CSE 4701

Project 1, Part 2

Mike Medved

October 4th, 2023

Table of Contents

Part 1, Modifying the Publisher Table

Last Assignment State

Here is the state as it was left from the previous assignment:

```
MariaDB [Book_Loan_DB]> DESCRIBE Publisher;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| Name    | varchar(255) | NO   | PRI | NULL    |       |
| Address | varchar(255) | NO   | PRI | NULL    |       |
| Phone   | varchar(14)  | NO   |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
3 rows in set (0.004 sec)
```

Figure 1: Last Assignment State

Adding the New Column

We can use the *ALTER* command to add another column to the *Publisher* table. The command is as follows:

```
01 |  ALTER TABLE Publisher ADD COLUMN City VARCHAR(255) NOT NULL;
```

As a result, we get the following table schema after executing the above command:

```
MariaDB [Book_Loan_DB]> DESCRIBE Publisher;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| Name    | varchar(255) | NO   | PRI | NULL    |       |
| Address | varchar(255) | NO   | PRI | NULL    |       |
| Phone   | varchar(14)  | NO   |     | NULL    |       |
| City    | varchar(255) | NO   |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
4 rows in set (0.012 sec)
```

Figure 2: Table Schema After Adding New Column

Additionally, this is the result we get when we execute a *SELECT* all operation against the table:

```
MariaDB [Book_Loan_DB]> SELECT * FROM Publisher;
+----------------------+-----------------------+----------------+------+
| Name                 | Address               | Phone          | City |
+----------------------+-----------------------+----------------+------+
| Picador              | 4300 Turkey Pen Road  | (223) 444-5521 |      |
| Simon & Schuster     | 4096 Emerson Road     | (421) 540-1675 |      |
| Tom Doherty Assoc Llc | 4321 Walnut Hill Drive | (345) 623-0356 |      |
+----------------------+-----------------------+----------------+------+
3 rows in set (0.005 sec)
```

Figure 3: Select Operation Result

Here, we can see that the new column is added to the table, and all of the values are blank, not *NULL*.

Part 2, Emptying the Tables

Deleting the Records

We can delete all records from the tables in our database by using the *DELETE FROM [table]* syntax as seen below:

```
DELETE FROM Book_Authors;
DELETE FROM Book_Copies;
DELETE FROM Book_Loans;
DELETE FROM Book;
DELETE FROM Borrower;
DELETE FROM Library_Branch;
DELETE FROM Publisher;
```

Do note, we need to delete records from tables in a specific order as to not violate foreign key constraints.

```
MariaDB [Book_Loan_DB]> DELETE FROM Book_Authors;
Query OK, 3 rows affected (0.013 sec)

MariaDB [Book_Loan_DB]> DELETE FROM Book_Copies;
Query OK, 3 rows affected (0.005 sec)

MariaDB [Book_Loan_DB]> DELETE FROM Book_Loans;
Query OK, 3 rows affected (0.007 sec)

MariaDB [Book_Loan_DB]> DELETE FROM Book;
Query OK, 3 rows affected (0.018 sec)

MariaDB [Book_Loan_DB]> DELETE FROM Borrower;
Query OK, 3 rows affected (0.014 sec)

MariaDB [Book_Loan_DB]> DELETE FROM Library_Branch;
Query OK, 3 rows affected (0.006 sec)

MariaDB [Book_Loan_DB]> DELETE FROM Publisher;
Query OK, 3 rows affected (0.005 sec)
```

Figure 4: Table Schema After Adding New Column

Verifying Deletion

Once all of the *DELETE* commands are executed, we can use *SELECT* to verify all data has been removed.

```
MariaDB [Book_Loan_DB]> SELECT * FROM Book;
Empty set (0.003 sec)

MariaDB [Book_Loan_DB]> SELECT * FROM Book_Authors;
Empty set (0.003 sec)

MariaDB [Book_Loan_DB]> SELECT * FROM Book_Copies;
Empty set (0.003 sec)

MariaDB [Book_Loan_DB]> SELECT * FROM Book_Loans;
Empty set (0.003 sec)

MariaDB [Book_Loan_DB]> SELECT * FROM Borrower;
Empty set (0.007 sec)

MariaDB [Book_Loan_DB]> SELECT * FROM Library_Branch;
Empty set (0.006 sec)

MariaDB [Book_Loan_DB]> SELECT * FROM Publisher;
Empty set (0.003 sec)
```

