

# SQL Databases

Name: \_\_\_\_\_ ( ) Class: \_\_\_\_\_ Date: \_\_\_\_\_

## **Lesson 5: Using DB Browser for SQLite**

---

### **Instructional Objectives:**

By the end of this task, you should be able to:

- Understand that Structured Query Language (SQL) is the standard language for relational database management systems (RDBMS)
- Understand the concept of type affinities found in SQLite and how it affects table columns
- State that the basic database operations in relational databases are CREATE, READ, UPDATE and DELETE (CRUD)
- Use DB Browser for SQLite to perform basic database operations:
  - create a database
  - create a table
  - change a table definition
  - insert a record
  - update a record
  - delete a record
  - remove a table
  - delete a database
- Apply constraints: UNIQUE, AUTOINCREMENT, NOT NULL, PRIMARY KEY, FOREIGN KEY in DB Browser for SQLite

# SQL Databases

---

## Structured Query Language

**Structured Query Language (SQL)** is a standard computer language for the operation and management of relational databases. It is a language used to query, insert, update and modify data.



### History of SQL

In 1970, Dr. E.F. Codd published "A Relational Model of Data for Large Shared Data Banks," an article that outlined a model for storing and manipulating data using tables. Using the article as a premise, Donald D. Chamberlin and Raymond F. Boyce of IBM began developing a relational database.

By 1986, SQL had become the defacto data query language used in such databases.

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987. Since then, the standard was updated several times. Most major relational databases support this standard but would have their own proprietary extensions.

For an introduction to relational databases and SQL, watch the following video:

Khan Academy (2018, March 23). *Intro to SQL: Querying and managing data*. Retrieved from <https://www.khanacademy.org/computing/computer-programming/sql/sql-basics/v/welcome-to-sql>

---

## SQLite

There are many types of SQL database engines. A database engine is the software that a database management system (DBMS) uses to create, read, update and delete (CRUD) data from a database.

SQLite is a widely used database engine. Python's IDLE comes with a built-in module for SQLite3.

# SQL Databases



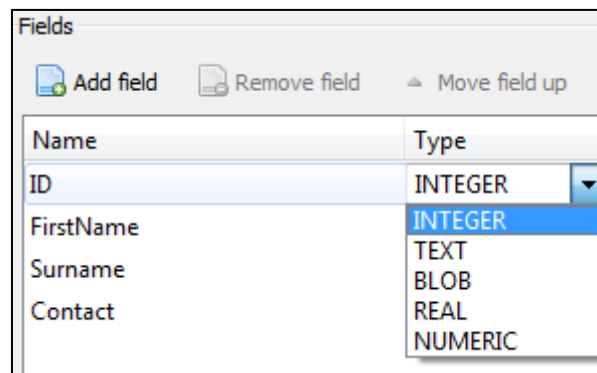
## Type Affinities

SQLite supports the concept of **type affinity** on columns. Type affinity refers to the preferred type for data stored in a column. This means that you can store any type of data in a column with the recommended types, but they are not enforced.

Each column in a SQLite table is assigned one of the following type affinities:

- INTEGER
- TEXT
- REAL
- NUMERIC
- BLOB

If you click on a table to view its columns in DB Browser for SQLite, you will notice these types as well.



# SQL Databases

In summary, you will only need to know and use these three types:

Types	Meaning
<b>INTEGER</b>	Used to store a signed integer value. The value is stored in 1, 2, 3, 4, 6, or 8 bytes depending on the magnitude of the value
<b>TEXT</b>	Used to store a text string using the database encoding (UTF-8, UTF-16BE or UTF-16LE)
<b>REAL</b>	Used to store a floating point value, as an 8-byte IEEE floating point number

BLOB stands for **Binary Large Object (BLOB)** which is used to store large binary data, such as images or multimedia in a database.

Type affinities are relevant in database migrations. This maximizes the compatibility between SQLite and other database management systems. For example, a column declared as VARCHAR in one database is assigned the type affinity of TEXT if it was migrated to and stored in a SQLite database instead. This feature helps to ensure portability across databases.

## Limitation of DB Browser for SQLite

Note that in DB Browser for SQLite, even if a column is assigned a specific type, you can still insert values of other types into the column!

**Table: class\_list**

	Index	Name
	Filter	Filter
1	1	Geraldine Tan
2	2	100
3	3	0.333

**Table column  
of a single  
type affinity**

# SQL Databases

## Typeof Function

SQLite provides the **typeof()** function that allows you to check the data type of an expression.

