**Лабораторная работа 6. Создание, наполнение баз данных в SQLite**

Цель: освоить азы инженерии данных, т.е. научиться создавать и наполнять базы данных с помощью SQL/DDL для СУБД SQLite.

Задачи:

* познакомиться с СУБД SQLite;
* освоить инструменты для работы с БД SQLite: консольная программа sqlite3 и графические оболочки DB Browser for SQLite или SQLiteStudio;
* изучить типы данных SQLite, научиться описывать и создавать таблицы на SQL;
* уметь определять ограничения целостности данных и ссылок на SQL;
* освоить инструкции SQL для добавления, изменения и удаления записей данных;
* научиться создавать представления (запросы) на SQL;
* научиться изменять таблицы и преставления;
* уметь очищать и удалять таблицы и представления.

***Общая характеристика и назначение СУБД SQLite***

SQLite представляет библиотеку, написанную на языке C (ANSI-C) и реализующую движок реляционных баз данных. Сегодня SQLite – широко распространённая СУБД, которую можно найти в каждом устройстве на Android, iOS, Mac, Windows 10/11, ее используются большинство распространенных браузеров - Firefox, Chrome, Safari и т.д.

SQLite в отличие от других СУБД (MS SQL Server, MySQL, Postgres) представляет встраиваемый движок БД, для которого не требуется сервер базы данных, он обращается напрямую к файлу БДх на диске. Поэтому для работы с БД нам не надо явно устанавливать или как-то настраивать SQLite.

SQLite имеет полноценную поддержку большинства возможностей, которыми обладают другие реляционные СУБД.

SQLite для работы с БД применяет язык SQL (), реализация которого в целом похожа на диалекты в других реляционных СУБД.

Формат файла базы данных является кроссплатформенным – переносимым между устройствами в разных операционных системах.

Для разработки приложений большинство языков программирования (Python, C#, Java, и т.д.) имеют поддержку SQLite, что позволяет использовать эту СУБД в различных сценариях и самых различных типах приложений.

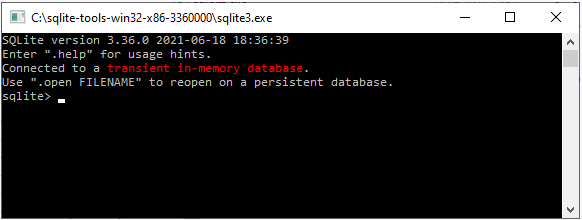
SQLite развивается как проект с открытым исходным кодом.

***Консольный клиент sqlite3***

Для работы с SQLite разработчики данной СУБД предоставляют консольный клиент sqlite3.

Прежде всего нам надо загрузить sqlite3 со страницы <https://www.sqlite.org/download.html>. Название необходимого нам пакета начинается с sqlite-tools. И на странице загрузки мы можем найти версии для Windows, Linux, MacOS.

В загруженной и распакованной папке мы сможем найти файл с названием sqlite3**,** который и представляет собственно консольную оболочку для работы с БД SQLite. Запустим ее:

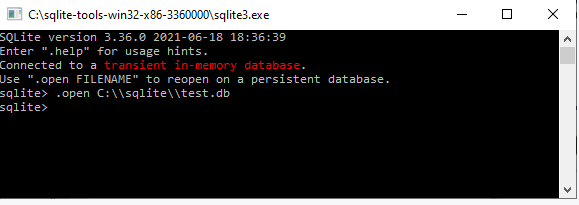


Открытие базы данных

Для открытия базы данных необходимо ввести команду **.open**, после которой указывается путь к базе данных. Например,

|  |
| --- |
| sqlite>.open test.db |

В данном случае будет открыта база данных под названием "test.db', которая находится в той же папке, что и консольная утилита. Если базы данных не существует, то она будет создаться.



Также можно передать абсолютный путь::

|  |
| --- |
| sqlite>.open C:\\sqlite\\test.db |

После открытия мы сможем работать с этой БД.

Создание таблицы

Для создания таблицы после открытия базы данных необходимо ввести команды **CREATE TABLE**, после которой указываются название таблицы и спецификация ее столбцов:

|  |
| --- |
| sqlite>create table users(name text, age integer); |

В данном случае создается таблица users, в которой два столбца: столбец name, который имеет тип text, и столбец age, который имеет тип integer. Обратите внимание, что команда завершается точкой с запятой, благодаря чему SQLite может идентифицировать, что выполняется SQL-команда.

