**A database management system (DBMS)** is system software for creating and managing [databases](https://searchsqlserver.techtarget.com/definition/database). The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage [data](https://searchdatamanagement.techtarget.com/definition/data).

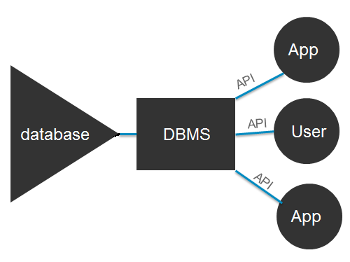
A DBMS makes it possible for end users to create, read, update and delete [data](https://searchdatamanagement.techtarget.com/definition/data) in a database. The DBMS essentially serves as an interface between the [database](https://searchsqlserver.techtarget.com/definition/database) and end users or [application programs](https://searchsoftwarequality.techtarget.com/definition/application), ensuring that data is consistently organized and remains easily accessible.

The DBMS is perhaps most useful for providing a centralized view of data that can be accessed by multiple users, from multiple locations, in a controlled manner. A DBMS can limit what data the end user sees, as well as how that end user can view the data, providing many views of a single database schema. End users and software programs are free from having to understand where the data is physically located or on what type of storage media it resides because the DBMS handles all requests.

The DBMS can offer both logical and physical data independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data ([storage](https://searchstorage.techtarget.com/definition/storage) and hardware). As long as programs use the application programming interface ([API](https://searchmicroservices.techtarget.com/definition/application-program-interface-API)) for the database that is provided by the DBMS, developers won't have to modify programs just because changes have been made to the database.

The DBMS is perhaps most useful for providing a centralized view of data that can be accessed by multiple users, from multiple locations, in a controlled manner. A DBMS can limit what data the end user sees, as well as how that end user can view the data, providing many views of a single database schema. End users and software programs are free from having to understand where the data is physically located or on what type of storage media it resides because the DBMS handles all requests.

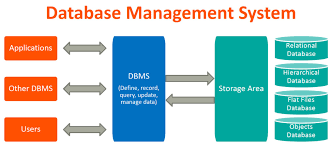
The DBMS can offer both logical and physical data independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data ([storage](https://searchstorage.techtarget.com/definition/storage) and hardware). As long as programs use the application programming interface ([API](https://searchmicroservices.techtarget.com/definition/application-program-interface-API)) for the database that is provided by the DBMS, developers won't have to modify programs just because changes have been made to the database.



Advantages of a DBMS

* Data abstraction and independence
* Data security
* A locking mechanism for concurrent access
* An efficient handler to balance the needs of multiple applications using the same data
* The ability to swiftly recover from crashes and errors, including restartability and recoverability
* Robust data integrity capabilities
* Logging and auditing of activity
* Simple access using a standard application programming interface (API)
* Uniform administration procedures for data

A DBMS can also provide many views of a single database schema. A view defines what data the user sees and how that user sees the data. The DBMS provides a level of abstraction between the conceptual schema that defines the logical structure of the database and the physical schema that describes the files, indexes and other physical mechanisms used by the database. When a DBMS is used, systems can be modified much more easily when business requirements change. New categories of data can be added to the database without disrupting the existing system and applications can be insulated from how data is structured and stored.

Of course, a DBMS must perform additional work to provide these advantages, thereby bringing with it the overhead. A DBMS will use more memory and [CPU](https://whatis.techtarget.com/definition/processor) than a simple file storage system. And, of course, different types of DBMSes will require different types and levels of system resources.

**SQL (Structured Query Language)** is a [domain-specific language](https://en.wikipedia.org/wiki/Domain-specific_language) used in programming and designed for managing data held in a [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS), or for stream processing in a [relational data stream management system](https://en.wikipedia.org/wiki/Relational_data_stream_management_system) (RDSMS). It is particularly useful in handling [structured data](https://en.wikipedia.org/wiki/Data_model) where there are relations between different entities/variables of the data. SQL offers two main advantages over older read/write [APIs](https://en.wikipedia.org/wiki/API) like [ISAM](https://en.wikipedia.org/wiki/ISAM) or [VSAM](https://en.wikipedia.org/wiki/VSAM). First, it introduced the concept of accessing many records with one single command; and second, it eliminates the need to specify *how* to reach a record, e.g. with or without an [index](https://en.wikipedia.org/wiki/Database_index).

