CSE 3330 Database System & File Structures

11/15/2024

Project 2 library DB Part 2

Group 13, Section 004

Shaheen Nijamudheen, 1002101057

James Paul Nguyen, 1001983168

Ali Shirazi-Nejad, 1002062834

TABLE OF CONTENTS

Task 1:

```
CREATE TABLE PUBLISHER (
   Publisher_Name VARCHAR(100) PRIMARY KEY,
   Phone VARCHAR(15),
   Address VARCHAR(255)
);

CREATE TABLE LIBRARY_BRANCH (
   Branch_Id INT PRIMARY KEY,
   Branch_Name VARCHAR(100),
   Branch_Address VARCHAR(255)
);

CREATE TABLE BORROWER (
   Card_No INT PRIMARY KEY,
   Name VARCHAR(100),
   Address VARCHAR(255),
   Phone VARCHAR(15)
);

CREATE TABLE BOOK (
   Book_Id INT PRIMARY KEY,
   Title VARCHAR(255),
   Publisher_Name VARCHAR(100),
   FOREIGN KEY (Publisher_Name) REFERENCES PUBLISHER(Publisher_Name)
);
```

```
CREATE TABLE BOOK_AUTHORS (
   Book_Id INT,
   Author_Name VARCHAR(100),
   PRIMARY KEY (Book_Id, Author_Name),
   FOREIGN KEY (Book_Id) REFERENCES BOOK(Book_Id)
);

CREATE TABLE BOOK_COPIES (
   Book_Id INT,
   Branch_Id INT,
   No_Of_Copies INT,
   PRIMARY KEY (Book_Id, Branch_Id),
   FOREIGN KEY (Book_Id) REFERENCES BOOK(Book_Id),
   FOREIGN KEY (Branch_Id) REFERENCES LIBRARY_BRANCH(Branch_Id)
);

CREATE TABLE BOOK_LOANS (
   Book_Id INT,
   Branch_Id INT,
   Card_No INT,
   Date_Out DATE,
   Due_Date DATE,
   Returned_date DATE,
   PRIMARY KEY (Book_Id, Branch_Id, Card_No),
   FOREIGN KEY (Book_Id) REFERENCES BOOK(Book_Id),
   FOREIGN KEY (Branch_Id) REFERENCES LIBRARY_BRANCH(Branch_Id),
   FOREIGN KEY (Card_No) REFERENCES BORROWER(Card_No)
);
```

- Task 1 commands, and if you have any comments to add, e.g., how you decided on specific data types, or constraints etc.

In the PUBLISHER table, the `Publisher_Name` is set as the primary key because it uniquely identifies each publisher in the system. I chose a `VARCHAR(15)` for `Phone` to accommodate different formats and lengths for phone numbers, especially for international numbers. The `Address` field is set to `VARCHAR(255)`, allowing it to store full, descriptive addresses comfortably.

For the LIBRARY_BRANCH table, `Branch_Id` is the primary key, giving each branch a unique identifier. The `Branch_Name` field is `VARCHAR(100)`, which should provide enough space for descriptive branch names without taking up excessive storage. Similarly, `Branch_Address` uses `VARCHAR(255)` to allow complete addresses, ensuring sufficient space for all necessary details.

In the BORROWER table, `Card_No` is the primary key that uniquely identifies each borrower, but without using `AUTO_INCREMENT`. This means that the unique card number will need to be assigned manually, giving more control over the values. `Name` is set to `VARCHAR(100)`, while `Address` and `Phone` are `VARCHAR(255)` and `VARCHAR(15)`, respectively, offering flexibility for various formats and levels of detail in each field.

The BOOK table uses `Book_Id` as a primary key, making it easy to reference each book uniquely. I set `Title` as `VARCHAR(255)` to provide ample space for book titles, even longer ones. The `Publisher_Name` in this table references the `Publisher_Name` in the PUBLISHER table, which means each book is directly linked to a publisher. This connection keeps the database organized and prevents entries from unlisted publishers.

In the BOOK_AUTHORS table, both `Book_Id` and `Author_Name` make up the primary key together, which means we can record multiple authors for a single book without duplicate entries. `Book_Id` here is a foreign key that references the BOOK table, keeping each author entry linked back to a book. This setup is practical because books often have multiple authors, and the composite key ensures that each book-author pair is unique.

The BOOK_COPIES table tracks how many copies of each book are available in different branches. I used a composite primary key consisting of `Book_Id` and `Branch_Id`, so each book’s copies can be counted uniquely at each branch. `No_Of_Copies` is set as an integer to store the quantity of copies without extra formatting. Both `Book_Id` and `Branch_Id` are foreign keys that link back to the BOOK and LIBRARY_BRANCH tables, respectively, creating a clear relationship between books and their availability in specific branches.

Finally, the BOOK_LOANS table logs each loan transaction in the library. Here, `Book_Id`, `Branch_Id`, and `Card_No` make up the composite primary key, uniquely identifying each individual loan. This table also includes `Date_Out`, `Due_Date`, and `Returned_date` fields to track the borrowing, due, and return dates. Each of these fields uses the `DATE` type, making it easy to handle date-based queries. The foreign keys link back to the BOOK, LIBRARY_BRANCH, and BORROWER tables, making sure each loan record is tied to a specific book, branch, and borrower. This structure is helpful for managing book loans across branches and ensures consistency by preventing records that don't match existing books, branches, or borrowers.

