Library Management System

PROJECT REPORT

Group Members

MAHNDSE23.2F-007	J.G. Sachini Gayanika
MADHNSE23.2F-008	W.G. Ashen Wishwa Geeth Jayarathna
MAHNDSE23.2F-009	W.A.K.Nuwan Darshana
MAHNDSE23.2F-010	G.I.Colombage

ADVANCED DATABASE MANAGEMENT SYSTEM

Assessment NO:04

Higher Diploma in Software Engineering

National Institute of Business Management

Under supervision

Ms. Kaushalya Dissanayake

Consultant/Lecture

DECLARATION

"I certify that this project does not incorporate without acknowledgement, any material previously submitted for a Diploma in any institution and to the best of my knowledge and belief ,it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my project report, if accepted, to be made available for photocopying and for interlibrary loans, and for the title and summary to be made available to outside organizations"

Index no	**Name**	**Signature**
MAHNDSE23.2F-007	J.G. Sachini Gayanika	……..
MAHNDSE23.2F-008	W.G. Ashen Wishwa Geeth Jayarathna	……..
MAHNDSE23.2F-009	W.A.K.Nuwan Darshana	…….
MAHNDSE23.2F-010	G.I.Colombage	…….

Acknowledgement

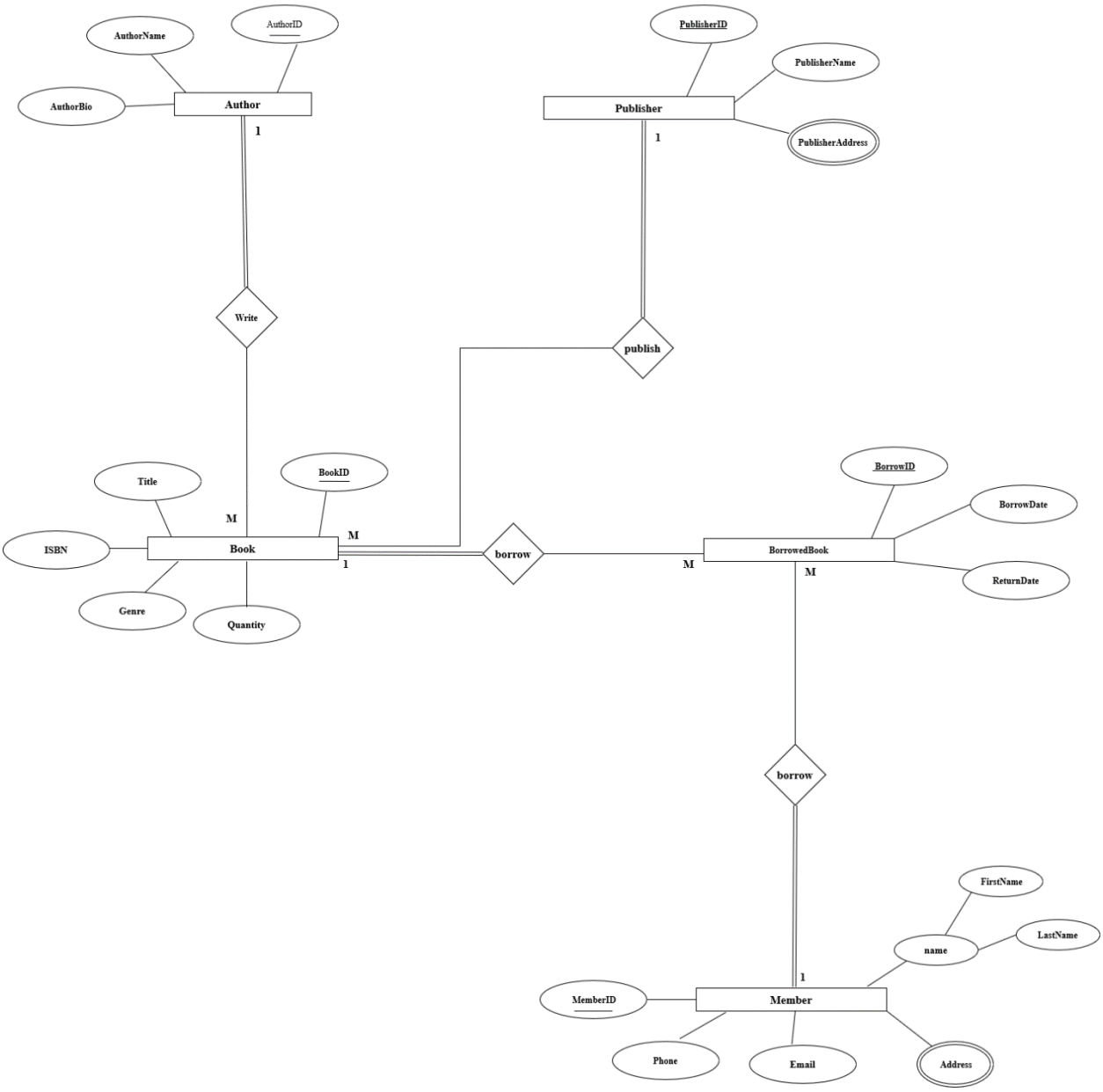
We would like to express our sincere appreciation to everyone who contributed to the successful completion of this assignment. First and foremost, we extend our gratitude to our lecturer, MS Kaushalya Dissanayake for providing us with the opportunity to explore and deepen our knowledge of Advanced database management and PL/SQL programming. We would also like to thank our fellow group members for their collaboration and shared commitment to the project. Their insights and hard work greatly enriched the final deliveries. Additionally, we are thankful for the support and resources provided by our institution, which made this assignment possible. Last but not least, we appreciate the patience and understanding of our friends and family during the intense work on this project. Their encouragement and support were invaluable throughout this journey.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

ER DIAGRAM


Library Managment System
AuthorName
AuthorID
AuthorBio
Author
1
Write
PublisherID
PublisherName
Publisher
1
PublisherAddress
publish
Title
BookID
M
ISBN
Book
M
1
Genre
Quantity
borrow
BorrowID
BorrowDate
BorrowedBook
M
M
ReturnDate
borrow
FirstName
LastName
name
1
MemberID
Member
Phone
Email
Address

Chapter 1
Introduction

A library management system is a software solution designed to help librarians and members to efficiently manage the operations of a library.

The primary goal of a library management system is to streamline the process of organizing and accessing library resources, including books, periodicals, multimedia materials, and digital resources. This involves tasks such as maintaining an updated inventory of library holdings, facilitating patron registration and borrowing, managing overdue fines and renewals, and generating reports for administrative purposes.

1.1 Background

The Library Management System project aims to provide an efficient and organized solution for managing a library's resources and member interactions. In this comprehensive database design, we have created tables for authors, books, library members, and borrowed books, each meticulously structured with primary keys, foreign keys, check constraints, and other integrity features. This database allows for the management of library assets, tracking of member activities, and seamless author and book data retrieval. The SQL Server-based implementation showcases various database operations, including data query language commands.

1.2 Problem Statement

The objective of this project is to design and develop a comprehensive Library Management System that streamlines and automates various management processes within a library. The system aims to improve overall efficiency, communication, and data management while providing a user-friendly system for librarian and members Control waste time unnecessarily.

- Unnecessary waste of money.
- Prevention of Data Loosing.
- Weaknesses in file usage

Aim and Objectives

This project's major goal is to provide a comprehensive and user-friendly system that makes it easier for librarian and members to manage and communicate effectively. The system tries to address issues with resource allocation, data management, and communication breakdowns the library.

