SQL PROJECT ON LIBRARY MANAGEMENT SYSTEM

"ORGANIZING LIBRARY RESOURCES WITH RELATIONAL DATABASES"

PROJECT OVERVIEW:

This library management system project uses SQL (MySQL) , a relational database management system to manage a library's operations through various interconnected tables.

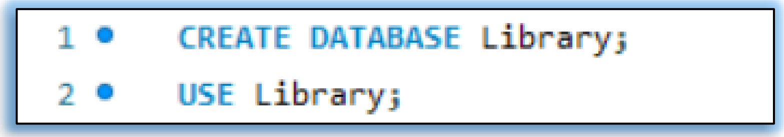
OPERATION PERFORMED:

Key operations include : Create, Read, Update, and Delete(CRUD).
SQL queries facilitate these operations:

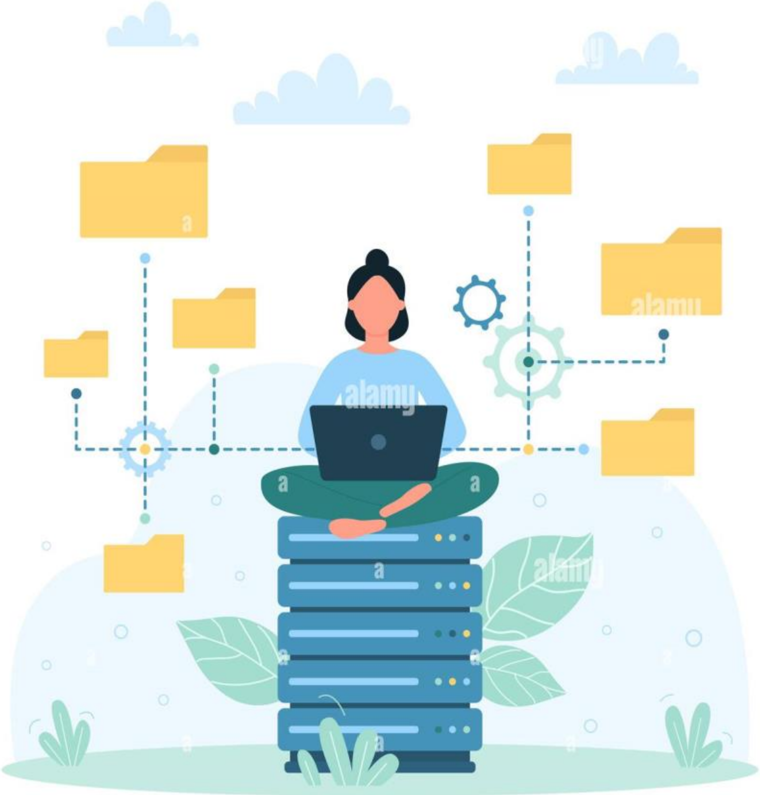
- Create: Add new books, borrowers, transactions, and reservations.
- Read: Retrieve book details, member information, borrowing history, and reservation status.
- Update: Modify book availability, borrower details, transaction records, and reservation status.
- Delete: Remove outdated or incorrect entries.

SQL queries like INSERT, SELECT, UPDATE, and DELETE manage and manipulate the library's data efficiently.

DATABASE NAME : Library



```sql
CREATE DATABASE Library;
USE Library;
```



DATABASE SCHEMA:

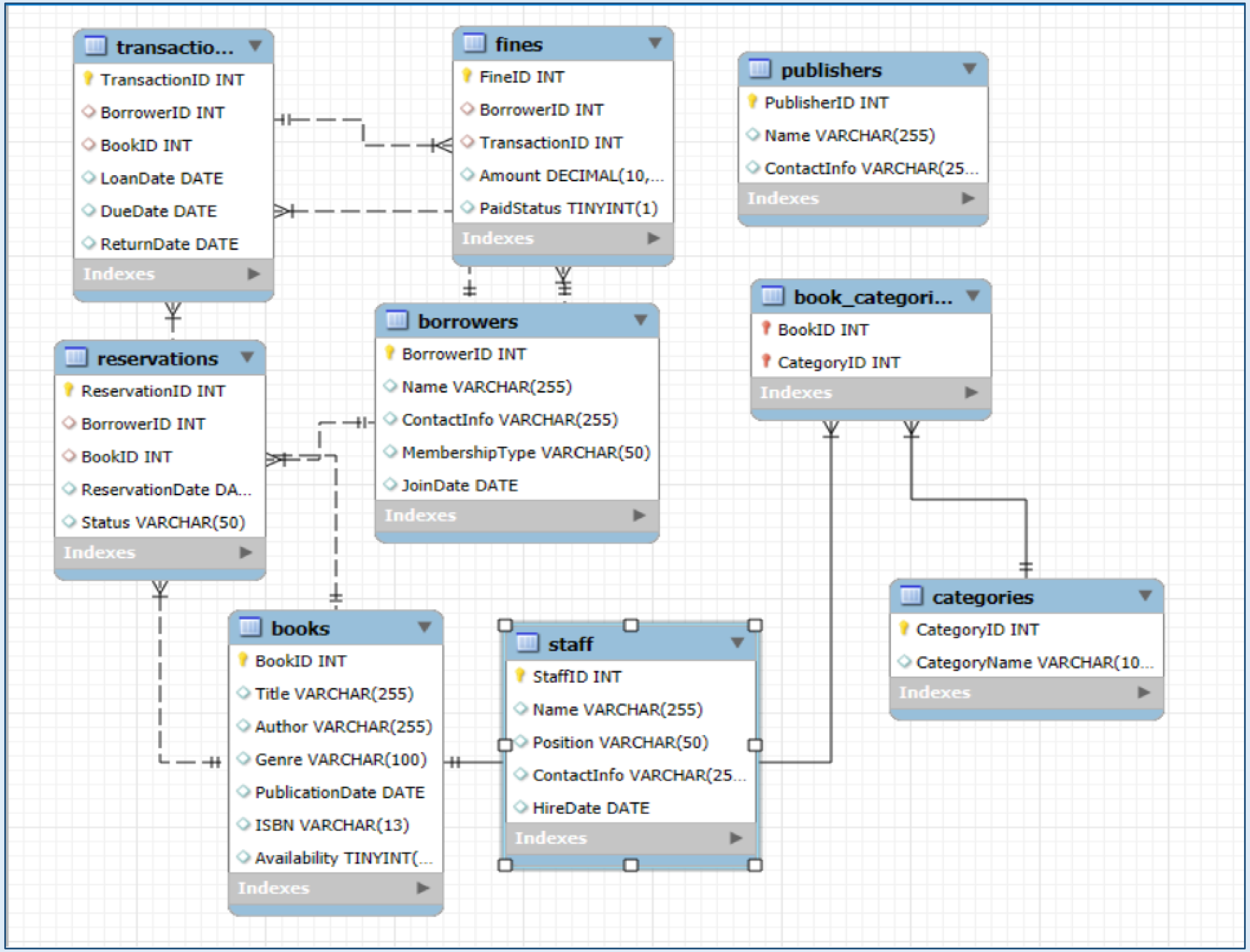
The library management system comprises key components including

- Books - Stores book information and availability status.
- Borrowers - Contains library members' personal details.
- Transactions - Logs book borrowing and returning activities
- Reservations - Manages borrowers' book reservation records.
- Staff - Stores information about library employees.
- Fines - Tracks overdue book fines details.
- Publishers - Contains information about book publishers.
- Categories - Lists different genres or book categories.
- Book_Categories - Links books to multiple categories.

