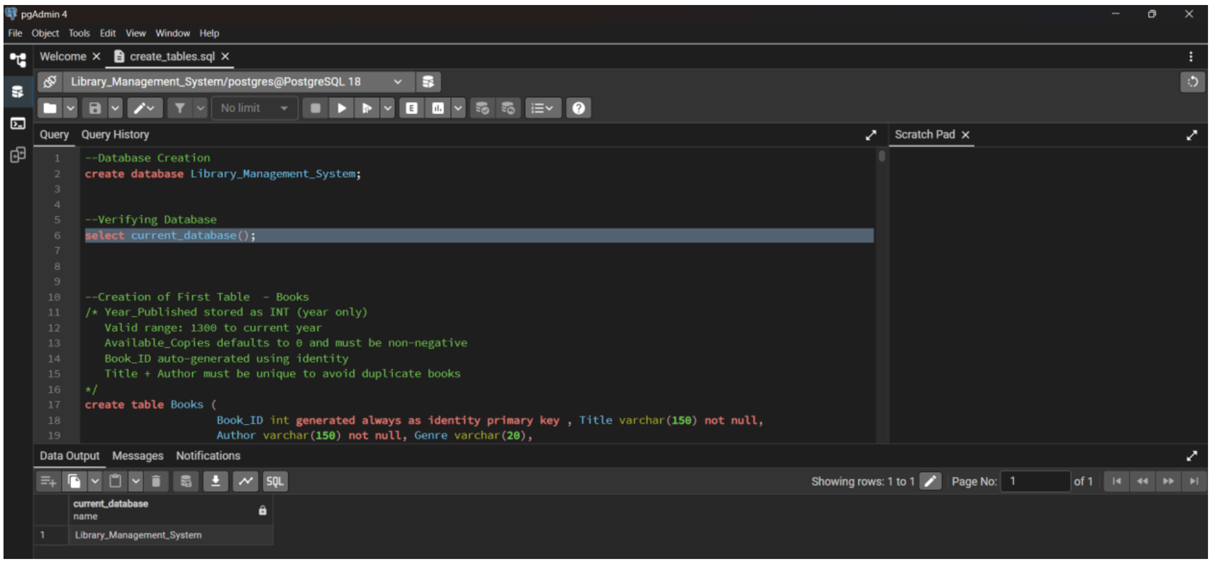
Task 1

Project Title: Library Management System (using SQL)

Project Description:

Design and develop a Library Management System using SQL. The project should involve three tables: Books, Members, BorrowingRecords. The system will manage book inventories, member details, and borrowing transactions.

