

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ  
Федеральное государственное автономное  
образовательное учреждение высшего  
образования  
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра  
инфокоммуникаций  
Институт цифрового  
развития**

**ОТЧЁТ**  
**по лабораторной работе №2.20**  
Дисциплина: «Основы программной инженерии»  
Тема: «Основы работы с SQLite3»

Выполнила: студентка 2  
курса группы Пиж-б-о-21-

1

Джолдошова Мээрим  
Бекболотовна

Ставрополь 2023

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

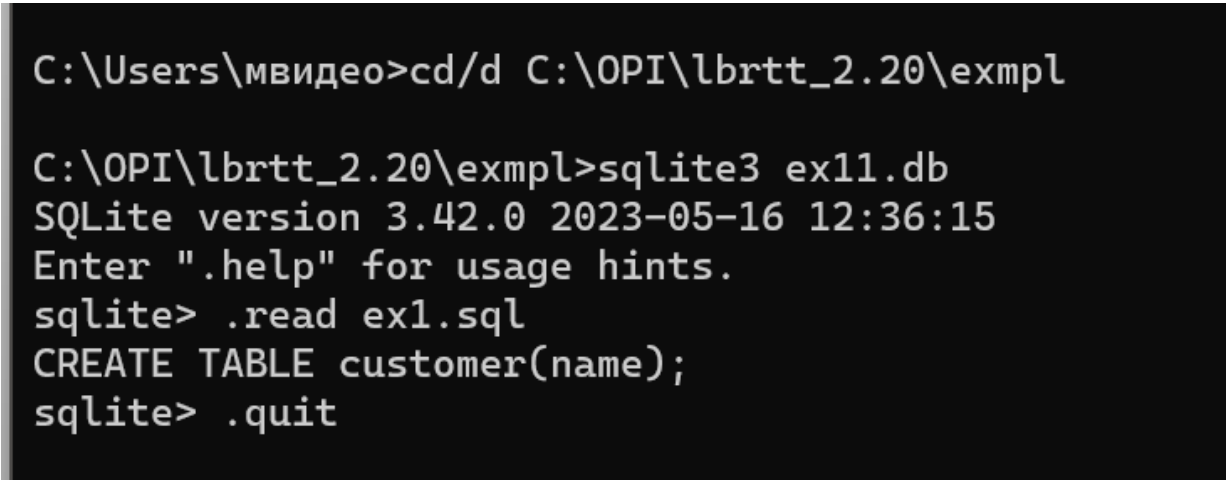
1. Решите задачу: выполните в песочнице команды:

```
create table customer(name);
```

```
select * from customer;
```

```
.schema customer
```

Что вернула команда .schema?



```
C:\Users\мвидео>cd/d C:\OPI\lbrtt_2.20\exmpl  
  
C:\OPI\lbrtt_2.20\exmpl>sqlite3 ex11.db  
SQLite version 3.42.0 2023-05-16 12:36:15  
Enter ".help" for usage hints.  
sqlite> .read ex1.sql  
CREATE TABLE customer(name);  
sqlite> .quit
```

Команда вернула список и структуру всех таблиц в базе.

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:

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

Например:

```
sqlite> .SOMETHING on
```

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

Какая команда должна быть вместо SOMETHING?

```

C:\OPI\lbrtt_2.20\exmpl>sqlite3 exx2.db
SQLite version 3.42.0 2023-05-16 12:36:15
Enter ".help" for usage hints.
sqlite> .read ex2.sql
Run Time: real 0.010 user 0.000000 sys 0.000000
Parse error near line 2: no such table: city
sqlite> .mode box
sqlite> .import --csv city.csv city
Error: cannot open "city.csv"
sqlite> .import --csv city.csv city
sqlite> .read ex2.sql

```

count(*)
1117

```

Run Time: real 0.000 user 0.000000 sys 0.000000
sqlite> |

```

Вместо something должна быть команда timer

3. 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?

```

C:\OPI\lbrtt_2.20\exmpl>sqlite3 exx3.db
SQLite version 3.42.0 2023-05-16 12:36:15
Enter ".help" for usage hints.
sqlite> .read ex3.sql

```

max(length(city))
25

```

sqlite> |

```

Он вернул число 25

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.

Вам поможет команда `.help import`. Всего должно получиться две команды:  
`do_something`

`.import city.csv city`

Какая команда должна быть вместо `do_something` ?

Должна быть команда `.mode csv`

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:

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`.

```
C:\OPI\lbrtt_2.20\exmpl>sqlite3 exx5.db
SQLite version 3.42.0 2023-05-16 12:36:15
Enter ".help" for usage hints.
sqlite> .read ex5.sql
```

timezone	city_count
UTC+3	101
UTC+4	41
UTC+5	58
UTC+6	6
UTC+7	86
UTC+8	22

```
sqlite> .quit
```

UTC+5 = 58

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.