### sqlite3.typeof()

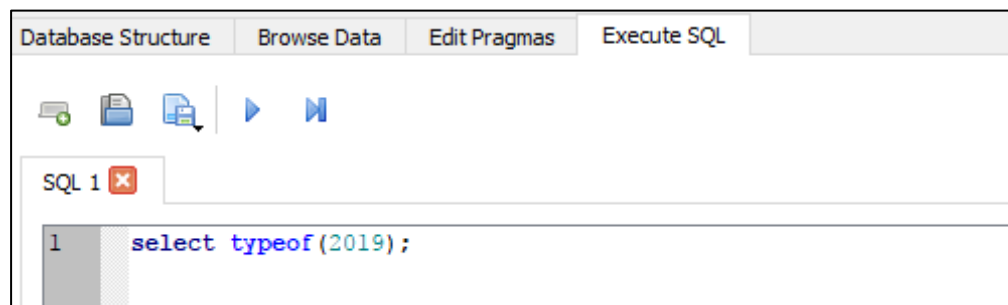
Returns a string that indicates the data type of an expression.

The only return values are: "null", "integer", "real", "text", or "blob".

## Task 1 – Using typeof Function

1. Open DB Browser for SQLite.
2. Create a new database.
3. Click on the Execute SQL tab.
4. Enter a simple query using **typeof()** function to find the type for each of the value below.

An example of how typeof is used is shown here.



Value	Type
i) 2019	
ii) '2019'	
iii) 345.2147	
iv) NULL	
v) 'True and False'	
vi) x'1001'	
vii) true	

# SQL Databases

## Boolean

SQLite does not have a type for Boolean values. Boolean values are stored as INTEGER with 0 (False) and 1 (True) values. For example, in the Light table below, the status of each LED lamp is of type INTEGER. 0 indicates that the light is not functioning, while 1 indicates that the light is functioning.

### Light

led_id	status
1001	0
1002	1
1003	0

---

## Quiz

For each sentence, write True or False.

	a. The typeof() function returns the data type of an expression. It returns the string 'No such column' if none of the data types are found.
	b. If there exists a table column with values ranging from 1 to 200 of type 'INT' in MySQL, the affinity type for it in SQLite is REAL.
	c. SQLite has type BOOLEAN.
	d. If data is imported from another database management system into DB Browser for SQLite, it is important to check on the type affinities as they may not be preserved during the migration.