Figure 5: Select Operation Result

Part 3, Importing CSV-Formatted Data

Importing the Data

We can use this command template to import all of the obtained CSV files into their respective tables.

```
LOAD DATA LOCAL INFILE
    ['/path/to/csv']
INTO TABLE
    [table]
FIELDS TERMINATED BY ',' ENCLOSED BY '"'
LINES TERMINATED BY '\r\n';
```

Resultant Tuple Counts

After importing all of the data from the provided CSV files, we can use the *SELECT* command to count the amount of rows in each table. The command template, and totals are shown below:

```
SELECT COUNT(*) FROM [table];
```

```
MariaDB [Book_Loan_DB]> SELECT COUNT(*) AS "Rows" FROM Book;
+------+
| Rows |
+------+
|    5 |
+------+
1 row in set (0.015 sec)

MariaDB [Book_Loan_DB]> SELECT COUNT(*) AS "Rows" FROM Book_Authors;
+------+
| Rows |
+------+
|    7 |
+------+
1 row in set (0.003 sec)

MariaDB [Book_Loan_DB]> SELECT COUNT(*) AS "Rows" FROM Book_Copies;
+------+
| Rows |
+------+
|    8 |
+------+
1 row in set (0.006 sec)

MariaDB [Book_Loan_DB]> SELECT COUNT(*) AS "Rows" FROM Book_Loans;
+------+
| Rows |
+------+
|    8 |
+------+
1 row in set (0.004 sec)

MariaDB [Book_Loan_DB]> SELECT COUNT(*) AS "Rows" FROM Borrower;
+------+
| Rows |
+------+
|    5 |
+------+
1 row in set (0.004 sec)

MariaDB [Book_Loan_DB]> SELECT COUNT(*) AS "Rows" FROM Library_Branch;
+------+
| Rows |
+------+
|    5 |
+------+
1 row in set (0.004 sec)

MariaDB [Book_Loan_DB]> SELECT COUNT(*) AS "Rows" FROM Publisher;
+------+
| Rows |
+------+
|    4 |
+------+
1 row in set (0.003 sec)
```

Figure 6: Tuple Counts

Part 4, Complex Query Exercises

a) Copies of a book at one branch

How many copies of the book 'The Lost Tribe' are owned by branch 'Sharpstown'?

```
01 |  SELECT No_of_copies FROM Book_Copies WHERE Book_id="B1" AND Branch_id="BR1";
```

```
+--------------+
| No_of_copies |
+--------------+
|           15 |
+--------------+
1 row in set (0.002 sec)
```

Figure 7: Number of Copies of 'The Lost Tribe' at 'Sharpstown'

b) Copies of a book at all branches

How many copies of the book 'The Lost Tribe' are owned by each branch?

```
01 |  SELECT Branch_id, No_of_copies FROM Book_Copies WHERE Book_id = "B1";
```

```
+-----------+--------------+
| Branch_id | No_of_copies |
+-----------+--------------+
| BR1       |           15 |
| BR2       |           25 |
+-----------+--------------+
2 rows in set (0.005 sec)
```

Figure 8: Number of Copies of 'The Lost Tribe' at All Branches

c) Borrowers with no books checked out

Retrieve the names of all borrowers who do not have any books checked out.

```
01 |  SELECT
02 |      b.Card_no,
03 |      b.Name
04 |  FROM Borrower b
05 |  LEFT JOIN Book_Loans l
06 |      ON b.Card_no = l.Card_no
07 |  WHERE l.Card_no IS NULL;
```

```
+---------+------------+
| Card_no | Name       |
+---------+------------+
| C5      | James Borg |
+---------+------------+
1 row in set (0.004 sec)
```

