Expliantion of querie 5

- `FROM Book b` → start with the Book table
- `JOIN loan l ON b.Book_ID = l.Book_ID` → link each book with its loans to count how many times it was borrowed
- `JOIN review r ON b.Book_ID = r.Book_ID` → link books to their reviews to calculate the average rating
- `GROUP BY b.Book_title, b.Book_ISBN, b.Book_Genre` → group the results **by each book**
- `COUNT(l.loan_id) AS total_loaned` → count how many times the book has been borrowed
- `AVG(r.review_rate) AS avg_review_rate` → calculate the average review rating
- `HAVING COUNT(l.loan_id) >= 3` → only include books borrowed 3 **or more times**

Expliantion of querie 6

1. `FROM loan l` → start with the loans table, because **every borrowing record is stored here**
2. `JOIN Members m ON l.Members_ID = m.Members_ID` → link each loan to the member who borrowed the book
3. `JOIN Book b ON l.Book_ID = b.Book_ID` → get information about the book for that loan
4. `LEFT JOIN review r ON r.Book_ID = b.Book_ID AND r.Members_ID = m.Members_ID` → attach review info **if it exists**, making sure the review is for **the same member and book**

- Using `LEFT JOIN` ensures that **if a member did not leave a review**, the loan will still appear in the result (the review columns will just be NULL).

Expliantion of querie 7

- `COUNT(l.loan_id)` → counts **all loans** in this genre
- `SUM(p.amount)` → adds up all fine amounts collected
- `AVG(CAST(p.amount AS float))` → calculates **average fine per loan**

Casting to float ensures the average is calculated correctly with decimals.