---

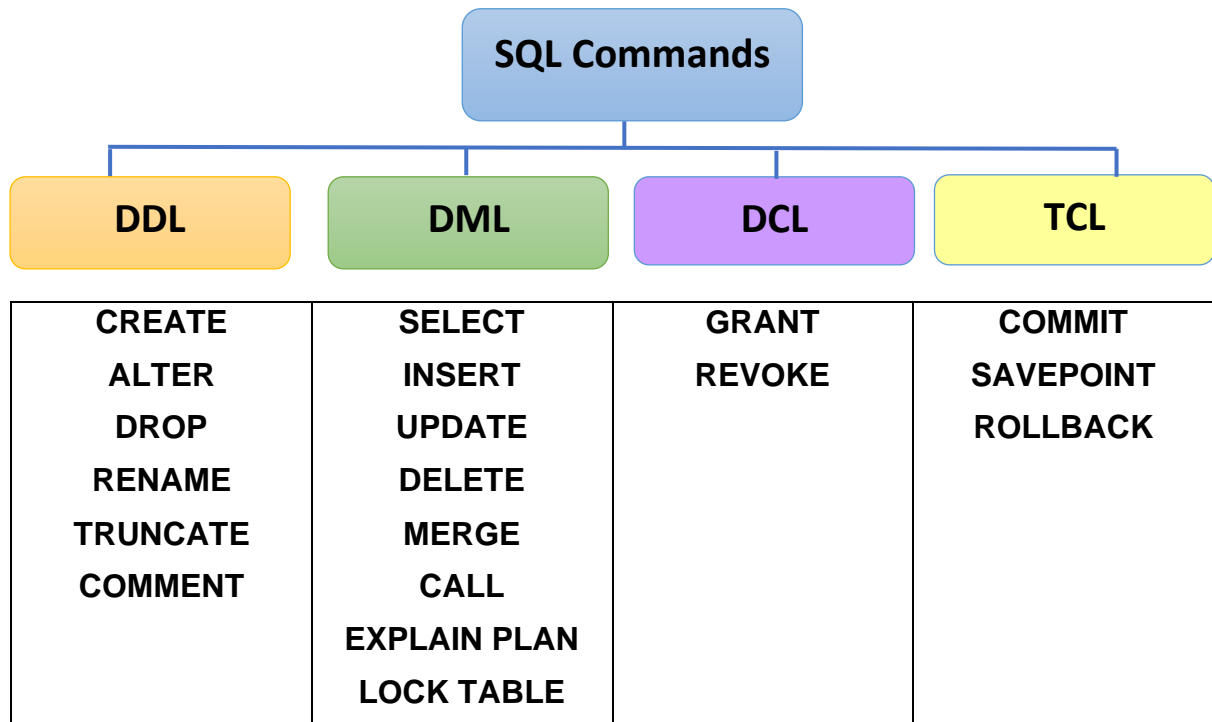
## Database Operations

In industry-based database applications, all four categories of SQL commands listed below will be required.

- Data Definition Language (DDL) defines database schemas.
- Data Manipulation Language (DML) is used to retrieve and modify data.
- Data Control Language (DCL) is used to control access to a database.

# SQL Databases

- Transaction Control Language (TCL) is used to manage changes to a database, usually at transactional level.



Some of the more advanced commands under DCL and TCL are more relevant to industry-specific roles, such as database administrators. For the purposes of our learning, you will need to understand and apply these basic CRUD database operations:

Operation	SQL Command
<u>C</u> REATE	INSERT
<u>R</u> EAD (RETRIEVE)	SELECT
<u>U</u> PDATE (MODIFY)	UPDATE
<u>D</u> ELETE (DESTROY)	DELETE

---

## DB Browser for SQLite

DB Browser for SQLite is a simple and easy to use Graphical User Interface (GUI) - based software for the creation and editing of database files compatible with SQLite. It abstracts and hides the details of complex SQL commands while providing an easy to user interface for performing the same database operations.

# SQL Databases

The downloadable version of it is available at <http://sqlitebrowser.org/>.

An online version of it known as SQLite Online is found at <https://sqliteonline.com/>

Users and developers who are familiar with SQLite3 may use command-line tools to manage SQLite database files.

## Note:

The command-line version of SQLite client is not approved for use in the school environment as it requires the running of an executable file, sqlite3.exe. Moreover, it is more commonly used to administer SQLite databases. Hence, you will need to be familiar with these two other ways of using SQLite:

- a. Using the features in DB Browser for SQLite software
- b. Using sqlite3 module in Python IDLE

---

## Task 2 - Create Database and Table

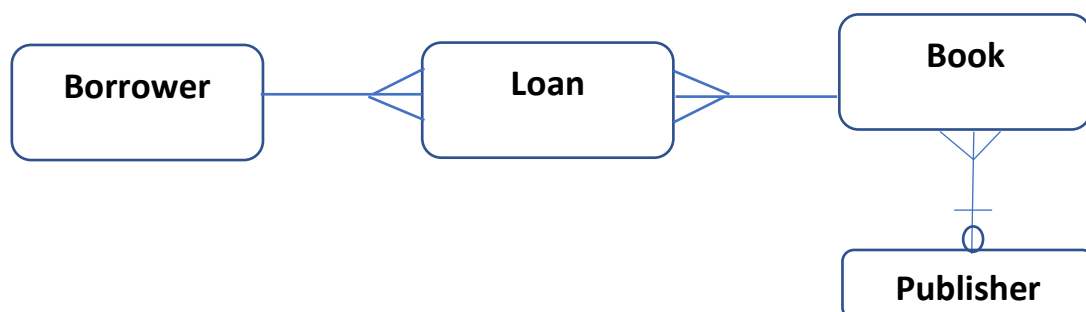
You will now create a library database with four tables using DB Browser for SQLite.

The library contains books that can be on loan to borrowers.

- A borrower can take one or more loans.
- Each loan record belongs to only one borrower.
- A book can be loaned many times.
- A publisher publishes one or more books.
- A book can be published by zero or one publisher.

For example, school lecture notes are not published by an official publishing house.

The Entity-Relationship (E-R) diagram below is provided to show the tables and the relationships between them.





# SQL Databases

The first table you will create is the Borrower table as shown below. After the creation of the table, you will apply some constraints on the table.

## Borrower

Column Name	Type
ID	INTEGER
FirstName	TEXT
Surname	TEXT
Contact	TEXT

## Table Constraints:

- ID is the **PRIMARY KEY** of the Borrower table  
This means that ID is used to identify a Borrower.
- The value of ID should be **AUTOINCREMENT**  
This means that the ID value increases automatically with each new record inserted.
- All fields are **NOT NULL**  
Each field cannot be empty.