Task 2:

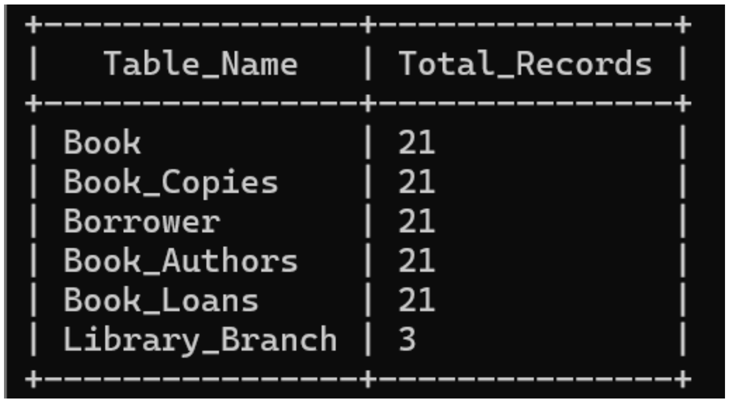
```
sqlite> .mode csv

sqlite> .import 'C:/Users/shahe/Downloads/LMSDataset/LMSDataset/Book.csv' Book

sqlite> .import 'C:/Users/shahe/Downloads/LMSDataset/LMSDataset/Book_Authors.csv'
Book_Authors

sqlite> .import 'C:/Users/shahe/Downloads/LMSDataset/LMSDataset/Book_Copies.csv' Book

_Copies

sqlite> .import 'C:/Users/shahe/Downloads/LMSDataset/LMSDataset/Book_Loans.csv' Book_

Loans

sqlite> .import 'C:/Users/shahe/Downloads/LMSDataset/LMSDataset/Borrower.csv' Borrowe

r

sqlite> .import 'C:/Users/shahe/Downloads/LMSDataset/LMSDataset/Library_Branch.csv'
Library_Branch

sqlite> .import 'C:/Users/shahe/Downloads/LMSDataset/LMSDataset/Publisher.csv' Publisher
```

Count Output

```
SELECT 'Book' AS Table_Name, COUNT(*) AS Total_Records FROM Book

UNION ALL

SELECT 'Book_Copies', COUNT(*) FROM Book_Copies

UNION ALL

SELECT 'Borrower', COUNT(*) FROM Borrower

UNION ALL

SELECT 'Book_Authors', COUNT(*) FROM Book_Authors

UNION ALL

SELECT 'Book_Loans', COUNT(*) FROM Book_Loans

UNION ALL

SELECT 'Library_Branch', COUNT(*) FROM Library_Branch;
```

Explanation of Commands:

- The .mode csv command ensures SQLite reads each file in CSV format.
- Each .import command loads a CSV file into its respective table.
- The SELECT query uses UNION ALL to list the total record count for each table, displaying them in a single output.



```
+----------------+---------------+
|   Table_Name   | Total_Records |
+----------------+---------------+
| Book           | 21            |
| Book_Copies    | 21            |
| Borrower       | 21            |
| Book_Authors   | 21            |
| Book_Loans     | 21            |
| Library_Branch | 3             |
+----------------+---------------+
```

Methodology

Preparation:

I used CSV files containing data for each table in the LMS database.

Each CSV file was prepared with column headers matching the table schema to ensure proper mapping during import.

Importing Data:

I used SQLite’s .import command in the SQLite Command Line to import data from each CSV file into the respective table. This allowed for efficient and direct insertion of data without needing to manually enter rows.

The .mode csv command set the mode to CSV, making it compatible with the CSV file format.

Verification:

After importing the data, I verified that each table was populated by running a count query on each table to check the number of records.

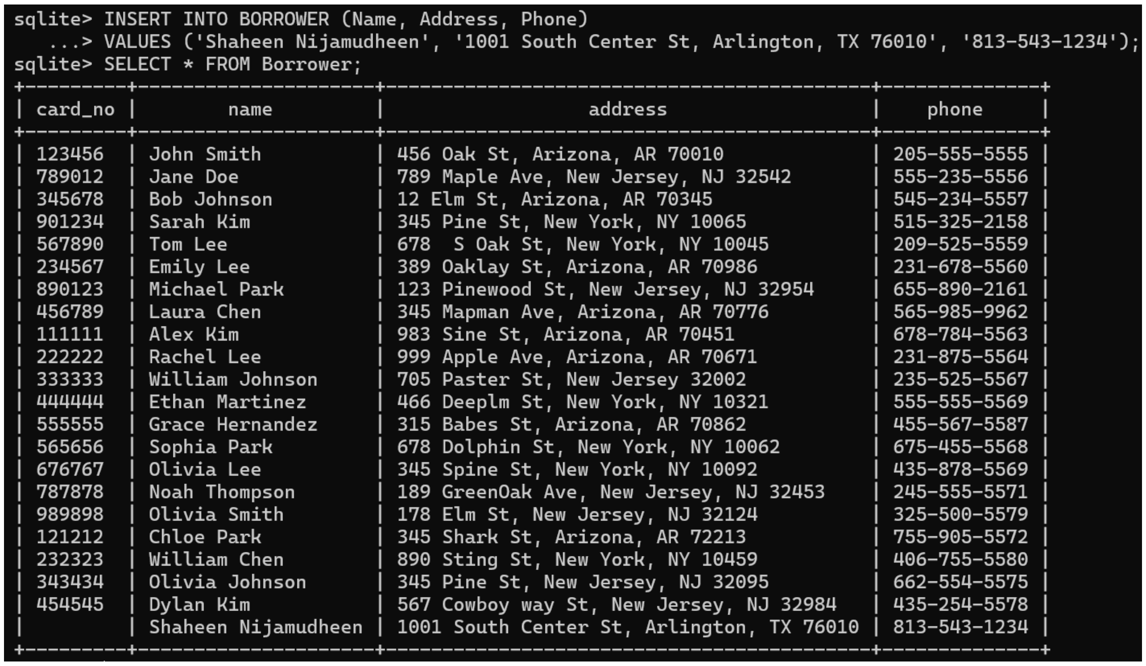
Challenges Encountered

- Data Format Consistency: Ensuring that each CSV file's data format matched the table schema was essential to avoid import errors. For instance, Book_Id should be an integer in both the CSV file and the database schema.
- Handling NULL Values: Some CSV files had missing values, which SQLite imported as NULL without issues. However, this required careful checking, especially in fields with constraints.

Task 3:

Question 1: Insert Yourself as a New Borrower

Description: Insert yourself as a new borrower. Don't provide the Card_No in the query.