Originally based upon [relational algebra](https://en.wikipedia.org/wiki/Relational_algebra) and [tuple relational calculus](https://en.wikipedia.org/wiki/Tuple_relational_calculus), SQL consists of many types of statements, which may be informally classed as [sublanguages](https://en.wikipedia.org/wiki/Sublanguage), commonly: a [data query language](https://en.wikipedia.org/wiki/Data_query_language) (DQL), a [data definition language](https://en.wikipedia.org/wiki/Data_definition_language) (DDL), a [data control language](https://en.wikipedia.org/wiki/Data_control_language) (DCL), and a [data manipulation language](https://en.wikipedia.org/wiki/Data_manipulation_language) (DML). The scope of SQL includes data query, data manipulation (insert, update and delete), data definition ([schema](https://en.wikipedia.org/wiki/Database_schema) creation and modification), and data access control. Although SQL is often described as, and to a great extent is, a [declarative language](https://en.wikipedia.org/wiki/Declarative_programming) ([4GL](https://en.wikipedia.org/wiki/4GL)), it also includes [procedural](https://en.wikipedia.org/wiki/Procedural_programming) elements.

SQL was one of the first commercial languages for [Edgar F. Codd](https://en.wikipedia.org/wiki/Edgar_F._Codd)'s [relational model](https://en.wikipedia.org/wiki/Relational_model). The model was described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks". Despite not entirely adhering to [the relational model as described by Codd](https://en.wikipedia.org/wiki/Codd%27s_12_rules), it became the most widely used database language.

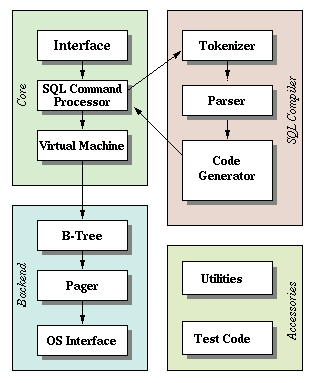
SQL became a [standard](https://en.wikipedia.org/wiki/Technical_standard) of the [American National Standards Institute](https://en.wikipedia.org/wiki/American_National_Standards_Institute) (ANSI) in 1986, and of the [International Organization for Standardization](https://en.wikipedia.org/wiki/International_Organization_for_Standardization) (ISO) in 1987.[[14]](https://en.wikipedia.org/wiki/SQL#cite_note-ISO/IEC-17) Since then, the standard has been revised to include a larger set of features. Despite the existence of such standards, most SQL code is not completely portable among different database systems without adjustments.