- **In short,**
- Organize library resources efficiently.
- Facilitate easy borrowing and returning of items.
- Provide user-friendly interface.
- Send automated notifications for due dates.
- Ensure data security.

CHAPTER 2

Design and Result

Authors Table

Table 1:Authors Table

NAME	DATA TYPE	CONSTRAINT
AuthorID	INT	PRIMARY KEY
AuthorName	NVARCHAR	NOT NULL
AuthorBio	NVARCHAR	

Publishers Table

Table 2:Publishers Table

NAME	DATA TYPE	CONSTRAINT
PublisherID	INT	PRIMARY KEY
PublisherName	NVARCHAR	NOT NULL
PublisherAddress	NVARCHAR	

Books Table

Table 3:Books Table

NAME	DATA TYPE	CONSTRAINT
BookID	INT	PRIMARY KEY
Title	NVARCHAR	NOT NULL
AuthorID	INT	
PublisherID	INT	
ISBN	NVARCHAR	
Genre	NVARCHAR	
Quantity	INT	NOT NULL

Members Table

Table 4:Members Table

NAME	DATA TYPE	CONSTRAINT
MemberID	INT	PRIMARY KEY
FirstName	NVARCHAR	NOT NULL
LastName	NVARCHAR	NOT NULL
Address	NVARCHAR	
Phone	NVARCHAR	
Email	NVARCHAR	
AuthorID	INT	FOREIGN KEY
PublisherID	INT	FOREIGN KEY

BorrowedBooks Table

Table 5:BorrowedBooks Table

NAME	DATA TYPE	CONSTRAINT
BorrowID	INT	PRIMARY KEY
MemberID	INT	NOT NULL
BookID	INT	NOT NULL
BorrowDate	DATE	
ReturnDate	DATE	
MemberID	INT	FOREIGN KEY
BookID	INT	FOREIGN KEY

1) The following features should be satisfied in each table.
    - Primary key and foreign key constraints
    - Check Constraint
    - Not Null Constraint
    - Auto-increment sequence for one of the tables