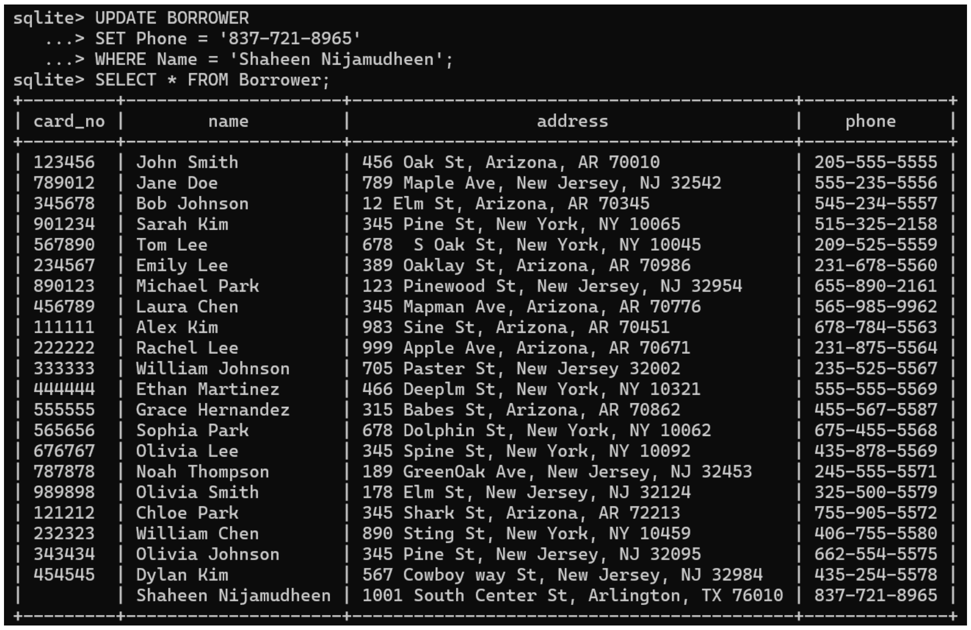


```
sqlite> INSERT INTO BORROWER (Name, Address, Phone)
   ...> VALUES ('Shaheen Nijamudheen', '1001 South Center St, Arlington, TX 76010', '813-543-1234');
sqlite> SELECT * FROM Borrower;
+---------+---------------------+-------------------------------------------+--------------+
| card_no |        name         |                  address                  |    phone     |
+---------+---------------------+-------------------------------------------+--------------+
| 123456  | John Smith          | 456 Oak St, Arizona, AR 70010             | 205-555-5555 |
| 789012  | Jane Doe            | 789 Maple Ave, New Jersey, NJ 32542       | 555-235-5556 |
| 345678  | Bob Johnson         | 12 Elm St, Arizona, AR 70345              | 545-234-5557 |
| 901234  | Sarah Kim           | 345 Pine St, New York, NY 10065           | 515-325-2158 |
| 567890  | Tom Lee             | 678  S Oak St, New York, NY 10045         | 209-525-5559 |
| 234567  | Emily Lee           | 389 Oaklay St, Arizona, AR 70986          | 231-678-5560 |
| 890123  | Michael Park        | 123 Pinewood St, New Jersey, NJ 32954     | 655-890-2161 |
| 456789  | Laura Chen          | 345 Mapman Ave, Arizona, AR 70776         | 565-985-9962 |
| 111111  | Alex Kim            | 983 Sine St, Arizona, AR 70451            | 678-784-5563 |
| 222222  | Rachel Lee          | 999 Apple Ave, Arizona, AR 70671          | 231-875-5564 |
| 333333  | William Johnson     | 705 Paster St, New Jersey 32002           | 235-525-5567 |
| 444444  | Ethan Martinez      | 466 Deeplm St, New York, NY 10321         | 555-555-5569 |
| 555555  | Grace Hernandez     | 315 Babes St, Arizona, AR 70862           | 455-567-5587 |
| 565656  | Sophia Park         | 678 Dolphin St, New York, NY 10062        | 675-455-5568 |
| 676767  | Olivia Lee          | 345 Spine St, New York, NY 10092          | 435-878-5569 |
| 787878  | Noah Thompson       | 189 GreenOak Ave, New Jersey, NJ 32453    | 245-555-5571 |
| 989898  | Olivia Smith        | 178 Elm St, New Jersey, NJ 32124          | 325-500-5579 |
| 121212  | Chloe Park          | 345 Shark St, Arizona, AR 72213           | 755-905-5572 |
| 232323  | William Chen        | 890 Sting St, New York, NY 10459          | 406-755-5580 |
| 343434  | Olivia Johnson      | 345 Pine St, New Jersey, NJ 32095         | 662-554-5575 |
| 454545  | Dylan Kim           | 567 Cowboy way St, New Jersey, NJ 32984   | 435-254-5578 |
|         | Shaheen Nijamudheen | 1001 South Center St, Arlington, TX 76010 | 813-543-1234 |
+---------+---------------------+-------------------------------------------+--------------+
```

Total Number of Records Affected: 1 new record inserted in BORROWER.

