Temasek Junior College

2023 JC2 H2 Computing

Database 3 – Basic Operations in DB Browser for SQLite

Syllabus Objectives

After completing this set of notes, you should be able to:

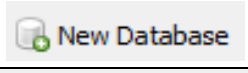
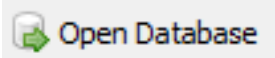
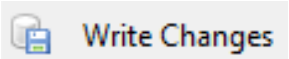
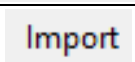
- Familiarise yourself with the basic operations of setting up a database in DB Browser for SQLite

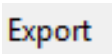
1 Introduction

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.

The table below summarrises the file handling features of the DB Browser.

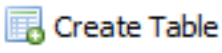
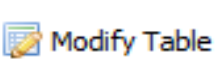
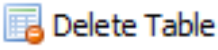
File Handling Features

Feature	Description
	• Creates a new SQLite database file. • File extensions are .db or .db3 or .sqlite or .sqlite3
	• Opens a SQLite database file.
	• Saves changes to database. • Equivalent to COMMIT operation.
	• Undo any changes made to the database. • Equivalent to ROLLBACK operation.
	• Imports either a CSV or TXT file as a table into the database or SQLite database file.

	• Exports one of the following:
	  ○ a table in the database as a new CSV file
	  ○ database as a SQLite file
	  ○ database as a JSON file

The table below summarrises the database modification features of the DB Browser.

Database Modification Features

Feature	Description
	• Creates a table in the database.
	• Modifies a table in the database, not limited to changing table name or field name, adding fields or constraints.
	• Deletes a table from the database.

Constraints are the rules enforced on data columns or tables. They are used to limit the type of data that can go into a column or table, ensuring 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.

The table below summarrises the constraints available in the DB Browser.

Constraints

Feature	Description
Not null	• Ensures that a column cannot have NULL value.
PK	• Sets a PRIMARY KEY constraint such that it uniquely identifies each row/record in a table.
AI	• Automatically increments the value of the attribute for each new record. • Works for integer values only.
U	• Ensures that all values in a column are unique.
Default	• Provides a default value for a column when none is specified.
Check	• 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.
Foreign Key	• Sets a FOREIGN KEY constraint where a column of a table can reference a column from another table or within the same table.

The table below summarrises the data browsing and modification features available in the DB Browser.

Data Browsing and Modification

Fearure	Description
	• Refresh data in the selected table.
	• Clear all filters.
New Record	• Creates a new record in the table.
Delete Record	• Deletes a record in the table.

2 Task 1 – 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 which may or may not be an official publishing house e.g school lecture notes are not published by an official publishing house.

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**. This means each field cannot be empty.

Steps

1. Create a folder called **DBTASK**. You will save all your files inside this folder.

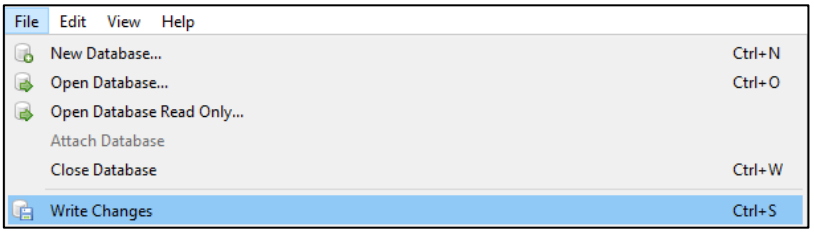
2. Open DB Browser for SQLite.

3. Click **File**, then **New Database**.

4. Save your database file using the filename **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.

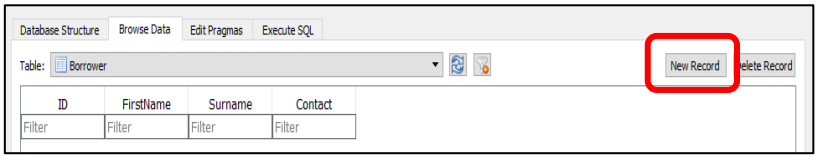
6. Click **Write Changes** or **CTRL + S** to save changes to the database.
   Note: In DB Browser for SQLite, the equivalent of the **COMMIT** command in SQLite is **Write Changes.** This feature saves changes but does not close the database file.



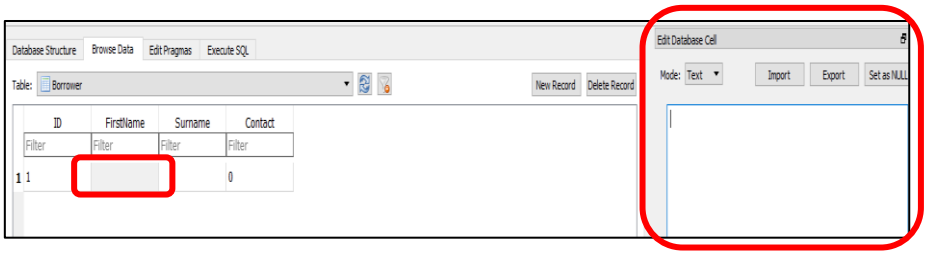
3 Task 2 – Insert Records

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

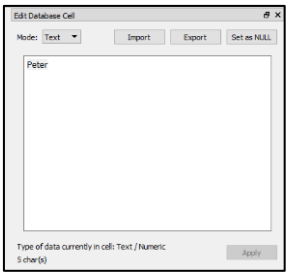
1. Under the **Browse Data** tab, click **New Record**.