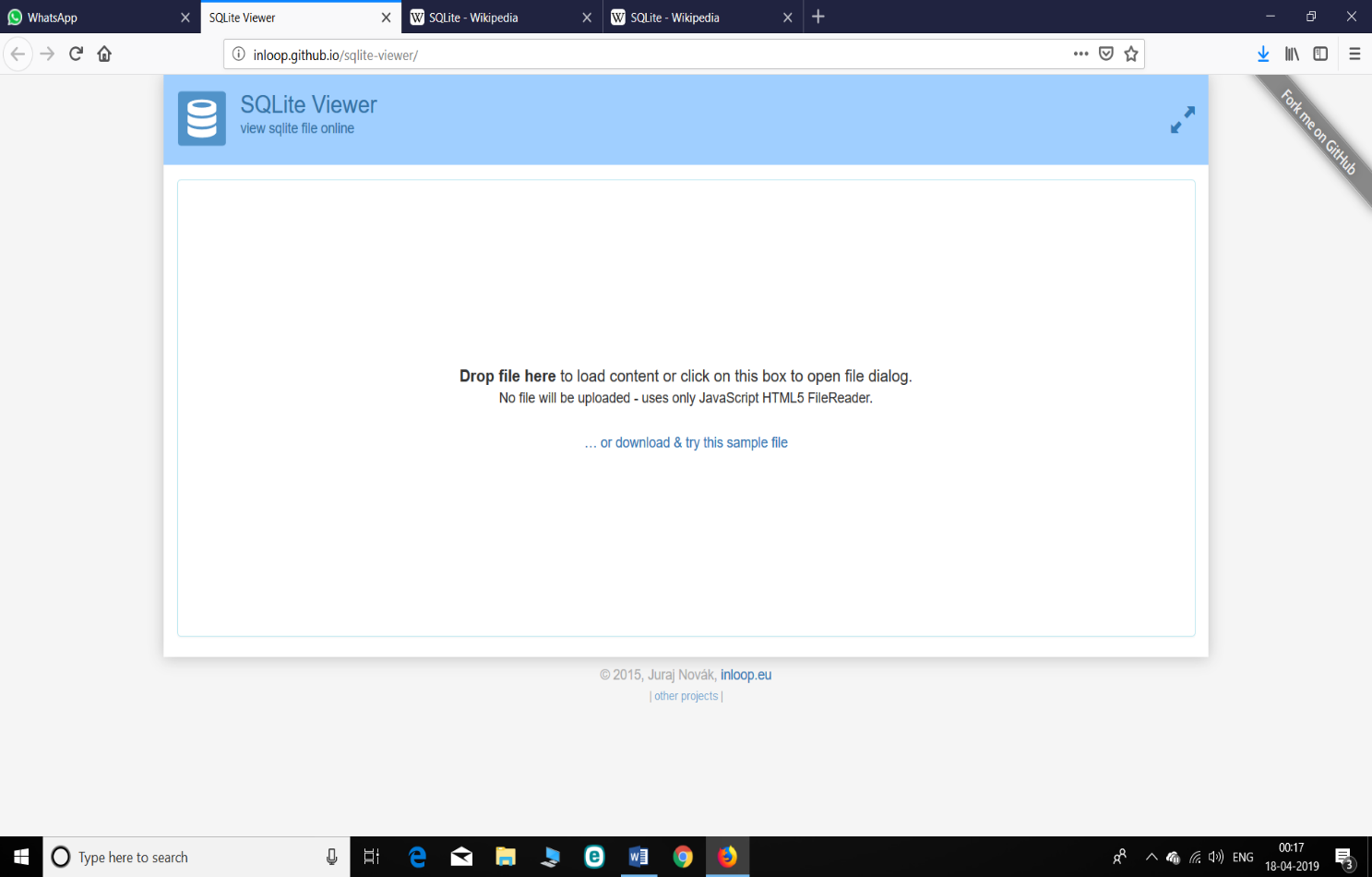
SQL is a [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) contained in a [C](https://en.wikipedia.org/wiki/C_(programming_language)) programming [library](https://en.wikipedia.org/wiki/Library_(computer_science)). In contrast to many other database management systems, SQLite is not a [client–server](https://en.wikipedia.org/wiki/Client%E2%80%93server) database engine. Rather, it is embedded into the end program.