TABLE RELATIONSHIPS AND INTERCONNECTIONS IN LIBRARY MANAGEMENT SYSTEM:

- Books - Reservations: Tracks reservations for each book.
- Books - Transactions: Tracks borrowed books.
- Borrowers - Transactions: Tracks books borrowed by members.
- Borrowers - Reservations: Manages members' book reservations.
- Transactions - Fines: Links overdue fines to transactions.
- Books - Publishers: Associates books with their publishers.
- Books - Book_Categories: Links books to multiple categories.
- Categories - Book_Categories: Classifies books into categories.

ENHANCED ENTITY-RELATIONSHIP DIAGRAM (EER):



transactio...
TransactionID INT
BorrowerID INT
BookID INT
LoanDate DATE
DueDate DATE
ReturnDate DATE
Indexes

fines
FineID INT
BorrowerID INT
TransactionID INT
Amount DECIMAL(10,...
PaidStatus TINYINT(1)
Indexes

publishers
PublisherID INT
Name VARCHAR(255)
ContactInfo VARCHAR(25...
Indexes

reservations
ReservationID INT
BorrowerID INT
BookID INT
ReservationDate DA...
Status VARCHAR(50)
Indexes

borrowers
BorrowerID INT
Name VARCHAR(255)
ContactInfo VARCHAR(255)
MembershipType VARCHAR(50)
JoinDate DATE
Indexes

book_categori...
BookID INT
CategoryID INT
Indexes

books
BookID INT
Title VARCHAR(255)
Author VARCHAR(255)
Genre VARCHAR(100)
PublicationDate DATE
ISBN VARCHAR(13)
Availability TINYINT(...
Indexes

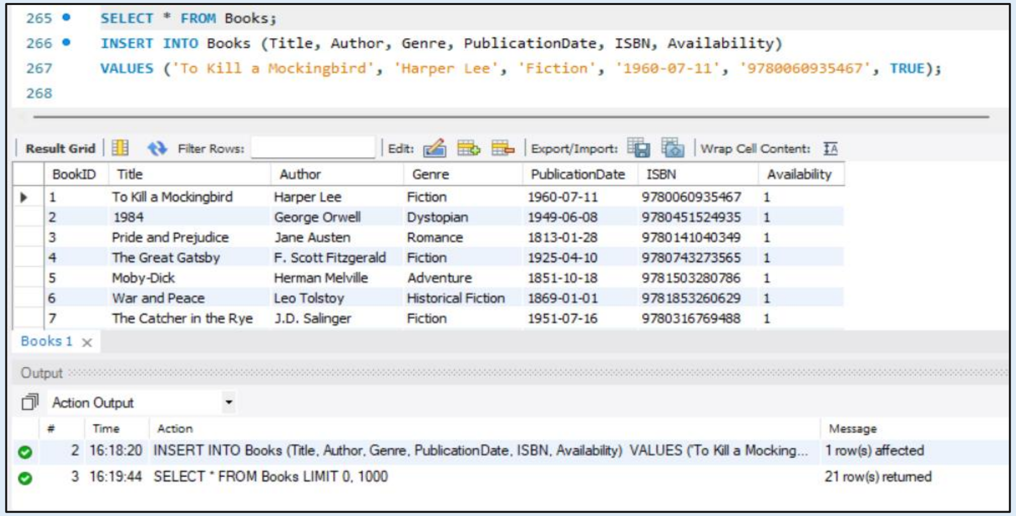
staff
StaffID INT
Name VARCHAR(255)
Position VARCHAR(50)
ContactInfo VARCHAR(25...
HireDate DATE
Indexes

categories
CategoryID INT
CategoryName VARCHAR(10...
Indexes

LIBRARY DATA INSIGHTS, ESSENTIAL SQL QUERIES:

BASIC CRUD OPERARTIONS:

1. Insert a new book into the Books table:

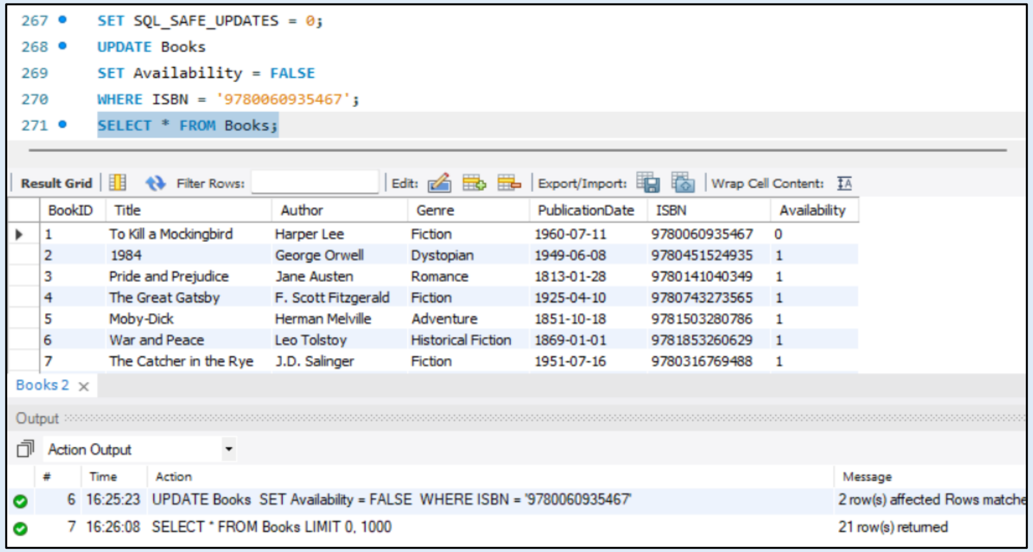


```sql
SELECT * FROM Books;
INSERT INTO Books (Title, Author, Genre, PublicationDate, ISBN, Availability)
VALUES ('To Kill a Mockingbird', 'Harper Lee', 'Fiction', '1960-07-11', '9780060935467', TRUE);
```

