Library Management System

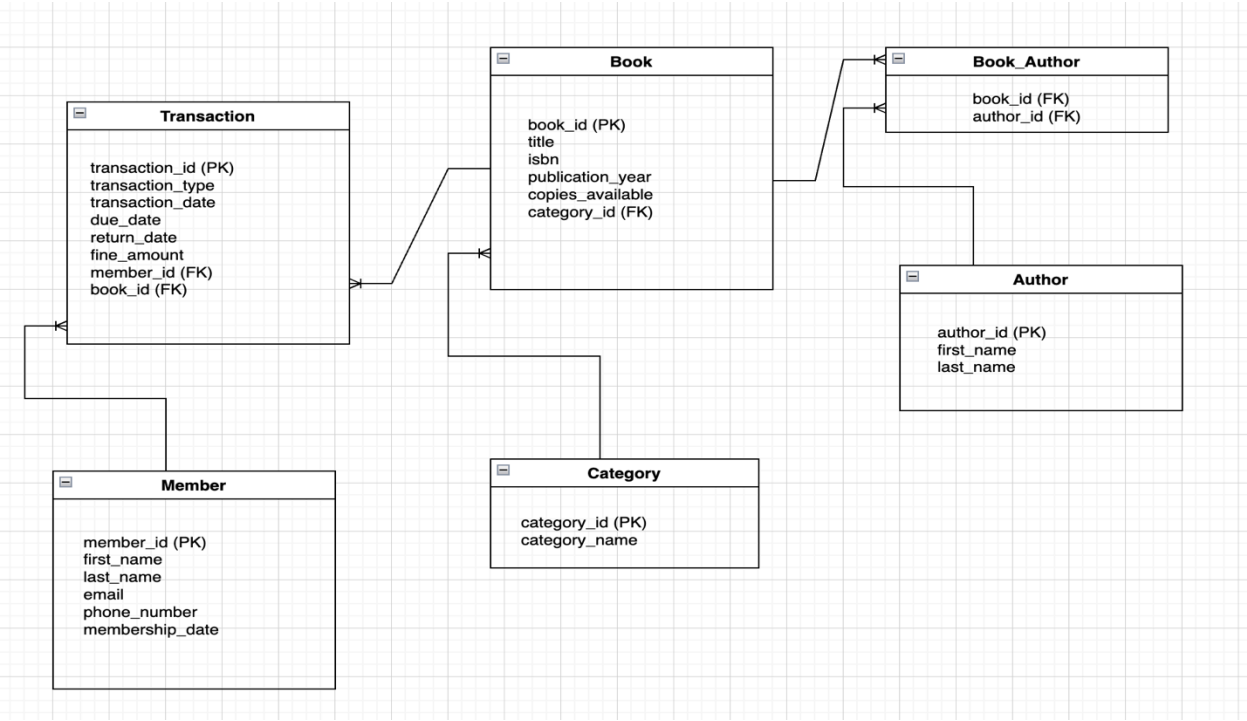
The main objective of this project is to develop a robust database structure that efficiently stores and organizes books within a library system. The project involves the use of SQL queries and joins to extract valuable insights from the library data.

Database System Design and Relationships

The database schema of this library management system is designed with key entities and their relationships to ensure efficient data management. Below, I have outlined the relationships between the various entities involved in the system:

Entities in the Database Design:

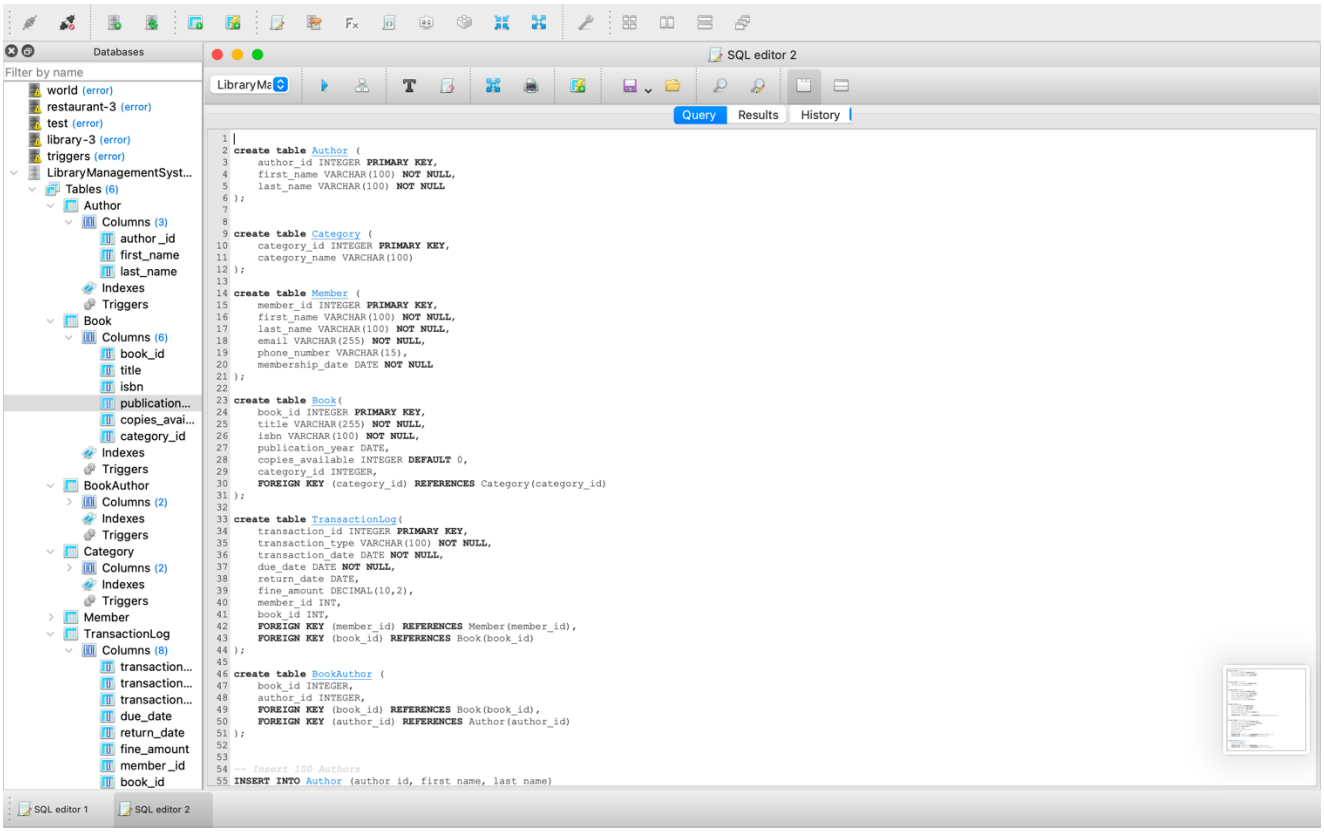
- **Transaction**
- **Book**
- **Book_Author**
- **Author**
- **Category**
- **Member**



Relationships Between Entities:

1. **Book and Category:**
    - **Relationship Type:** One-to-Many
    - **Explanation:** Each book belongs to a specific category, such as Fiction, Non-Fiction, Fantasy, etc. The Category entity helps in organizing books into genres, and this relationship ensures that each book is linked to a particular category.
2. **Book and Author (via Book_Author):**
    - **Relationship Type:** Many-to-Many
    - **Explanation:** A book can have multiple authors, and an author can write multiple books. To manage this relationship, we introduced a *Book_Author* entity, which acts as a junction table connecting the Book and Author entities.
3. **Member and Transaction:**
    - **Relationship Type:** One-to-Many
    - **Explanation:** A member can make multiple transactions, such as borrowing and returning books. Each transaction record is linked to a specific member.
4. **Book and Transaction:**
    - **Relationship Type:** One-to-Many
    - **Explanation:** A book can be part of multiple transactions, such as being borrowed and returned multiple times by different members. Each transaction is linked to a specific book.
5. **Author and Book_Author:**
    - **Relationship Type:** One-to-Many
    - **Explanation:** An author can write multiple books. The Book_Author entity maintains the mapping of authors to their written books.