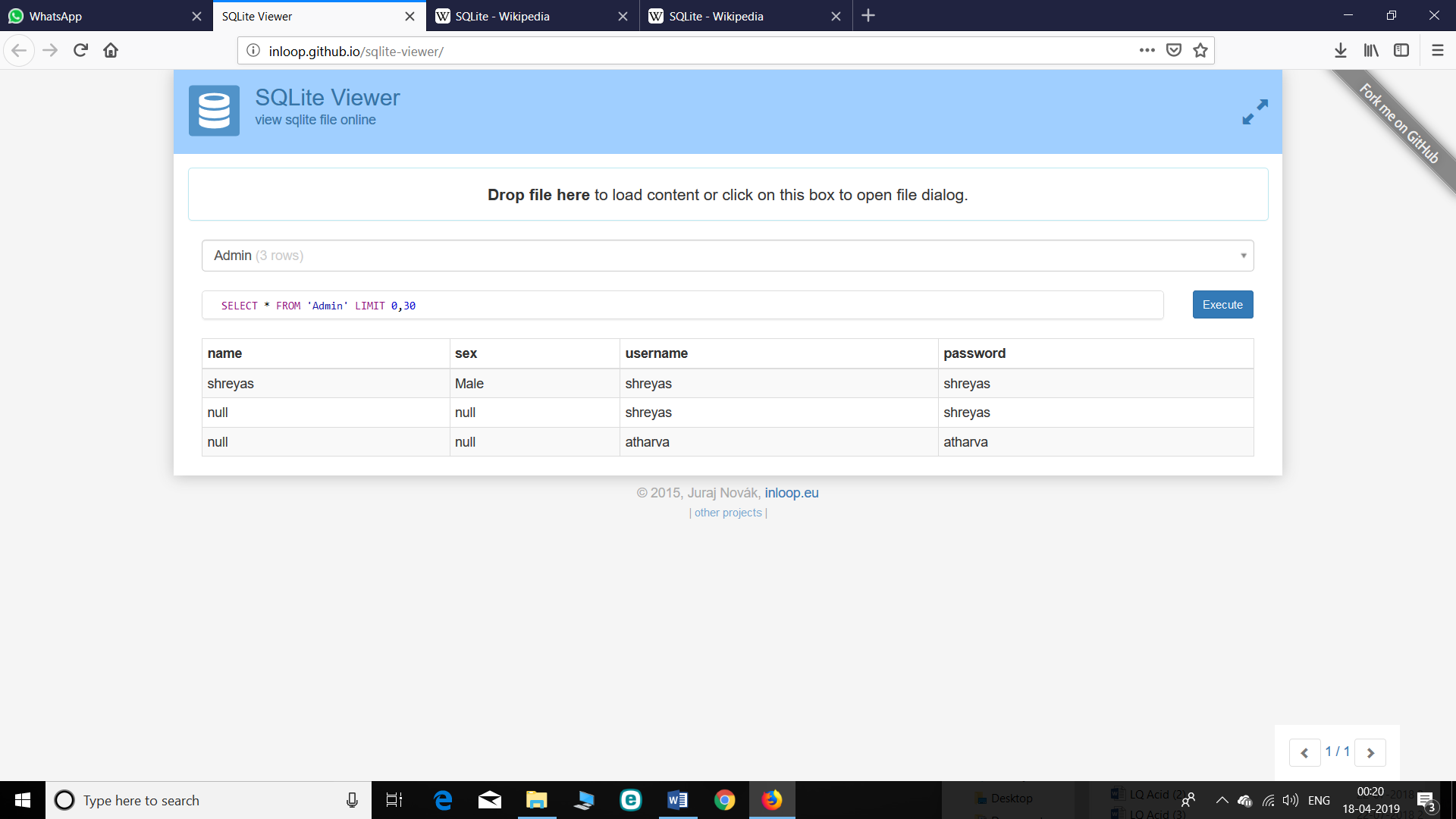
SQLite is [ACID](https://en.wikipedia.org/wiki/ACID)-compliant and implements most of the [SQL](https://en.wikipedia.org/wiki/SQL) standard, generally following [PostgreSQL](https://en.wikipedia.org/wiki/PostgreSQL) syntax. However, SQLite uses a dynamically and weakly [typed](https://en.wikipedia.org/wiki/Data_type) SQL [syntax](https://en.wikipedia.org/wiki/Syntax) that does not guarantee the [domain integrity](https://en.wikipedia.org/wiki/Domain_integrity). This means that one can, for example, insert a string into a column defined as an integer. SQLite will attempt to convert data between formats where appropriate, the string "123" into an integer in this case, but does not guarantee such conversions, and will store the data as-is if such a conversion is not possible.