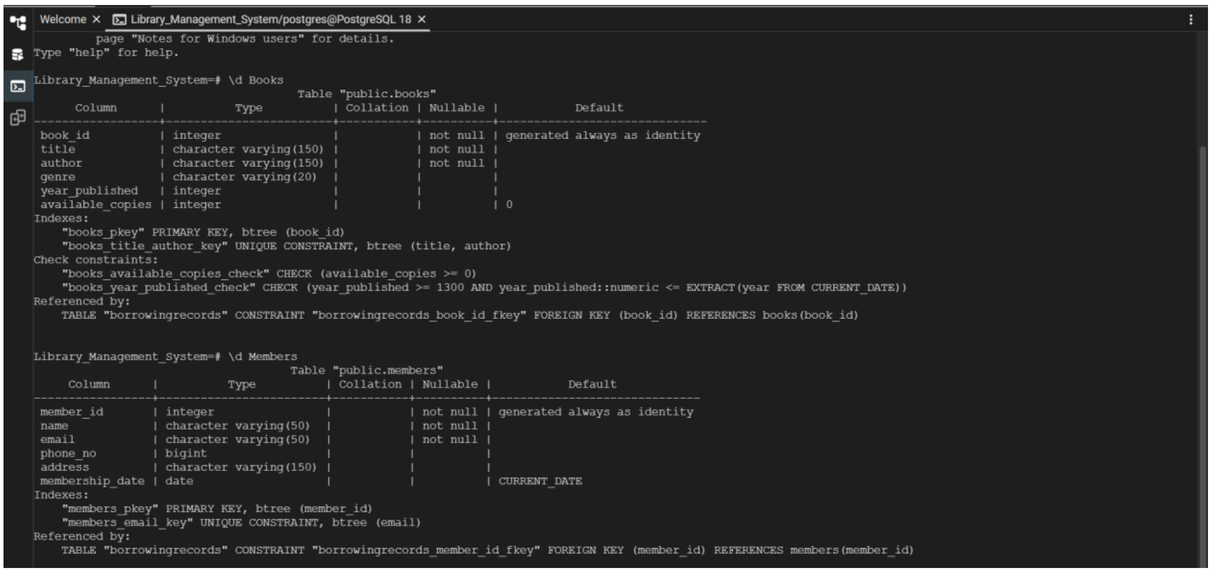
Database created



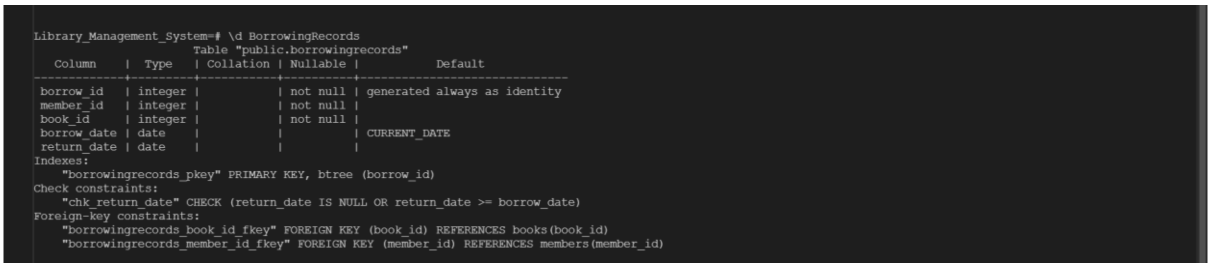
```sql
--Database Creation
create database Library_Management_System;


--Verifying Database
select current_database();



--Creation of First Table  - Books
/* Year_Published stored as INT (year only)
   Valid range: 1300 to current year
   Available_Copies defaults to 0 and must be non-negative
   Book_ID auto-generated using identity
   Title + Author must be unique to avoid duplicate books
*/
create table Books (
                    Book_ID int generated always as identity primary key , Title varchar(150) not null,
                    Author varchar(150) not null, Genre varchar(20),
```

	current_database name
1	Library_Management_System

Tables created



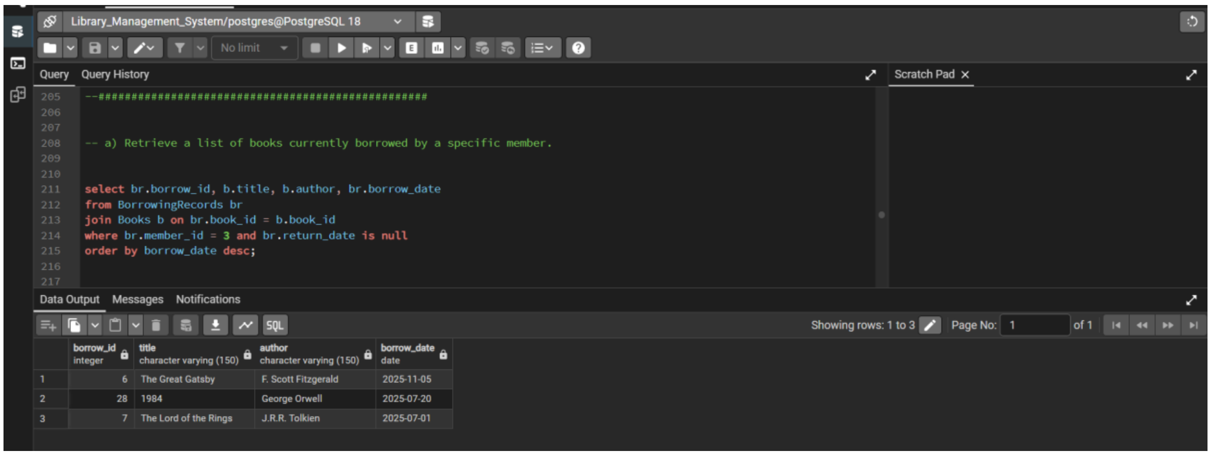
```
        page "Notes for Windows users" for details.
Type "help" for help.

Library_Management_System=# \d Books
                                    Table "public.books"
      Column      |          Type          | Collation | Nullable |           Default
------------------+------------------------+-----------+----------+------------------------------
 book_id          | integer                |           | not null | generated always as identity
 title            | character varying(150) |           | not null |
 author           | character varying(150) |           | not null |
 genre            | character varying(20)  |           |          |
 year_published   | integer                |           |          |
 available_copies | integer                |           |          | 0
Indexes:
    "books_pkey" PRIMARY KEY, btree (book_id)
    "books_title_author_key" UNIQUE CONSTRAINT, btree (title, author)
Check constraints:
    "books_available_copies_check" CHECK (available_copies >= 0)
    "books_year_published_check" CHECK (year_published >= 1300 AND year_published::numeric <= EXTRACT(year FROM CURRENT_DATE))
Referenced by:
    TABLE "borrowingrecords" CONSTRAINT "borrowingrecords_book_id_fkey" FOREIGN KEY (book_id) REFERENCES books(book_id)


Library_Management_System=# \d Members
                                    Table "public.members"
     Column      |          Type          | Collation | Nullable |           Default
-----------------+------------------------+-----------+----------+------------------------------
 member_id       | integer                |           | not null | generated always as identity
 name            | character varying(50)  |           | not null |
 email           | character varying(50)  |           | not null |
 phone_no        | bigint                 |           |          |
 address         | character varying(150) |           |          |
 membership_date | date                   |           |          | CURRENT_DATE
Indexes:
    "members_pkey" PRIMARY KEY, btree (member_id)
    "members_email_key" UNIQUE CONSTRAINT, btree (email)
Referenced by:
    TABLE "borrowingrecords" CONSTRAINT "borrowingrecords_member_id_fkey" FOREIGN KEY (member_id) REFERENCES members(member_id)
```



```
Library_Management_System=# \d BorrowingRecords
                     Table "public.borrowingrecords"
   Column    |  Type   | Collation | Nullable |           Default
-------------+---------+-----------+----------+------------------------------
 borrow_id   | integer |           | not null | generated always as identity
 member_id   | integer |           | not null |
 book_id     | integer |           | not null |
 borrow_date | date    |           |          | CURRENT_DATE
 return_date | date    |           |          |
Indexes:
    "borrowingrecords_pkey" PRIMARY KEY, btree (borrow_id)
Check constraints:
    "chk_return_date" CHECK (return_date IS NULL OR return_date >= borrow_date)
Foreign-key constraints:
    "borrowingrecords_book_id_fkey" FOREIGN KEY (book_id) REFERENCES books(book_id)
    "borrowingrecords_member_id_fkey" FOREIGN KEY (member_id) REFERENCES members(member_id)
```