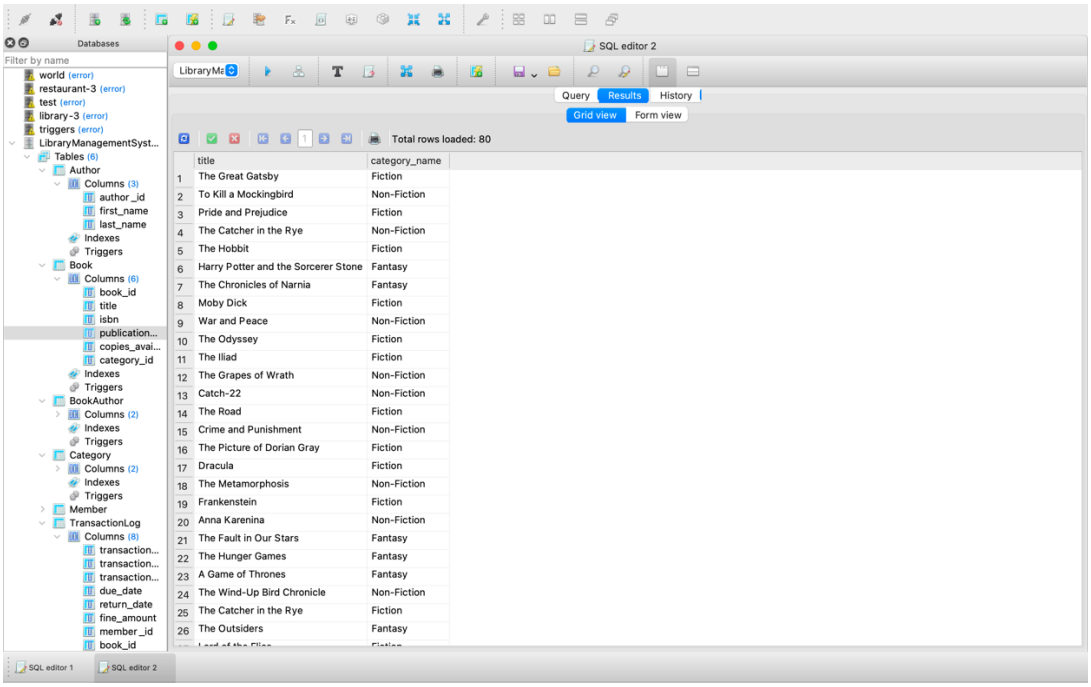
Database Design Implementation



Data Analysis and Query Implementation

1. List all the books that falls under the categories fantasy, nonfiction and fiction.

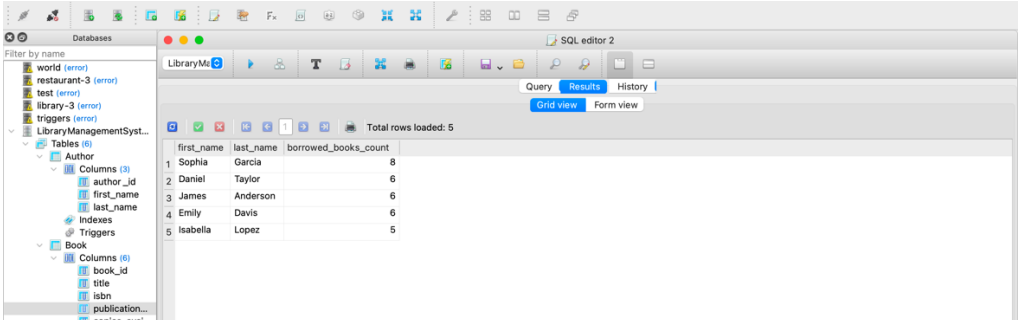
```
SELECT b.title, c.category_name
FROM book b
JOIN category c
ON b.category_id = c.category_id
WHERE c.category_name IN ('Fantasy', 'Non-Fiction', 'Fiction');
```



	title	category_name
1	The Great Gatsby	Fiction
2	To Kill a Mockingbird	Non-Fiction
3	Pride and Prejudice	Fiction
4	The Catcher in the Rye	Non-Fiction
5	The Hobbit	Fiction
6	Harry Potter and the Sorcerer Stone	Fantasy
7	The Chronicles of Narnia	Fantasy
8	Moby Dick	Fiction
9	War and Peace	Non-Fiction
10	The Odyssey	Fiction
11	The Iliad	Fiction
12	The Grapes of Wrath	Non-Fiction
13	Catch-22	Non-Fiction
14	The Road	Fiction
15	Crime and Punishment	Non-Fiction
16	The Picture of Dorian Gray	Fiction
17	Dracula	Fiction
18	The Metamorphosis	Non-Fiction
19	Frankenstein	Fiction
20	Anna Karenina	Non-Fiction
21	The Fault in Our Stars	Fantasy
22	The Hunger Games	Fantasy
23	A Game of Thrones	Fantasy
24	The Wind-Up Bird Chronicle	Non-Fiction
25	The Catcher in the Rye	Fiction
26	The Outsiders	Fantasy

