Fine Calculation: Calculate the total fines owed by each member, considering overdue books and a daily fine rate (e.g., $0.25 per day).

```sql
-- Test Query 1
SELECT
    u.Name AS Member_Name,
    SUM(DATEDIFF(CURDATE(), b.Due_Date) * 0.25) AS Total_Fines
FROM
    Borrow b
        JOIN Transaction t ON b.Transaction_ID = t.Transaction_ID
        JOIN Performs p ON t.Transaction_ID = p.Transaction_ID
        JOIN Member m ON p.User_ID = m.User_ID
        JOIN User u ON m.User_ID = u.User_ID
WHERE
    b.Due_Date < CURDATE() AND b.Return_Date IS NULL
GROUP BY
    m.User_ID, u.Name;
```

Output | Test Query 1

	Member_Name	Total_Fines
1	Alice Smith	29.75
2	Bob Johnson	29.75
3	Charlie Brown	37.25
4	Daisy Adams	36.00
5	Eve Turner	44.75
6	Frank Davis	1.50
7	Grace Lee	2.25
8	Hannah Martin	1.25
9	Ian Parker	0.25

Explanation of Query:

This query joins Borrow, Transaction, Performs, Member, and User to find the users who are members and have borrowed books from the library. Then, filters by borrows where the due date is past the current date and the return date is NULL. Next, these results are grouped by the User_ID and Name. Finally the total fines are calculated by multiplying .25 by the total days past the due date of each borrow for each user.

Book Availability: Display a list of all available books (not currently borrowed) within a specific genre.

```sql
-- Test Query 2
SELECT
    b.Title,
    g.Genre_Name,
    i.Availability_Status
FROM
    Book b
        JOIN Belongs bl ON b.Item_ID = bl.Item_ID
        JOIN Genre g ON bl.Genre_Name = g.Genre_Name
        JOIN Item i ON b.Item_ID = i.Item_ID
WHERE
    i.Availability_Status = 'Available'
  AND g.Genre_Name = 'Mystery';
```

Output | Test Query 2

	Title	Genre_Name	Availability_Status
1	The Hound of the Baskervilles	Mystery	Available
2	Murder on the Orient Express	Mystery	Available
3	The Girl with the Dragon Tattoo	Mystery	Available
4	Gone Girl	Mystery	Available
5	The Da Vinci Code	Mystery	Available
6	And Then There Were None	Mystery	Available
7	The Big Sleep	Mystery	Available
8	The Silence of the Lambs	Mystery	Available
9	The Maltese Falcon	Mystery	Available
10	The No. 1 Ladies' Detective Agency	Mystery	Available
11	Whispers in the Wind	Mystery	Available

Explanation of Query:

The query selects from the join of the *Book, Belongs, Genre,* and *Item* relations. The 'Title' attribute resides in *Book*, the 'Genre_Name' resides in *Belongs* which has a foreign key from *Genre*, and the 'Availability_Status' resides in *Item*. We select items where the Item.Availability_Status='Available' and Genre.Genre_Name='Mystery'.

The query returns the book title, genre, and availability of all eleven available mystery books

Frequent Borrowers of a Specific Genre: Identify the members who have borrowed the most books in a particular genre (e.g., "Mystery") in the last year.

```
43
44  SELECT
45      u.Name AS Member_Name,
46      COUNT(*) AS Books_Borrowed
47  FROM
48      Borrow b
49          JOIN Transaction t ON b.Transaction_ID = t.Transaction_ID
50          JOIN Performs p ON t.Transaction_ID = p.Transaction_ID
51          JOIN Member m ON p.User_ID = m.User_ID
52          JOIN User u ON m.User_ID = u.User_ID
53          JOIN Transaction_Involves_Item tii ON t.Transaction_ID = tii.Transaction_ID
54          JOIN Belongs bl ON tii.Item_ID = bl.Item_ID
55  WHERE
56      bl.Genre_Name = 'Mystery'
57    AND t.Transaction_Date >= DATE_SUB(CURDATE(), INTERVAL 1 YEAR)
58  GROUP BY
59      m.User_ID, u.Name
60  ORDER BY
61      Books_Borrowed DESC
62  LIMIT 3;
63
```

Output | Result 38

	Member_Name	Books_Borrowed
1	John Watson	14
2	Hercule Poirot	8
3	Nancy Drew	5

Explanation of Query:

This query joins Borrow, Transaction, Performs, Member, User, Transaction_Involves_Item, and Belongs to find all borrows by members along with the genre of each item borrowed. Then, this result is filtered by Mystery books that have been borrowed in the past year. Next, this result is grouped by user ID and Name. Finally the top 3 Member Names and the COUNT of Mystery books they have borrowed are displayed.