Figure 9: Borrowers with No Books Checked Out

d) Information about all books loaned on 1/3/2023

Assume today is 1/3/2023. For each book that is loaned out from the Sharpstown branch, retrieve the book title, the borrower's name, and the borrower's address.

```
01 |  SELECT
02 |      b.Title,
03 |      u.Name,
04 |      u.Address
05 |  FROM Book_Loans l
06 |  INNER JOIN Book b
07 |      ON b.Book_id = l.Book_id
08 |  INNER JOIN Borrower u
09 |      ON l.Card_no = u.Card_no
10 |  WHERE Date_out < '2023-01-03'
11 |      AND Due_date > '2023-01-03'
12 |      AND Branch_id = "BR1";
```

```
+----------------+--------------+--------------------------+
| Title          | Name         | Address                  |
+----------------+--------------+--------------------------+
| The Lost Tribe | John Smith   | 731 Fondren, Houoston, TX |
| It             | Franklin Wong | 638 Voss, Houston, TX    |
+----------------+--------------+--------------------------+
2 rows in set (0.003 sec)
```

Figure 10: Books Loaned on 1/3/2023

e) For each branch get the number of books loaned

For each branch, retrieve the branch name and the total number of books loaned out from that branch.

```
01 |  SELECT
02 |      l.Branch_id,
03 |      b.Branch_name,
04 |      COUNT(l.Book_id) AS "Books Loaned"
05 |  FROM Book_Loans l
06 |  INNER JOIN Library_Branch b
07 |      ON l.Branch_id = b.Branch_id
08 |  GROUP BY l.Branch_id;
```

```
+-----------+-------------------+--------------+
| Branch_id | Branch_name       | Books Loaned |
+-----------+-------------------+--------------+
| BR1       | Sharpstown        |            2 |
| BR3       | Drive Lake Charles |            2 |
| BR4       | Mansfield Center  |            1 |
| BR5       | Storrs            |            3 |
+-----------+-------------------+--------------+
4 rows in set (0.005 sec)
```

Figure 11: Books Loaned by Branch

f) Retrieve info about borrowers with more than two books loaned

Retrieve the names, addresses, and number of books checked out for all borrowers who have more than two books checked out.

```
SELECT
    b.Name,
    b.Address,
    COUNT(l.Book_id) AS "Books Out"
FROM Book_Loans l
INNER JOIN Borrower b
    ON l.Card_no = b.Card_no
GROUP BY l.Card_no
HAVING COUNT(*) > 2;
```

```
+----------------+-------------------------+-----------+
| Name           | Address                 | Books Out |
+----------------+-------------------------+-----------+
| Franklin Wong  | 638 Voss, Houston, TX   |         3 |
| Ramesh Narayan | 975 Fire Oak, Humble, TX |         3 |
+----------------+-------------------------+-----------+
2 rows in set (0.003 sec)
```

Figure 12: Borrowers with More than Two Books Loaned

g) Retrieve info about books (co)authored by Stephen King at the Central branch

For each book authored (or coauthored) by Stephen King, retrieve the title and the number of copies owned by the 'Central' branch.

```
SELECT
    b.Title,
    c.No_of_copies
FROM Book_Authors a
INNER JOIN Book b ON
    a.Book_id = b.Book_id
INNER JOIN Book_Copies c ON
    b.Book_id = c.Book_id
WHERE a.Author_name = "Stephen King" AND c.Branch_id = "BR2";
```

```
+-------+--------------+
| Title | No_of_copies |
+-------+--------------+
| It    |           10 |
+-------+--------------+
1 row in set (0.006 sec)
```

Figure 13: Books (Co)Authored by Stephen King at the Central Branch

h) Find books that cannot be loaned on 2/2/2023

Assume today is 2/2/2023. Find book(s) that cannot be loaned because all copies in the library branch have been completely loaned out. Show book title and branch name.

```sql
SELECT
    b.Title,
    p.Branch_name,
    c.No_of_copies
FROM Book_Copies c
INNER JOIN Book b
    ON c.Book_id = b.Book_id
INNER JOIN Book_Loans l
    ON c.Book_id = l.Book_id
        AND c.Branch_id = l.Branch_id
INNER JOIN Library_Branch p
    ON c.Branch_id = p.Branch_id
GROUP BY c.Branch_id
HAVING COUNT(l.Book_id) >= c.No_of_copies;
```

```
+---------------+--------------------+--------------+
| Title         | Branch_name        | No_of_copies |
+---------------+--------------------+--------------+
| Event Horizon | Drive Lake Charles |            2 |
+---------------+--------------------+--------------+
1 row in set (0.004 sec)
```