2. Write a query to find the top 5 members who have borrowed the most books.

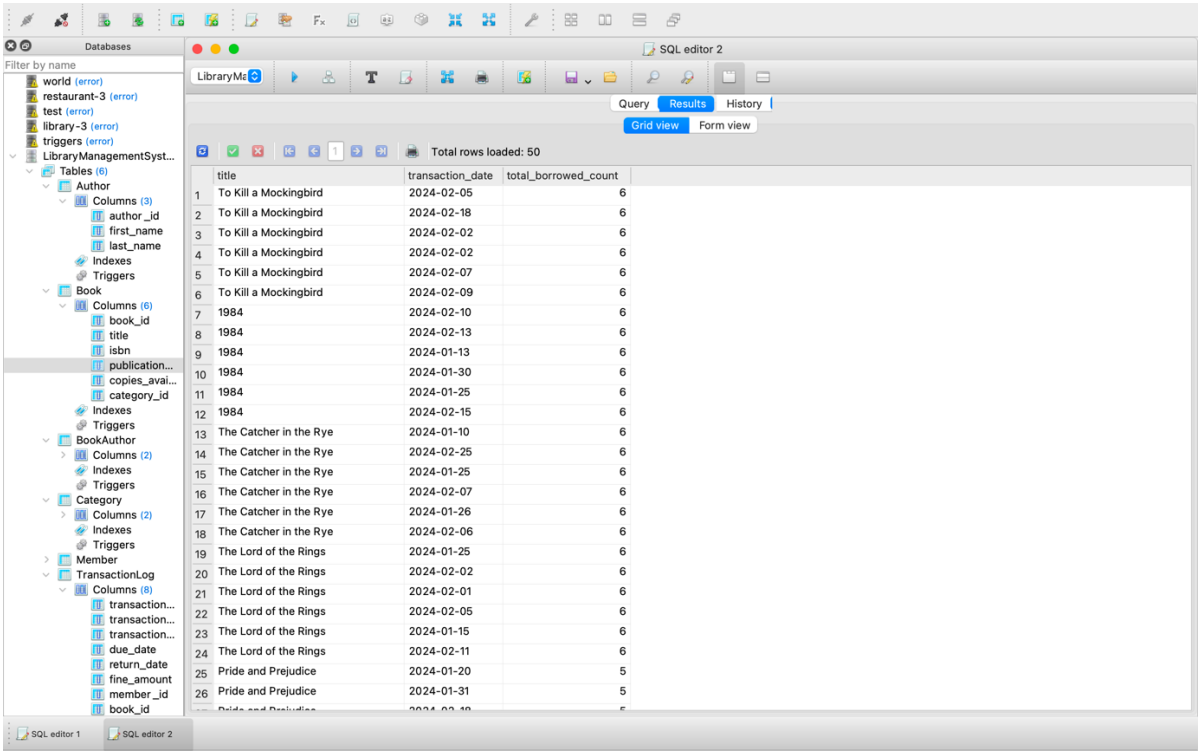
```
select m.first_name, m.last_name, count(*) as borrowed_books_count
from member m
inner join transactionlog t
on m.member_id = t.member_id
where t.transaction_type = 'Borrow'
group by m.member_id
order by borrowed_books_count desc
LIMIT 5;
```



	first_name	last_name	borrowed_books_count
1	Sophia	Garcia	8
2	Daniel	Taylor	6
3	James	Anderson	6
4	Emily	Davis	6
5	Isabella	Lopez	5

3. Write a SQL query to find the total number of times each book has been borrowed, along with the transaction dates, ordered by the total borrow count in descending order.

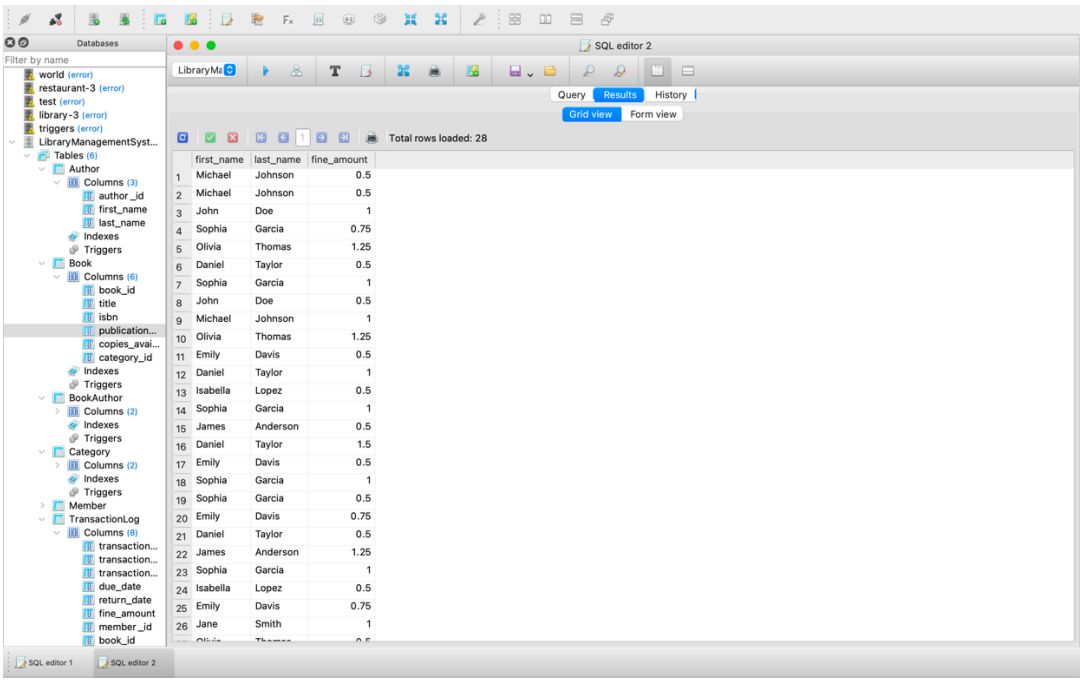
```sql
select b.title, t.transaction_date, count(*) over (partition by b.book_id) as total_borrowed_count
from Book b
join transactionlog t
on b.book_id = t.book_id
where t.transaction_type = 'Borrow'
order by total_borrowed_count DESC;
```



