change	Cases in the last 7 days/1M pop	Deaths in
41 unique values					
Albania	66	168	-61.0	23	0
Andorra	39	0	3900.0	503	0
Austria	13095	17026	-23.0	1444	33
Belgium	2780	3913	-29.0	238	47
Bosnia and Herzegovina	87	91	-4.0	27	4
Bulgaria	748	980	-24.0	109	21
Channel Islands	193	365	-47.0	1094	2
Croatia	699	1073	-35.0	172	59
Czechia	1759	2185	-19.0	164	22

Рисунок 11 – Выбранный датасет с Kaggle

```
sqlite> select Country, min(CasesLast7days) from covid2;
```

Рисунок 12 – Запрос на минимальное количество больных за последние 7 дней

```
sqlite> select country, DeathsLast7days from covid2 where DeathsLast7days between 1 and 10;
[{"Country": "Malta", "DeathsLast7days": "1"},
{"Country": "Montenegro", "DeathsLast7days": "1"},
{"Country": "North Macedonia", "DeathsLast7days": "10"}]
```

Рисунок 13 – Запрос для выбора смертей в промежутке от 1 до 10 за последние 7 дней

```
sqlite> select sum(DeathsLast7days) from covid2;
```

sum(DeathsLast7days)
2451

Рисунок 14 – Сумма смертей за последние 7 дней

```
sqlite> select country, WeeklyCaseChange from covid2 where WeeklyCaseChange >= 0 order by WeeklyCaseChange limit 5
```

Country	WeeklyCaseChange
Poland	1.0
Moldova	26.0
Andorra	3900.0
Russia	41.0
Spain	7.0

Рисунок 15 – Запрос на еженедельные изменение % случаев ≥ 0

```
sqlite> select country, DeathsPreceding7days, Population from covid2 where DeathsPreceding7days != 0 order by Population desc limit 3;
```

Country	DeathsPreceding7days	Population
Hungary	32	9606259
Austria	55	9066710
Switzerland	14	8773637

Рисунок 16 – Запрос на смерти за прошедшие 7 дней не равные 0

```
sqlite> select country, CasesLast7days, DeathsLast7days1M, Population from covid2 where country = 'Russia' or country = 'Spain';
```

Country	CasesLast7days	DeathsLast7days1M	Population
Russia	34692	2.0	145805947
Spain	9871	2.0	46719142

```
sqlite>
```

Рисунок 17 – Запрос на смерти за последние 7 дней, и на 1 миллион населения в России и Испании

Вывод: в результате выполнения лабораторной работы были исследованы на практике базовые возможности системы управления базами данных SQLite3.

Контрольные вопросы:

1. Каково назначение реляционных баз данных и СУБД?

Главная функция СУБД – это управление данными (которые могут быть как во внешней, так и в оперативной памяти). СУБД обязательно поддерживает языки баз данных, а также отвечает за копирование и восстановление информации после каких-либо сбоев.

2. Каково назначение языка SQL?

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.

3. Из чего состоит язык SQL?

Язык SQL состоит из операторов, инструкций и вычисляемых функций.

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

С помощью SQLite создаются базы данных, представляющие собой один кроссплатформенный текстовый файл. Файл базы данных, в отличие от SQLite, не встраивается в приложение, не становится его частью, он существует отдельно. Так можно создать базу данных, пользуясь консольным sqlite3, после чего использовать ее в программе с помощью библиотеки SQLite языка программирования. При этом файл базы данных также хранится на локальной машине.

5. Как установить SQLite в Windows и Linux?

В Ubuntu установить sqlite3 можно командой `sudo apt install sqlite3`. Для операционной системы Windows скачивают свой архив (sqlite- tools-win32-*.zip) и распаковывают.

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.

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

Выяснить, какая база данных является текущей, можно с помощью команды `.databases` утилиты `sqlite3`.

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?

`PRIMARY KEY` – ограничитель, который заставляет СУБД проверять уникальность значения данного поля у каждой добавляемой записи.

10. Как сделать первичный ключ таблицы автоинкрементным?

Добавить `AUTOINCREMENT` в столбце при создании таблицы.

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

Ограничитель `NOT NULL` используют, чтобы запретить оставление поля пустым. `DEFAULT` задает значение 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.

Чтобы включить поддержку внешних ключей в `sqlite3`, надо выполнить команду `PRAGMA foreign_keys = ON`. После этого добавить в таблицу запись, в которой внешний ключ не совпадает ни с одним первичным из другой таблицы, не получится.

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 осуществляется выборочный просмотр данных из таблицы.

15. Как ограничить выборку данных с помощью условия WHERE?

Условие WHERE используется не только с оператором SELECT, также с UPDATE и DELETE. С помощью WHERE определяются строки, которые будут выбраны, обновлены или удалены. По сути это фильтр.

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 выборки 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 результатзапроса.

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 citygroup 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