Books Due Soon: Generate a report of all books due within the next week, sorted by due date.

```sql
-- Test Query 4
SELECT
    b.Title,
    br.Due_Date
FROM
    Borrow br
        JOIN Transaction t ON br.Transaction_ID = t.Transaction_ID
        JOIN Transaction_Involves_Item tii ON t.Transaction_ID = tii.Transaction_ID
        JOIN Book b ON tii.Item_ID = b.Item_ID
WHERE
    br.Due_Date BETWEEN CURDATE() AND DATE_ADD(CURDATE(), INTERVAL 7 DAY)
ORDER BY
    br.Due_Date;
```

Output | Test Query 4

	Title	Due_Date
1	Enchanted	2024-12-07
2	Fireball	2024-12-08
3	The Maze Runner	2024-12-08
4	Farenheit 451	2024-12-09
5	Pig Pen	2024-12-09
6	Book 0	2024-12-10
7	Harry Potter and the Sorcerer's Stone	2024-12-10
8	The Hunger Games	2024-12-10
9	Twilight	2024-12-11
10	To Kill A Mockingbird	2024-12-11
11	The Great Gatsby	2024-12-12
12	The Fault in Our Stars	2024-12-12
13	1984	2024-12-13

Explanation of Query:

The query selects the title and due date of books from the join of *Borrow, Transaction, Transaction_Involves_Item*, and *Book,* where the due date is in the next week. We start at the *Borrow* relation to get the 'Due_Date' attribute. Then we join *Transaction* to get the 'Transaction_ID', and join *Transaction_Involves_Item* to associate the 'Transaction_ID' with an 'Item_ID'. Next, join *Book* to get the 'Title'. The query returns the title and due date of all books due within the next week, ordered by due date with the closest due dates at the top.

Members with Overdue Books: List all members who currently have at least one overdue book, along with the titles of the overdue books.

```sql
-- Test Query 5
SELECT
    u.Name AS Member_Name,
    b.Title AS Overdue_Book_Title
FROM
    Borrow br
        JOIN Transaction t ON br.Transaction_ID = t.Transaction_ID
        JOIN Performs p ON t.Transaction_ID = p.Transaction_ID
        JOIN Member m ON p.User_ID = m.User_ID
        JOIN User u ON m.User_ID = u.User_ID
        JOIN Transaction_Involves_Item tii ON t.Transaction_ID = tii.Transaction_ID
        JOIN Book b ON tii.Item_ID = b.Item_ID
WHERE
    br.Due_Date < CURDATE()
  AND br.Return_Date IS NULL;
```