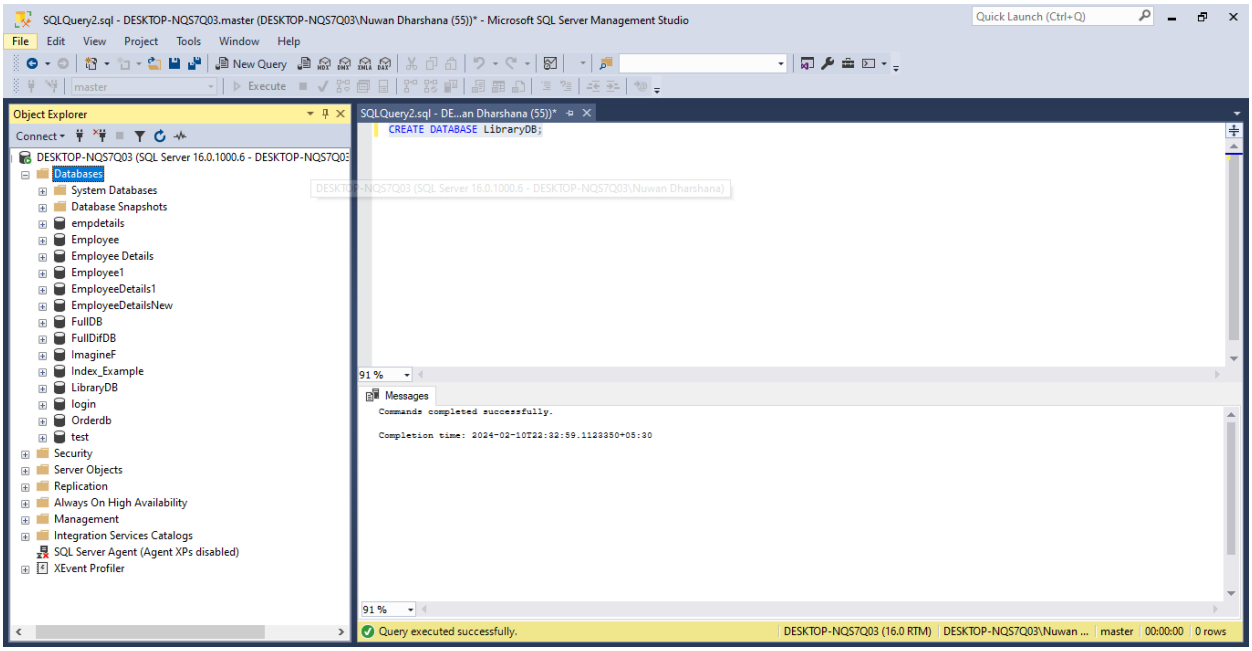


Figure 1:Create Database

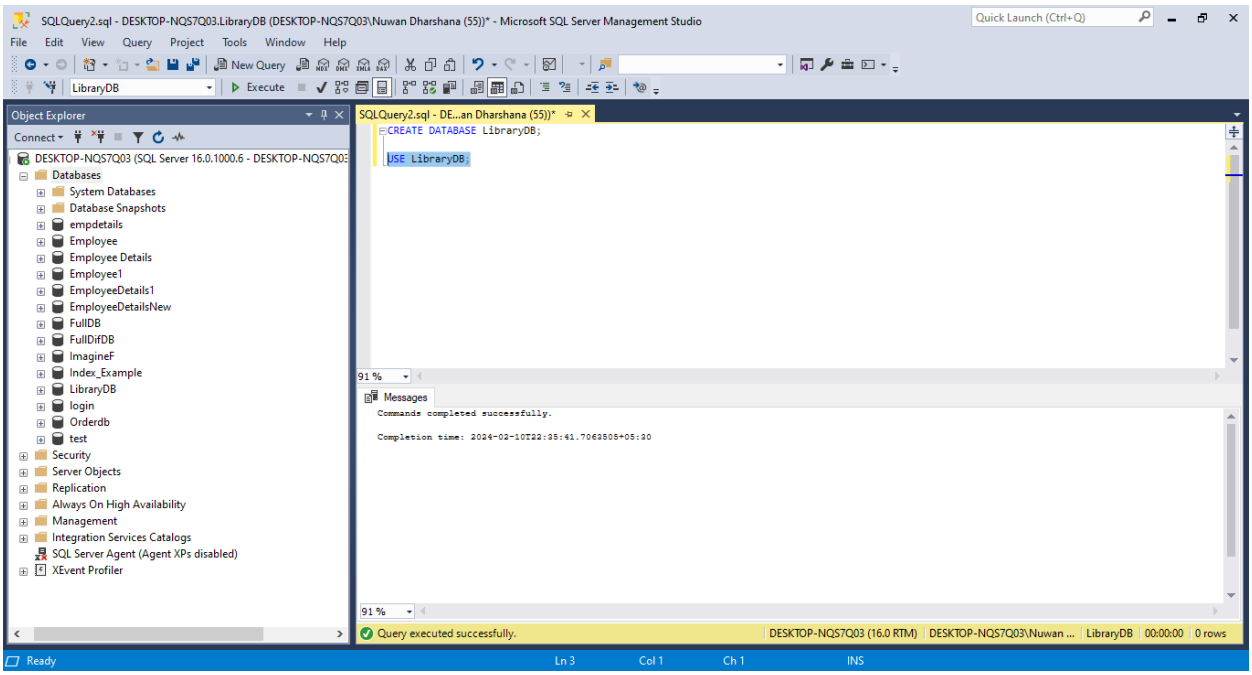


Figure 2:Use Database

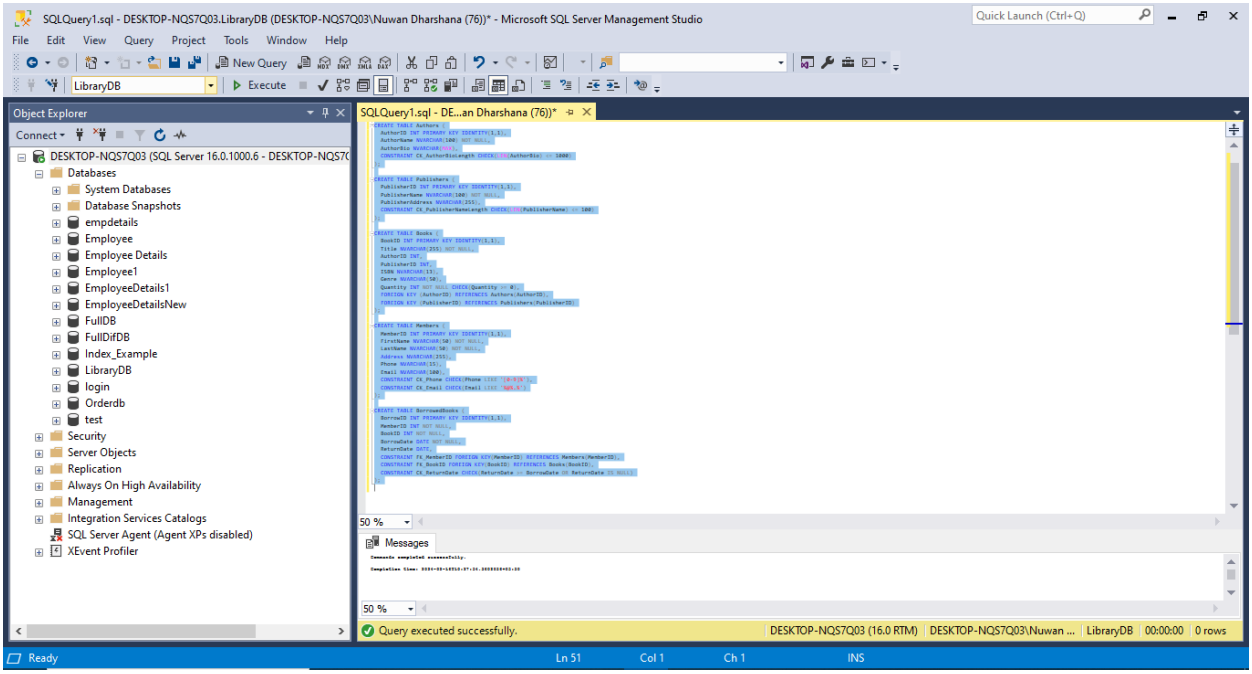


Figure 3:Create Tables

2) All the tables must have a minimum of 10 records associated with them.