Information Retrieval:

a) Retrieve a list of books currently borrowed by a specific member.



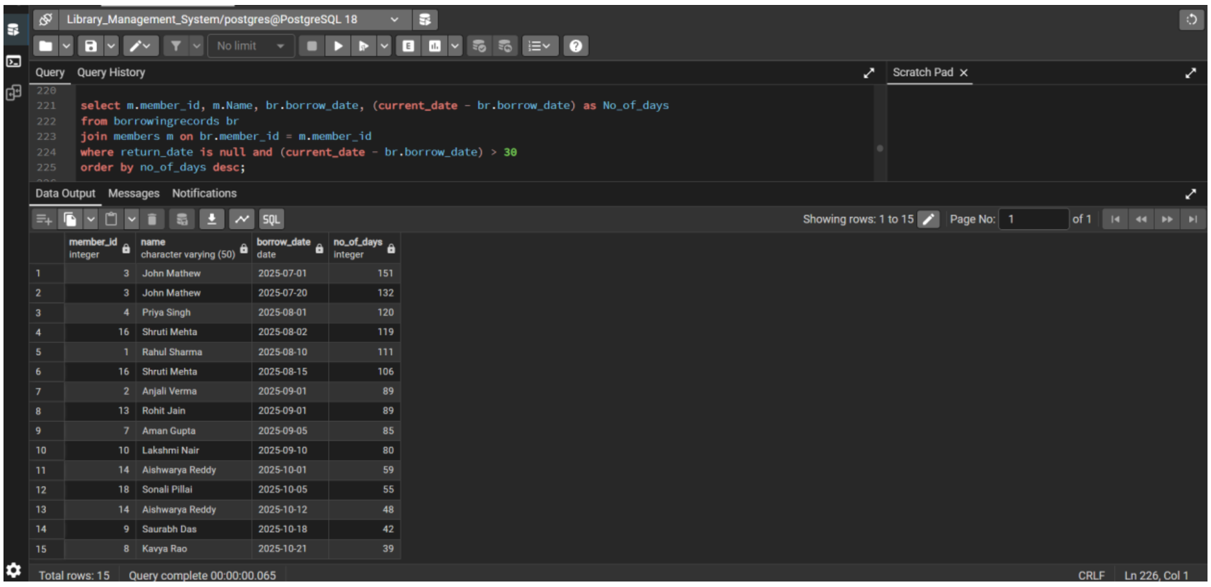
Library_Management_System/postgres@PostgreSQL 18

```sql
--################################################

-- a) Retrieve a list of books currently borrowed by a specific member.

select br.borrow_id, b.title, b.author, br.borrow_date
from BorrowingRecords br
join Books b on br.book_id = b.book_id
where br.member_id = 3 and br.return_date is null
order by borrow_date desc;
```

Showing rows: 1 to 3

	borrow_id integer	title character varying (150)	author character varying (150)	borrow_date date
1	6	The Great Gatsby	F. Scott Fitzgerald	2025-11-05
2	28	1984	George Orwell	2025-07-20
3	7	The Lord of the Rings	J.R.R. Tolkien	2025-07-01

b) Find members who have overdue books (borrowed more than 30 days ago, not returned).