BookID	Title	Author	Genre	PublicationDate	ISBN	Availability
1	To Kill a Mockingbird	Harper Lee	Fiction	1960-07-11	9780060935467	1
2	1984	George Orwell	Dystopian	1949-06-08	9780451524935	1
3	Pride and Prejudice	Jane Austen	Romance	1813-01-28	9780141040349	1
4	The Great Gatsby	F. Scott Fitzgerald	Fiction	1925-04-10	9780743273565	1
5	Moby-Dick	Herman Melville	Adventure	1851-10-18	9781503280786	1
6	War and Peace	Leo Tolstoy	Historical Fiction	1869-01-01	9781853260629	1
7	The Catcher in the Rye	J.D. Salinger	Fiction	1951-07-16	9780316769488	1

```
#  Time      Action                                                                                       Message
2  16:18:20  INSERT INTO Books (Title, Author, Genre, PublicationDate, ISBN, Availability) VALUES ('To Kill a Mocking...  1 row(s) affected
3  16:19:44  SELECT * FROM Books LIMIT 0, 1000                                                            21 row(s) returned
```

2. Update a book's availability status in the Books table:

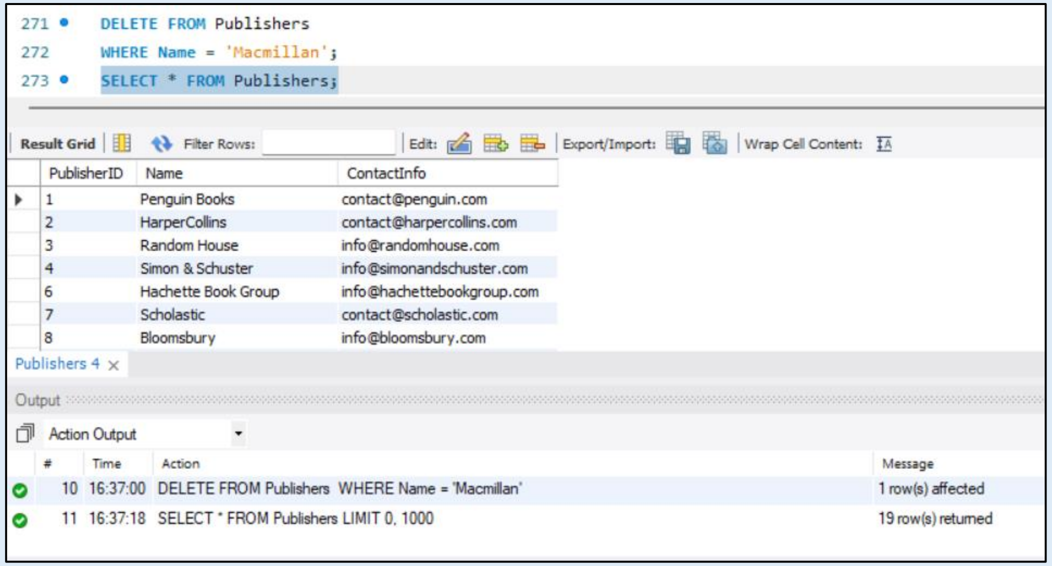


```sql
SET SQL_SAFE_UPDATES = 0;
UPDATE Books
SET Availability = FALSE
WHERE ISBN = '9780060935467';
SELECT * FROM Books;
```

BookID	Title	Author	Genre	PublicationDate	ISBN	Availability
1	To Kill a Mockingbird	Harper Lee	Fiction	1960-07-11	9780060935467	0
2	1984	George Orwell	Dystopian	1949-06-08	9780451524935	1
3	Pride and Prejudice	Jane Austen	Romance	1813-01-28	9780141040349	1
4	The Great Gatsby	F. Scott Fitzgerald	Fiction	1925-04-10	9780743273565	1
5	Moby-Dick	Herman Melville	Adventure	1851-10-18	9781503280786	1
6	War and Peace	Leo Tolstoy	Historical Fiction	1869-01-01	9781853260629	1
7	The Catcher in the Rye	J.D. Salinger	Fiction	1951-07-16	9780316769488	1

```
#  Time      Action                                                        Message
6  16:25:23  UPDATE Books  SET Availability = FALSE  WHERE ISBN = '9780060935467'  2 row(s) affected Rows matche
7  16:26:08  SELECT * FROM Books LIMIT 0, 1000                             21 row(s) returned
```

3. Delete from publishers table :

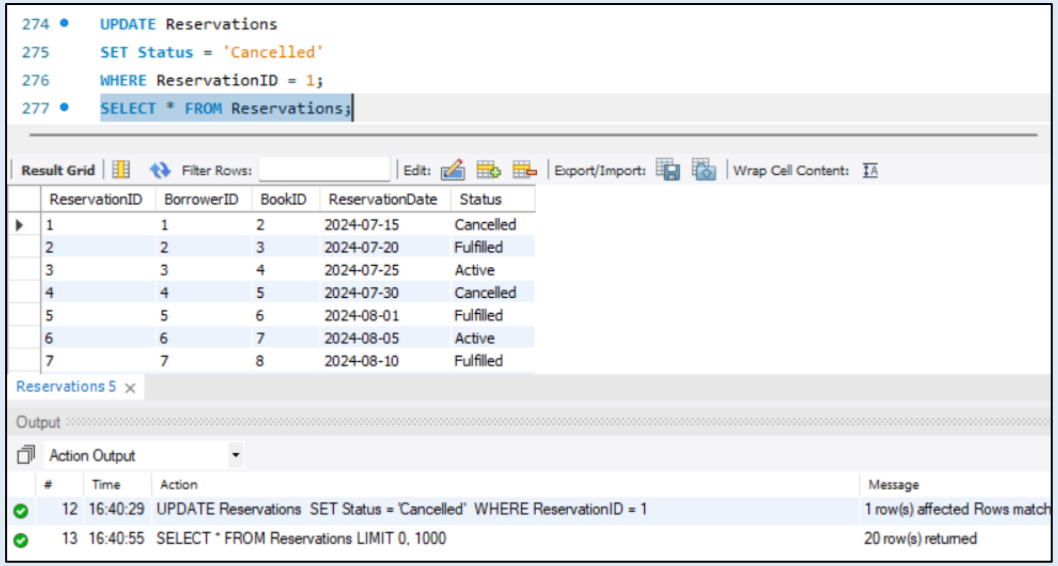


```sql
DELETE FROM Publishers
WHERE Name = 'Macmillan';
SELECT * FROM Publishers;
```