Question 2: Update Your Phone Number

Description: Update your phone number to (837) 721-8965 in the BORROWER table.



```
sqlite> UPDATE BORROWER
   ...> SET Phone = '837-721-8965'
   ...> WHERE Name = 'Shaheen Nijamudheen';
sqlite> SELECT * FROM Borrower;
+---------+---------------------+------------------------------------------+--------------+
| card_no |        name         |                 address                  |    phone     |
+---------+---------------------+------------------------------------------+--------------+
| 123456  | John Smith          | 456 Oak St, Arizona, AR 70010            | 205-555-5555 |
| 789012  | Jane Doe            | 789 Maple Ave, New Jersey, NJ 32542      | 555-235-5556 |
| 345678  | Bob Johnson         | 12 Elm St, Arizona, AR 70345             | 545-234-5557 |
| 901234  | Sarah Kim           | 345 Pine St, New York, NY 10065          | 515-325-2158 |
| 567890  | Tom Lee             | 678  S Oak St, New York, NY 10045        | 209-525-5559 |
| 234567  | Emily Lee           | 389 Oaklay St, Arizona, AR 70986         | 231-678-5560 |
| 890123  | Michael Park        | 123 Pinewood St, New Jersey, NJ 32954    | 655-890-2161 |
| 456789  | Laura Chen          | 345 Mapman Ave, Arizona, AR 70776        | 565-985-9962 |
| 111111  | Alex Kim            | 983 Sine St, Arizona, AR 70451           | 678-784-5563 |
| 222222  | Rachel Lee          | 999 Apple Ave, Arizona, AR 70671         | 231-875-5564 |
| 333333  | William Johnson     | 705 Paster St, New Jersey 32002          | 235-525-5567 |
| 444444  | Ethan Martinez      | 466 Deeplm St, New York, NY 10321        | 555-555-5569 |
| 555555  | Grace Hernandez     | 315 Babes St, Arizona, AR 70862          | 455-567-5587 |
| 565656  | Sophia Park         | 678 Dolphin St, New York, NY 10062       | 675-455-5568 |
| 676767  | Olivia Lee          | 345 Spine St, New York, NY 10092         | 435-878-5569 |
| 787878  | Noah Thompson       | 189 GreenOak Ave, New Jersey, NJ 32453   | 245-555-5571 |
| 989898  | Olivia Smith        | 178 Elm St, New Jersey, NJ 32124         | 325-500-5579 |
| 121212  | Chloe Park          | 345 Shark St, Arizona, AR 72213          | 755-905-5572 |
| 232323  | William Chen        | 890 Sting St, New York, NY 10459         | 406-755-5580 |
| 343434  | Olivia Johnson      | 345 Pine St, New Jersey, NJ 32095        | 662-554-5575 |
| 454545  | Dylan Kim           | 567 Cowboy way St, New Jersey, NJ 32984  | 435-254-5578 |
|         | Shaheen Nijamudheen | 1001 South Center St, Arlington, TX 76010 | 837-721-8965 |
+---------+---------------------+------------------------------------------+--------------+
```

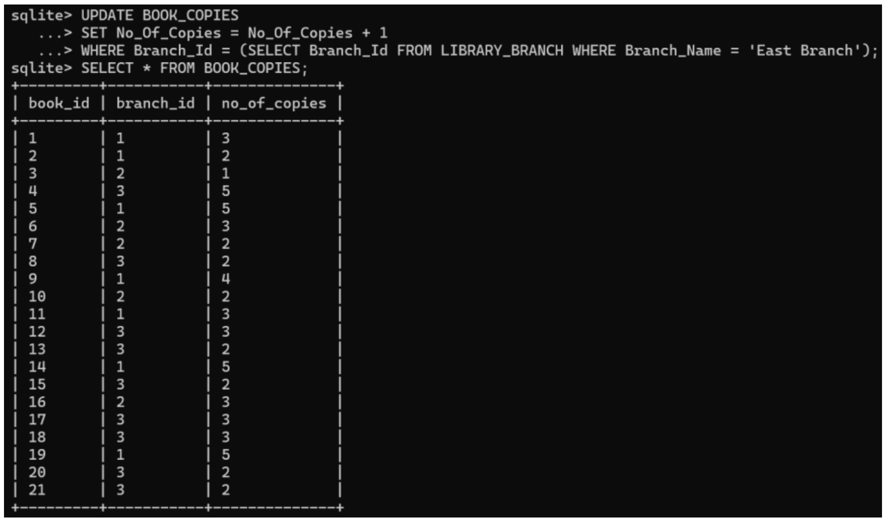
Challenges:

Ensure that the Name field uniquely identifies you in the table. If multiple borrowers have the same name, consider adding additional conditions to the WHERE clause (e.g., by checking Address as well) to avoid updating multiple rows by mistake.

Total Number of Records Affected: 1 record updated in BORROWER.

Question 3: Increase the Number of Book Copies by 1 for the 'East Branch'

Description: Increase the No_Of_Copies by 1 for all books located in the "East Branch."



```
sqlite> UPDATE BOOK_COPIES
   ...> SET No_Of_Copies = No_Of_Copies + 1
   ...> WHERE Branch_Id = (SELECT Branch_Id FROM LIBRARY_BRANCH WHERE Branch_Name = 'East Branch');
sqlite> SELECT * FROM BOOK_COPIES;
+---------+-----------+--------------+
| book_id | branch_id | no_of_copies |
+---------+-----------+--------------+
| 1       | 1         | 3            |
| 2       | 1         | 2            |
| 3       | 2         | 1            |
| 4       | 3         | 5            |
| 5       | 1         | 5            |
| 6       | 2         | 3            |
| 7       | 2         | 2            |
| 8       | 3         | 2            |
| 9       | 1         | 4            |
| 10      | 2         | 2            |
| 11      | 1         | 3            |
| 12      | 3         | 3            |
| 13      | 3         | 2            |
| 14      | 1         | 5            |
| 15      | 3         | 2            |
| 16      | 2         | 3            |
| 17      | 3         | 3            |
| 18      | 3         | 3            |
| 19      | 1         | 5            |
| 20      | 3         | 2            |
| 21      | 3         | 2            |
+---------+-----------+--------------+
```

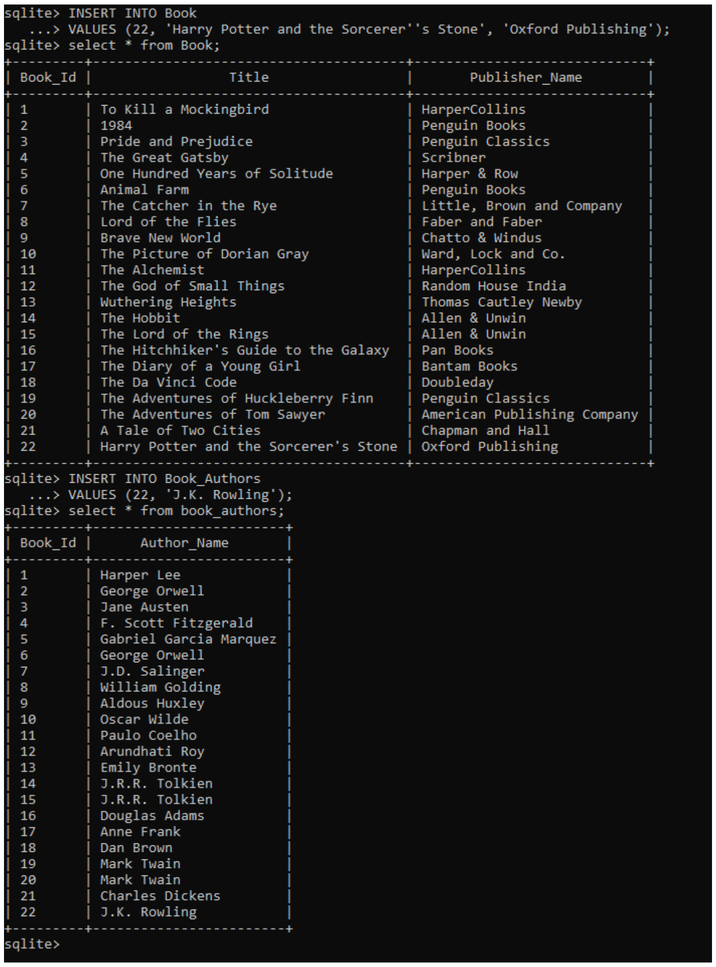
Challenges:

- Ensure that Branch_Name is correctly spelled and exists in LIBRARY_BRANCH.
- If there are multiple branches with the same name, adjust the WHERE clause to include additional identifiers like Branch_Address to prevent incorrect updates.

Total Number of Records Affected: 9 records updated in BOOK_COPIES.

Question 4-a: Insert a new BOOK with the following info: Title: ‘Harry Potter and the Sorcerer's Stone’ ; Book_author: ‘J.K. Rowling’ ; Publisher_name: ‘Oxford Publishing’

Description: Insert the entry for ‘Harry Potter and the Sorcerer's Stone’ as well as its author ‘J.K. Rowling’.



```
sqlite> INSERT INTO Book
   ...> VALUES (22, 'Harry Potter and the Sorcerer''s Stone', 'Oxford Publishing');
sqlite> select * from Book;
+---------+---------------------------------------+-----------------------------+
| Book_Id |                 Title                 |       Publisher_Name        |
+---------+---------------------------------------+-----------------------------+
| 1       | To Kill a Mockingbird                 | HarperCollins               |
| 2       | 1984                                  | Penguin Books               |
| 3       | Pride and Prejudice                   | Penguin Classics            |
| 4       | The Great Gatsby                      | Scribner                    |
| 5       | One Hundred Years of Solitude         | Harper & Row                |
| 6       | Animal Farm                           | Penguin Books               |
| 7       | The Catcher in the Rye                | Little, Brown and Company   |
| 8       | Lord of the Flies                     | Faber and Faber             |
| 9       | Brave New World                       | Chatto & Windus             |
| 10      | The Picture of Dorian Gray            | Ward, Lock and Co.          |
| 11      | The Alchemist                         | HarperCollins               |
| 12      | The God of Small Things               | Random House India          |
| 13      | Wuthering Heights                     | Thomas Cautley Newby        |
| 14      | The Hobbit                            | Allen & Unwin               |
| 15      | The Lord of the Rings                 | Allen & Unwin               |
| 16      | The Hitchhiker's Guide to the Galaxy  | Pan Books                   |
| 17      | The Diary of a Young Girl             | Bantam Books                |
| 18      | The Da Vinci Code                     | Doubleday                   |
| 19      | The Adventures of Huckleberry Finn    | Penguin Classics            |
| 20      | The Adventures of Tom Sawyer          | American Publishing Company |
| 21      | A Tale of Two Cities                  | Chapman and Hall            |
| 22      | Harry Potter and the Sorcerer's Stone | Oxford Publishing           |
+---------+---------------------------------------+-----------------------------+
sqlite> INSERT INTO Book_Authors
   ...> VALUES (22, 'J.K. Rowling');
sqlite> select * from book_authors;
+---------+------------------------+
| Book_Id |      Author_Name       |
+---------+------------------------+
| 1       | Harper Lee             |
| 2       | George Orwell          |
| 3       | Jane Austen            |
| 4       | F. Scott Fitzgerald    |
| 5       | Gabriel Garcia Marquez |
| 6       | George Orwell          |
| 7       | J.D. Salinger          |
| 8       | William Golding        |
| 9       | Aldous Huxley          |
| 10      | Oscar Wilde            |
| 11      | Paulo Coelho           |
| 12      | Arundhati Roy          |
| 13      | Emily Bronte           |
| 14      | J.R.R. Tolkien         |
| 15      | J.R.R. Tolkien         |
| 16      | Douglas Adams          |
| 17      | Anne Frank             |
| 18      | Dan Brown              |
| 19      | Mark Twain             |
| 20      | Mark Twain             |
| 21      | Charles Dickens        |
| 22      | J.K. Rowling           |
+---------+------------------------+
sqlite>
```

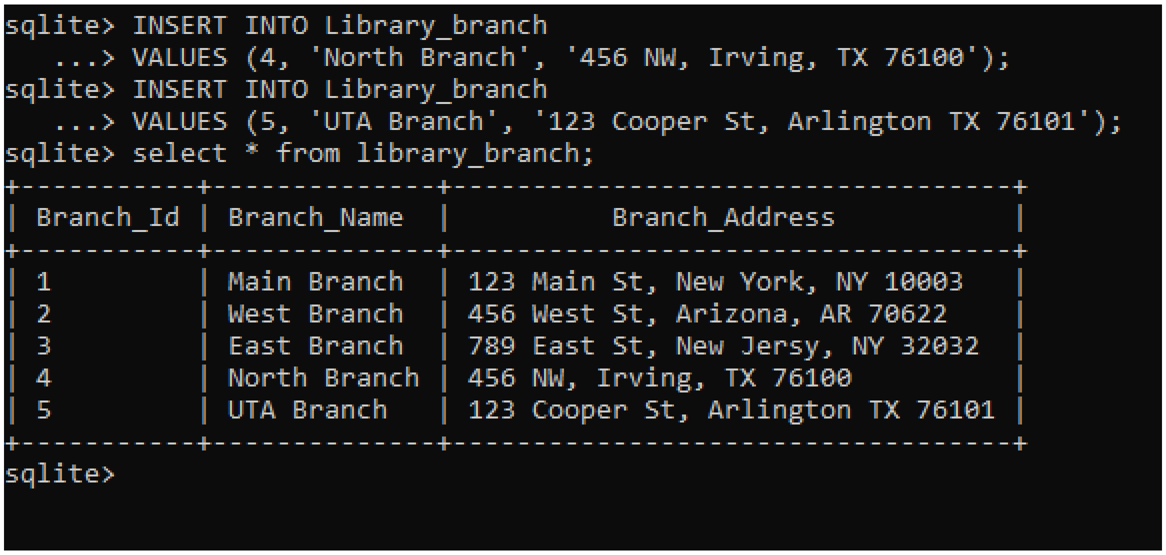
Records affected: 1 new entry in BOOK and BOOK_AUTHORS

Challenges:

- Having the apostrophe in the book’s title. Circumvented by adding two apostrophes where it should go.