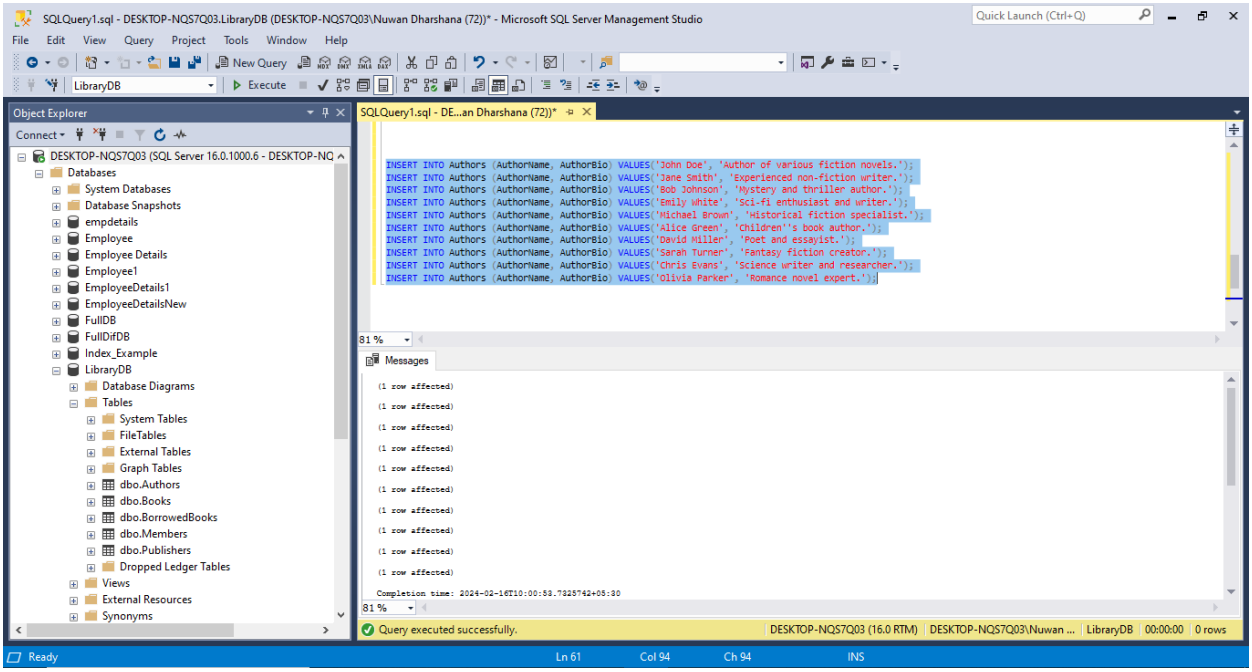


Figure 4:Insert Data

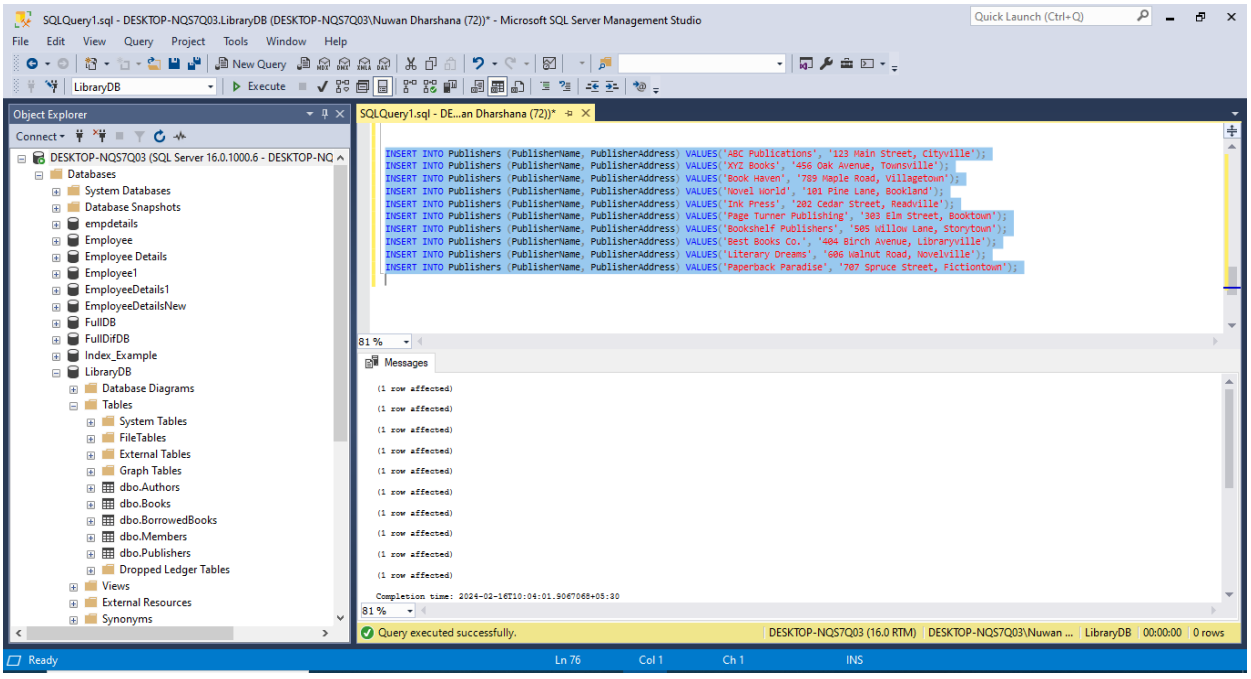


Figure 5:Insert Data

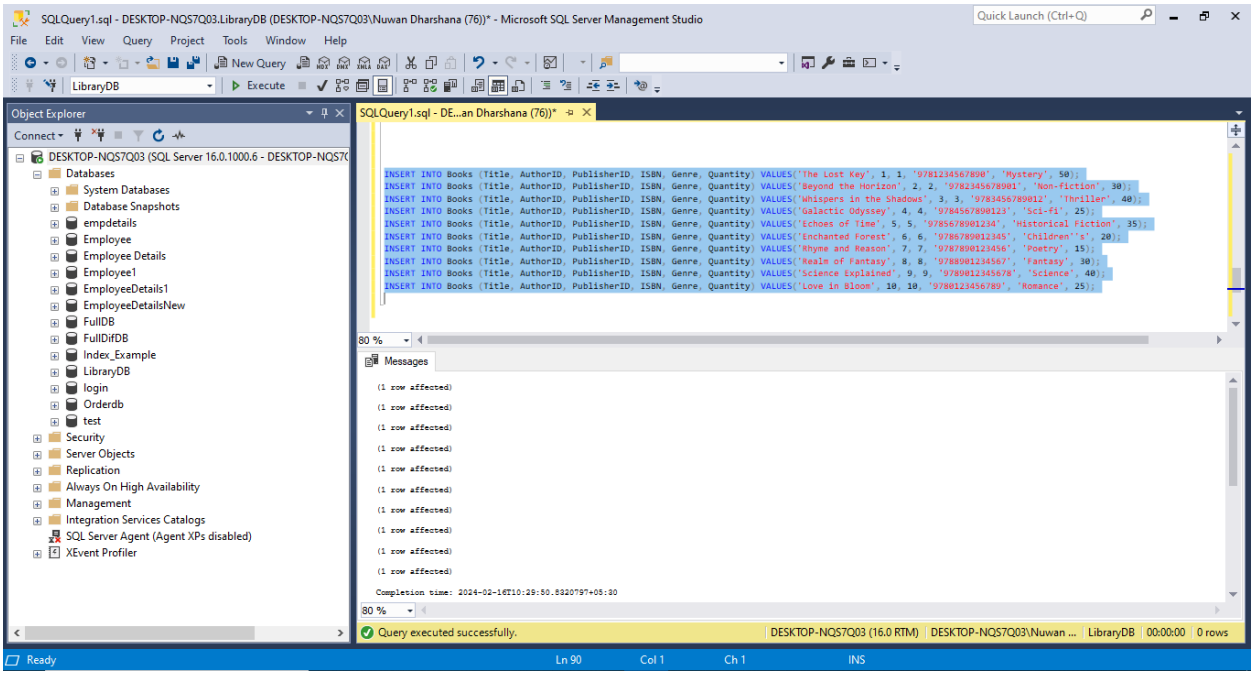


Figure 6:Insert Data

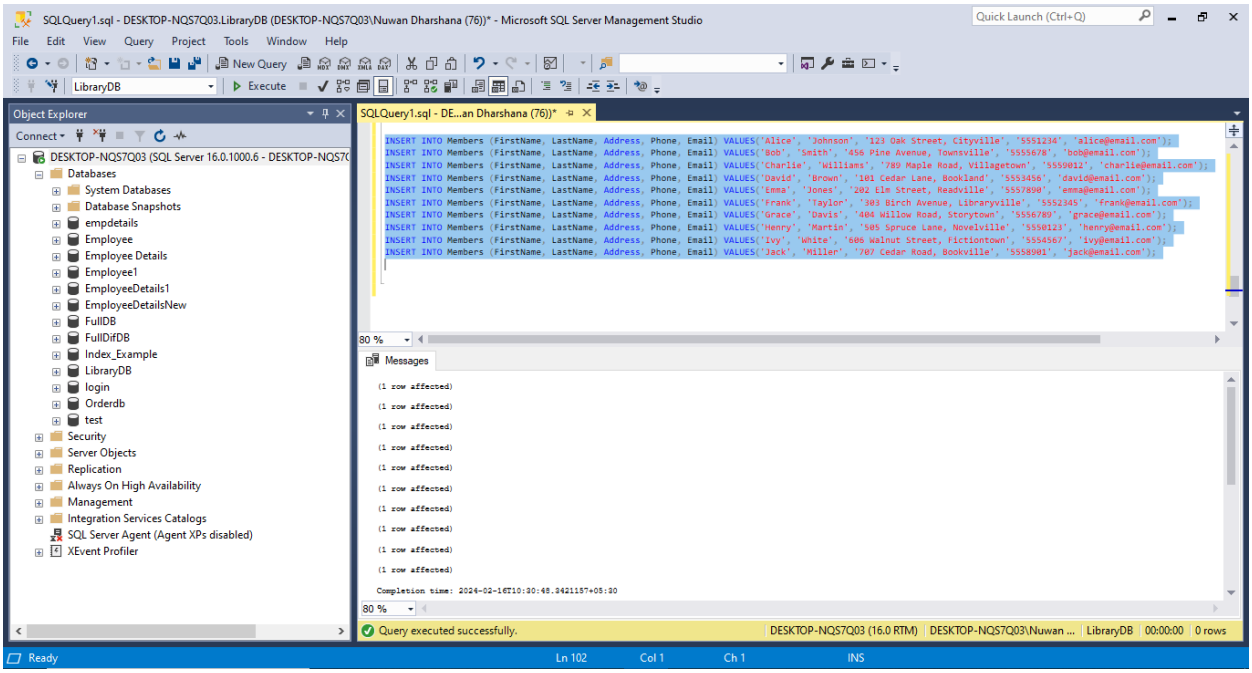


Figure 7:Insert Data

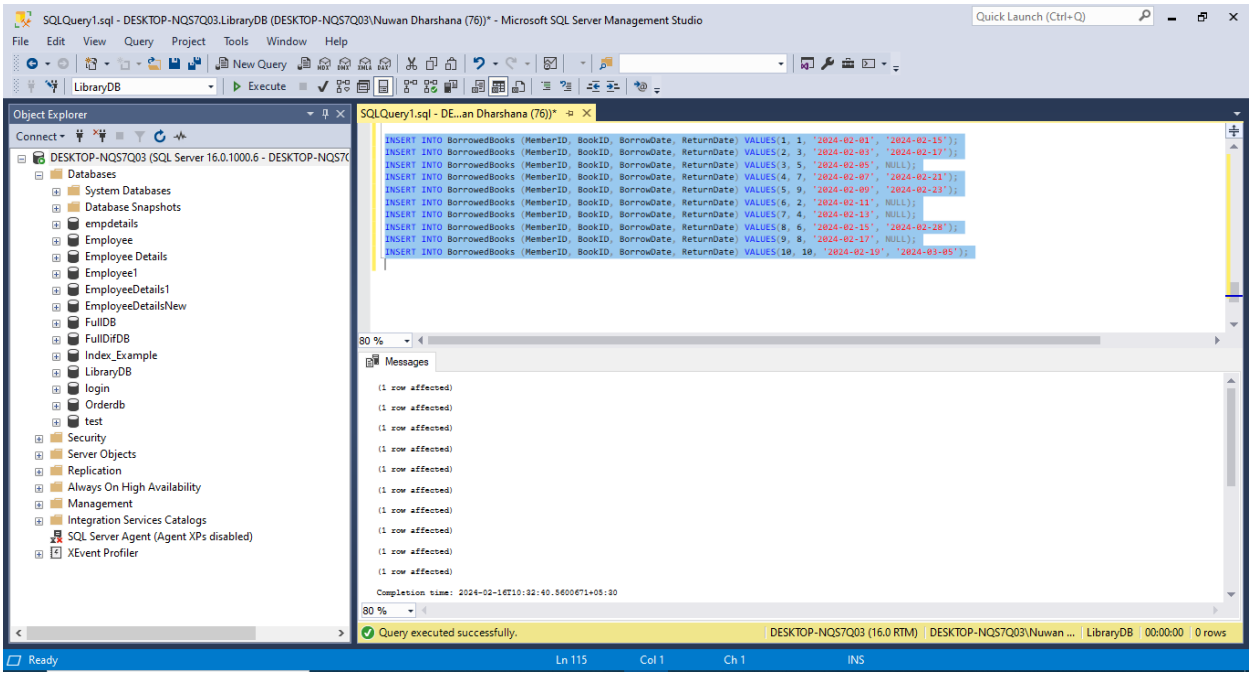


Figure 8:Insert Data

3) Write twenty DQL [Data Query Language] Command with covering following criteria. • Where • Group by • Having • Order by