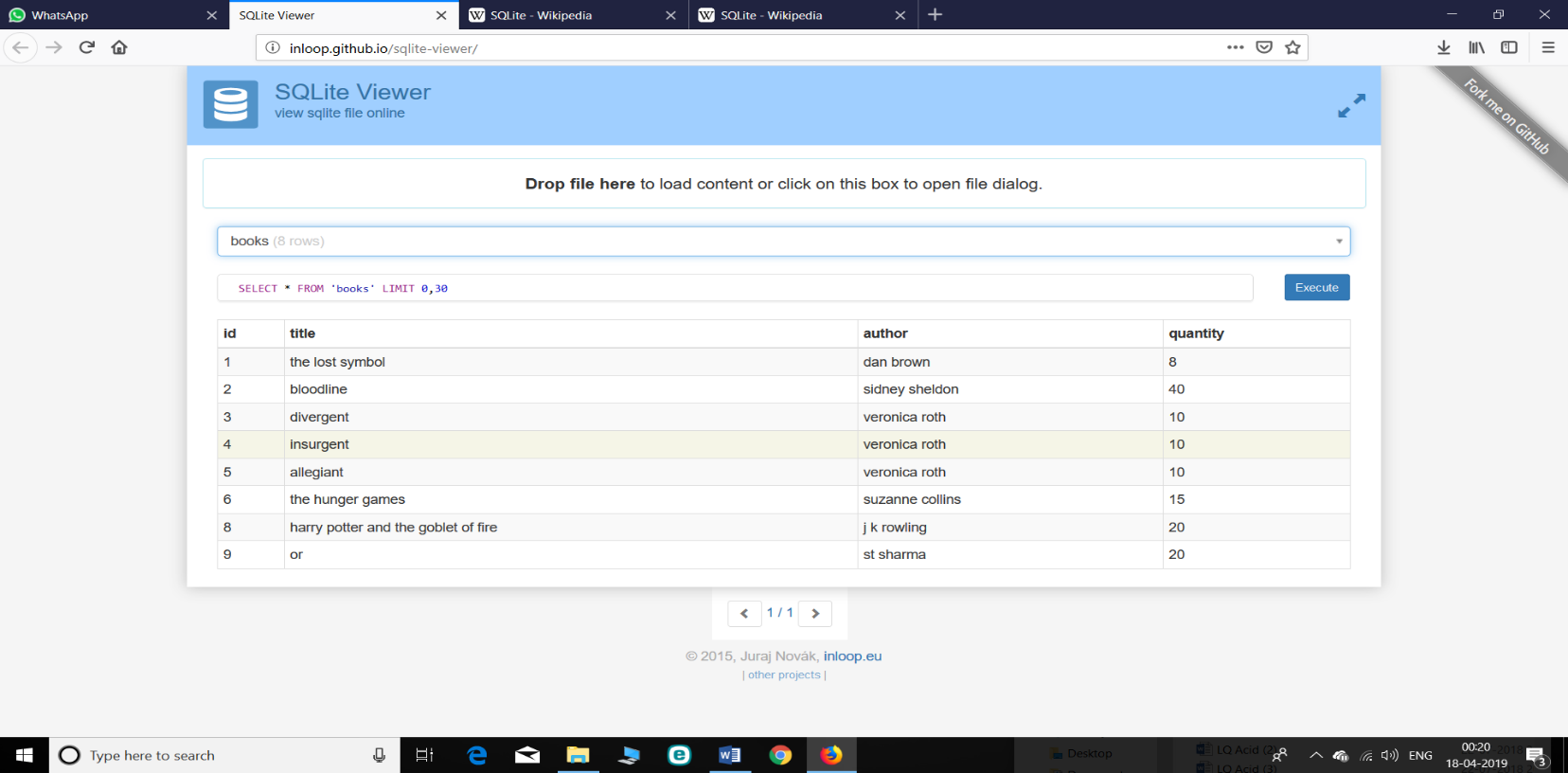
SQLite is a popular choice as [embedded database](https://en.wikipedia.org/wiki/Embedded_database) software for local/client storage in [application software](https://en.wikipedia.org/wiki/Application_software) such as [web browsers](https://en.wikipedia.org/wiki/Web_browser). It is arguably the most widely deployed [database engine](https://en.wikipedia.org/wiki/Database_engine), as it is used today by several widespread browsers, [operating systems](https://en.wikipedia.org/wiki/Operating_system), and [embedded systems](https://en.wikipedia.org/wiki/Embedded_system) (such as mobile phones), among others. SQLite has [bindings](https://en.wikipedia.org/wiki/Language_binding) to many programming languages.

