Create database

```
C:\Users\lenovo\OneDrive\Desktop\Database>sqlite3 practicedb
SQLite version 3.50.4 2025-07-30 19:33:53
Enter ".help" for usage hints.
sqlite> .database
```

Create table Author

```
sqlite> create table Author (authorID integer primarykey Autoincreament,
(x1...> Name Varchar(255)  not null,
(x1...> Country varchar(255));
sqlite> select * from author;
sqlite> .tables
Author
```

Create table books

```
sqlite> CREATE TABLE Books (
(x1...>     BookId    INTEGER PRIMARY KEY AUTOINCREMENT,
(x1...>     Title     TEXT NOT NULL,
(x1...>     AuthorId  INTEGER NOT NULL,
(x1...>     Published INTEGER,
(x1...>     FOREIGN KEY (AuthorId) REFERENCES Author(AuthorId)
(x1...> );
```

Create table Borrower

```
sqlite> CREATE TABLE Borrowers (
(x1...>     Id     INTEGER PRIMARY KEY AUTOINCREMENT,
(x1...>     Name   TEXT NOT NULL,
(x1...>     Email  TEXT UNIQUE
(x1...> );
sqlite> .tables
Author     Books      Borrowers
sqlite>
```

Insert into Author

```
sqlite> INSERT INTO Author (Name, Country)
   ...> VALUES ('Suyasha', 'Nepal'),
   ...>        ('Shambhavi', 'Nepal');
```

```
sqlite> select * from author;
|Suyasha|Nepal
|Shambhavi|Nepal
```

Insert into books

```
sqlite> INSERT INTO Books (Title, AuthorId, Published)
   ...> VALUES
   ...>     ('Mystery of the River', 1, 2020),
   ...>     ('Adventures in Space', 2, 2021),
   ...>     ('Legends of the Forest', 2, 2019);
```

Select all form books

```
sqlite> select * from books
   ...> ;
1|Mystery of the River|1|2020
2|Adventures in Space|2|2021
3|Legends of the Forest|2|2019
```

Select from books with author id as 1

```
sqlite> select * from books where AuthorId=1;
1|Mystery of the River|1|2020
sqlite>
```

Insert into borrowers

```
sqlite> INSERT INTO Borrowers (Email, Name)
   ...> VALUES
   ...>     ('bandana@example.com', 'Bandana'),
   ...>     ('kalidash@example.com', 'Kalidash');
```

Update data of borrower

```
sqlite> update borrowers set email='ban.koirala@gmail.com' where name='Bandana';
sqlite> select* from borrowers
```

```
sqlite> select* from borrowers
   ...> ;
1|Bandana|ban.koirala@gmail.com
2|Kalidash|kalidash@example.com
sqlite>
```

Delete data b=from book where ID is 1

```
sqlite> delete from books where bookid=1;
sqlite> select * form book;
```

```
sqlite> select * from books;
2|Adventures in Space|2|2021
3|Legends of the Forest|2|2019
sqlite>
```

Normalization

Business rule

- One-to-Many: An Author has many Books.
- Many-to-Many: A Borrower can borrow many Books, and a Book can be borrowed by many Borrowers.

UNF

Book {

BookId, Title, ISBN, PublishedYear,

AuthorId, AuthorName, AuthorCountry,

{BorrowerId, BorrowerName, BorrowerEmail} -- repeating group

}

1NF

Book-{ BookId, Title, ISBN, AuthorId, PublishedYear, Name, Country }

Borrower-{BookID, BorrowerID,Name, Email }

2NF//remove partial dependancy

Borrower-{BookID,Name, Email }

Borrower_book-{BookID,BorrowerID}

Book-{ BookId, Title, ISBN, AuthorId, PublishedYear}

Author-{ AuthorId, Name, Country }

3NF//no transient dependancy

Borrower-{BookID,Name, Email }

Borrower_book-{BookID,BorrowerID}

Book-{ BookId, Title, ISBN, AuthorId, PublishedYear}

Author-{ AuthorId, Name, Country }