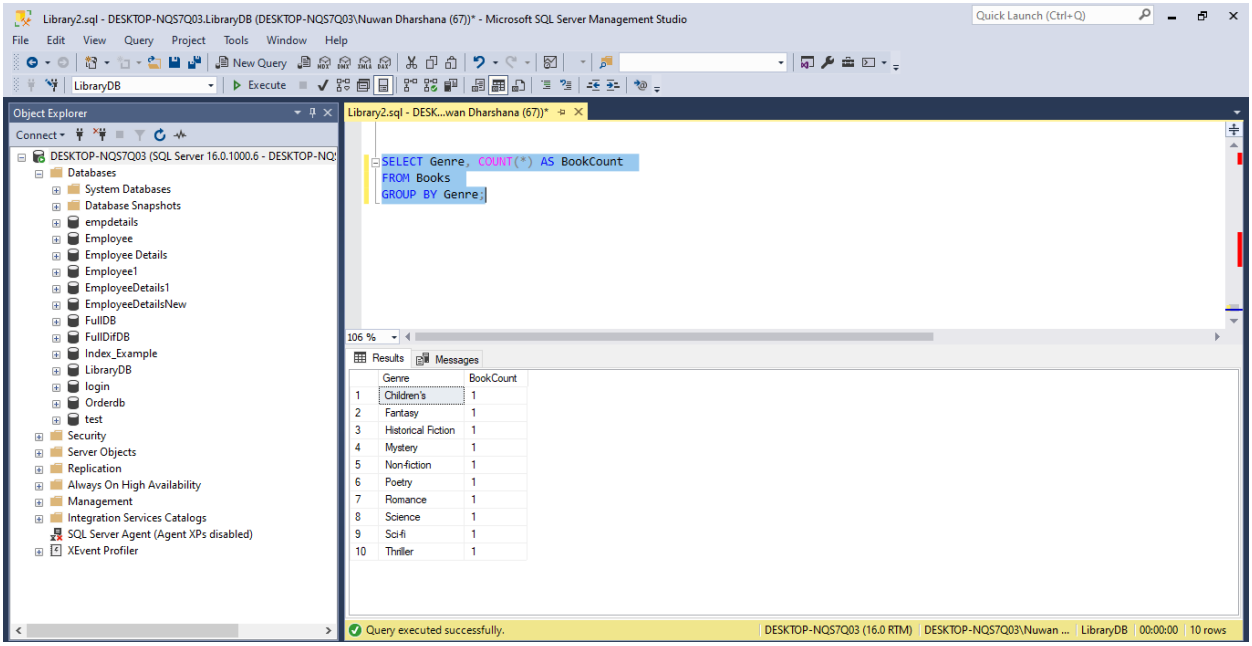


Figure 9:DQL Query

Count the number of books in each genre from the "Books" table, displaying the genre and the count of books for each genre.

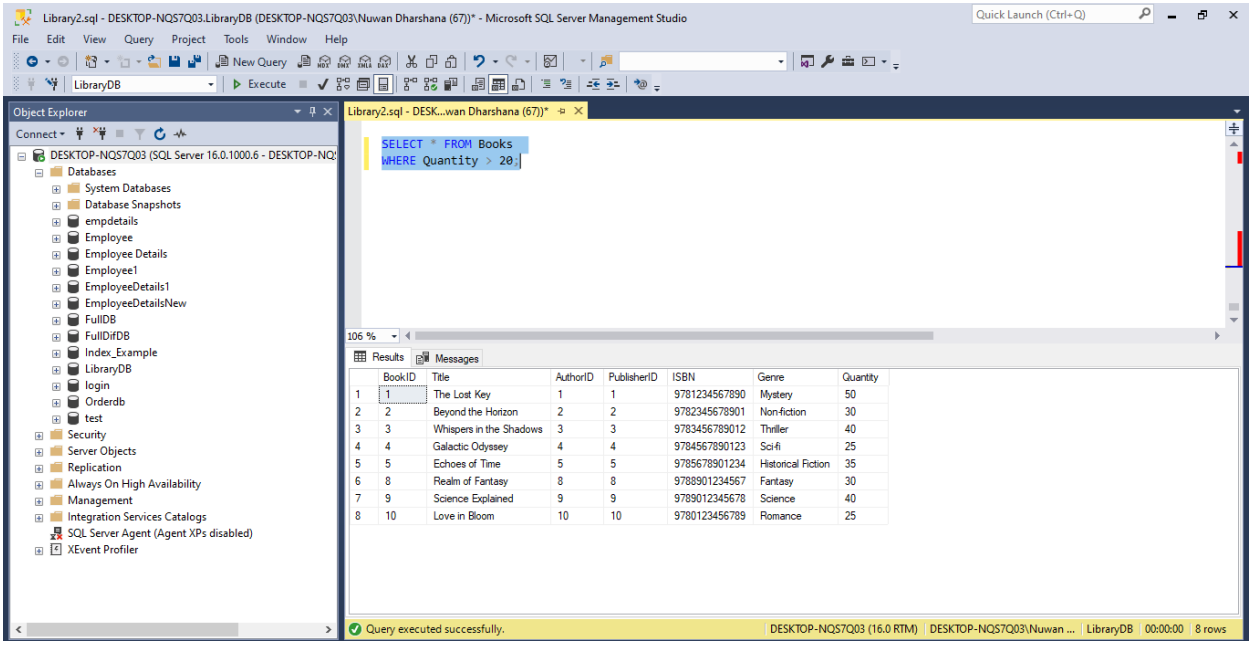


Figure 10:DQL Query

Retrieve all columns from the "Books" table where the quantity of books is greater than 20.