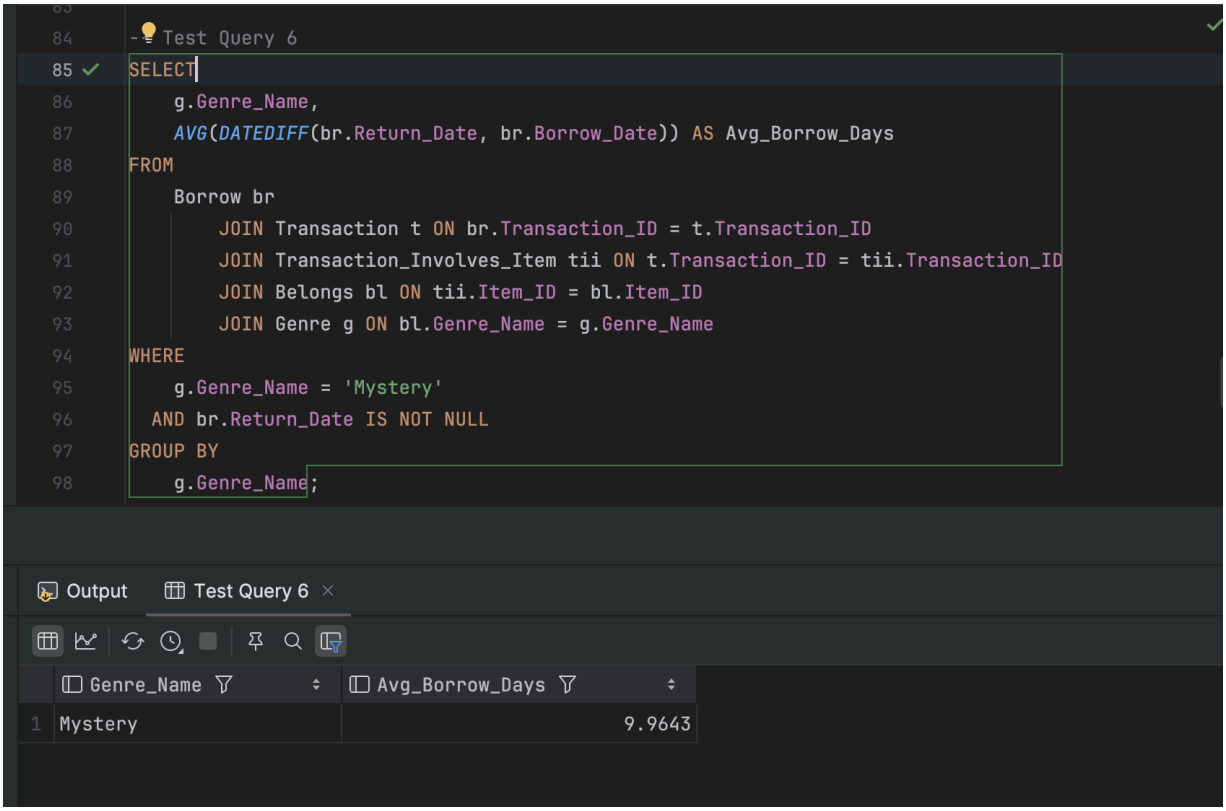
Output | Test Query 5

	Member_Name	Overdue_Book_Title
1	Alice Smith	To Kill a Mockingbird
2	Bob Johnson	1984
3	Charlie Brown	The Great Gatsby
4	Daisy Adams	The Catcher in the Rye
5	Eve Turner	Sapiens: A Brief History of Humankind
6	Alice Smith	Pride and Prejudice
7	Bob Johnson	The Art of War
8	Frank Davis	Jane Eyre
9	Grace Lee	The Selfish Gene
10	Hannah Martin	Guns, Germs, and Steel
11	Ian Parker	The Girl with the Dragon Tattoo

Explanation of Query:

The query needs to return the *User*.Name and *Book*.Title, and it returns tuples where 'Due_Date' is less than the current date and the 'Return_Date' is null, so we need those four attributes. We select from the join of *Borrow, Transaction, Performs, Member, User, Transaction_Involves_Item*, and *Book*. We start at *Borrow* to get the 'Due_Date' and 'Return_Date' attributes. Then, we join *Transaction* to get to *Performs* to get the associated 'User_ID', and join *Member* to get the associated 'Current_Borrows', and join *User* to get the 'Name' of the user. To get the 'Title', we join *Transaction_Involves_Item* to get the associated 'Item_ID', and join *Book* to get the 'Title'. The query returns the user name of all members with at least one overdue book, and the titles of the overdue books.

Average Borrowing Time: Calculate the average number of days members borrow books for a specific genre.



```
-- Test Query 6
SELECT
    g.Genre_Name,
    AVG(DATEDIFF(br.Return_Date, br.Borrow_Date)) AS Avg_Borrow_Days
FROM
    Borrow br
        JOIN Transaction t ON br.Transaction_ID = t.Transaction_ID
        JOIN Transaction_Involves_Item tii ON t.Transaction_ID = tii.Transaction_ID
        JOIN Belongs bl ON tii.Item_ID = bl.Item_ID
        JOIN Genre g ON bl.Genre_Name = g.Genre_Name
WHERE
    g.Genre_Name = 'Mystery'
  AND br.Return_Date IS NOT NULL
GROUP BY
    g.Genre_Name;
```

Output | Test Query 6

	Genre_Name	Avg_Borrow_Days
1	Mystery	9.9643

Explanation of Query:

This query joins Borrow, Transaction, Transaction_Involves_Item, Belongs, and Genre to obtain a relation of all borrows and what genre the item involved in the borrow is. Then, this relation is filtered by Mystery books that have been returned. Next, group all of these results and calculate the average days borrowed. Finally, display the Genre name and average days borrowed.