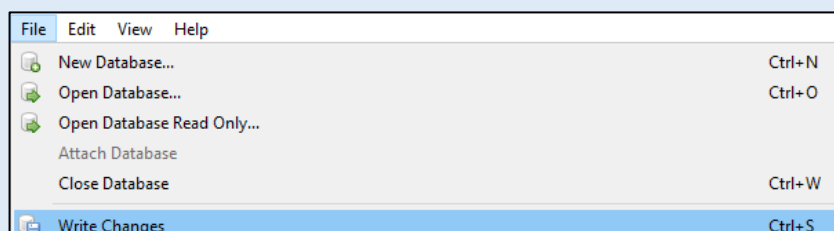
1. Create a folder called **DBTASK**. You will save all your files inside this folder.
2. Open DB Browser for SQLite.
3. Click **File**, then **New Database**.
4. Save your database file as library. The default extension is .db.

**Note: other database file extensions are sqlite/sqlite3/db3**

5. Create a table called Borrower with the fields and constraints listed above.
6. Click **Write Changes** or **CTRL + S** to save changes to the database.

### COMMIT in DB Browser for SQLite

In DB Browser for SQLite, the equivalent of the COMMIT command in SQLite is **Write Changes**. This feature saves changes but does not close the database file.

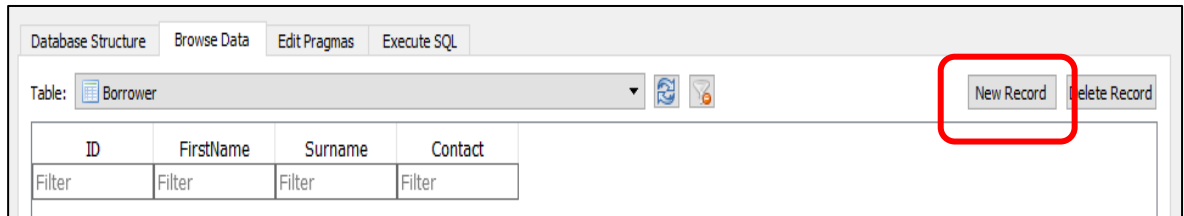


# SQL Databases

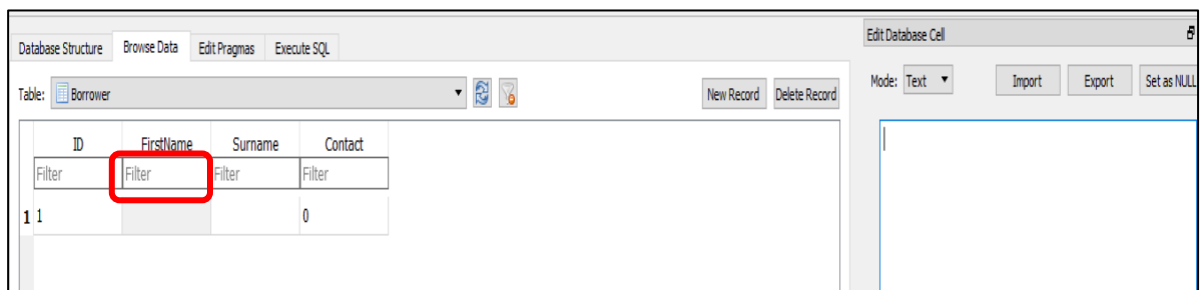
## Task 3 - Insert Records [CREATE]

After creating the library database and Borrower table, you will now add four records to the table.

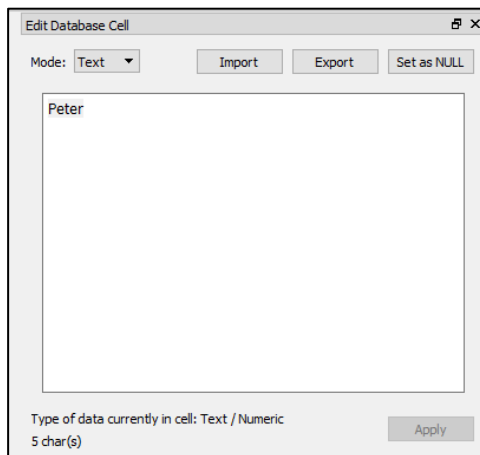
1. Under the Browse Data tab, click **New Record**.