Question 4-b: You also need to insert the following branches:
North Branch | 456 NW, Irving, TX 76100
UTA Branch | 123 Cooper St, Arlington TX 76101

Description: Insert two branches to library branches, the North Branch and UTA Branch.

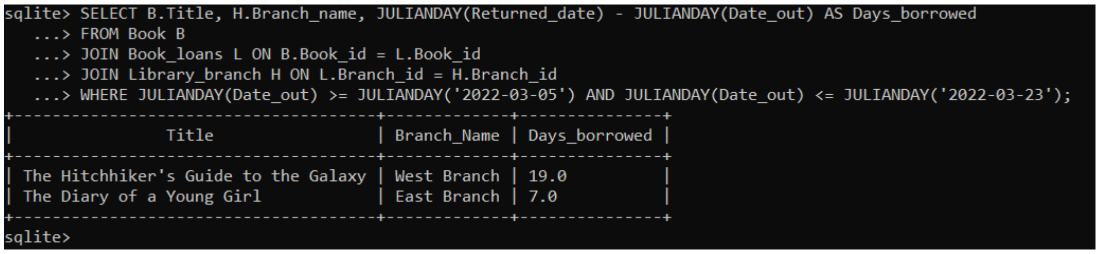


```
sqlite> INSERT INTO Library_branch
   ...> VALUES (4, 'North Branch', '456 NW, Irving, TX 76100');
sqlite> INSERT INTO Library_branch
   ...> VALUES (5, 'UTA Branch', '123 Cooper St, Arlington TX 76101');
sqlite> select * from library_branch;
+-----------+--------------+----------------------------------+
| Branch_Id | Branch_Name  |          Branch_Address          |
+-----------+--------------+----------------------------------+
| 1         | Main Branch  | 123 Main St, New York, NY 10003  |
| 2         | West Branch  | 456 West St, Arizona, AR 70622   |
| 3         | East Branch  | 789 East St, New Jersy, NY 32032 |
| 4         | North Branch | 456 NW, Irving, TX 76100         |
| 5         | UTA Branch   | 123 Cooper St, Arlington TX 76101 |
+-----------+--------------+----------------------------------+
sqlite>
```

Records affected: 2 new entries in LIBRARY_BRANCH

Question 5: Return all Books that were loaned between March 5, 2022 until March 23, 2022. List Book title and Branch name, and how many days it was borrowed for.

Description: Show loaned books from March 5 – 23, 2022, branch it came from, and the days borrowed

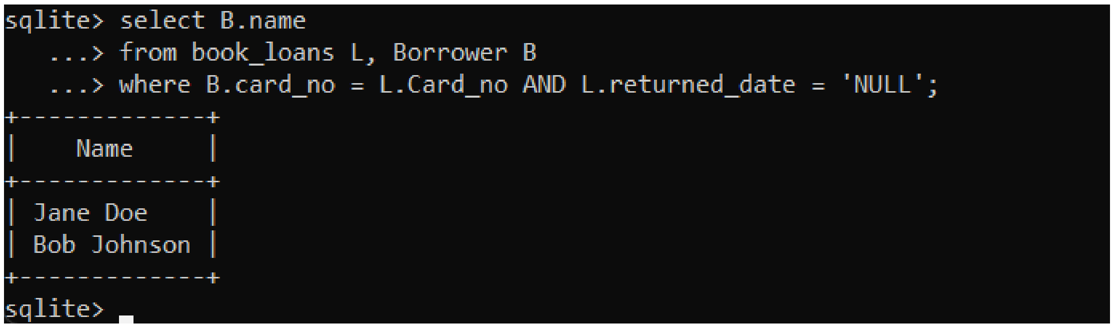


```
sqlite> SELECT B.Title, H.Branch_name, JULIANDAY(Returned_date) - JULIANDAY(Date_out) AS Days_borrowed
   ...> FROM Book B
   ...> JOIN Book_loans L ON B.Book_id = L.Book_id
   ...> JOIN Library_branch H ON L.Branch_id = H.Branch_id
   ...> WHERE JULIANDAY(Date_out) >= JULIANDAY('2022-03-05') AND JULIANDAY(Date_out) <= JULIANDAY('2022-03-23');
+-------------------------------------+-------------+---------------+
|                Title                | Branch_Name | Days_borrowed |
+-------------------------------------+-------------+---------------+
| The Hitchhiker's Guide to the Galaxy | West Branch | 19.0          |
| The Diary of a Young Girl           | East Branch | 7.0           |
+-------------------------------------+-------------+---------------+
sqlite>
```

Records returned: 2

Question 6: Return a List borrower names, that have books not returned.

Description: Find the names of borrowers who didn't return their books.

