Лабораторна робота №1.

**SQLite. Загальні відомості. Типи даних. Створення БД. Створення таблиці. Резервне копіювання та відновлення. Імпорт-експорт даних.**

1. Загальні теоретичні відомості.

SQLite — це програмна бібліотека, що реалізує самодостатню, не потребуючу сервера, транзакційну систему управління реляційними базами даних SQL. SQLite - це одна з найбільш швидко зростаючих СУБД. Вихідний код SQLite є у вільному доступі. Це база даних, яка не потребує конфігурування, ще означає, що вам не потрібно її налаштовувати під вашу операційну систему. SQLite-движок не є окремим процесом, як і інші бази даних, ви можете поєднати його статично або динамічно відповідно до ваших вимог з вашим додатком. SQLite отримує прямий доступ безпосередньо до файлів даних.

Чому SQLite?

• SQLite не вимагає окремого серверного процесу або систему для роботи (serverless).

• SQLite поставляється з нульової конфігурації, який означає, що не потрібні установки або адміністрування.

• Вся база даних SQLite зберігається у одному крос платформному файлі.

• SQLite - дуже мала неважка, менше, ніж 400KiB у повній комплектації або менше, ніж 250KiB з виключенням деяких функції.

• SQLite є самодостатньою, що означає, що немає зовнішніх залежностей.

• SQLite транзакції повністю задовольняють вимогам ACID, дозволяючи безпечний доступ з декількох процесів або потоків.

• SQLite підтримує більшість функцій мови запитів, знайдені в SQL92 (SQL2) стандарті.

• SQLite написано в ANSI-C та забезпечує простий і легкий у використанні API.

• SQLite доступна на UNIX (Linux, Mac OS X, Android, iOS) і Windows (Win32, WinCE, WinRT).

1. Інсталяція та запуск SQLite.

Крок 1: перейдіть на сторінку завантаження SQLite http://www.sqlite.org/download.html в розділ Windows.

Крок 2: Завантажити sqlite-shell-win32\*.zip і sqlite-dll-win32-\*.zip архіви.

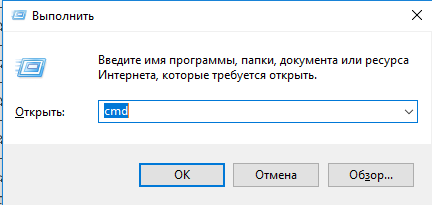
Крок 3: Створити папку C:\>sqlite і розпакувати завантажені архіви у цю папку (мають бути sqlite3.def, sqlite3.dll та sqlite3.exe файли).

Крок 4: Додавати C:\>sqlite у змінній оточення PATH

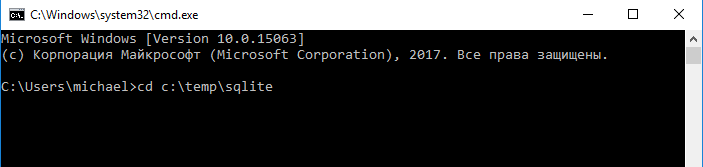
Крок 5: Перейти до командного рядка і виконати команду sqlite3.

При виконанні роботи в комп’ютерному класі:

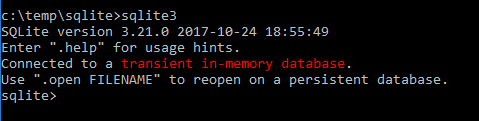
* Завантажити з інтернету (або скопіювати в локальній мережі – уточнити у викладача де саме) файли sqlite3.def, sqlite3.dll та sqlite3.exe у папку c:\temp\sqlite (або у будь-яку іншу, де є права на запис даних).
* Запустити командний рядок Windows->run (win+r) ->cmd



* Перейти до c:\sqlite (виконати команду c:\sqlite)



* Запустити sqlite (виконати команду sqlite3)



* Далі введемо команду .help для отримання відомостей щодо переліку управляючих команд та .exit для завершення першого сеансу роботи із sqlite.

1. Типи даних

Тип даних SQLite є атрибутом, який визначає тип даних будь-якого об'єкта. Кожен стовпець, змінна і вираз має пов'язані тип даних у SQLite. Ви повинні використовувати ці типи даних під час створення таблиці. Але при цьому слід розуміти, що SQLite використовує більш загальну динамічну систему типів. У SQLite тип даних значення асоціюється з самим значенням, не з її контейнером.