PublisherID	Name	ContactInfo
1	Penguin Books	contact@penguin.com
2	HarperCollins	contact@harpercollins.com
3	Random House	info@randomhouse.com
4	Simon & Schuster	info@simonandschuster.com
6	Hachette Book Group	info@hachettebookgroup.com
7	Scholastic	contact@scholastic.com
8	Bloomsbury	info@bloomsbury.com

```
#   Time      Action                                          Message
10  16:37:00  DELETE FROM Publishers  WHERE Name = 'Macmillan'  1 row(s) affected
11  16:37:18  SELECT * FROM Publishers LIMIT 0, 1000           19 row(s) returned
```

4. Update a reservation status in the Reservations table:

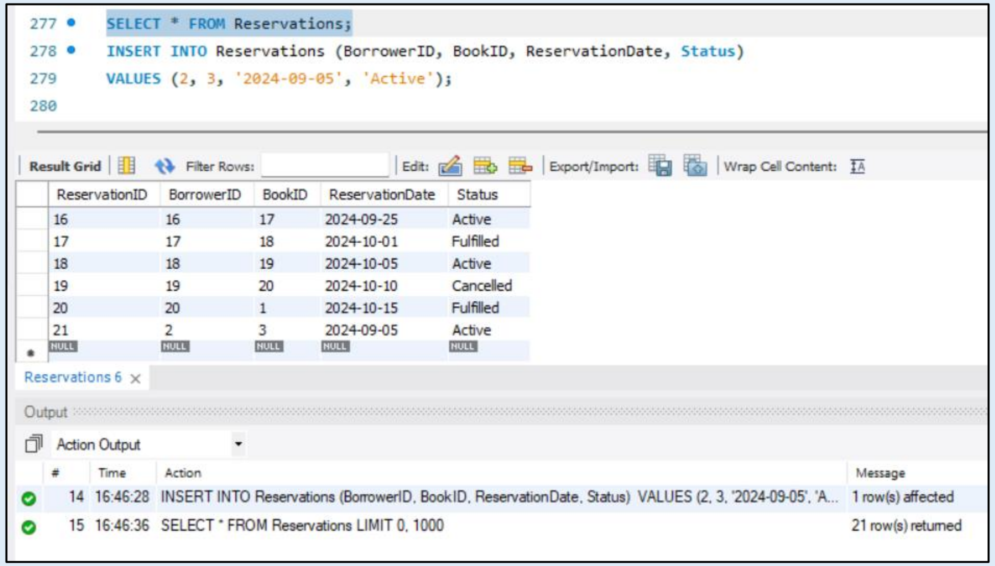


```sql
UPDATE Reservations
SET Status = 'Cancelled'
WHERE ReservationID = 1;
SELECT * FROM Reservations;
```

ReservationID	BorrowerID	BookID	ReservationDate	Status
1	1	2	2024-07-15	Cancelled
2	2	3	2024-07-20	Fulfilled
3	3	4	2024-07-25	Active
4	4	5	2024-07-30	Cancelled
5	5	6	2024-08-01	Fulfilled
6	6	7	2024-08-05	Active
7	7	8	2024-08-10	Fulfilled

```
#   Time      Action                                                       Message
12  16:40:29  UPDATE Reservations  SET Status = 'Cancelled'  WHERE ReservationID = 1  1 row(s) affected Rows match
13  16:40:55  SELECT * FROM Reservations LIMIT 0, 1000                     20 row(s) returned
```

5. Insert a new reservation into the Reservations table:



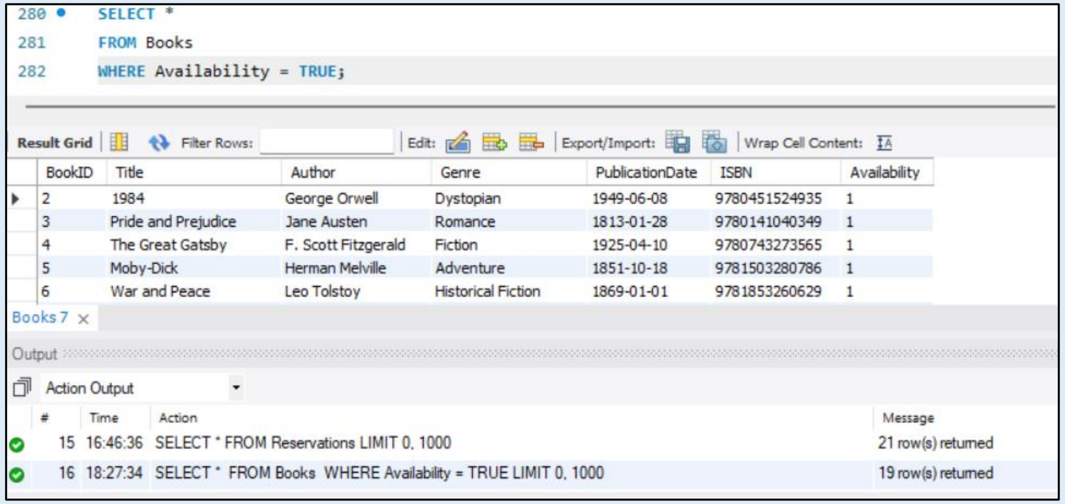
```sql
SELECT * FROM Reservations;
INSERT INTO Reservations (BorrowerID, BookID, ReservationDate, Status)
VALUES (2, 3, '2024-09-05', 'Active');
```

ReservationID	BorrowerID	BookID	ReservationDate	Status
16	16	17	2024-09-25	Active
17	17	18	2024-10-01	Fulfilled
18	18	19	2024-10-05	Active
19	19	20	2024-10-10	Cancelled
20	20	1	2024-10-15	Fulfilled
21	2	3	2024-09-05	Active
NULL	NULL	NULL	NULL	NULL

```
#   Time      Action                                                                                   Message
14  16:46:28  INSERT INTO Reservations (BorrowerID, BookID, ReservationDate, Status)  VALUES (2, 3, '2024-09-05', 'A...  1 row(s) affected
15  16:46:36  SELECT * FROM Reservations LIMIT 0, 1000                                                 21 row(s) returned
```

STANDARD DATA RETRIEVAL:

6. Select all available books:

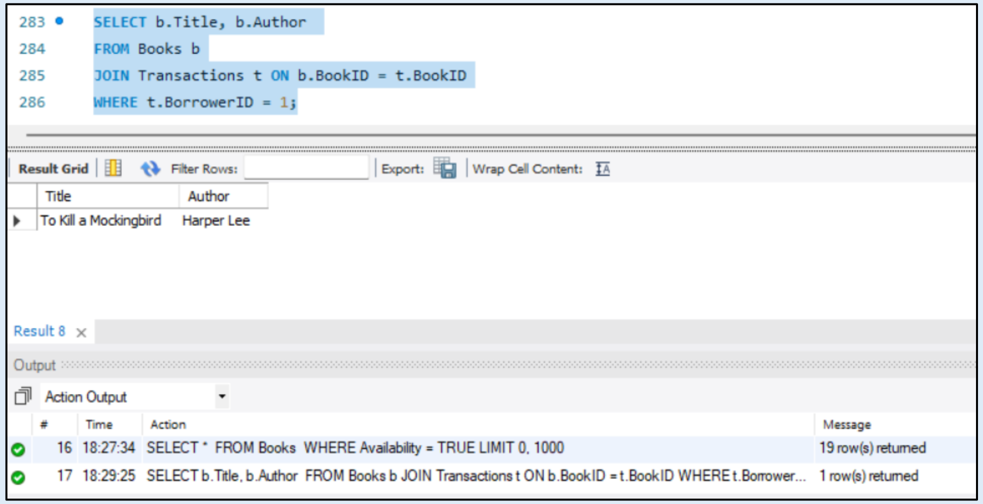


```sql
SELECT *
FROM Books
WHERE Availability = TRUE;
```

BookID	Title	Author	Genre	PublicationDate	ISBN	Availability
2	1984	George Orwell	Dystopian	1949-06-08	9780451524935	1
3	Pride and Prejudice	Jane Austen	Romance	1813-01-28	9780141040349	1
4	The Great Gatsby	F. Scott Fitzgerald	Fiction	1925-04-10	9780743273565	1
5	Moby-Dick	Herman Melville	Adventure	1851-10-18	9781503280786	1
6	War and Peace	Leo Tolstoy	Historical Fiction	1869-01-01	9781853260629	1

```
#   Time      Action                                                   Message
15  16:46:36  SELECT * FROM Reservations LIMIT 0, 1000                 21 row(s) returned
16  18:27:34  SELECT *  FROM Books  WHERE Availability = TRUE LIMIT 0, 1000   19 row(s) returned
```

7. Select all books borrowed by a specific borrower:

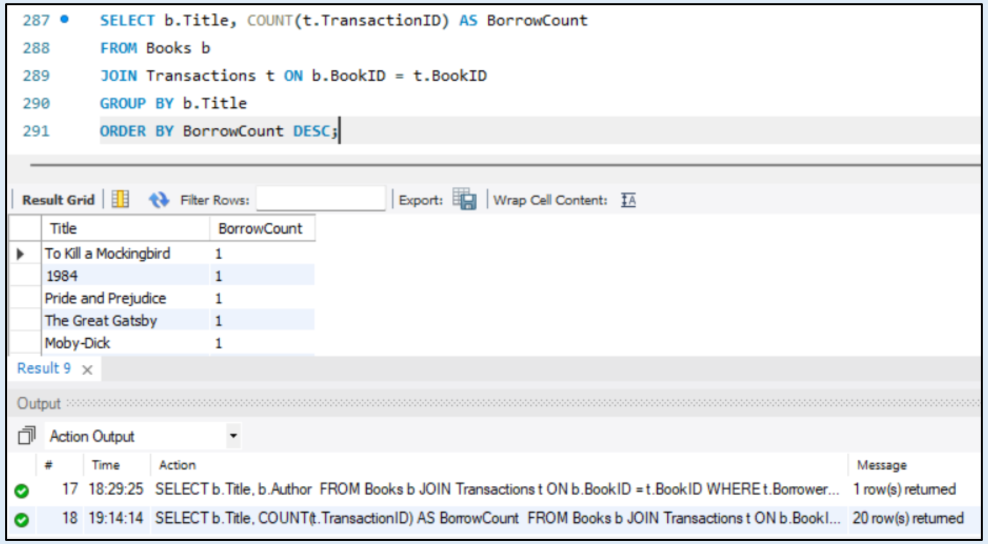


```sql
SELECT b.Title, b.Author
FROM Books b
JOIN Transactions t ON b.BookID = t.BookID
WHERE t.BorrowerID = 1;
```

Title	Author
To Kill a Mockingbird	Harper Lee

```
#   Time      Action                                                                           Message
16  18:27:34  SELECT *  FROM Books  WHERE Availability = TRUE LIMIT 0, 1000                    19 row(s) returned
17  18:29:25  SELECT b.Title, b.Author  FROM Books b JOIN Transactions t ON b.BookID = t.BookID WHERE t.Borrower...   1 row(s) returned
```

8. Find the most borrowed books:

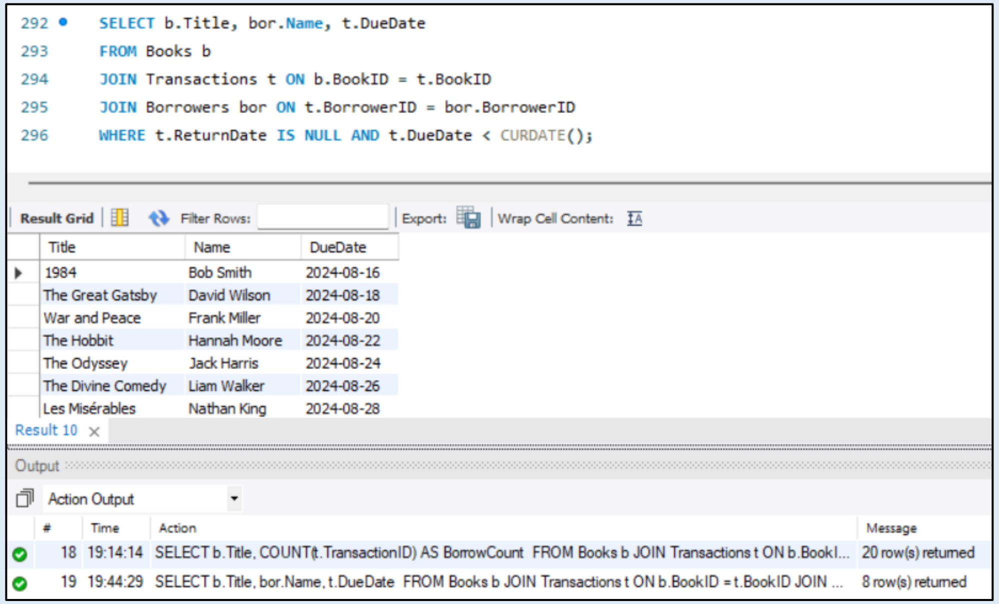


```sql
SELECT b.Title, COUNT(t.TransactionID) AS BorrowCount
FROM Books b
JOIN Transactions t ON b.BookID = t.BookID
GROUP BY b.Title
ORDER BY BorrowCount DESC;
```