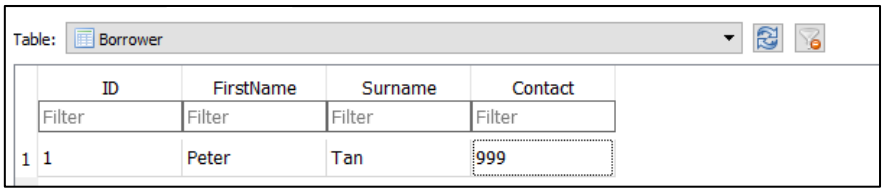
2. Click on the **FirstName** cell of the first record.



3. Under **Edit Database Cell** (see illustration above), 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:



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

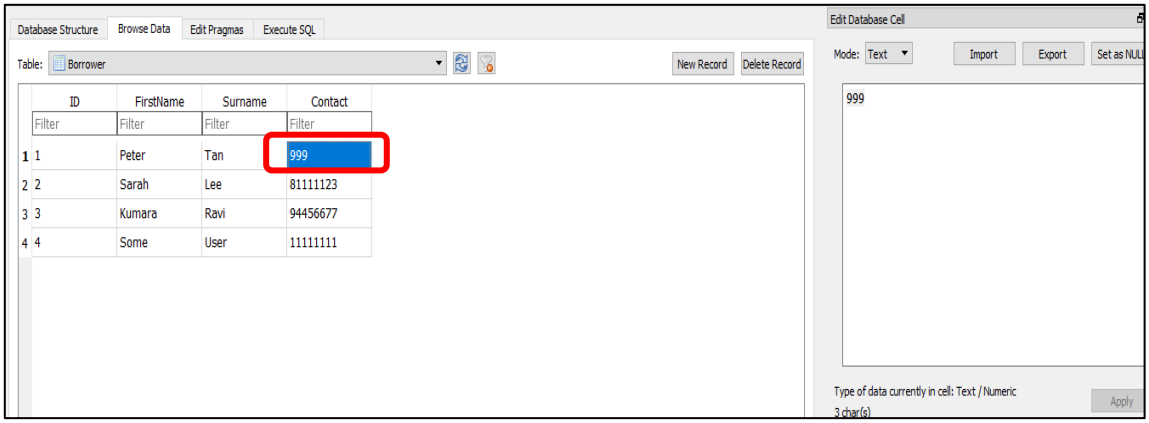
Borrower

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

6. **Write Changes** to the database.

4 Task 3 – Update Records

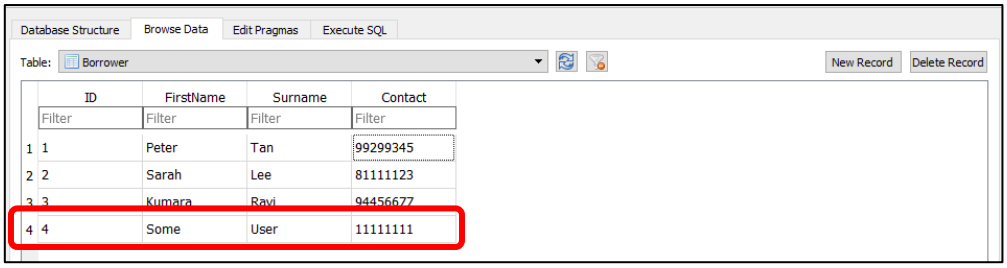
The contact number of one of the borrowers, Peter Tan, is incorrect. You will update values in the **Borrower** table using **Edit Database Cell** in DB Browser for SQLite.



1. Click on the **Contact** cell of the first record.
2. Under **Edit Database Cell**, update the value to 99299345.
3. Click **Apply**.

5 Task 4 – Delete Records

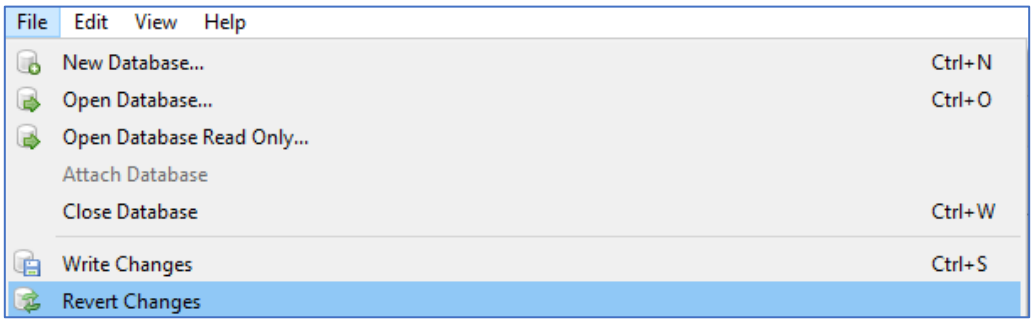
One of the records in the **Borrower** table is redundant, hence remove it.