Library_Management_System/postgres@PostgreSQL 18

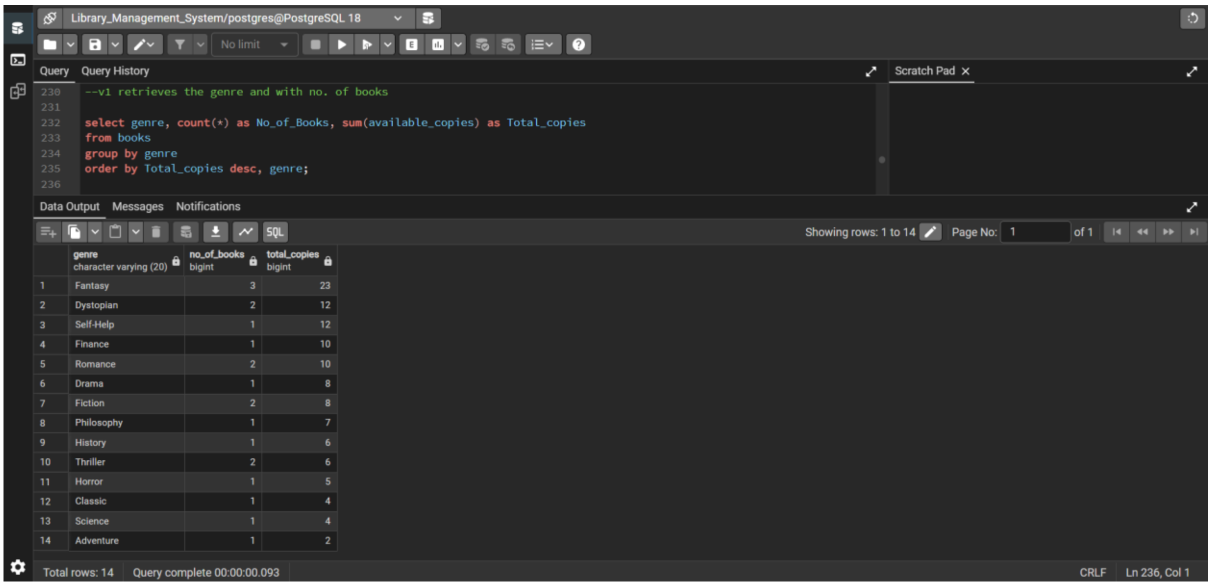
```sql
select m.member_id, m.Name, br.borrow_date, (current_date - br.borrow_date) as No_of_days
from borrowingrecords br
join members m on br.member_id = m.member_id
where return_date is null and (current_date - br.borrow_date) > 30
order by no_of_days desc;
```

Showing rows: 1 to 15

	member_id integer	name character varying (50)	borrow_date date	no_of_days integer
1	3	John Mathew	2025-07-01	151
2	3	John Mathew	2025-07-20	132
3	4	Priya Singh	2025-08-01	120
4	16	Shruti Mehta	2025-08-02	119
5	1	Rahul Sharma	2025-08-10	111
6	16	Shruti Mehta	2025-08-15	106
7	2	Anjali Verma	2025-09-01	89
8	13	Rohit Jain	2025-09-01	89
9	7	Aman Gupta	2025-09-05	85
10	10	Lakshmi Nair	2025-09-10	80
11	14	Aishwarya Reddy	2025-10-01	59
12	18	Sonali Pillai	2025-10-05	55
13	14	Aishwarya Reddy	2025-10-12	48
14	9	Saurabh Das	2025-10-18	42
15	8	Kavya Rao	2025-10-21	39