Операции с данными

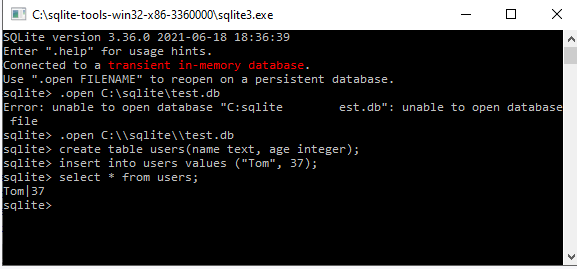
Для добавления данных применяется команда **INSERT INTO**. Например, добавим в таблицу users одну строку:

|  |
| --- |
| sqlite>insert into users values ('Tom', 37); |

Теперь получим ранее добавленные данные. Для этого используем команду **SELECT**:

|  |
| --- |
| sqlite>select \* from users |

И SQLite выведет нам все данные из таблицы users:



Для получения справки о командах введите:

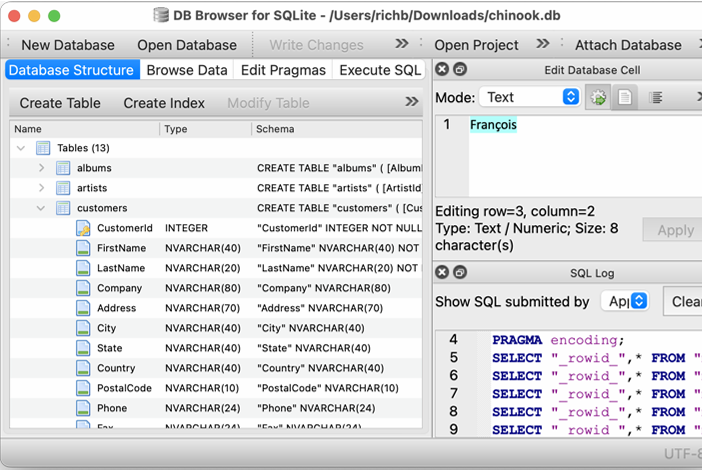
|  |
| --- |
| sqlite>.help |

Для завершения работы с утилитой введите следующую команду:

|  |
| --- |
| sqlite>.quit |

**Браузер БД для SQLite**

DB Browser for SQLite (DB4S) – это высококачественный, визуальный инструмент с открытым исходным кодом предназначен для тех, кто хочет создавать, искать и редактировать файлы баз данных SQLite. DB4S предоставляет знакомый интерфейс, похожий на электронные таблицы, для работы с базой данных в дополнение к предоставлению полного функционала SQL-запросов. Он работает с и Windows, macOS, и большинство версий Linux и Unix. Документация по программе находится на wiki.



DB4S предоставляет доступ ко всем возможностям базовой базы данных SQLite.

Элементы управления и мастера доступны для:

* Создание и сжатие файлов базы данных
* Создание, определение, изменение и удаление таблиц
* Создание, определение и удаление индексов
* Просмотр, редактирование, добавление и удаление записей
* Сортировка и поиск записей Импорт и экспорт записей в виде текста
* Импорт и экспорт таблиц из/в файлы CSV
* Импорт и экспорт баз данных из/в файлы дампа SQL
* Выполнение SQL-запросов и проверка результатов
* Просмотр журнала всех SQL-команд, выдаваемых приложением
* Построение простых графиков на основе данных таблиц или запросов.

Текущей официальной, выпущенной версией DB4S является 3.13.0. Скачать с: <https://sqlitebrowser.org/dl>

**Определение структуры данных в SQLite**

***Создание и удаление таблицы. Прикрепление базы данных***

Для создания таблиц используется команда **CREATE TABLE**. Общий формальный синтаксис команды CREATE TABLE:

|  |
| --- |
| **CREATE TABLE** *название\_таблицы*  (*название\_столбца1 тип\_данных свойства\_столбца1*,  *название\_столбца2 тип\_данных свойства\_столбца2*,   ................................................  *название\_столбцаN тип\_данных свойства\_столбцаN*,  *свойства\_уровня\_таблицы*  ) |

После команды CREATE TABLE указывается название таблицы. Имя таблицы выполняет роль ее идентификатора в базе данных, поэтому оно должно быть уникальным. Кроме того, оно не должно начинаться на "sqlite\_", поскольку названия таблиц, которые начинаются на "sqlite\_", зарезервированы для внутреннего пользования.

Затем после названия таблицы в скобках перечисляются названия столбцов, их типы данных и атрибуты. В самом конце можно определить атрибуты для всей таблицы. Атрибуты столбцов, а также атрибуты таблицы указывать необязательно.

Создадим простейшую таблицу. Перед выполнением команды CREATE TABLE вне зависимости, что мы используем - консольный клиент sqlite3, графический клиент DB Browser for SQLite иликакой-то другой клиент, вначале откроем базу данных, где мы хотим создать таблицу.