Класи для збереження даних в SQLite

Кожне значення, що зберігається в БД SQLite з має один із наступтих класів.

* NULL - це значення NULL.
* INTEGER - значення є цілим, зберігаються в 1, 2, 3, 4, 6 або 8 байтів в залежності від величини значення.
* REAL – значення числа з плаваючою точкою, збережені як IEEE 8-байтове число з рухомою точкою.
* TEXT - це текстовий рядок, збережений використовуючи кодову сторінку БД (UTF8, UTF-16BE або UTF-16LE)
* BLOB значення blob, які зберігаються точно, як були уведені.

SQLite клас зберігання даних є трохи більш загальним, ніж тип даних. Клас зберігання ціле число, наприклад, включає в себе 6 різних цілих типів даних різної довжини.

SQLite підтримує концепцію спорідненості типів для колонок. Будь яка колонка може зберігати будь який тип даних, але клас, якому надається перевага має назву спорідненого. Кожна колонка таблиці в SQLite3 БД віднесена до одного із наступних споріднених типів: TEXT, NUMERIC, INTEGER, REAL, NONE.

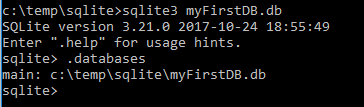
SQLite не має окремого булевого класу зберігання. Замість цього, логічні значення зберігаються як цілі числа 0 (невірно) і 1 (вірно).

1. Створення бази даних.

Для створення БД за допомогою SQLite не потрібно превілегій або додаткових команд. Все, що потрібно зробити – при запуску sqlite3 вказати ім’я БД. Якщо БД існує, робота буде продовжена в контексті вказаної БД, в іншому випадку вона буде автоматично створена. В командному рядку (див.п.2) виконаємо команду sqlite3 myFirstDB.db



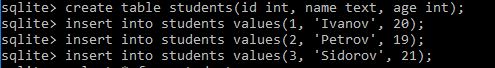
Ця команда створить файл myFirstDB.db в поточному каталозі. Цей файл буде використовуватися як поточна база даних SQLite. Якщо ви помітили, після успішного створення бази даних, sqlite3 виводить запрошення sqlite > для вводу наступної команди.



Після створення бази даних, ви можете перевірити у списку баз даних, використовуючи команду .databases.

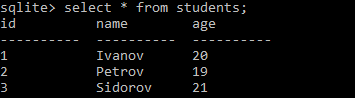


1. Створимо таблицю students, та додамо до неї декілька рядків



перевіримо вміст таблиці, попередньо виконавши налаштування для відображення імен колонок виводу

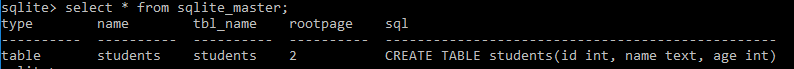




1. Для відображення схеми поточної БД можемо виконати команду .schema



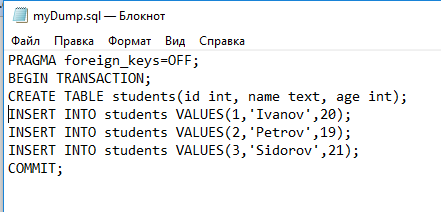
Також уявлення про перелік об’єктів БД можна отримати звернувшись до системної таблиці sqlite\_master.



1. За допомогою команди .dump можна виконати резервну копію БД у вигляді скрипта на створення об’єктів БД та заповнення їх даними.



Наведемо вміст створеного файлу myDump.sql



Дамп БД також може бути створений з командного рядку за допомогою sqlite3



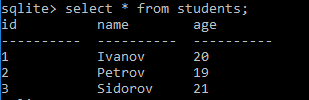
1. Маючи резервну копію можемо у разі необхідності відновити БД. Видалимо файл myFirstDB.db та виконаємо команду



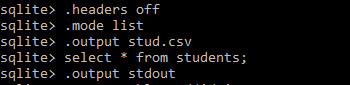
Далі завантажимо БД myFirstDB та пересвідчимося у наявності таблиці students та даних у ній







1. Виконаємо експорт даних таблиці students до зовнішнього файлу stud.csv

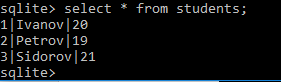


Видалимо дані з таблиці students та пересвідчимося, що таблиця порожня



завантажимо дані з файлу до таблиці students та перевіримо її вміст





1. Кінець роботи