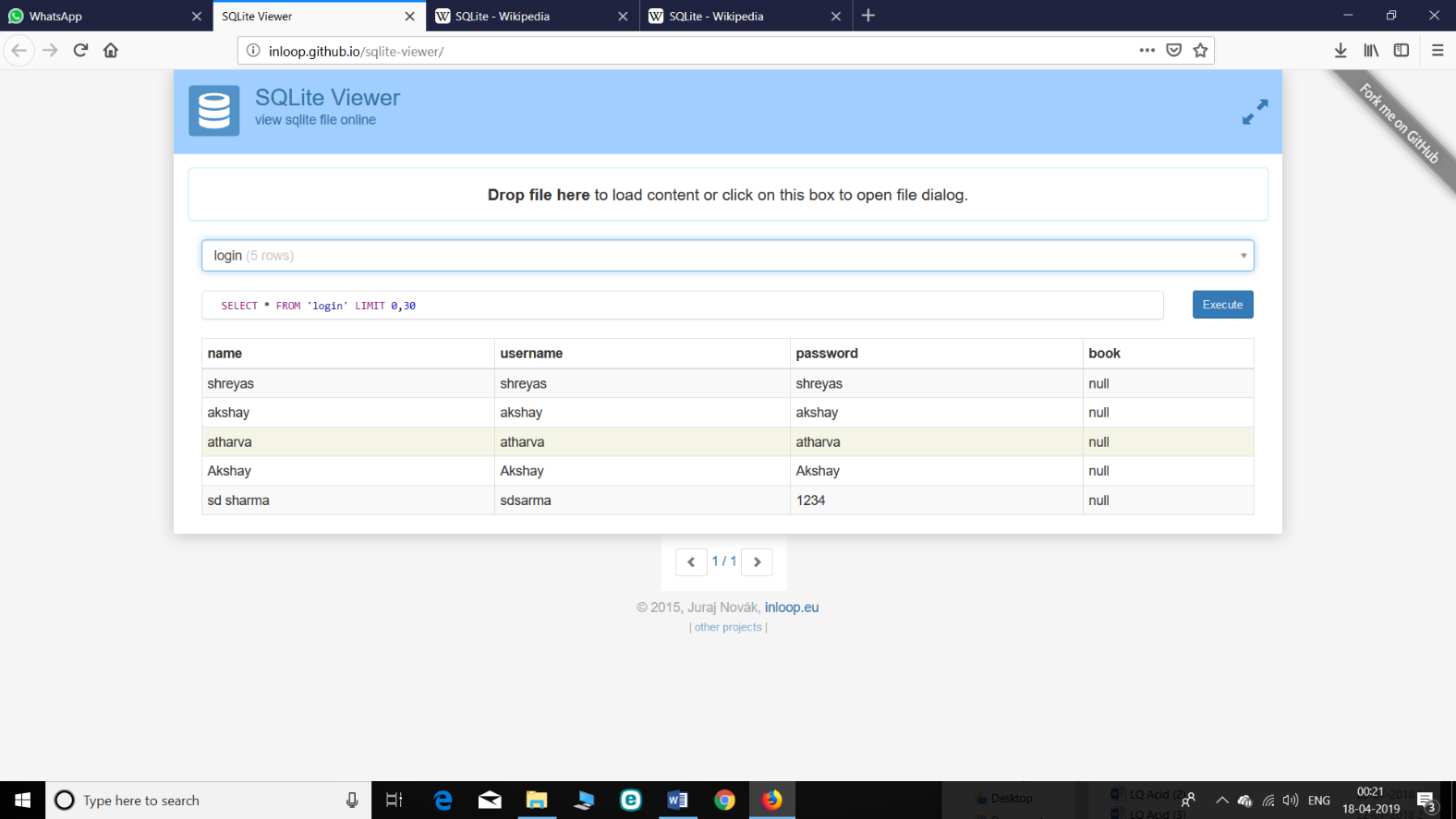
Unlike client–server database management systems, the SQLite engine has no standalone [processes](https://en.wikipedia.org/wiki/Process_(computing)) with which the application program communicates. Instead, the SQLite [library](https://en.wikipedia.org/wiki/Library_(computing)) is [linked](https://en.wikipedia.org/wiki/Linker_(computing)) in and thus becomes an integral part of the application program. Linking may be [static](https://en.wikipedia.org/wiki/Static_library) or [dynamic](https://en.wikipedia.org/wiki/Dynamic_linker). The application program uses SQLite's functionality through simple [function calls](https://en.wikipedia.org/wiki/Subroutine), which reduce [latency](https://en.wikipedia.org/wiki/Latency_(engineering)) in database access: function calls within a single process are more efficient than [inter-process communication](https://en.wikipedia.org/wiki/Inter-process_communication).