	title	transaction_date	total_borrowed_count
1	To Kill a Mockingbird	2024-02-05	6
2	To Kill a Mockingbird	2024-02-18	6
3	To Kill a Mockingbird	2024-02-02	6
4	To Kill a Mockingbird	2024-02-02	6
5	To Kill a Mockingbird	2024-02-07	6
6	To Kill a Mockingbird	2024-02-09	6
7	1984	2024-02-10	6
8	1984	2024-02-13	6
9	1984	2024-01-13	6
10	1984	2024-01-30	6
11	1984	2024-01-25	6
12	1984	2024-02-15	6
13	The Catcher in the Rye	2024-01-10	6
14	The Catcher in the Rye	2024-02-25	6
15	The Catcher in the Rye	2024-01-25	6
16	The Catcher in the Rye	2024-02-07	6
17	The Catcher in the Rye	2024-01-26	6
18	The Catcher in the Rye	2024-02-06	6
19	The Lord of the Rings	2024-01-25	6
20	The Lord of the Rings	2024-02-02	6
21	The Lord of the Rings	2024-02-01	6
22	The Lord of the Rings	2024-02-05	6
23	The Lord of the Rings	2024-01-15	6
24	The Lord of the Rings	2024-02-11	6
25	Pride and Prejudice	2024-01-20	5
26	Pride and Prejudice	2024-01-31	5

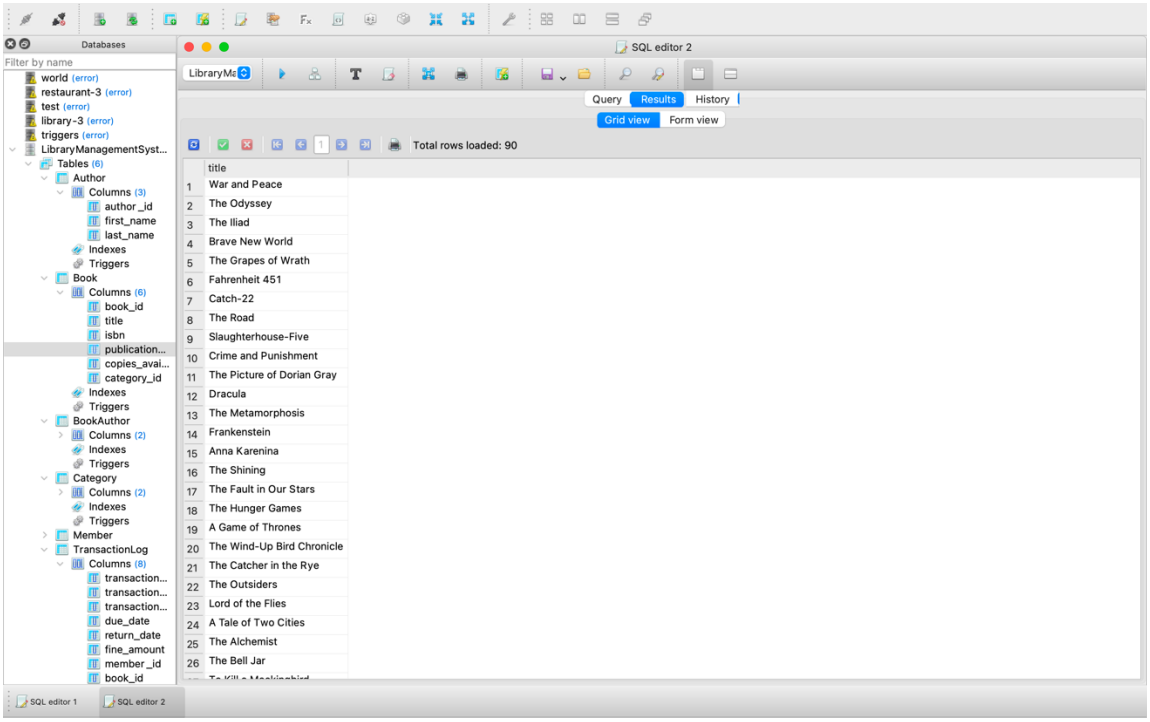
4. List the members who have returned books late along with the fine amount.

```sql
select m.first_name, m.last_name, t.fine_amount
from member m
join transactionlog t
on m.member_id = t.member_id
where t.fine_amount is not null and t.fine_amount <> 0;
```