Most Popular Author in the Last Month: Determine the author whose books have been borrowed the most in the last month.

```sql
-- Test Query 7
SELECT
    a.Name AS Author_Name,
    COUNT(*) AS Books_Borrowed
FROM
    Borrow br
        JOIN Transaction t ON br.Transaction_ID = t.Transaction_ID
        JOIN Transaction_Involves_Item tii ON t.Transaction_ID = tii.Transaction_ID
        JOIN Author_Writes_Item awi ON tii.Item_ID = awi.Item_ID
        JOIN Author a ON awi.Author_ID = a.Author_ID
WHERE
    t.Transaction_Date >= DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
GROUP BY
    a.Author_ID, a.Name
ORDER BY
    Books_Borrowed DESC
LIMIT 1;
```

Output | Test Query 7

	Author_Name	Books_Borrowed
1	J.K. Rowling	14

Explanation of Query:

The query returns the name of the author whose books have been borrowed the most in the past month, and the number of times they've been borrowed. It selects from the join of *Borrow, Transaction, Transaction_Involves_Item, Author_Writes_Item,* and *Author,* where the Transaction Date is within the past month. It groups tuples by Author ID and Author Name, and orders by Books Borrowed in descending order, with a limit of one tuple.

Collection Analysis Report:

```sql
-- Queries for Demo

-- Distribution of Books by Genre
SELECT
    g.Genre_Name AS Genre,
    COUNT(*) AS Book_Count
FROM
    Book b
        JOIN Belongs bl ON b.Item_ID = bl.Item_ID
        JOIN Genre g ON bl.Genre_Name = g.Genre_Name
GROUP BY
    g.Genre_Name
ORDER BY
    Book_Count DESC;
```

Output | Distribution of Books by Genre

	Genre	Book_Count
1	Fiction	20
2	Non-Fiction	20
3	Fantasy	13
4	Mystery	12
5	Romance	12
6	Action	11
7	Western	7
8	Adventure	6
9	Drama	6
10	Thriller	6

1. Distribution of Books by Genre

- **Key Findings**:
    - The library collection is dominated by Fiction and Non-Fiction, each with 20 books.
    - Fantasy (13), Mystery (12), and Romance (12) are the next most represented genres.
    - Adventure, Drama, and Thriller genres have some of the lowest representation (6 books each), followed by Western (7 books).
- **Insights**:
    - There is an opportunity to expand genres such as Adventure, Drama, and Thriller to better balance the collection.
    - Increasing the variety in less-represented genres like Western can appeal to less common readers.

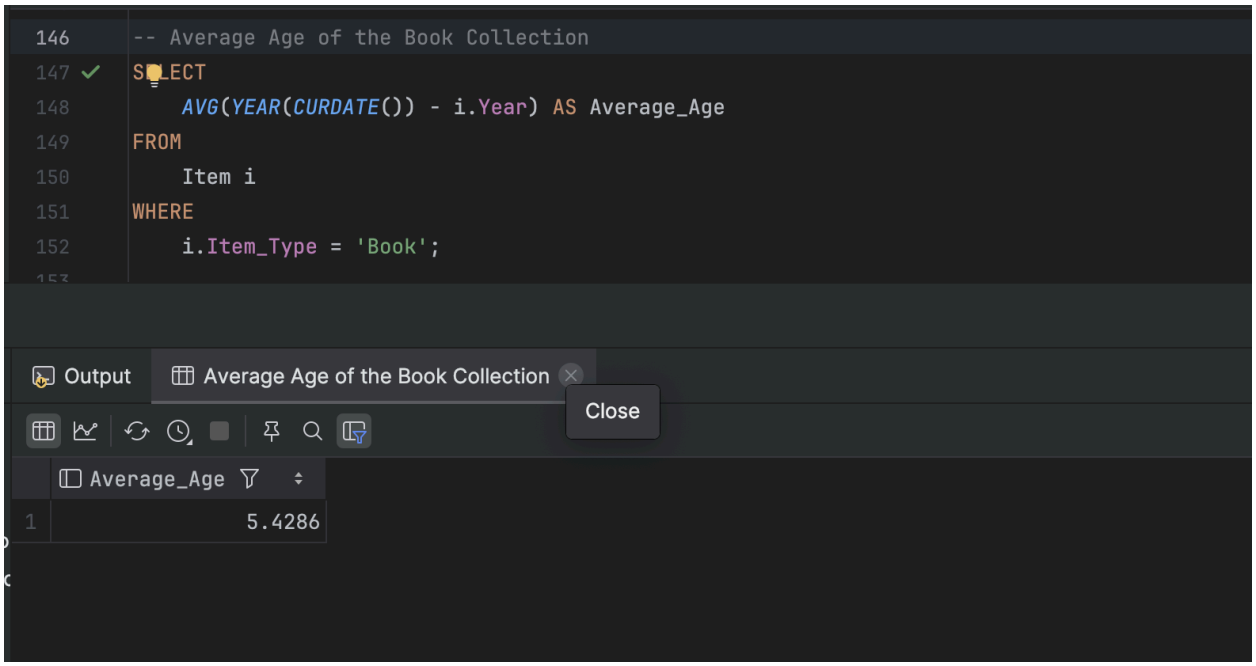
```sql
-- Trends in Acquisition Over the Past 5 Years
SELECT
    i.Year AS Publication_Year,
    COUNT(*) AS Books_Acquired
FROM
    Item i
WHERE
    i.Item_Type = 'Book' AND i.Year >= YEAR(CURDATE()) - 5
GROUP BY
    i.Year
ORDER BY
    i.Year DESC;
```