Figure 14: Books that Cannot be Loaned on 2/2/2023

i) Retrieve info about the borrower who loaned all the books by Henry Kissinger

Find the name and address of the borrower who loaned all the books authored by Henry A Kissinger.

```sql
SELECT
    u.Name,
    u.Address
FROM Book_Loans l
INNER JOIN Book b
    ON l.Book_id = b.Book_id
INNER JOIN Borrower u
    ON l.Card_no = u.Card_no
INNER JOIN Book_Authors a
    ON b.Book_id = a.Book_id
WHERE a.Author_name = "Henry A Kissinger"
GROUP BY u.Name
HAVING COUNT(l.Book_id) = (
    SELECT
        COUNT(Book_id)
    FROM
        Book_Authors
    WHERE Author_name = "Henry A Kissinger"
);
```

```
+---------------+--------------------------+
| Name          | Address                  |
+---------------+--------------------------+
| Ramesh Narayan | 975 Fire Oak, Humble, TX |
+---------------+--------------------------+
1 row in set (0.005 sec)
```

Figure 15: Borrower who Loaned all Books by Henry Kissinger

Part 5, Setting Up Constraints

Due Date Validation

We are able to add a constraint to validate the *Due_date* column in the *Book_Loans* table such that a due date cannot be set to occur before the recorded *Date_out* date.

In order to add it, we can execute the following SQL command to add the constraint to the table:

```sql
ALTER TABLE Book_Loans ADD CONSTRAINT chk_due_date CHECK (Due_date >= Date_out);
```

After adding the constraint, we can verify it was indeed successfully added by using the *SHOW* command to interrogate the table creation command from the system schema. In the below output, we can see on the second to last line of the command our new constraint *chk_due_date* was added.

```
MariaDB [Book_Loan_DB]> SHOW CREATE TABLE Book_Loans;
+------------+------------------------------------------------------------------------------------------
| Table      | Create Table
+------------+------------------------------------------------------------------------------------------
| Book_Loans | CREATE TABLE `Book_Loans` (
  `Book_id` varchar(2) NOT NULL,
  `Branch_id` varchar(3) NOT NULL,
  `Card_no` varchar(2) NOT NULL,
  `Date_out` date NOT NULL,
  `Due_date` date NOT NULL,
  PRIMARY KEY (`Book_id`,`Branch_id`,`Card_no`),
  KEY `Branch_id` (`Branch_id`),
  KEY `Card_no` (`Card_no`),
  CONSTRAINT `Book_Loans_ibfk_1` FOREIGN KEY (`Book_id`) REFERENCES `Book` (`Book_id`),
  CONSTRAINT `Book_Loans_ibfk_2` FOREIGN KEY (`Branch_id`) REFERENCES `Library_Branch` (`Branch_id`),
  CONSTRAINT `Book_Loans_ibfk_3` FOREIGN KEY (`Card_no`) REFERENCES `Borrower` (`Card_no`),
  CONSTRAINT `chk_due_date` CHECK (`Due_date` ≥ `Date_out`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci |
+------------+------------------------------------------------------------------------------------------
1 row in set (0.004 sec)
```

Figure 16: Constraint Verification

Now that the constraint is indeed applied to the table, we can try inserting a record with an invalid due date and ensure it fails.

```
MariaDB [Book_Loan_DB]> INSERT INTO Book_Loans VALUES ("B5", "BR1", "C1", '2023-01-01', '2022-12-25');
ERROR 4025 (23000): CONSTRAINT `chk_due_date` failed for `Book_Loan_DB`.`Book_Loans`
```

Figure 17: Constraint Failure on Invalid Due Date