Total rows: 15 Query complete 00:00:00.065

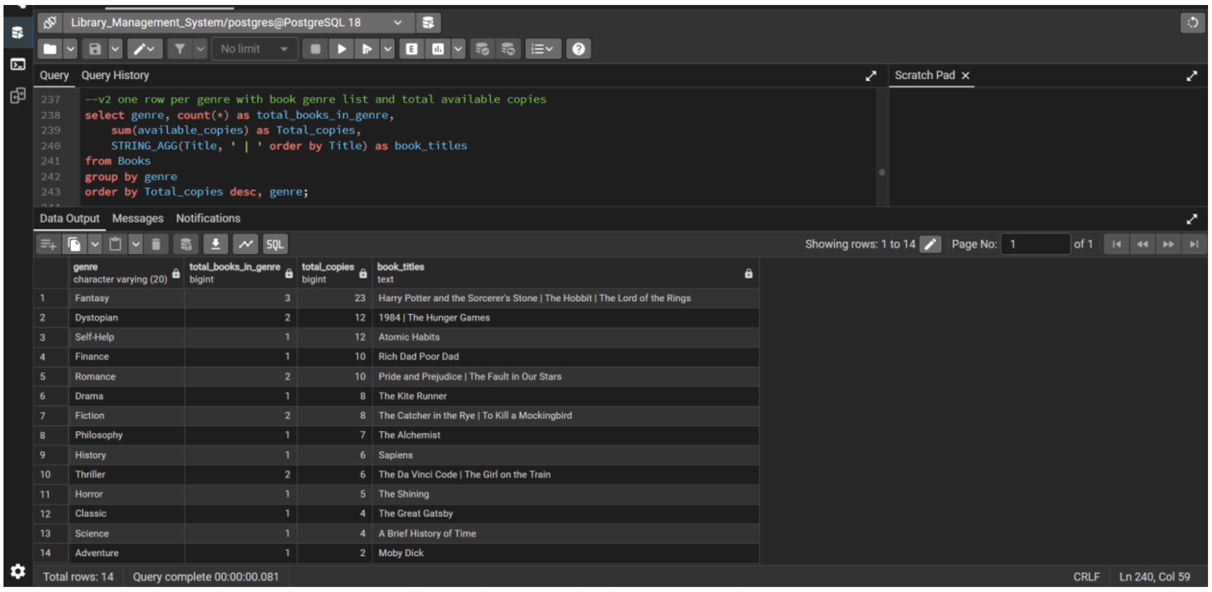
c) **Retrieve books by genre along with the count of available copies.**



```sql
--v1 retrieves the genre and with no. of books

select genre, count(*) as No_of_Books, sum(available_copies) as Total_copies
from books
group by genre
order by Total_copies desc, genre;
```

	genre character varying (20)	no_of_books bigint	total_copies bigint
1	Fantasy	3	23
2	Dystopian	2	12
3	Self-Help	1	12
4	Finance	1	10
5	Romance	2	10
6	Drama	1	8
7	Fiction	2	8
8	Philosophy	1	7
9	History	1	6
10	Thriller	2	6
11	Horror	1	5
12	Classic	1	4
13	Science	1	4
14	Adventure	1	2

Total rows: 14 Query complete 00:00:00.093

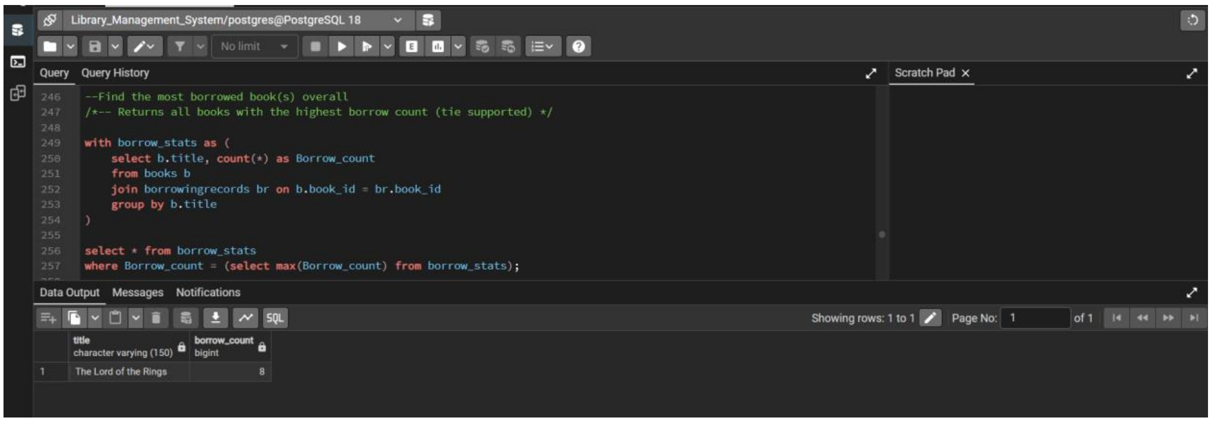


```sql
--v2 one row per genre with book genre list and total available copies
select genre, count(*) as total_books_in_genre,
    sum(available_copies) as Total_copies,
    STRING_AGG(Title, ' | ' order by Title) as book_titles
from Books
group by genre
order by Total_copies desc, genre;
```