Output | Trends in Acquisitio...Over the Past 5 Years

	Publication_Year	Books_Acquired
1	2024	7
2	2023	11
3	2022	14
4	2021	18
5	2020	22
6	2019	17

2. Trends in Acquisition Over the Past 5 Years

- **Key Findings**:
    - Book acquisitions have been steady, with a gradual decline over the past five years:
        - 2020: 22 books
        - 2021: 18 books
        - 2022: 14 books
        - 2023: 11 books
        - 2024: 7 books
    - The highest acquisition was in 2020 (22 books).
- **Insights:**
    - The declining trend suggests a need for increased investment in acquisitions to maintain an up-to-date and appealing collection.
    - Consider focusing future acquisitions on under-represented or emerging genres.



3. Average Age of the Book Collection

- **Key Findings**:
  - The average age of the collection is 5.43 years, suggesting the collection is relatively modern.
  - However, older materials may still require review for relevance and condition.
- **Insights**:
  - Perform an inventory check to identify outdated or worn books for removal or replacement.
  - Highlight older classic titles for preservation if they remain popular or culturally significant.

```sql
-- Identify Books with Zero Circulation
SELECT
    b.Title AS Book_Title,
    a.Name AS Author_Name
FROM
    Book b
        LEFT JOIN Transaction_Involves_Item tii ON b.Item_ID = tii.Item_ID
        LEFT JOIN Transaction t ON tii.Transaction_ID = t.Transaction_ID
        LEFT JOIN Author_Writes_Item awi ON b.Item_ID = awi.Item_ID
        LEFT JOIN Author a ON awi.Author_ID = a.Author_ID
WHERE
    t.Transaction_ID IS NULL;
```

Output | Identify Books with Zero Circulation

	Book_Title	Author_Name
1	Legends Rising	Pincas Cowthart
2	Legends Rising	Hermie Corwood
3	Legends Rising	Myrilla Finlow
4	Introduction to Algorithms	<null>
5	Dreams of Infinity	<null>
6	The Forgotten Path	Tucker Ummfrey
7	Beyond the Horizon	Ethelind Callear
8	Beyond the Horizon	Nevin Spottiswoode
9	Beyond the Horizon	Nonie Lemmon
10	Beyond the Horizon	Leann Papaminas
11	Chronicles of the Future	Park Kopje
12	Chronicles of the Future	Correna Kingdom
13	Chronicles of the Future	Alister Livett

4. Books with Zero Circulation

- **Key Findings**:
    - The following titles have no recorded circulation:
        - The Forgotten Path by Tucker Ummfrey
        - Beyond the Horizon by multiple authors (Ethelind Callear, Nevin Spottiswoode, Nonie Lemmon, Leann Papaminas)
        - Chronicles of the Future by multiple authors (Park Kopje, Correna Kingdom, Alister Livett)
        - Revelations in Darkness (no author listed)
        - Whispers in the Wind by multiple authors (Cathy Byas, Jdavie Aspray)
        - Embers of Eternity by Vivian Tockell.
- **Insights**:
    - Titles with zero circulation may indicate misalignment with reader preferences or insufficient promotion.
    - Consider promoting these titles through displays, events, or book clubs.

```sql
-- Analyze Borrowing Patterns
SELECT
    g.Genre_Name AS Genre,
    COUNT(*) AS Borrowed_Count
FROM
    Borrow br
        JOIN Transaction t ON br.Transaction_ID = t.Transaction_ID
        JOIN Transaction_Involves_Item tii ON t.Transaction_ID = tii.Transaction_ID
        JOIN Belongs bl ON tii.Item_ID = bl.Item_ID
        JOIN Genre g ON bl.Genre_Name = g.Genre_Name
WHERE
    t.Transaction_Date >= DATE_SUB(CURDATE(), INTERVAL 1 YEAR)
GROUP BY
    g.Genre_Name
ORDER BY
    Borrowed_Count DESC;
```