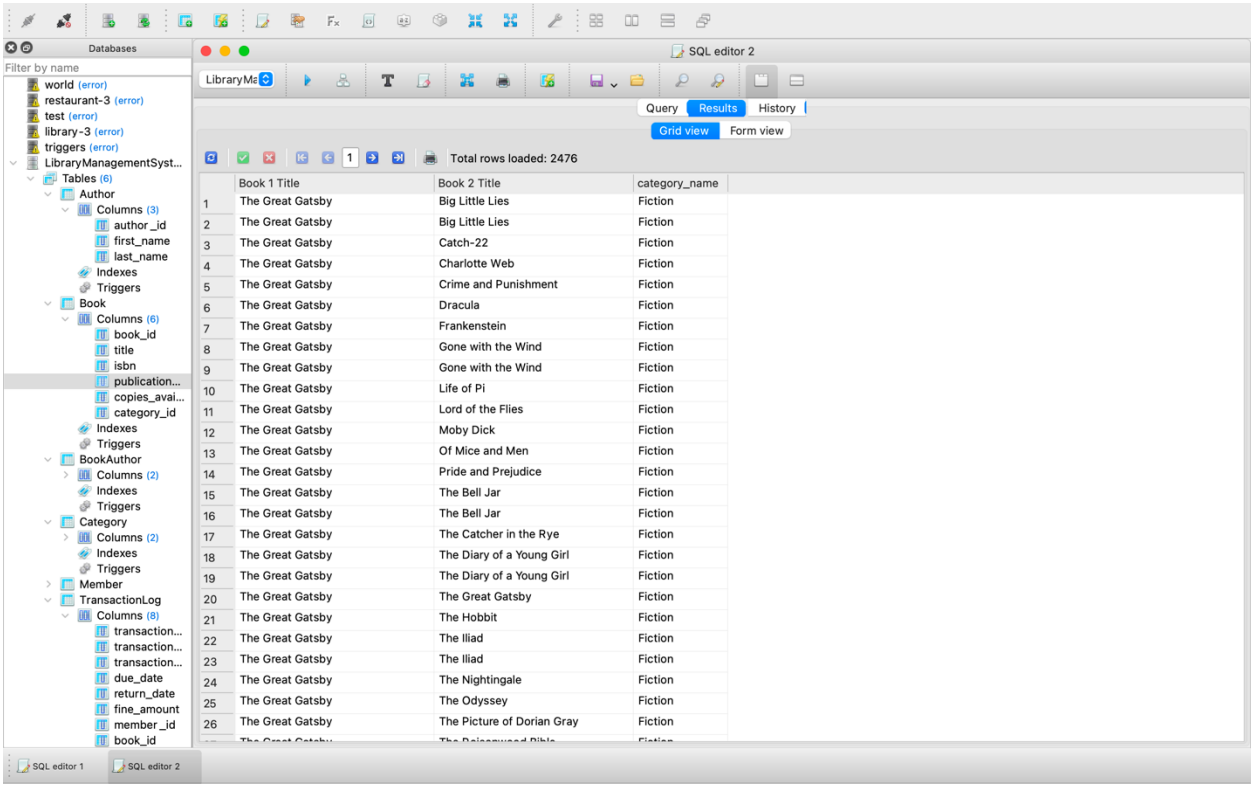
5. Find the books that have never been borrowed.

```
select b.title
from Book b
left join transactionlog t
on b.book_id = t.book_id
where t.book_id is null;
```



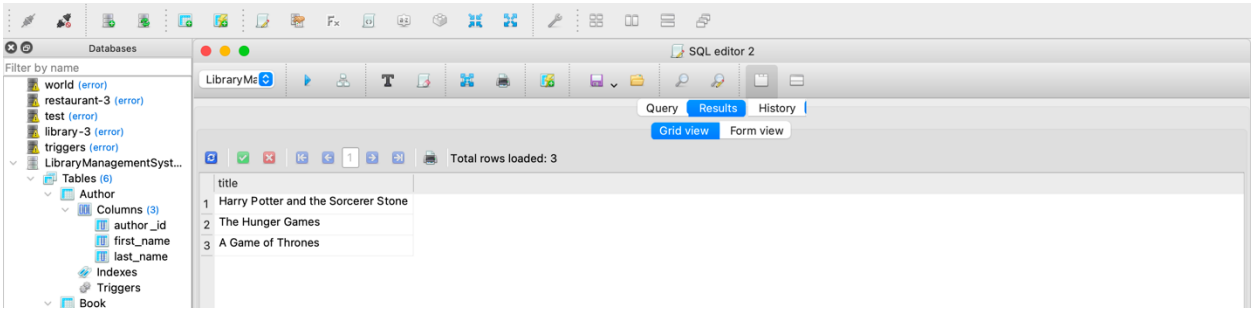
6. Find the books that have the same genre as another book.

```
select b1.title as "Book 1 Title", b2.title as "Book 2 Title", c.category_name
from book b1
join book b2 on b1.book_id <> b2.book_id
join category c on b1.category_id = c.category_id and b2.category_id = c.category_id;
```