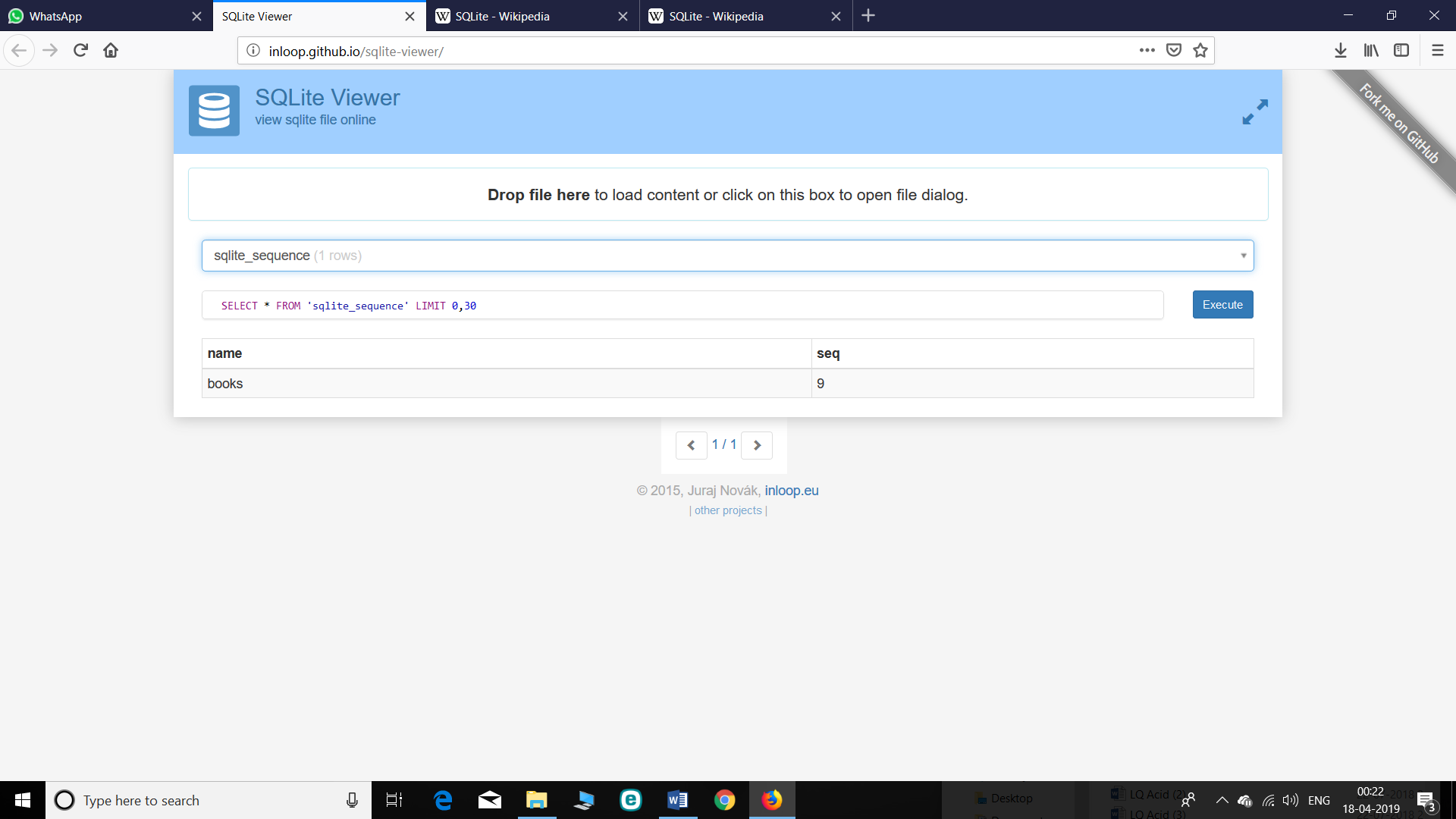


SQLITE VIEWER:

ADMIN PAGE :

BOOKS AVAILABLE:

LOGINS:

SEQUENCE:

**Python** offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.  
To create a tkinter:

1. Importing the module – tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

**tkinter** also offers access to the geometric configuration of the widgets which can organize the widgets in the parent windows. There are mainly three geometry manager classes class.

1. **pack() method:**It organizes the widgets in blocks before placing in the parent widget.
2. **grid() method:**It organizes the widgets in grid (table-like structure) before placing in the parent widget.
3. **place() method:**It organizes the widgets by placing them on specific positions directed by the programmer.

**Button**:To add a button in your application, this widget is used.  
The general syntax is:

w=Button(master, option=value)

master is the parameter used to represent the parent window.  
There are number of options which are used to change the format of the Buttons. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **activebackground**: to set the background color when button is under the cursor.
* **activeforeground**: to set the foreground color when button is under the cursor.
* **bg**: to set he normal background color.
* **command**: to call a function.
* **font**: to set the font on the button label.
* **image**: to set the image on the button.
* **width**: to set the width of the button.
* **height**: to set the height of the button.

**Canvas:** It is used to draw pictures and other complex layout like graphics, text and widgets.  
The general syntax is:

w = Canvas(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

1. **bd**: to set the border width in pixels.
2. **bg**: to set the normal background color.
3. **cursor**: to set the cursor used in the canvas.
4. **highlightcolor**: to set the color shown in the focus highlight.
5. **width**: to set the width of the widget.
6. **height**: to set the height of the widget.

**CheckButton:** To select any number of options by displaying a number of options to a user as toggle buttons. The general syntax is:

w = CheckButton(master, option=value)

There are number of options which are used to change the format of this widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **Title**: To set the title of the widget.
* **activebackground**: to set the background color when widget is under the cursor.
* **activeforeground**: to set the foreground color when widget is under the cursor.
* **bg**: to set he normal backgrouSteganography

Break

Secret Code:

Attach a File:nd color.

* **command**: to call a function.
* **font**: to set the font on the button label.
* **image**: to set the image on the widget.