Например:

Нижний Новгород, Москва, Владивосток

Чтобы посчитать расстояние между двумя городами, используйте формулу из школьного курса геометрии:

$$distance^2 = (lat_1 - lat_2)^2 + (lon_1 - lon_2)^2$$

Где –  $(lat_1, lon_1)$  – координаты первого города, а  $(lat_2, lon_2)$  – координаты второго.

```
sqlite> .read ex6.sql
```

city	distance
Новокуйбышевск	0.18569700863441
Чапаевск	0.358068603404667
Кинель	0.528066220190501

```
sqlite> |
```

Три ближайших к Самаре города: Новокуйбышевск, Чапаевск, Кинель

7. 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+4	xx
UTC+5	xx
UTC+6	x
UTC+7	xx
UTC+8	xx

```
sqlite> .read ex7.sql
```

timezone	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> |
```

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> .read ex7__1.sql
```

```
timezone|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> |
```

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

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.

```
C:\OPI\lbrtt_2.20\indiv>sqlite3 idv1.db
SQLite version 3.42.0 2023-05-16 12:36:15
Enter ".help" for usage hints.
sqlite> .mod box
sqlite> .import --csv ds_salaries.csv ds_salaries
sqlite> SELECT job_title, experience_level, count() as count FROM
M ds_salaries
...> WHERE job_title == 'Data Scientist'
...> GROUP BY experience_level
...> ;
```

job_title	experience_level	count
Data Scientist	EN	59
Data Scientist	EX	12
Data Scientist	MI	161
Data Scientist	SE	608

Рисунок 1 – Запрос Количество Data Scientist по уровню опыта



year	average_salary
2021	544163.252
2020	386352.75
2022	165421.017
2023	160381.481

Рисунок 2 – Запрос Средняя зарплата по году с округлением до тысячных

```
sqlite> .read indiv3.sql
Parse error near line 1: near "LIMITED": syntax error
AS difference FROM ds_salaries GROUP BY job_title LIMITED 30;
error here ---^
sqlite> .read indiv3.sql
```

job_title	min_salary	max_salary	difference
3D Computer Vision Researcher	10000	50000	40000
AI Developer	100000	80000	-20000
AI Programmer	40000	70000	30000
AI Scientist	12000	55000	43000
Analytics Engineer	100000	87000	-13000
Applied Data Scientist	100000	80000	-20000
Applied Machine Learning Engineer	130000	65000	-65000
Applied Machine Learning Scientist	108000	90000	-18000
Applied Scientist	110680	72000	-38680
Autonomous Vehicle Technician	45555	7000	-38555
Azure Data Engineer	100000	100000	0
BI Analyst	110000	78000	-32000
BI Data Analyst	100000	98000	-2000
BI Data Engineer	60000	60000	0
BI Developer	100000	197000	97000
Big Data Architect	124999	125000	1
Big Data Engineer	100000	85000	-15000
Business Data Analyst	100000	95000	-5000
Business Intelligence Engineer	129300	225000	95700
Cloud Data Architect	250000	250000	0
Cloud Data Engineer	12000	160000	148000
Cloud Database Engineer	115000	190000	75000
Compliance Data Analyst	30000	30000	0
Computer Vision Engineer	10000	60000	50000
Computer Vision Software Engineer	120000	81000	-39000
Data Analyst	10000	99750	89750
Data Analytics Consultant	113000	50000	-63000
Data Analytics Engineer	110000	67000	-43000
Data Analytics Lead	1440000	405000	-1035000
Data Analytics Manager	105400	204500	99100

```
sqlite> |
```

Рисунок 3 – Запрос минимальная и максимальная зарплата среди служащих и их разница

job_title	salary	company_size
Data Scientist	90700	M
Data Modeler	90700	M
Data Strategist	90000	M
Data Engineer	90700	M
Data Engineer	81500	M
Data Engineer	94300	M
Data Engineer	85000	M
Data Scientist	83270	M
Data Engineer	95000	M
Data Analyst	85000	M
Data Engineer	81500	M
Data Quality Analyst	80000	M
Data Engineer	85000	M
Data Analyst	80000	M
Data Manager	86000	M

```
sqlite>
```

Рисунок 4 – Запрос на заработную плату выше 80000 в средних компаниях среди работников

```
sqlite> .read indiv5.sql
```

[illegible]

Рисунок 5 – Запрос Должностей в больших компаниях, где выполненная удаленно работа превышает 0%

## Контрольные вопросы

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

Реляционные базы данных используются для хранения, организации, управления и доступа к данным в приложениях и информационных системах. Они организуют данные в таблицы, состоящие из строк и столбцов, где каждая строка представляет собой отдельную запись, а каждый столбец представляет собой отдельный атрибут.

### 2. Каково назначение языка SQL?

Язык SQL (Structured Query Language) является стандартным языком запросов для работы с данными в реляционных базах данных и предоставляет мощные возможности для создания, изменения, извлечения и управления данными.