2. Click on the FirstName cell of the first record.



3. Under Edit Database Cell, type the value for FirstName. Click **Apply**.



4. Repeat the above two steps for Surname and Contact.

If the record has been entered correctly, you should see the following in the table:

# SQL Databases

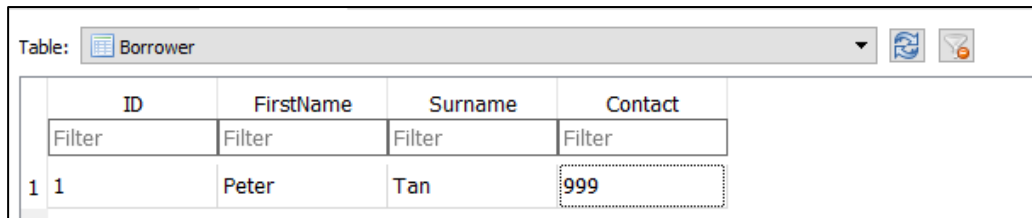


Table: Borrower

ID	FirstName	Surname	Contact
1	Peter	Tan	999

- Click New Record to enter values for the next few records.

## Borrower

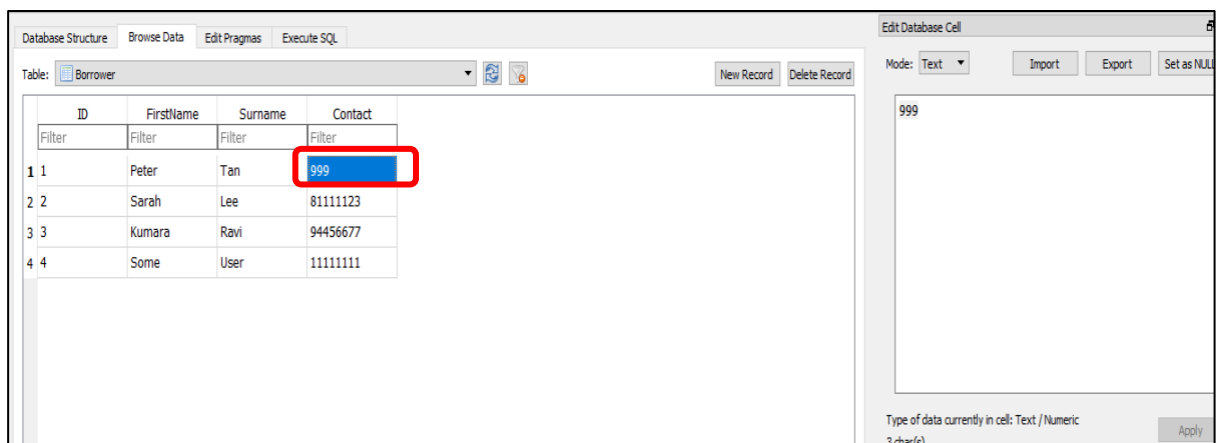
ID	FirstName	Surname	Contact
1	Peter	Tan	999
2	Sarah	Lee	81111123
3	Kumara	Ravi	94456677
4	Some	User	11111111

- Write changes to the database.

## Task 4 - Update Records [UPDATE]

The contact number of one of the borrowers, Peter Tan, is incorrect. You will update values in the Borrower table using Edit Database Cell in DB Browser for SQLite.

- Click on the Contact cell of the first record.



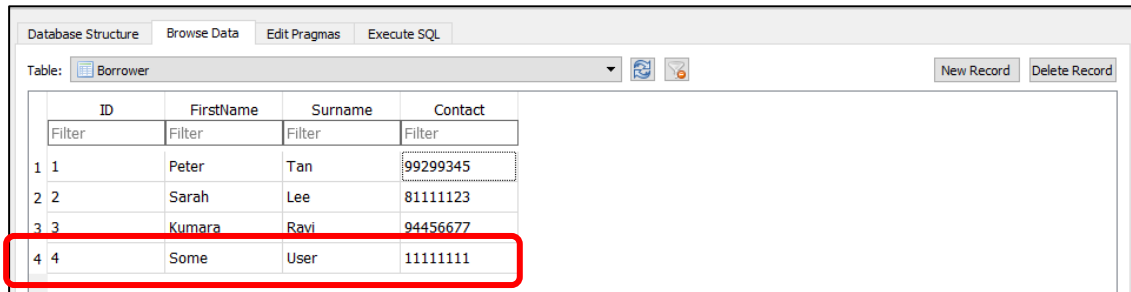
- Under Edit Database Cell, update the value to 99299345.
- Click **Apply**.

# SQL Databases

## Task 5 - Delete Records [DELETE]

One of the records in the Borrower table is redundant, hence remove it.

1. Select record 4.



The screenshot shows the DB Browser for SQLite interface. The 'Table: Borrower' is selected, and the data is displayed in a table. Record 4 is highlighted with a red rectangle.