1. Select record 4.
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?
6. **Write Changes** to the database.

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 at a later stage of learning.

6 Task 5 – Creating More Tables

After creating the **library** database and **Borrower** table, you will now create the **Publisher** and **Book** tables and apply their 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.

Rule
Tables with foreign keys should only be created after the referenced tables are created.

1. Create the **Publisher** table with the following types and constraints.

   Publisher

Column Name	Type
ID	INTEGER
Name	TEXT

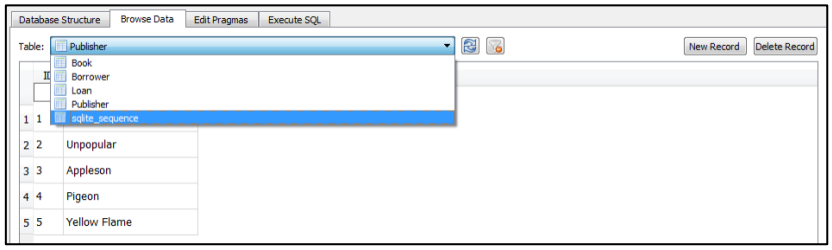
   Table Constraints

   - **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.



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

Book

Column Name	Type
ID	INTEGER
Title	TEXT
PublisherID	INTEGER
Damaged	INTEGER

Table Constraints

- **ID** is the **PRIMARY KEY** of the **Book** table.
- **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,
        - A value of 1 means that the book is damaged.
- **PublisherID** is a **FOREIGN KEY** to **ID** in the **Publisher** table.
- The **Book** table can only be created after the **Publisher** table because of the foreign key reference 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.

6 Task 6 – Creating Table Using Import

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

Loan

Column Name	Type
ID	INTEGER
BorrowerID	INTEGER
BookID	INTEGER
DateBorrowed	TEXT

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.

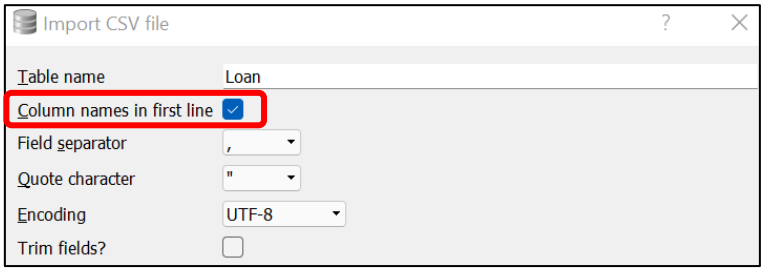
- **BorrowerID** is a **FOREIGN KEY** to _____ in the _____ table

- **BookID** is a **FOREIGN KEY** to _____ in the _____ table.

1. Create the **Loan** table using the **Import** feature. This feature allows importing of .txt and .csv files.

2. Import the text file Loan.txt from where it is stored in.

3. Select the option **Column names in first line**. This will create the column names using the first line of the text file.

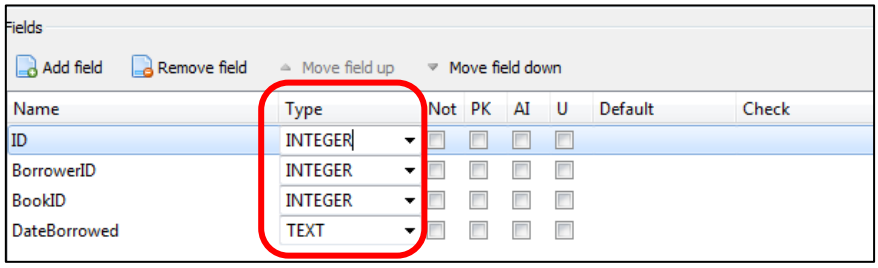


4. Click **OK**.

5. Click **Modify Table**.

6. Edit the types according to the description above.

   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.

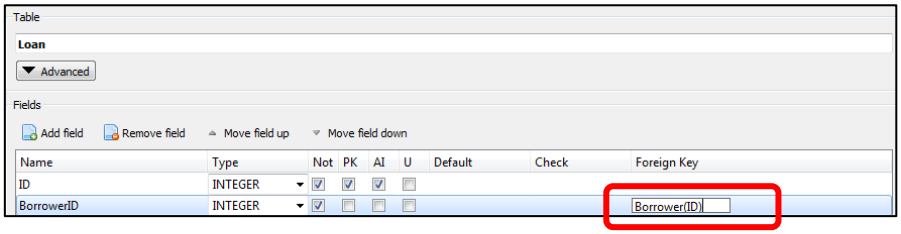


7. Check the table constraints accordingly.

8. 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.



9. Repeat the above step for **BookID** to create the foreign key reference.

10. 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

11. **Write Changes** to the database.