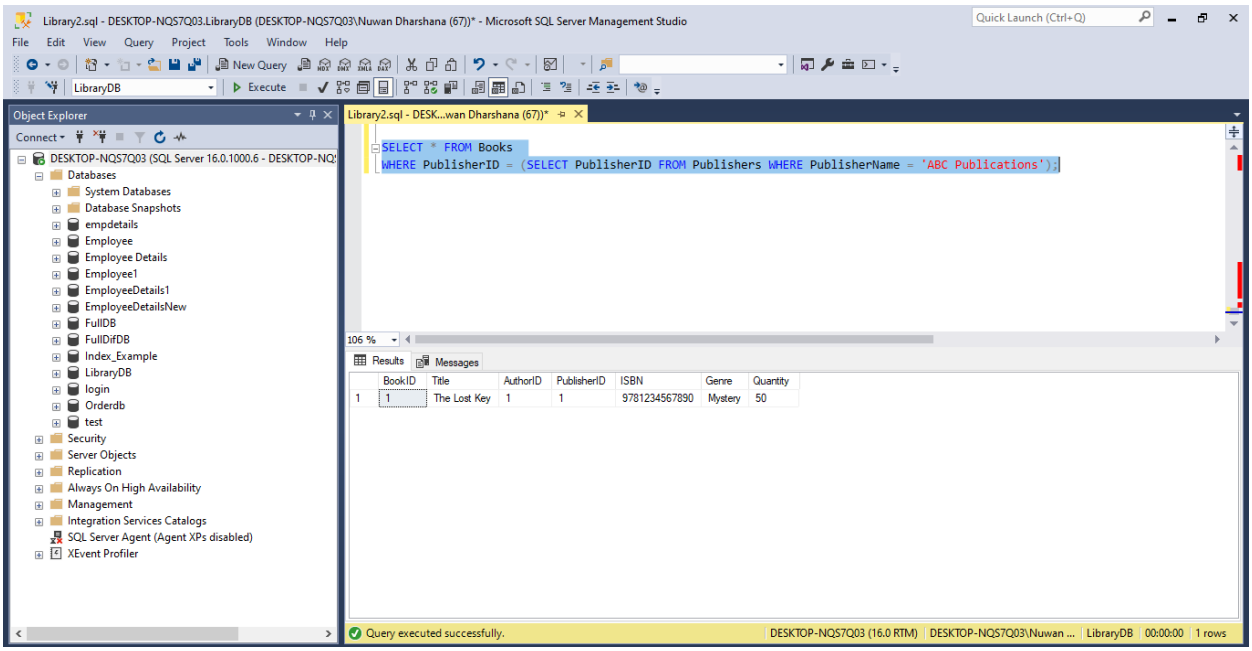


Figure 11:DQL Query

Retrieve all columns from the "Books" table where the publisher ID matches the ID of the publisher with the name 'ABC Publications' from the "Publishers" table.

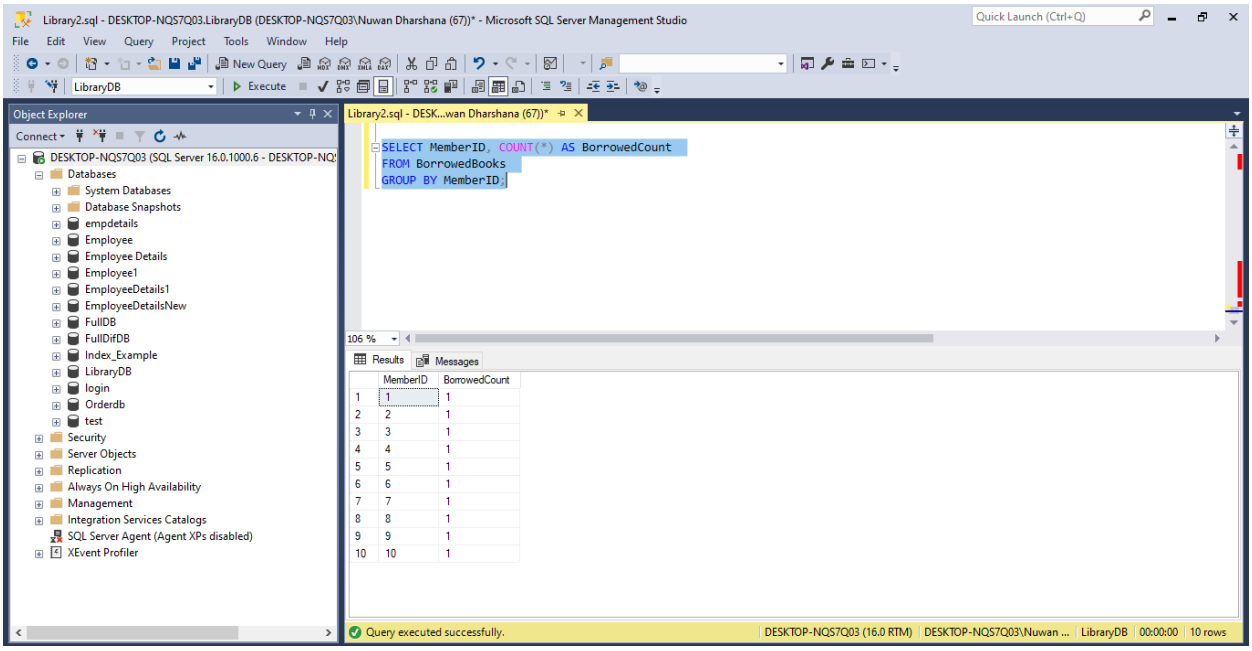


Figure 12:DQL Query

Count the number of books borrowed by each member from the "BorrowedBooks" table, displaying the member ID and the count of books borrowed for each member.

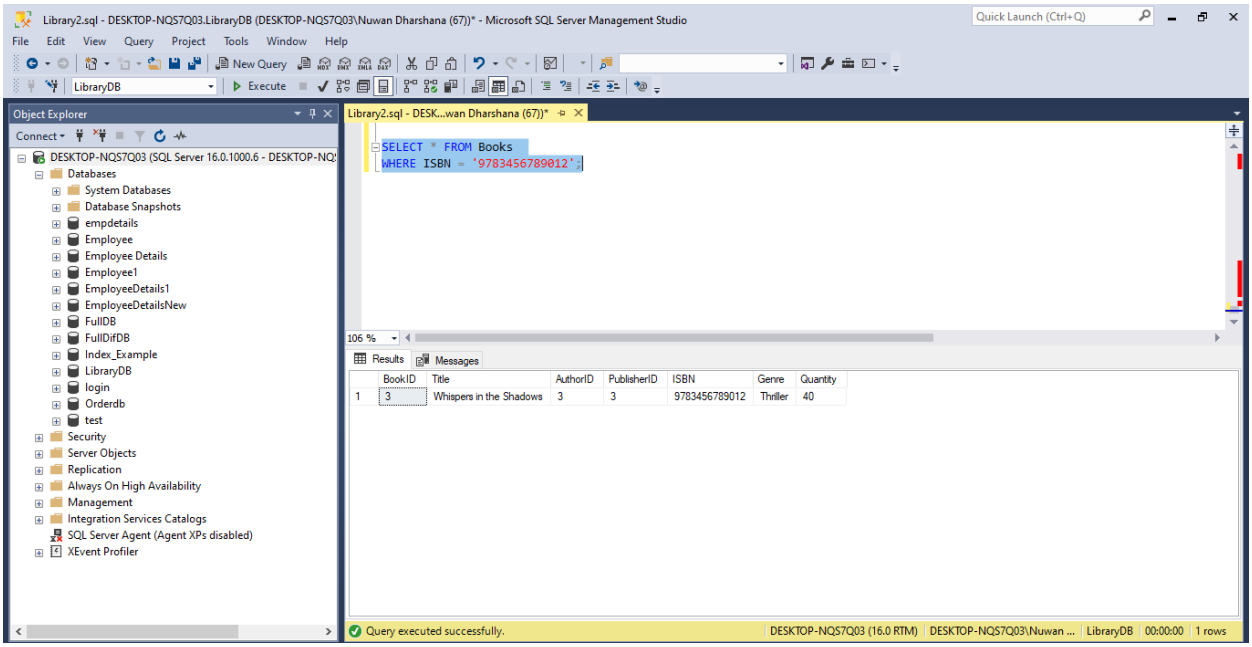


Figure 13:DQL Query

Retrieve all columns from the "Books" table where the ISBN matches '9783456789012'.