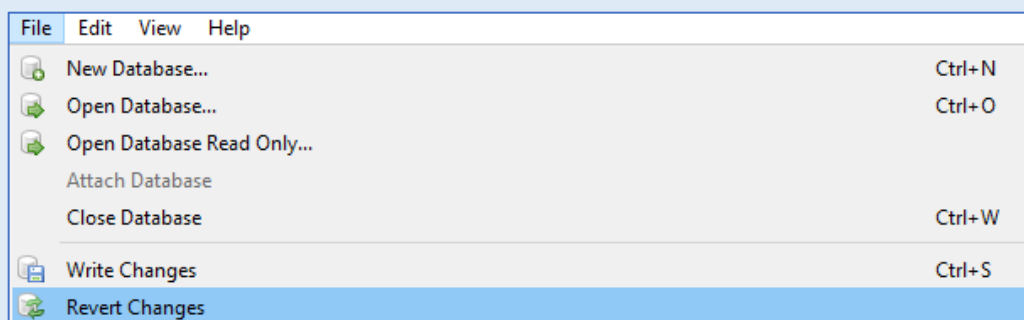
	ID	FirstName	Surname	Contact
1	1	Peter	Tan	99299345
2	2	Sarah	Lee	81111123
3	3	Kumara	Ravi	94456677
4	4	Some	User	11111111

2. Click **Delete Record**.
3. Write changes to the database.
4. Now add another record for Borrower. Type in ID, FirstName, Surname and Contact of your choice.
5. Click **Revert Changes**.

What do you observe?

### ROLLBACK in DB Browser for SQLite

In DB Browser for SQLite, the equivalent of the ROLLBACK command in SQLite is **Revert Changes**. This feature is useful for undoing any modifications to the database since the last saved state.



More details on ROLLBACK will be provided in later lessons.

6. Write changes to the database.

# SQL Databases

## Task 6 – Creating More Tables [CREATE, INSERT]

After creating the library database and Borrower table, you will now create the **Publisher** and **Book** tables and apply the relevant constraints. You will need to take note of special constraints which help to maintain inter-table dependencies and the integrity of related data in different tables. They will affect the order in which tables are created.

1. Using DB Browser for SQLite, create the **Publisher** table with the following types and constraints.

### Publisher

Column Name	Type
ID	INTEGER
Name	TEXT

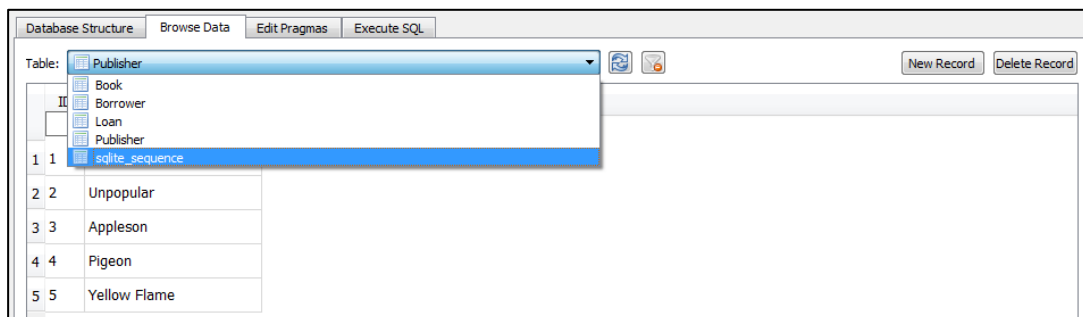
### Table Constraints:

- ID is the **PRIMARY KEY** of the Publisher table
- The value of ID should be **AUTOINCREMENT**
- All fields are **NOT NULL**

2. Insert the following records into the **Publisher** table.

ID	Name
1	NPH
2	Unpop
3	Appleson
4	Squirrel
5	Yellow Flame

3. If you have successfully created the Publisher table, you can view it under the Browse Data tab.



# SQL Databases

4. Create the **Book** table with the following types and constraints.

**Note:**

The Publisher table has to be created before the Book table because of the foreign key reference to ID in the Publisher table.

**Rule:**

**Tables with foreign keys have to be created after the referenced tables are created.**

## Book

Column Name	Type
ID	INTEGER
Title	TEXT
PublisherID	INTEGER
Damaged	INTEGER

Table Constraints:

- ID is the **PRIMARY KEY** of the Book table.
- The value of ID should be **AUTOINCREMENT**.
- ID, Title and Damaged fields are **NOT NULL**  
Damaged is an attribute that tracks the condition of the book.  
A value of 0 means that the book is not damaged, while a value of 1 means that the book is damaged.
- PublisherID is a **FOREIGN KEY** to ID in the Publisher table.