Title	BorrowCount
To Kill a Mockingbird	1
1984	1
Pride and Prejudice	1
The Great Gatsby	1
Moby-Dick	1

```
#   Time      Action                                                                           Message
17  18:29:25  SELECT b.Title, b.Author  FROM Books b JOIN Transactions t ON b.BookID = t.BookID WHERE t.Borrower...   1 row(s) returned
18  19:14:14  SELECT b.Title, COUNT(t.TransactionID) AS BorrowCount  FROM Books b JOIN Transactions t ON b.BookI...   20 row(s) returned
```

9. Find all overdue books and their borrowers:

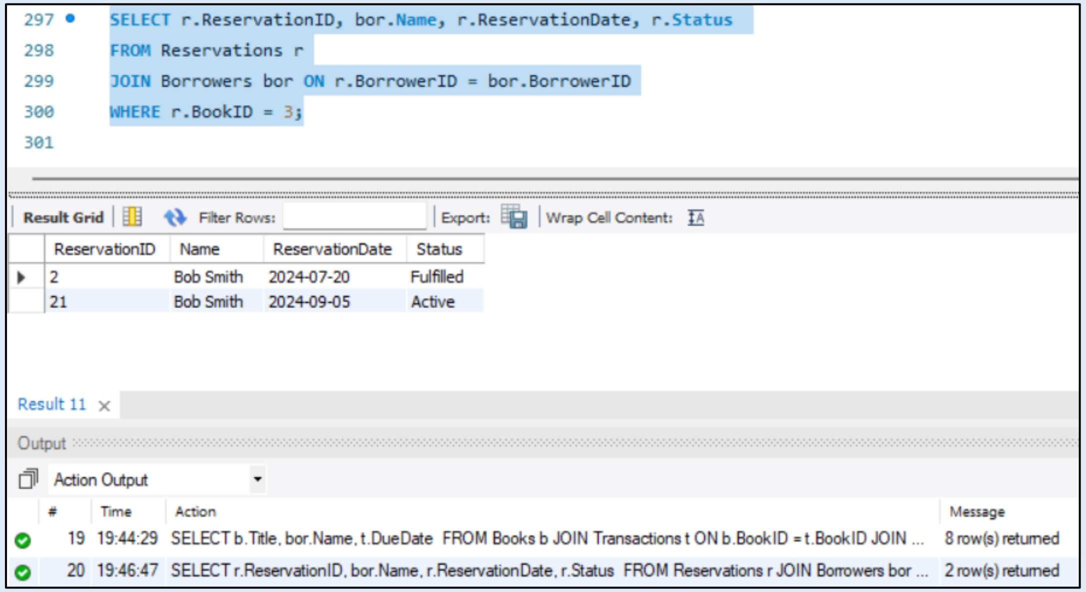


```sql
SELECT b.Title, bor.Name, t.DueDate
FROM Books b
JOIN Transactions t ON b.BookID = t.BookID
JOIN Borrowers bor ON t.BorrowerID = bor.BorrowerID
WHERE t.ReturnDate IS NULL AND t.DueDate < CURDATE();
```

Title	Name	DueDate
1984	Bob Smith	2024-08-16
The Great Gatsby	David Wilson	2024-08-18
War and Peace	Frank Miller	2024-08-20
The Hobbit	Hannah Moore	2024-08-22
The Odyssey	Jack Harris	2024-08-24
The Divine Comedy	Liam Walker	2024-08-26
Les Misérables	Nathan King	2024-08-28

	#	Time	Action	Message
✓	18	19:14:14	SELECT b.Title, COUNT(t.TransactionID) AS BorrowCount FROM Books b JOIN Transactions t ON b.BookI...	20 row(s) returned
✓	19	19:44:29	SELECT b.Title, bor.Name, t.DueDate FROM Books b JOIN Transactions t ON b.BookID = t.BookID JOIN ...	8 row(s) returned

10. Select all reservations for a specific book:



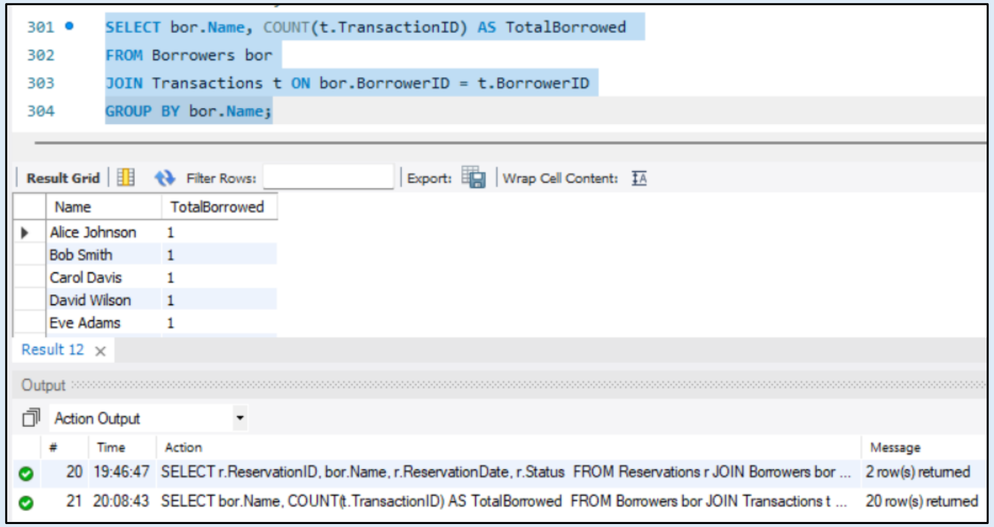
```sql
SELECT r.ReservationID, bor.Name, r.ReservationDate, r.Status
FROM Reservations r
JOIN Borrowers bor ON r.BorrowerID = bor.BorrowerID
WHERE r.BookID = 3;
```

ReservationID	Name	ReservationDate	Status
2	Bob Smith	2024-07-20	Fulfilled
21	Bob Smith	2024-09-05	Active