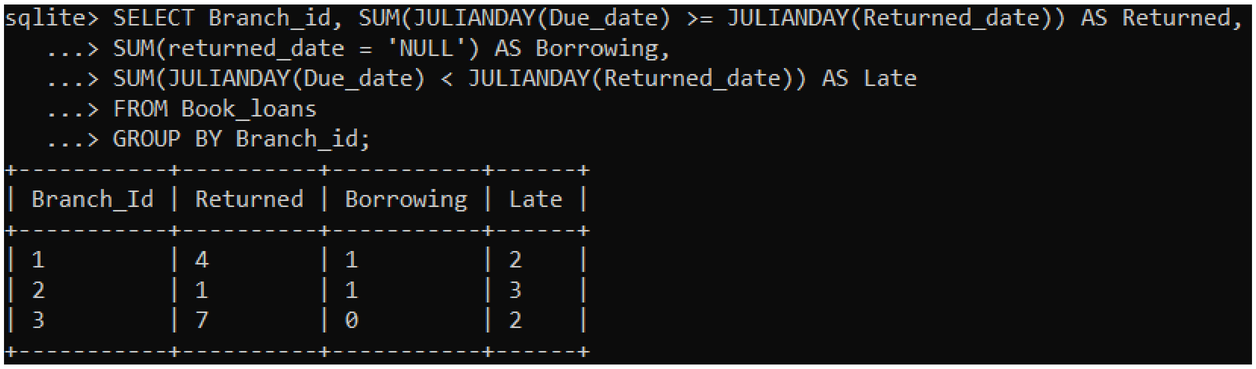


```
sqlite> select B.name
   ...> from book_loans L, Borrower B
   ...> where B.card_no = L.Card_no AND L.returned_date = 'NULL';
+-------------+
|    Name     |
+-------------+
| Jane Doe    |
| Bob Johnson |
+-------------+
sqlite>
```

Records returned: 2

Question 7: Create a report that will return all branches with the number of books borrowed per branch separated by if they have been returned, still borrowed, or late.

Description: Show all the branches and sort books by whether they are borrowed, returned, or late



```
sqlite> SELECT Branch_id, SUM(JULIANDAY(Due_date) >= JULIANDAY(Returned_date)) AS Returned,
   ...> SUM(returned_date = 'NULL') AS Borrowing,
   ...> SUM(JULIANDAY(Due_date) < JULIANDAY(Returned_date)) AS Late
   ...> FROM Book_loans
   ...> GROUP BY Branch_id;
+-----------+----------+-----------+------+
| Branch_Id | Returned | Borrowing | Late |
+-----------+----------+-----------+------+
| 1         | 4        | 1         | 2    |
| 2         | 1        | 1         | 3    |
| 3         | 7        | 0         | 2    |
+-----------+----------+-----------+------+
```

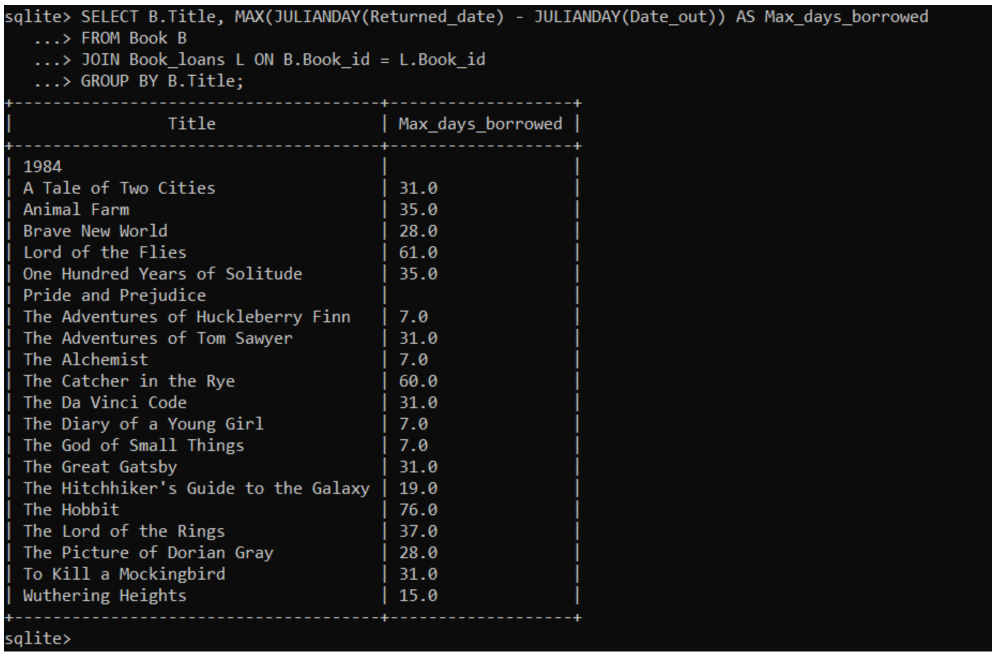
Records returned: 3

Challenges:

- Getting the correct entry for borrowed, returned and late books

Question 8: List all the books (title) and the maximum number of days that they were borrowed.

Description: Show each book and its highest number of days it was borrowed

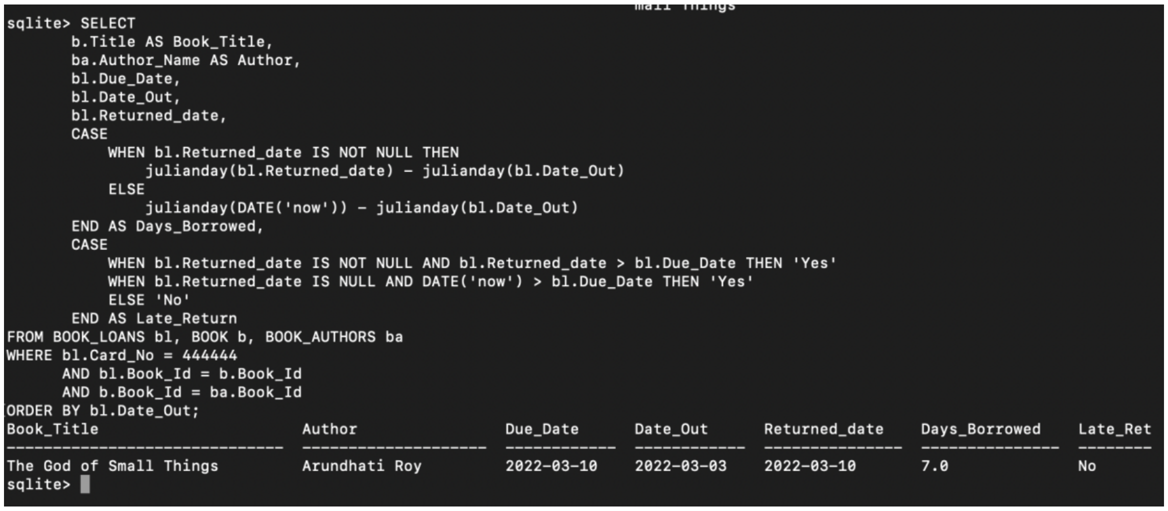


```
sqlite> SELECT B.Title, MAX(JULIANDAY(Returned_date) - JULIANDAY(Date_out)) AS Max_days_borrowed
   ...> FROM Book B
   ...> JOIN Book_loans L ON B.Book_id = L.Book_id
   ...> GROUP BY B.Title;
+------------------------------------+------------------+
|               Title                | Max_days_borrowed |
+------------------------------------+------------------+
| 1984                               |                  |
| A Tale of Two Cities               | 31.0             |
| Animal Farm                        | 35.0             |
| Brave New World                    | 28.0             |
| Lord of the Flies                  | 61.0             |
| One Hundred Years of Solitude      | 35.0             |
| Pride and Prejudice                |                  |
| The Adventures of Huckleberry Finn | 7.0              |
| The Adventures of Tom Sawyer       | 31.0             |
| The Alchemist                      | 7.0              |
| The Catcher in the Rye             | 60.0             |
| The Da Vinci Code                  | 31.0             |
| The Diary of a Young Girl          | 7.0              |
| The God of Small Things            | 7.0              |
| The Great Gatsby                   | 31.0             |
| The Hitchhiker's Guide to the Galaxy | 19.0           |
| The Hobbit                         | 76.0             |
| The Lord of the Rings              | 37.0             |
| The Picture of Dorian Gray         | 28.0             |
| To Kill a Mockingbird              | 31.0             |
| Wuthering Heights                  | 15.0             |
+------------------------------------+------------------+
sqlite>
```