5. Insert records to Book table as follows:

ID	Title	PublisherID	Damaged
1	The Lone Gatsby	5	0
2	A Winter's Slumber	4	1
3	Life of Pie	4	0
4	A Brief History Of Primates	3	0
5	To Praise a Mocking Bird	2	0
6	The Catcher in the Eye	1	1
123	H2 Computing Ten Year Series	NULL	0

6. Write changes to the database.

---

## Task 7 – Creating Table Using Import [CREATE, INSERT]

You will now create the **Loan** table by importing a text file into the library database.  
The types and constraints are described below.

# SQL Databases

## Loan

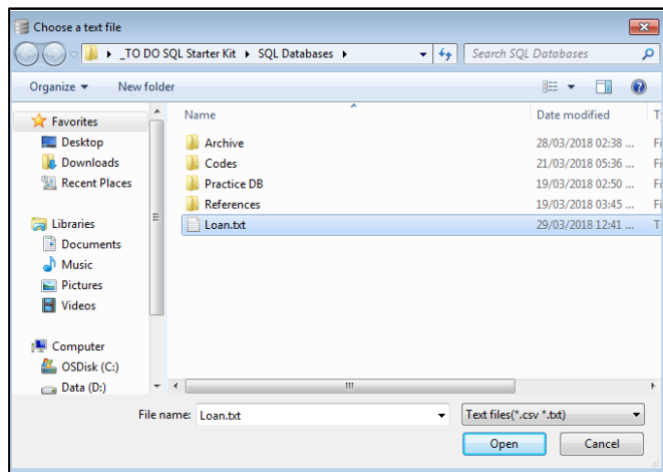
Column Name	Type
ID	INTEGER
BorrowerID	INTEGER
BookID	INTEGER
DateBorrowed	TEXT

1. Create the **Loan** table using the **Import** feature.

This feature also allows importing of .TXT and .CSV files.



2. Select Loan.txt.



3. Click **Open**.
4. Tick the option **Column names in first line**.

# SQL Databases

	ID	BorrowerID	BookID	Date Borrowed
1	1	3	2	20180220
2	2	3	1	20171215
3	3	2	3	20171231
4	4	1	5	20180111

5. Click OK.
6. Click **Modify Table**.
7. Edit the types according to the description above.

## Note:

The types for every column in the table are defaulted to TEXT during an import. Hence it is important that you check on the types after an import.

Name	Type	Not	PK	AI	U	Default	Check
ID	INTEGER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
BorrowerID	INTEGER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
BookID	INTEGER	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
DateBorrowed	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		

8. Tick the constraints as according to below.

## Table Constraints:

- ID is the **PRIMARY KEY** of the Loan table
- The value of ID should be **AUTOINCREMENT**
- ID, BorrowerID and BookID fields are **NOT NULL**

For BorrowerID and BookID of the Loan table, identify the **FOREIGN KEY** constraints.

- BorrowerID is a **FOREIGN KEY** to \_\_\_\_\_ in the \_\_\_\_\_ table
- BookID is a **FOREIGN KEY** to \_\_\_\_\_ in the \_\_\_\_\_ table.



# SQL Databases

9. To create the foreign key for BorrowerID, highlight the BorrowerID attribute. Type **Borrower(ID)** under Foreign Key column.

This creates a foreign key reference to ID in the Borrower table.

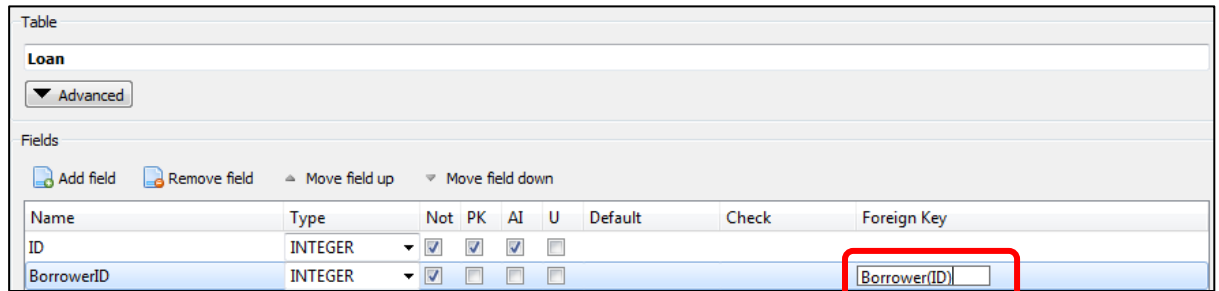


Table: Loan

Advanced

Fields

Add field Remove field Move field up Move field down

Name	Type	Not	PK	AI	U	Default	Check	Foreign Key
ID	INTEGER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>			
BorrowerID	INTEGER	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>			Borrower(ID)