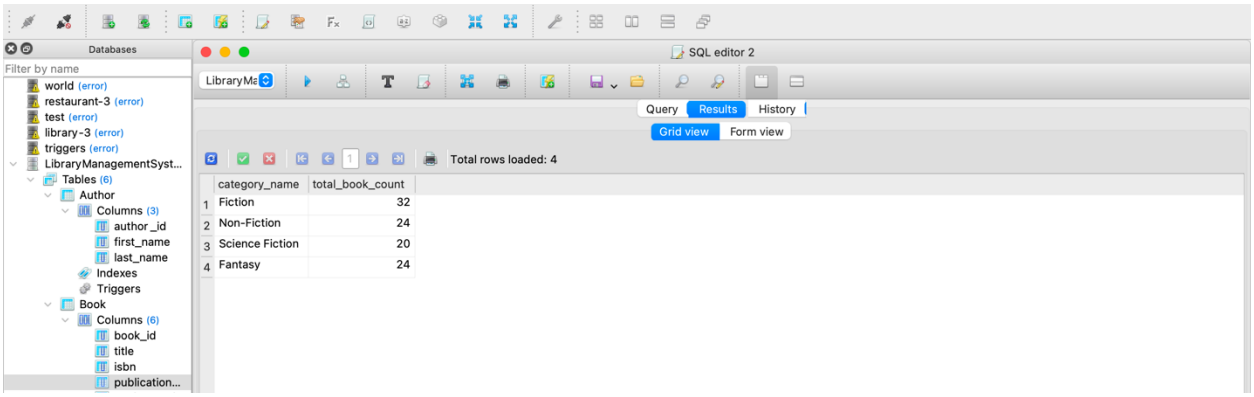
7. Find all book titles in the Book table that contain the word "Harry" and "Game" anywhere in the title.

```
select b.title
from Book b
where b.title like '%Harry%' or b.title like '%Game%';
```



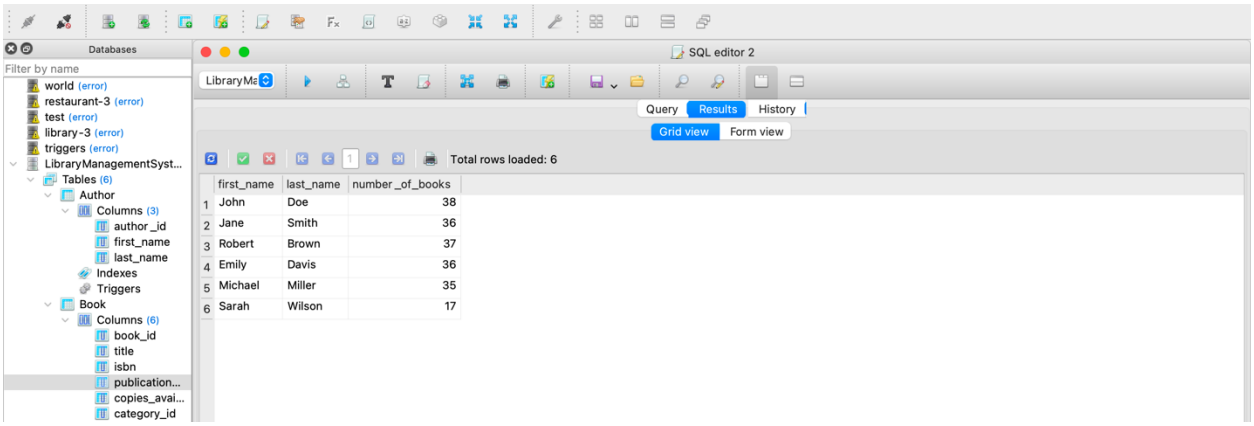
8. Write a SQL query to categorize books based on their genres and count how many books belong to each genre.

```sql
select c.category_name, count(*) as "total_book_count"
from book  b
inner join category c
on b.category_id = c.category_id
group by b.category_id;
```