Records returned: 21

Question 9: Create a report for Ethan Martinez with all the books they borrowed. List the book title and author. Also, calculate the number of days each book was borrowed for and if any book is late being returned. Order the results by the date_out.

Description: Show the books borrowed, its author, the number of days borrowed, and whether or not the book was returned on time for Ethan Martinez. Ordered by the date loaned



```
                                                                   mall Things
sqlite> SELECT
       b.Title AS Book_Title,
       ba.Author_Name AS Author,
       bl.Due_Date,
       bl.Date_Out,
       bl.Returned_date,
       CASE
           WHEN bl.Returned_date IS NOT NULL THEN
               julianday(bl.Returned_date) - julianday(bl.Date_Out)
           ELSE
               julianday(DATE('now')) - julianday(bl.Date_Out)
       END AS Days_Borrowed,
       CASE
           WHEN bl.Returned_date IS NOT NULL AND bl.Returned_date > bl.Due_Date THEN 'Yes'
           WHEN bl.Returned_date IS NULL AND DATE('now') > bl.Due_Date THEN 'Yes'
           ELSE 'No'
       END AS Late_Return
FROM BOOK_LOANS bl, BOOK b, BOOK_AUTHORS ba
WHERE bl.Card_No = 444444
      AND bl.Book_Id = b.Book_Id
      AND b.Book_Id = ba.Book_Id
ORDER BY bl.Date_Out;
Book_Title                      Author               Due_Date      Date_Out      Returned_date    Days_Borrowed    Late_Ret
------------------------------  -------------------  ------------  ------------  ---------------  ---------------  --------
The God of Small Things         Arundhati Roy        2022-03-10    2022-03-03    2022-03-10       7.0              No
sqlite> ▌
```

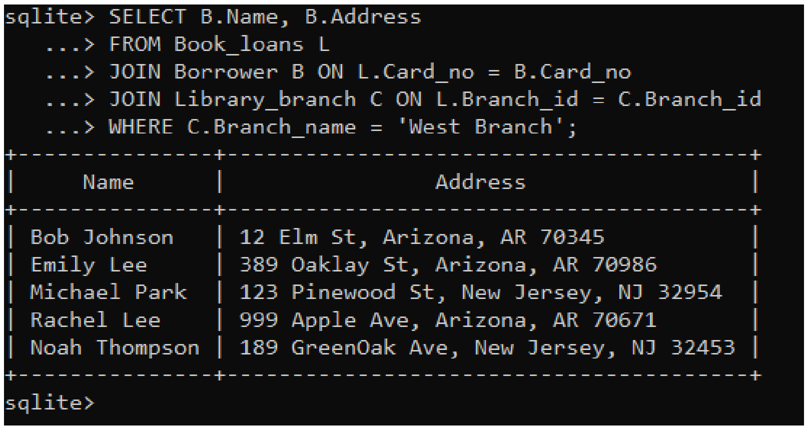
Records returned: 1

Challenges:

- Ensure that the person targeted is Ethan Martinez.

Question 10: Return the names of all borrowers that borrowed a book from the West Branch include their addresses.

Description: Show the name and addresses of people who borrowed from West Branch



```
sqlite> SELECT B.Name, B.Address
   ...> FROM Book_loans L
   ...> JOIN Borrower B ON L.Card_no = B.Card_no
   ...> JOIN Library_branch C ON L.Branch_id = C.Branch_id
   ...> WHERE C.Branch_name = 'West Branch';
+---------------+--------------------------------------+
|     Name      |               Address                |
+---------------+--------------------------------------+
| Bob Johnson   | 12 Elm St, Arizona, AR 70345         |
| Emily Lee     | 389 Oaklay St, Arizona, AR 70986     |
| Michael Park  | 123 Pinewood St, New Jersey, NJ 32954 |
| Rachel Lee    | 999 Apple Ave, Arizona, AR 70671     |
| Noah Thompson | 189 GreenOak Ave, New Jersey, NJ 32453 |
+---------------+--------------------------------------+
sqlite>
```

Records returned: 5

CONTRIBUTION LIST

Shaheen Nijamudheen – Task 1, Task 2, Task 3: Q1, Q2, Q3
James Paul Nguyen - Task 3: Q4-Q10
Ali Shirazi-Nejad – Task3: Checked Q1-10, Task3: Q9

HONOR CODE

We pledge, on our honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence.

We promise that we will submit only work that we personally created or that we contribute to group collaborations, and we will appropriately reference any work from other sources. We will follow the highest standards of integrity and uphold the spirit of the Honor Code.