	genre character varying (20)	total_books_in_genre bigint	total_copies bigint	book_titles text
1	Fantasy	3	23	Harry Potter and the Sorcerer's Stone | The Hobbit | The Lord of the Rings
2	Dystopian	2	12	1984 | The Hunger Games
3	Self-Help	1	12	Atomic Habits
4	Finance	1	10	Rich Dad Poor Dad
5	Romance	2	10	Pride and Prejudice | The Fault in Our Stars
6	Drama	1	8	The Kite Runner
7	Fiction	2	8	The Catcher in the Rye | To Kill a Mockingbird
8	Philosophy	1	7	The Alchemist
9	History	1	6	Sapiens
10	Thriller	2	6	The Da Vinci Code | The Girl on the Train
11	Horror	1	5	The Shining
12	Classic	1	4	The Great Gatsby
13	Science	1	4	A Brief History of Time
14	Adventure	1	2	Moby Dick

Total rows: 14 Query complete 00:00:00.081

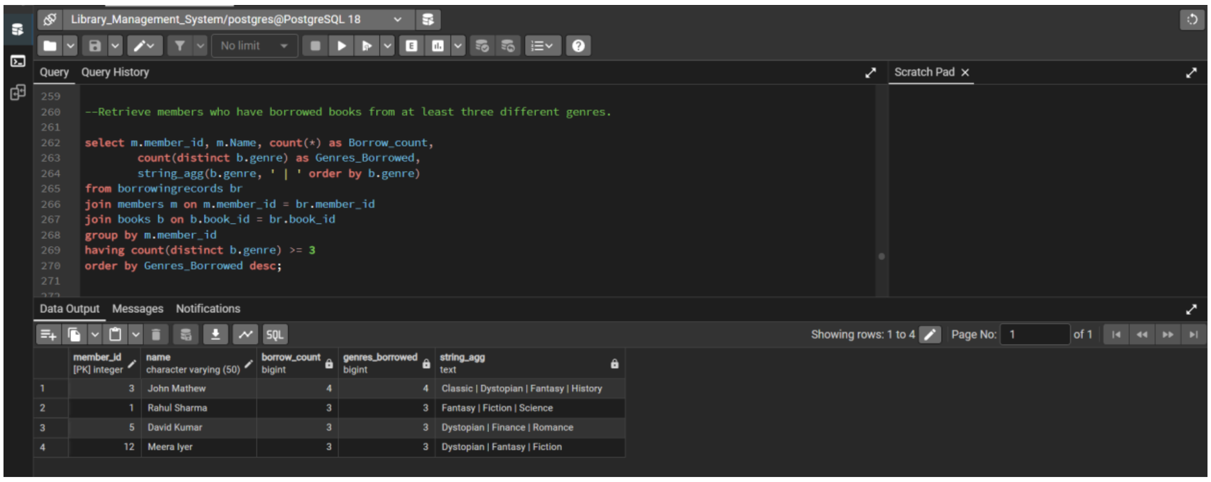
d) **Find the most borrowed book(s) overall.**



```sql
--Find the most borrowed book(s) overall
/*-- Returns all books with the highest borrow count (tie supported) */

with borrow_stats as (
    select b.title, count(*) as Borrow_count
    from books b
    join borrowingrecords br on b.book_id = br.book_id
    group by b.title
)

select * from borrow_stats
where Borrow_count = (select max(Borrow_count) from borrow_stats);
```

	title character varying (150)	borrow_count bigint
1	The Lord of the Rings	8

e) Retrieve members who have borrowed books from at least three different genres.