**Entry:**It is used to input the single line text entry from the user.. For multi-line text input, Text widget is used.  
The general syntax is:

w=Entry(master, option=value)

master is the parameter used to represent the parent window.  
There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bd**: to set the border width in pixels.
* **bg**: to set the normal background color.
* **cursor**: to set the cursor used.
* **command**: to call a function.
* **highlightcolor**: to set the color shown in the focus highlight.
* **width**: to set the width of the button.
* **height**: to set the height of the button.

**Frame:** It acts as a container to hold the widgets. It is used for grouping and organizing the widgets. The general syntax is:

w = Frame(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **highlightcolor**: To set the color of the focus highlight when widget has to be focused.
* **bd**: to set the border width in pixels.
* **bg**: to set the normal background color.
* **cursor**: to set the cursor used.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.

**Label**: It refers to the display box where you can put any text or image which can be updated any time as per the code.  
The general syntax is:

w=Label(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bg**: to set he normal background color.
* **bg** to set he normal background color.
* **command**: to call a function.
* **font**: to set the font on the button label.
* **image**: to set the image on the button.
* **width**: to set the width of the button.
* **height**” to set the height of the button.

**MenuButton**: It is a part of top-down menu which stays on the window all the time. Every menubutton has its own functionality. The general syntax is:

w = MenuButton(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **activebackground**: To set the background when mouse is over the widget.
* **activeforeground**: To set the foreground when mouse is over the widget.
* **bg**: to set he normal background color.
* **bd**: to set the size of border around the indicator.
* **cursor**: To appear the cursor when the mouse over the menubutton.
* **image**: to set the image on the widget.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.
* **highlightcolor**: To set the color of the focus highlight when widget has to be focused.

**Menu**: It is used to create all kinds of menus used by the application.  
The general syntax is:

w = Menu(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of this widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **title**: To set the title of the widget.
* **activebackground**: to set the background color when widget is under the cursor.
* **activeforeground**: to set the foreground color when widget is under the cursor.
* **bg**: to set he normal background color.
* **command**: to call a function.
* **font**: to set the font on the button label.
* **image**: to set the image on the widget.

**Message**: It refers to the multi-line and non-editable text. It works same as that of Label.  
The general syntax is:

w = Message(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bd**: to set the border around the indicator.
* **bg**: to set he normal background color.
* **font**: to set the font on the button label.
* **image**: to set the image on the widget.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.

**RadioButton:** It is used to offer multi-choice option to the user. It offers several options to the user and the user has to choose one option.  
The general syntax is:

w = RadioButton(master, option=value)

There are number of options which are used to change the format of this widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **activebackground**: to set the background color when widget is under the cursor.
* **activeforeground**: to set the foreground color when widget is under the cursor.
* **bg**: to set he normal background color.
* **command**: to call a function.
* **font**: to set the font on the button label.
* **image**: to set the image on the widget.
* **width**: to set the width of the label in characters.
* **height**: to set the height of the label in characters

**Scrollbar**: It refers to the slide controller which will be used to implement listed widgets.  
The general syntax is:

w = Scrollbar(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **width**: to set the width of the widget.
* **activebackground**: To set the background when mouse is over the widget.
* **bg**: to set he normal background color.
* **bd**: to set the size of border around the indicator.
* **cursor**: To appear the cursor when the mouse over the menubutton.

**Text:** To edit a multi-line text and format the way it has to be displayed.  
The general syntax is:

w =Text(master, option=value)

There are number of options which are used to change the format of the text. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **highlightcolor**: To set the color of the focus highlight when widget has to be focused.
* **insertbackground**: To set the background of the widget.
* **bg**: to set he normal background color.
* **font**: to set the font on the button label.
* **image**: to set the image on the widget.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.

**TopLevel:** This widget is directly controlled by the window manager. It don’t need any parent window to work on.The general syntax is:

w = TopLevel(master, option=value)

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bg**: to set he normal background color.
* **bd**: to set the size of border around the indicator.
* **cursor**: To appear the cursor when the mouse over the menubutton.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.

**SpinBox:** It is an entry of ‘Entry’ widget. Here, value can be input by selecting a fixed value of numbers.The general syntax is:

w = SpinBox(master, option=value)

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bg**: to set he normal background color.
* **bd**: to set the size of border around the indicator.
* **cursor**: To appear the cursor when the mouse over the menubutton.
* **command**: To call a function.
* **width**: to set the width of the widget.
* **activebackground**: To set the background when mouse is over the widget.
* **disabledbackground**: To disable the background when mouse is over the widget.
* **from\_**: To set the value of one end of the range.
* **to**: To set the value of the other end of the range.