Output | Analyze Borrowing Patterns

	Genre	Borrowed_Count
1	Mystery	27

5. Borrowing Patterns by Genre

- **Key Findings**:
    - Mystery books have the highest borrowing rate in the past year (27 instances).
    - Other popular genres for borrowing were not specified but could be inferred from circulation trends.
- **Insights**:
    - Expanding the Mystery genre could cater to high reader demand.
    - Further analysis of borrowing trends across other genres may reveal additional patterns.

```sql
-- Under-represented Genres or Authors (Low Borrow Counts)
SELECT
    g.Genre_Name AS Genre,
    COUNT(*) AS Borrowed_Count
FROM
    Belongs bl
        JOIN Genre g ON bl.Genre_Name = g.Genre_Name
        LEFT JOIN Transaction_Involves_Item tii ON bl.Item_ID = tii.Item_ID
        LEFT JOIN Borrow br ON tii.Transaction_ID = br.Transaction_ID
WHERE
    br.Transaction_ID IS NULL
GROUP BY
    g.Genre_Name
ORDER BY
    Borrowed_Count ASC;
```

Output | Under-represented Ge...s (Low Borrow Counts)

	Genre	Borrowed_Count
1	Horror	1
2	Science Fiction	2
3	Historical	3
4	Comedy	4
5	Mystery	5
6	Thriller	8
7	Adventure	9
8	Drama	9

6. Under-represented Genres

- **Key Findings**:
    - Genres with low borrow counts include:
        - Horror (1 borrow)
        - Science Fiction (2 borrows)
        - Historical (3 borrows)
        - Comedy (4 borrows).
- **Insights**:
    - Promote under-borrowed genres through targeted marketing or curated displays.
    - Consider reducing acquisitions in these genres unless specific demand arises.

```sql
-- Under-represented Genres or Authors (Low Borrow Counts)
SELECT
    a.Name AS Author,
    COUNT(br.Transaction_ID) AS Borrowed_Count
FROM
    Author a
        LEFT JOIN Author_Writes_Item awi ON a.Author_ID = awi.Author_ID
        LEFT JOIN Transaction_Involves_Item tii ON awi.Item_ID = tii.Item_ID
        LEFT JOIN Borrow br ON tii.Transaction_ID = br.Transaction_ID
GROUP BY
    a.Author_ID, a.Name
ORDER BY
    Borrowed_Count ASC;
```

Output | Under-represented Ge...s (Low Borrow Counts)

	Author	Borrowed_Count
1	Doti Matts	0
2	Tucker Ummfrey	0
3	Pincas Cowthart	0
4	Mike Mouse	0
5	Davita Pulteneye	0
6	Fidelity Minucci	0
7	Corney Boat	0
8	Valentijn Hambleton	0
9	Harlan Connachan	0
10	Ethelind Callear	0
11	Elisa Iskowicz	0

7. Under-represented Authors

- **Key Findings**:
    - Authors with zero borrows include:
        - Doti Matts
        - Tucker Ummfrey
        - Pincas Cowthart
        - Mike Mouse
        - Davita Pulteneye
        - Fidelity Minucci
        - and more.
- **Insights**:
    - Evaluate whether these authors align with reader interests or the library's acquisition strategy.
    - Conduct a user survey to determine interest in these authors or their genres.

Recommendations for Collection Development

1. **Expand:**
    - Focus on under-represented genres such as Adventure, Drama, Western, and Thriller.
    - Increase the collection of popular genres like Mystery to align with borrowing trends.
2. **Refresh:**
    - Identify outdated or rarely borrowed materials to get rid of or replace.
    - Prioritize refreshing genres with high user demand.
3. **Promote:**
    - Market low-circulation titles and genres through events, recommendations, or digital channels.
    - Host book talks or discussions to generate interest in lesser-known works.
4. **Acquisition Strategy**:
    - Maintain steady acquisitions, focusing on contemporary and popular titles.
    - Balance acquisitions to address gaps in the collection and appeal to diverse audiences.