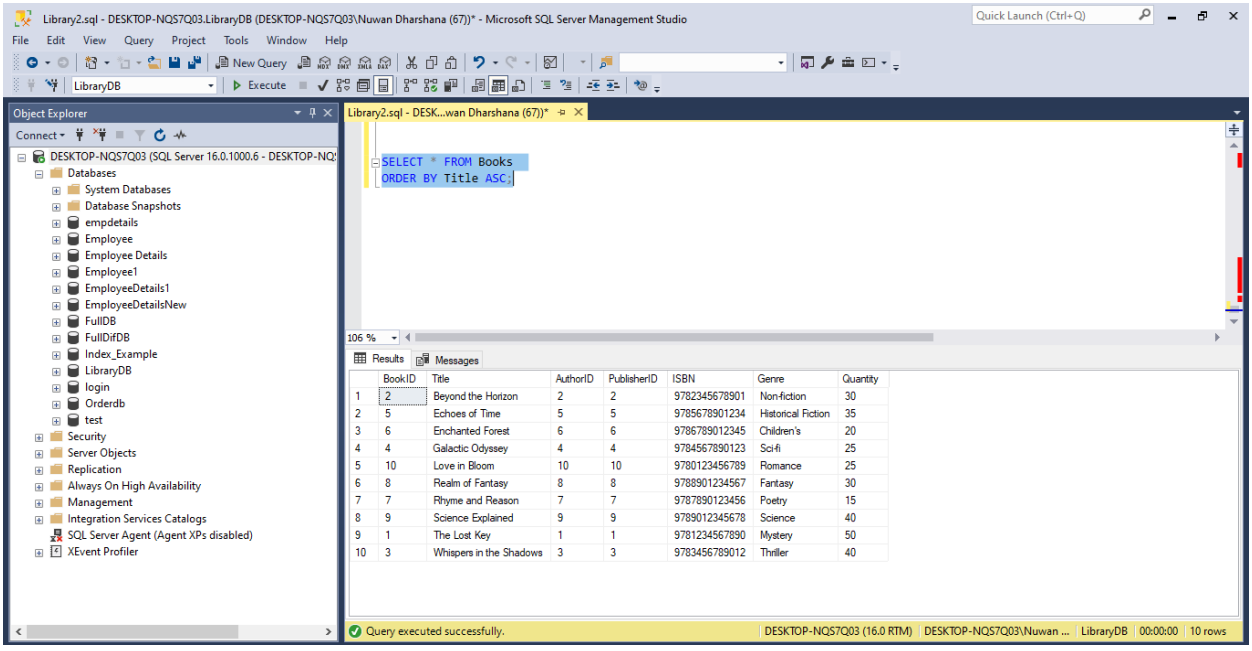


Figure 14:DQL Query

Retrieve all columns from the "Books" table, ordering the results by title in ascending order.

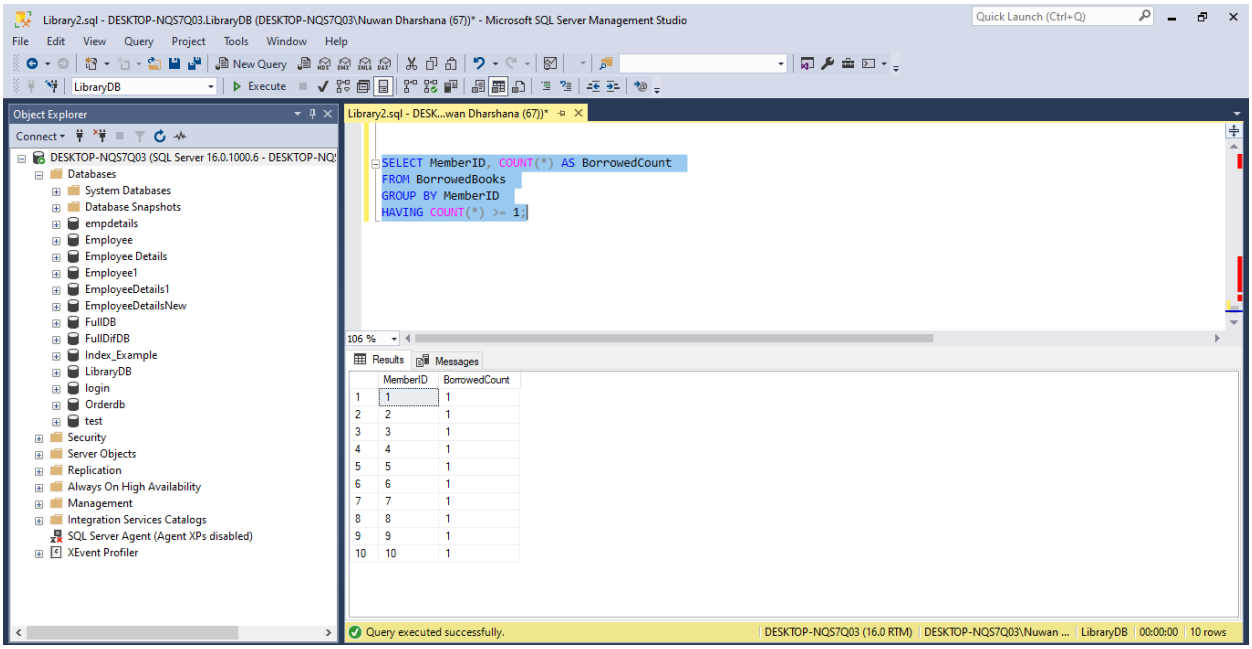


Figure 15:DQL Query

Count the number of books borrowed by each member from the "BorrowedBooks" table, displaying the member ID and the count of books borrowed for members who have borrowed at least one book.