Library_Management_System/postgres@PostgreSQL 18

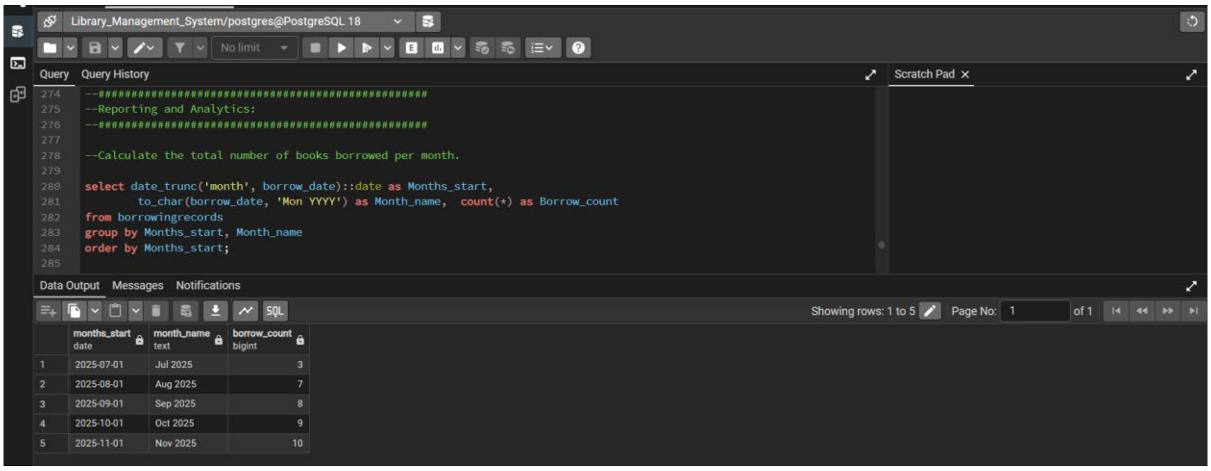
```sql
--Retrieve members who have borrowed books from at least three different genres.

select m.member_id, m.Name, count(*) as Borrow_count,
        count(distinct b.genre) as Genres_Borrowed,
        string_agg(b.genre, ' | ' order by b.genre)
from borrowingrecords br
join members m on m.member_id = br.member_id
join books b on b.book_id = br.book_id
group by m.member_id
having count(distinct b.genre) >= 3
order by Genres_Borrowed desc;
```

Showing rows: 1 to 4 Page No: 1 of 1

	member_id [PK] integer	name character varying (50)	borrow_count bigint	genres_borrowed bigint	string_agg text
1	3	John Mathew	4	4	Classic | Dystopian | Fantasy | History
2	1	Rahul Sharma	3	3	Fantasy | Fiction | Science
3	5	David Kumar	3	3	Dystopian | Finance | Romance
4	12	Meera Iyer	3	3	Dystopian | Fantasy | Fiction

Reporting and Analytics:

a) Calculate the total number of books borrowed per month.



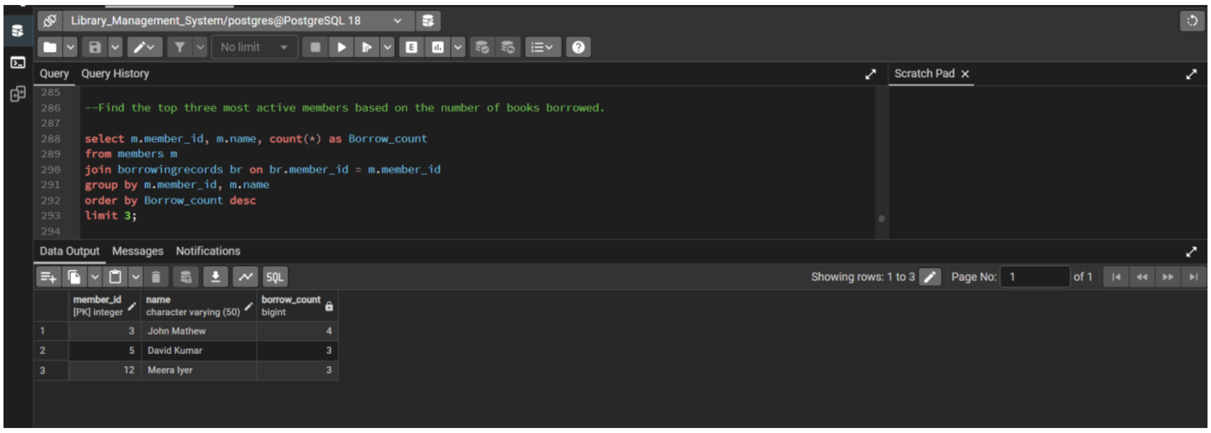
Library_Management_System/postgres@PostgreSQL 18

```sql
--###################################################
--Reporting and Analytics:
--###################################################

--Calculate the total number of books borrowed per month.

select date_trunc('month', borrow_date)::date as Months_start,
        to_char(borrow_date, 'Mon YYYY') as Month_name,  count(*) as Borrow_count
from borrowingrecords
group by Months_start, Month_name
order by Months_start;
```

Showing rows: 1 to 5 Page No: 1 of 1

	months_start date	month_name text	borrow_count bigint
1	2025-07-01	Jul 2025	3
2	2025-08-01	Aug 2025	7
3	2025-09-01	Sep 2025	8
4	2025-10-01	Oct 2025	9
5	2025-11-01	Nov 2025	10

b) Find the top three most active members based on the number of books borrowed.