### 3. Из чего состоит язык SQL?

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

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

Отличие между СУБД SQLite и клиент-серверными СУБД:

- SQLite: Встраиваемая СУБД, работает локально внутри приложения, не требует отдельного сервера. Ограничен однопользовательским доступом и не масштабируется для больших проектов.
- Клиент-серверные СУБД: Разделение на клиентскую и серверную части. Сервер управляет базой данных, клиенты подключаются удаленно по сети. Поддерживает многопользовательский доступ и масштабируется для обработки больших объемов данных.

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

В Ubuntu установить sqlite3 можно командой `sudo apt install sqlite3`. В

этом случае утилита вызывается командой `sqlite3`. Также можно скачать с сайта <https://sqlite.org> архив с последней версией библиотеки, распаковать и вызвать в терминале утилиту.

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?

В реляционных базах данных, первичный ключ (Primary Key) – это уникальное идентифицирующее поле или набор полей в таблице. Он служит для однозначной идентификации каждой записи (строки) в таблице.

#### 10. Как сделать первичный ключ таблицы автоинкрементным?

Для создания автоинкрементного первичного ключа в таблице SQLite, вы можете использовать тип данных INTEGER и атрибут AUTOINCREMENT.

Пример:

```
CREATE TABLE your_table_name (  
    id INTEGER PRIMARY KEY AUTOINCREMENT...);
```

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

Инструкция NOT NULL указывает, что столбец не может содержать NULL (пустое) значение, обеспечивая целостность данных.

Инструкция DEFAULT позволяет установить значение по умолчанию для столбца, которое будет использоваться, если явно не указано другое значение при вставке данных.

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

Внешние ключи в таблице используются для установления связей между двумя таблицами в реляционных базах данных. Они определяют отношения между записями в разных таблицах.

Пример:

```
CREATE TABLE pages (  
    ...  
    FOREIGN KEY (theme) REFERENCES sections(_id)  
    ...);
```

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

С помощью оператора INSERT.

```
INSERT INTO <table_name>  
(<column_name1>, <column_name2>, ...)
```

## VALUES

(<value1>, <value2>, ...);

### 14. Как выбрать данные из таблицы SQLite?

С помощью оператора SELECT.

SELECT \* FROM ;

### 15. Как ограничить выборку данных с помощью условия WHERE?

С помощью WHERE определяются строки, которые будут выбраны, обновлены или удалены. По сути, это фильтр.

После ключевого слова WHERE записывается логическое выражение, которое может быть как простым (содержащим операторы = или ==, >, =, <=, !=, BETWEEN), так и сложным (AND, OR, NOT, IN, NOT IN).

### 16. Как упорядочить выбранные данные?

С помощью оператора ORDER BY.

ORDER BY column1 ASC/DESC, column2 ASC/DESC, ...

### 17. Как выполнить обновление записей в таблице SQLite?

Для обновления записей в таблице SQLite используйте оператор UPDATE с указанием имени таблицы, столбцов и новых значений, а также условия WHERE для определения, какие строки обновить.

UPDATE your\_table

SET column1 = new\_value1, column2 = new\_value2;

### 18. Как удалить записи из таблицы SQLite?

Для удаления записей из таблицы SQLite вы можете использовать оператор DELETE. Он позволяет удалить одну или несколько строк, удовлетворяющих заданному условию.

DELETE FROM your\_table;



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?

Подзапросы в SQLite позволяют выполнять вложенные запросы внутри основного запроса для выполнения дополнительных вычислений, фильтрации или связывания данных.

Шаблоны в SQLite позволяют определить временную таблицу, которая может быть использована внутри запроса для создания и манипулирования данными во время выполнения запроса.

Подзапросы используются как внутренние выражения внутри операторов SELECT, FROM, WHERE, HAVING и других частей запроса.

Шаблоны определяются с помощью оператора WITH и могут быть использованы в основном запросе, как обычная таблица.

Оба инструмента предоставляют дополнительные возможности для обработки данных в SQLite, позволяя выполнять более сложные операции и повышая гибкость запросов.

### 23. Каково назначение представлений VIEW в SQLite?

Представления (VIEW) в SQLite создают виртуальные таблицы, которые представляют результат выполнения запроса. Они упрощают выполнение сложных запросов, абстрагируют данные, обеспечивают безопасность, позволяют повторное использование запросов и обновление данных.

### 24. Какие существуют средства для импорта данных в SQLite?

```
.import --csv data.csv data
```

### 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?

С помощью ORDER BY и GROUP BY.

### 27. Каково назначение "табличных выражений" в SQLite?

Табличные выражения в SQLite представляют результаты выполнения подзапросов или вложенных запросов, которые можно использовать как виртуальные таблицы в основном запросе.

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

Устанавливается вид данных с помощью `.mode csv/json/...`, направляет вывод результатов запроса в файл `.once data.csv` для одного запроса и `.output data.csv`, действующий, пока не произведется отмена.

29. Какие еще форматы для экспорта данных Вам известны? SQL-скрипты, JSON, XML