9. Count the number of books each author has written.

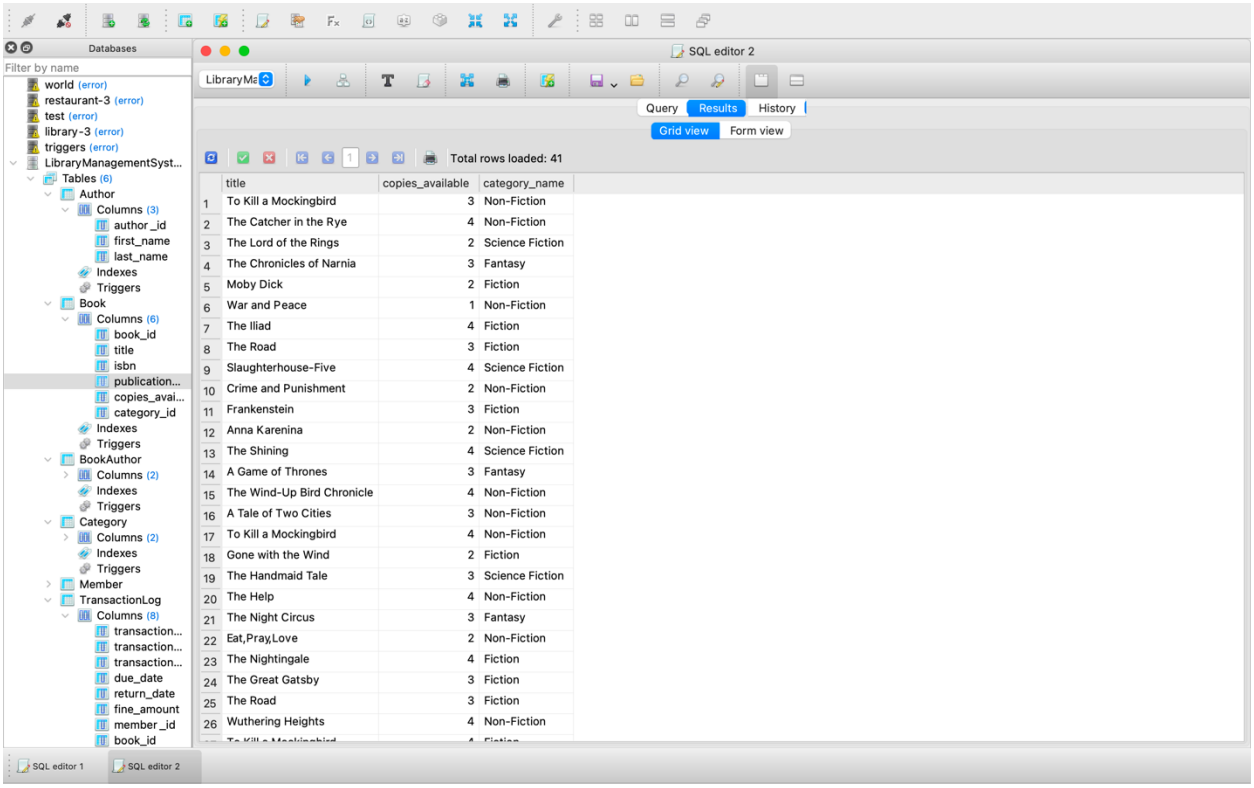
```sql
select a.first_name, a.last_name, count(*) as number_of_books
from author a
join bookauthor b
on a.author_id = b.author_id
group by a.author_id;
```



10. Write a SQL query to find books with limited availability (less than 5 copies) and categorize them based on their category.

```sql
with book_categories as (
    select b.title, b.copies_available, c.category_name,
        case when c.category_name='Fiction' then 'Limited stock in Fiction category'
        when c.category_name='Non-Fiction' then 'Limited stock in NonFiction category'
        when c.category_name='Science Fiction' then 'Limited stock in ScienceFiction category'
        when c.category_name='Fantasy' then 'Limited stock in Fantasy category'
        when c.category_name='Biography' then 'Limited stock in Biography category'
        when c.category_name='History' then 'Limited stock in History category'
        when c.category_name='Technology' then 'Limited stock in Technology category'
        when c.category_name='Self-help' then 'Limited stock in Self Help category'
        when c.category_name='Health' then 'Limited stock in Health category'
        else 'Limited stock in Romance category'
        end as stock_status
    from book b
    inner join category c on b.category_id = c.category_id
)

select title, copies_available, category_name
from book_categories
where copies_available < 5;
```



	title	copies_available	category_name
1	To Kill a Mockingbird	3	Non-Fiction
2	The Catcher in the Rye	4	Non-Fiction
3	The Lord of the Rings	2	Science Fiction
4	The Chronicles of Narnia	3	Fantasy
5	Moby Dick	2	Fiction
6	War and Peace	1	Non-Fiction
7	The Iliad	4	Fiction
8	The Road	3	Fiction
9	Slaughterhouse-Five	4	Science Fiction
10	Crime and Punishment	2	Non-Fiction
11	Frankenstein	3	Fiction
12	Anna Karenina	2	Non-Fiction
13	The Shining	4	Science Fiction
14	A Game of Thrones	3	Fantasy
15	The Wind-Up Bird Chronicle	4	Non-Fiction
16	A Tale of Two Cities	3	Non-Fiction
17	To Kill a Mockingbird	4	Non-Fiction
18	Gone with the Wind	2	Fiction
19	The Handmaid Tale	3	Science Fiction
20	The Help	4	Non-Fiction
21	The Night Circus	3	Fantasy
22	Eat,Pray,Love	2	Non-Fiction
23	The Nightingale	4	Fiction
24	The Great Gatsby	3	Fiction
25	The Road	3	Fiction
26	Wuthering Heights	4	Non-Fiction