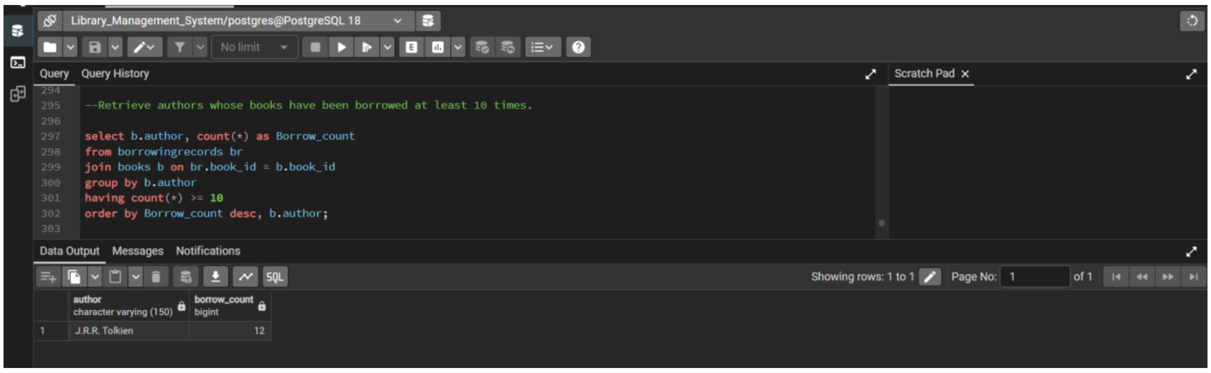
Library_Management_System/postgres@PostgreSQL 18

```sql
--Find the top three most active members based on the number of books borrowed.

select m.member_id, m.name, count(*) as Borrow_count
from members m
join borrowingrecords br on br.member_id = m.member_id
group by m.member_id, m.name
order by Borrow_count desc
limit 3;
```

Showing rows: 1 to 3 Page No: 1 of 1

	member_id [PK] integer	name character varying (50)	borrow_count bigint
1	3	John Mathew	4
2	5	David Kumar	3
3	12	Meera Iyer	3

c) Retrieve authors whose books have been borrowed at least 10 times.



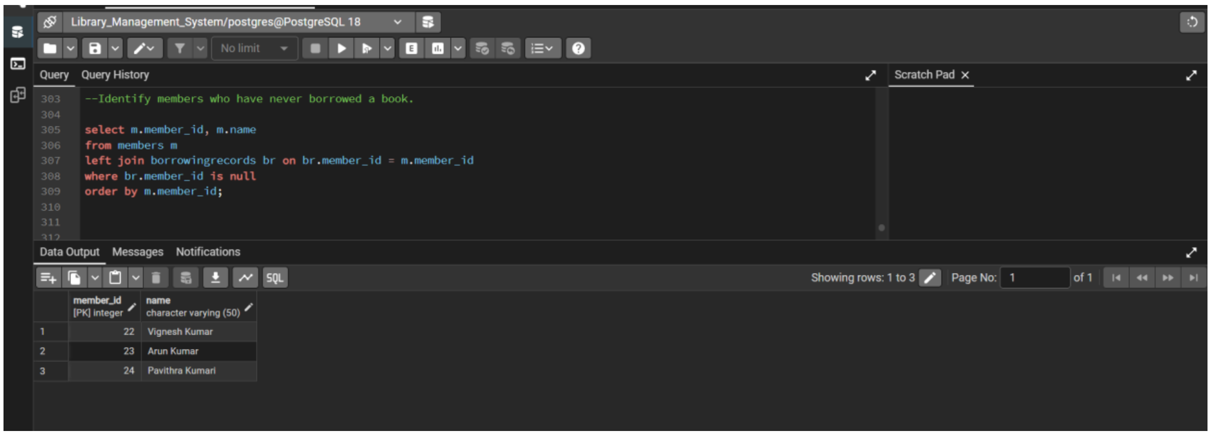
Library_Management_System/postgres@PostgreSQL 18

```
--Retrieve authors whose books have been borrowed at least 10 times.

select b.author, count(*) as Borrow_count
from borrowingrecords br
join books b on br.book_id = b.book_id
group by b.author
having count(*) >= 10
order by Borrow_count desc, b.author;
```

Showing rows: 1 to 1 Page No: 1 of 1

	author character varying (150)	borrow_count bigint
1	J.R.R. Tolkien	12

d) Identify members who have never borrowed a book.



Library_Management_System/postgres@PostgreSQL 18

```
--Identify members who have never borrowed a book.

select m.member_id, m.name
from members m
left join borrowingrecords br on br.member_id = m.member_id
where br.member_id is null
order by m.member_id;
```

Showing rows: 1 to 3 Page No: 1 of 1

	member_id [PK] integer	name character varying (50)
1	22	Vignesh Kumar
2	23	Arun Kumar
3	24	Pavithra Kumari