	#	Time	Action	Message
✓	19	19:44:29	SELECT b.Title, bor.Name, t.DueDate FROM Books b JOIN Transactions t ON b.BookID = t.BookID JOIN ...	8 row(s) returned
✓	20	19:46:47	SELECT r.ReservationID, bor.Name, r.ReservationDate, r.Status FROM Reservations r JOIN Borrowers bor ...	2 row(s) returned

ADVANCED QUERIES AND JOINS:

11.Find the total number of books borrowed by each borrower:

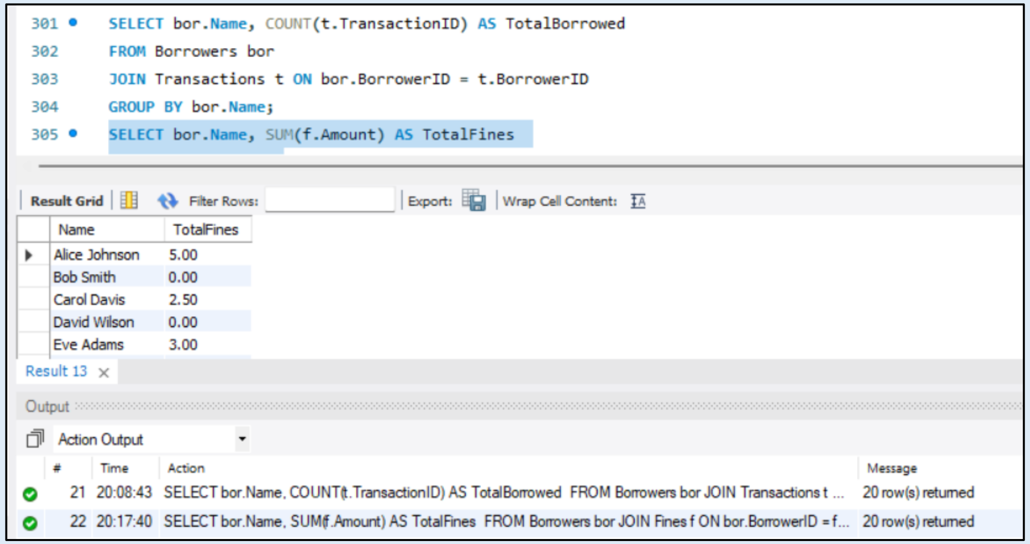


```sql
SELECT bor.Name, COUNT(t.TransactionID) AS TotalBorrowed
FROM Borrowers bor
JOIN Transactions t ON bor.BorrowerID = t.BorrowerID
GROUP BY bor.Name;
```

Name	TotalBorrowed
Alice Johnson	1
Bob Smith	1
Carol Davis	1
David Wilson	1
Eve Adams	1

```
#   Time      Action                                                                                         Message
20  19:46:47  SELECT r.ReservationID, bor.Name, r.ReservationDate, r.Status  FROM Reservations r JOIN Borrowers bor ...  2 row(s) returned
21  20:08:43  SELECT bor.Name, COUNT(t.TransactionID) AS TotalBorrowed  FROM Borrowers bor JOIN Transactions t ...    20 row(s) returned
```

12. Calculate the total fines for each borrower:

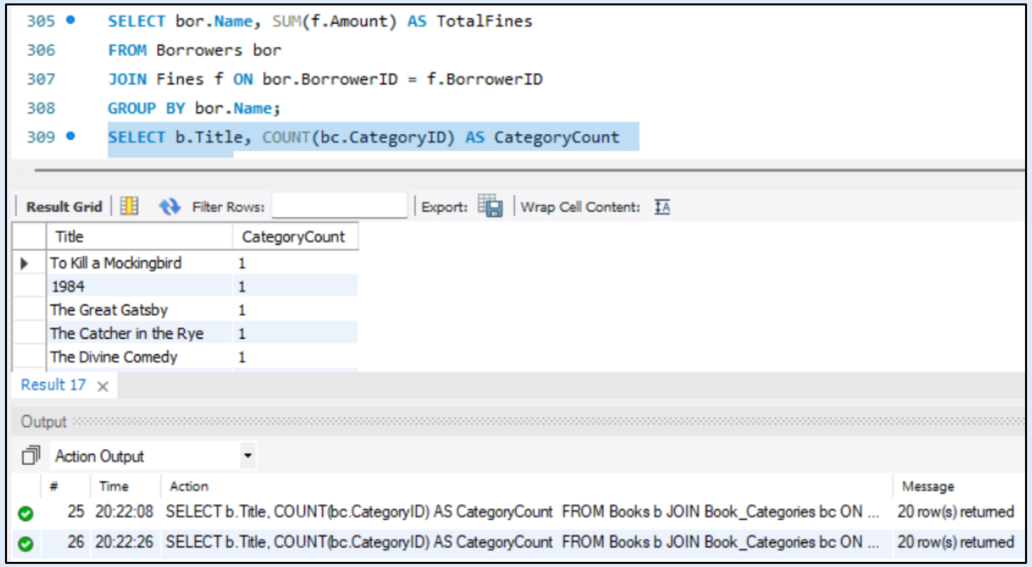


```sql
SELECT bor.Name, COUNT(t.TransactionID) AS TotalBorrowed
FROM Borrowers bor
JOIN Transactions t ON bor.BorrowerID = t.BorrowerID
GROUP BY bor.Name;
SELECT bor.Name, SUM(f.Amount) AS TotalFines
```

Name	TotalFines
Alice Johnson	5.00
Bob Smith	0.00
Carol Davis	2.50
David Wilson	0.00
Eve Adams	3.00

```
#   Time      Action                                                                                       Message
21  20:08:43  SELECT bor.Name, COUNT(t.TransactionID) AS TotalBorrowed  FROM Borrowers bor JOIN Transactions t ...  20 row(s) returned
22  20:17:40  SELECT bor.Name, SUM(f.Amount) AS TotalFines  FROM Borrowers bor JOIN Fines f ON bor.BorrowerID = f...  20 row(s) returned
```