10. Repeat the above step for BookID to create the foreign key reference.
11. View the Loan table from **Browse Data** tab. You should see the following data:





ID	BorrowerID	BookID	DateBorrowed
1	3	2	20180220
2	3	1	20171215
3	2	3	20171231
4	1	5	20180111

12. Write changes to the database.




# SQL Databases

## Summary of DB Browser for SQLite Features

### File Features

Name	Description
 New Database	Creates a new SQLite database file. File extensions are .db or .db3 or .sqlite or .sqlite3
 Open Database	Opens a SQLite database file.
 Write Changes	Saves changes to database. Equivalent to COMMIT operation.
 Revert Changes	Undo any changes made to the database. Equivalent to ROLLBACK operation.
Import	Imports either a CSV or TXT file as a table into the database or SQLite database file.
Export	Exports one of the following: <ul style="list-style-type: none"><li>• a table in the database as a new CSV file</li><li>• database as a SQLite file</li><li>• database as a JSON file</li></ul>





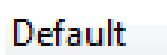
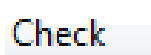
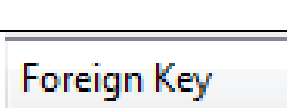
### Database Structure

Name	Description
 Create Table	Creates a table in the database.
 Modify Table	Modifies a table in the database, not limited to changing table name or field name, adding fields or constraints.
 Delete Table	Deletes a table from the database.



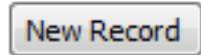
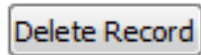
# SQL Databases

Constraints are the rules enforced on data columns or table. These are used to limit the type of data that can go into a column or table. This ensures the accuracy and reliability of the data in the database. Constraints could be at a column level or table level. Column level constraints are applied only to one column, whereas table level constraints are applied to the whole table.

## Constraints

Name	Description
	Ensures that a column cannot have NULL value.
	Sets a PRIMARY KEY constraint such that it uniquely identifies each row/record in a table.
	Automatically increments the value of the attribute for each new record. Works for integer values only.
	Ensures that all values in a column are unique.
	Provides a default value for a column when none is specified.
	Ensures that all values in a column satisfies certain conditions. If the condition evaluates to false, the record violates the constraint and isn't entered into the table.
	Sets a FOREIGN KEY constraint where a column of a table can reference a column from another table or within the same table.

## Browse Data

Name	Description
	Refresh data in the selected table.
	Clear all filters.
	Creates a new record in the table.
	Deletes a record in the table.

# SQL Databases

## References

Content	Link
DB Browser for SQLite Online	<a href="https://sqliteonline.com/">https://sqliteonline.com/</a>
SQL	<p>Vertabelo Academy, SQL Online Interactive Courses <a href="https://academy.vertabelo.com/">https://academy.vertabelo.com/</a></p> <p>w3schools.com SQL Tutorial <a href="https://www.w3schools.com/sql/default.asp">https://www.w3schools.com/sql/default.asp</a></p> <p>Intro to SQL: Querying and managing data <a href="https://www.khanacademy.org/computing/computer-programming/sql">https://www.khanacademy.org/computing/computer-programming/sql</a></p> <p>SQL Zoo <a href="http://sqlzoo.net/">http://sqlzoo.net/</a></p> <p>SQLCourse.com Interactive Online SQL training <a href="http://www.sqlcourse.com/">http://www.sqlcourse.com/</a></p> <p>SQL Fiddle <a href="http://sqlfiddle.com/">http://sqlfiddle.com/</a></p> <p>SQL Style Guide <a href="http://www.sqlstyle.guide/">http://www.sqlstyle.guide/</a></p> <p>Codd's 12 Rule Relational Database Definition <a href="https://www.w3resource.com/sql/sql-basic/codd-12-rule-relation.php">https://www.w3resource.com/sql/sql-basic/codd-12-rule-relation.php</a></p>
Singapore Datasets	<p>Singapore's public data <a href="https://data.gov.sg/">https://data.gov.sg/</a></p> <p>Visualizing Singapore <a href="http://www.viz.sg/">http://www.viz.sg/</a></p>