Для создания таблицы выполним следующий скрипт:

|  |
| --- |
| **CREATE TABLE Users**  **(Id INTEGER, Name TEXT, Age INTEGER);** |

В данном случае таблица называется "Users". В ней определено три столбца: Id, Age, Name. Первые два столбца представляют идентификатор пользователя и его возраст и имеют тип INTEGER, то есть будут хранить числовые значения. Столбец "Name" представляет имя пользователя и имеет тип TEXT, то есть представляет строку. В данном случае для каждого столбца определены имя и тип данных, при этом атрибуты столбцов и таблицы в целом отсутствуют.

И в результате выполнения этой команды будет создана таблица Users с тремя столбцами (рис.1).

***Создание таблицы при ее отсутствии***

Если мы повторно выполним выше определенную sql-команду для создания таблицы Users, то мы столкнемся с ошибкой - ведь мы уже создали таблицу с таким названием. Но могут быть ситуации, когда мы можем точно не знать или быть не уверены, есть ли в базе данных такая таблица (например, когда мы пишем приложение на каком-нибудь языке программирования и используем базу данных, которая не нами создана). И чтобы избежать ошибки, с помощью выражения **IF NOT EXISTS** мы можем задать создание таблицы, если она не существует:

|  |  |
| --- | --- |
|  | **CREATE TABLE IF NOT EXISTS Users**  **(Id INTEGER, Name TEXT, Age INTEGER);**  Далее смотрите учебные материалы на сайте <https://metanit.com/sql/sqlite/> |

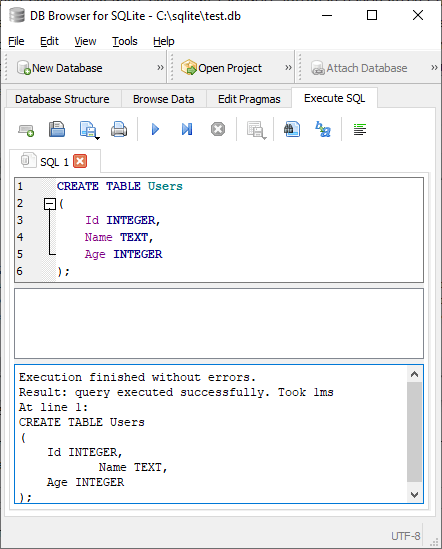
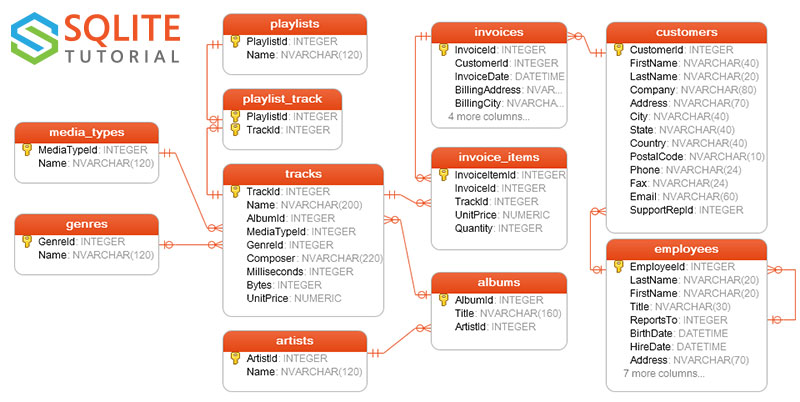


Рис. 1. Создание таблицы

**Пример базы данных SQLite**

Мы предоставляем пример базы данных Chinook, который хорош для практики с SQLite.

На следующей схеме базы данных показаны таблицы базы данных и их взаимосвязи.



**Примеры таблиц базы данных Chinook**

Образец базы данных содержит 11 таблиц, а именно:

* В таблице employees хранятся данные о сотрудниках, такие как ID, фамилия, имя и т. д. В нем также есть поле с именем ReportsTo, указывающее, кто кому отчитывается.
* В таблице customers хранятся данные о клиентах.
* Таблицы invoices & invoice\_items хранят данные счетов. В таблице invoices хранятся данные заголовка счета, а в таблице. invoice\_items — данные о позициях счета
* В таблице  artists  хранятся данные об исполнителях. Это простая таблица, которая содержит id и имя.
* В таблице  albums  хранятся данные о списке треков. Каждый альбом принадлежит одному исполнителю, но у исполнителя может быть несколько альбомов.
* В таблице media\_types  хранятся такие типы мультимедиа, как аудиофайлы MPEG и аудиофайлы AAC.
* В таблице genres  хранятся такие музыкальные типы, как рок, джаз, металл и т.д.
* В таблице  tracks  хранятся данные о песнях. Каждый трек принадлежит одному альбому.
* Таблицы: playlists & playlist\_track  Таблица playlists хранит данные о плейлистах. Каждый плейлист содержит список треков. Каждый трек может входить в несколько плейлистов.

Отношение между таблицами— это «многие-ко-многим». Таблица playlist\_track используется для отражения этой взаимосвязи.