13. List all books that belong to multiple categories:

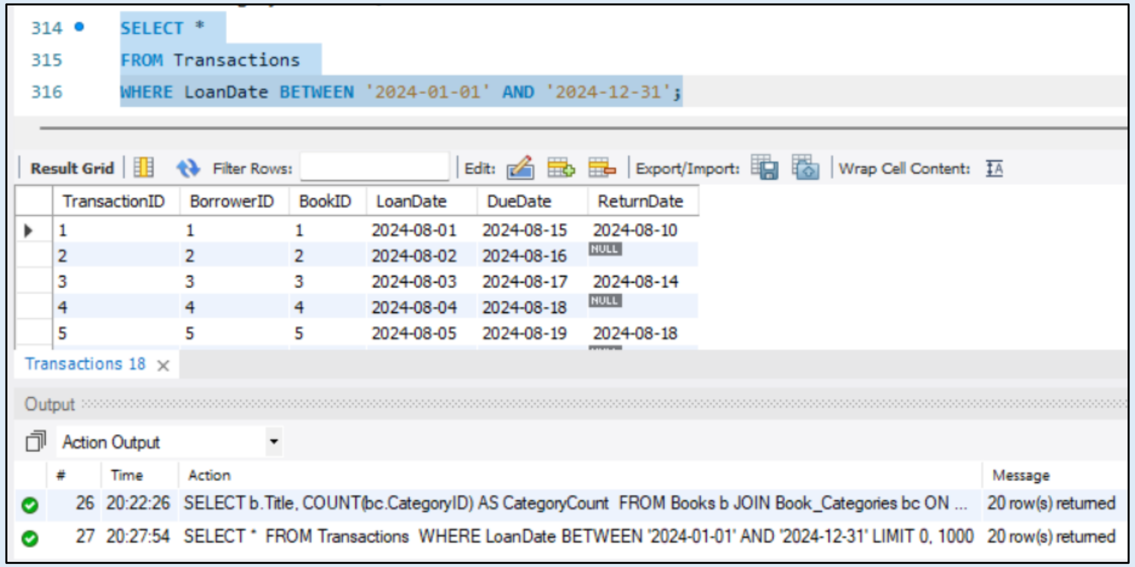


```sql
SELECT bor.Name, SUM(f.Amount) AS TotalFines
FROM Borrowers bor
JOIN Fines f ON bor.BorrowerID = f.BorrowerID
GROUP BY bor.Name;
SELECT b.Title, COUNT(bc.CategoryID) AS CategoryCount
```

Title	CategoryCount
To Kill a Mockingbird	1
1984	1
The Great Gatsby	1
The Catcher in the Rye	1
The Divine Comedy	1

#	Time	Action	Message
25	20:22:08	SELECT b.Title, COUNT(bc.CategoryID) AS CategoryCount FROM Books b JOIN Book_Categories bc ON ...	20 row(s) returned
26	20:22:26	SELECT b.Title, COUNT(bc.CategoryID) AS CategoryCount FROM Books b JOIN Book_Categories bc ON ...	20 row(s) returned

14. Find all transactions within a specific date range:

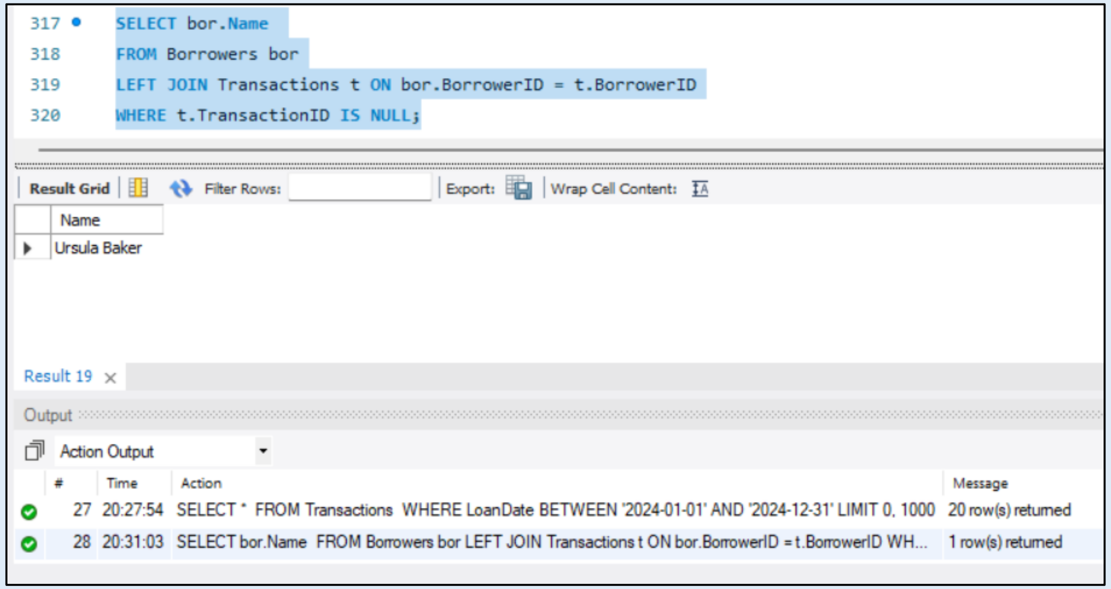


```sql
SELECT *
FROM Transactions
WHERE LoanDate BETWEEN '2024-01-01' AND '2024-12-31';
```

TransactionID	BorrowerID	BookID	LoanDate	DueDate	ReturnDate
1	1	1	2024-08-01	2024-08-15	2024-08-10
2	2	2	2024-08-02	2024-08-16	NULL
3	3	3	2024-08-03	2024-08-17	2024-08-14
4	4	4	2024-08-04	2024-08-18	NULL
5	5	5	2024-08-05	2024-08-19	2024-08-18

#	Time	Action	Message
26	20:22:26	SELECT b.Title, COUNT(bc.CategoryID) AS CategoryCount FROM Books b JOIN Book_Categories bc ON ...	20 row(s) returned
27	20:27:54	SELECT * FROM Transactions WHERE LoanDate BETWEEN '2024-01-01' AND '2024-12-31' LIMIT 0, 1000	20 row(s) returned

15. List all borrowers who have never borrowed a book:



```sql
SELECT bor.Name
FROM Borrowers bor
LEFT JOIN Transactions t ON bor.BorrowerID = t.BorrowerID
WHERE t.TransactionID IS NULL;
```

Name
Ursula Baker

#	Time	Action	Message
27	20:27:54	SELECT * FROM Transactions WHERE LoanDate BETWEEN '2024-01-01' AND '2024-12-31' LIMIT 0, 1000	20 row(s) returned
28	20:31:03	SELECT bor.Name FROM Borrowers bor LEFT JOIN Transactions t ON bor.BorrowerID = t.BorrowerID WH...	1 row(s) returned

CONCLUSION:

- This Library Management System project showcases the effectiveness of SQL in managing library operations, including book inventory, borrower details, transactions, reservations, and fines.
- SQL's robust querying capabilities allow for efficient data management and retrieval, supporting critical library functions and enhancing decision-making.
- The use of SQL ensures data accuracy, optimizes operational workflows, and provides valuable insights into library usage, making it a powerful tool for any library management system.