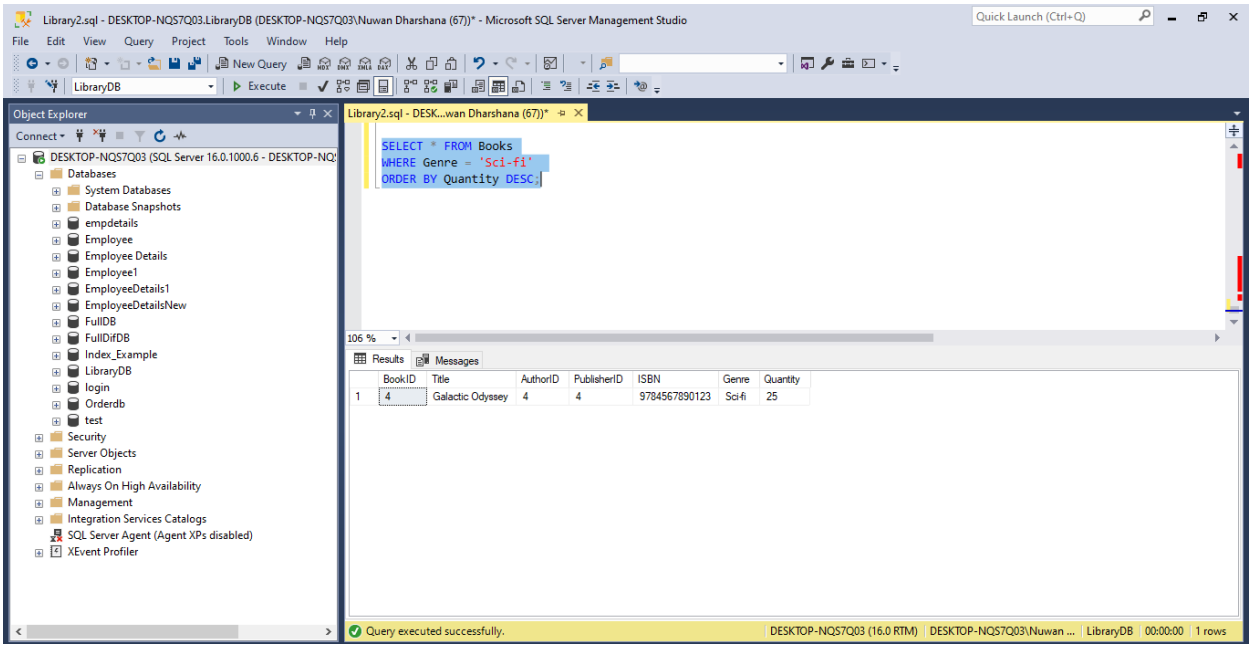


Figure 16:DQL Query

Retrieve all columns from the "Books" table where the genre is 'Sci-fi', ordering the results by quantity of books in descending order.

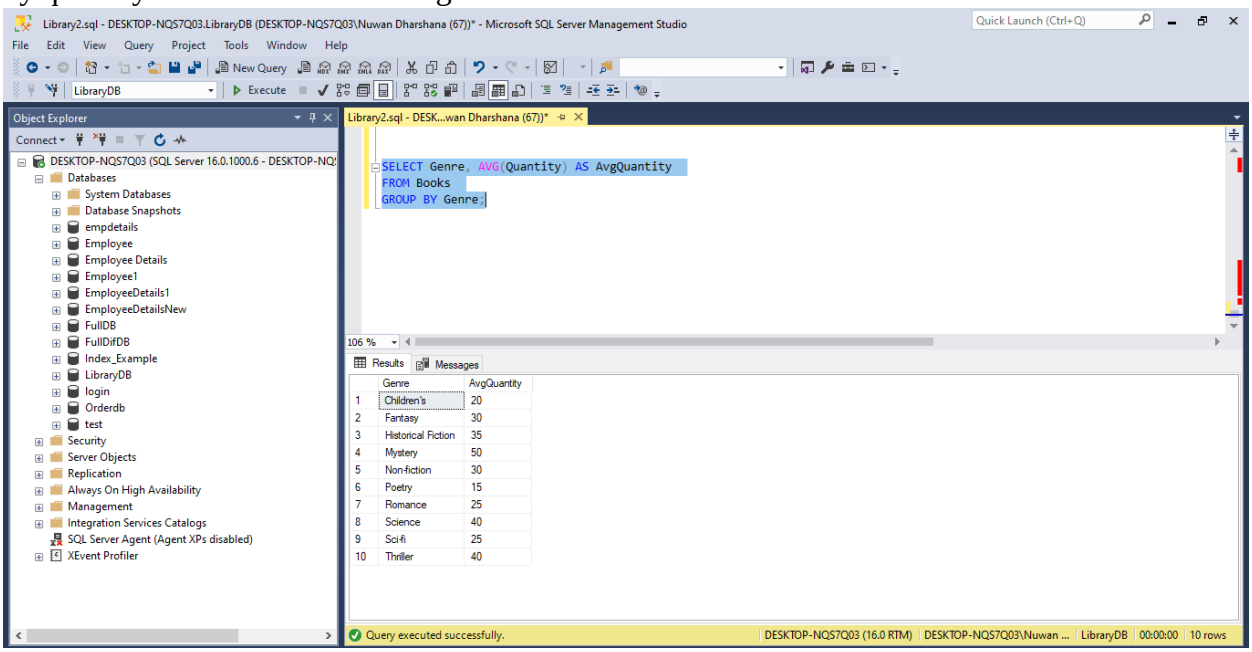


Figure 17:DQL Query

Calculate the average quantity of books for each genre from the "Books" table, displaying the genre and the average quantity.

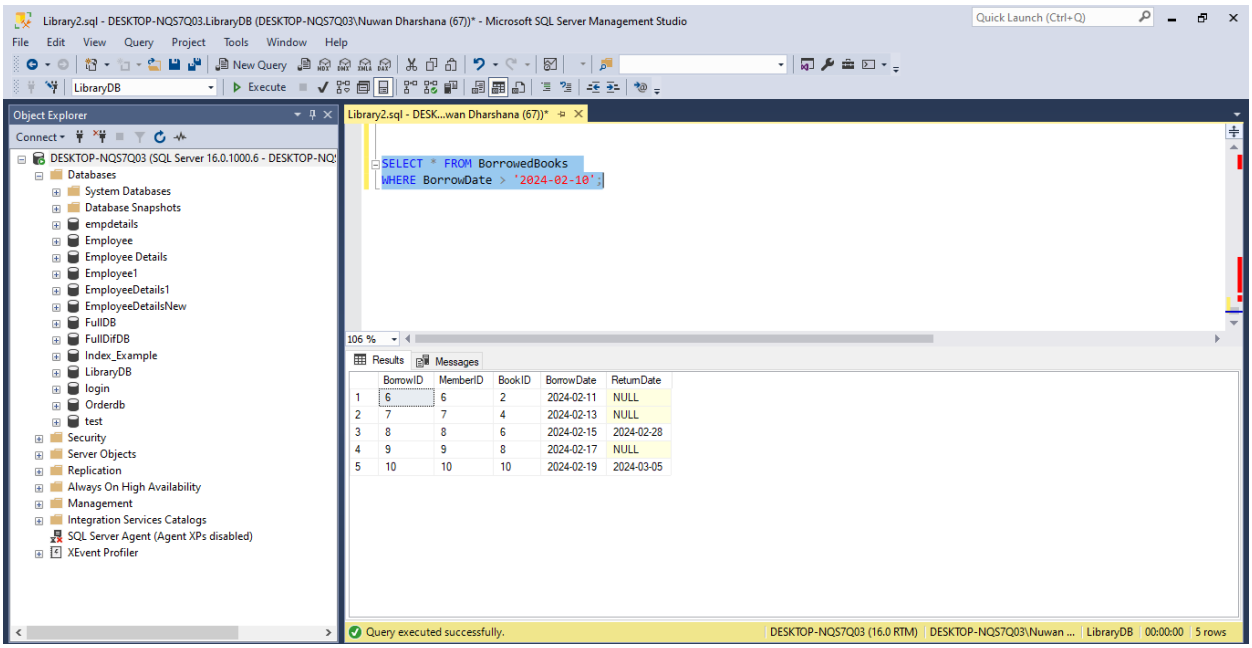


Figure 18:DQL Query

Retrieve all columns from the "BorrowedBooks" table where the borrow date is after '2024-02-10'.

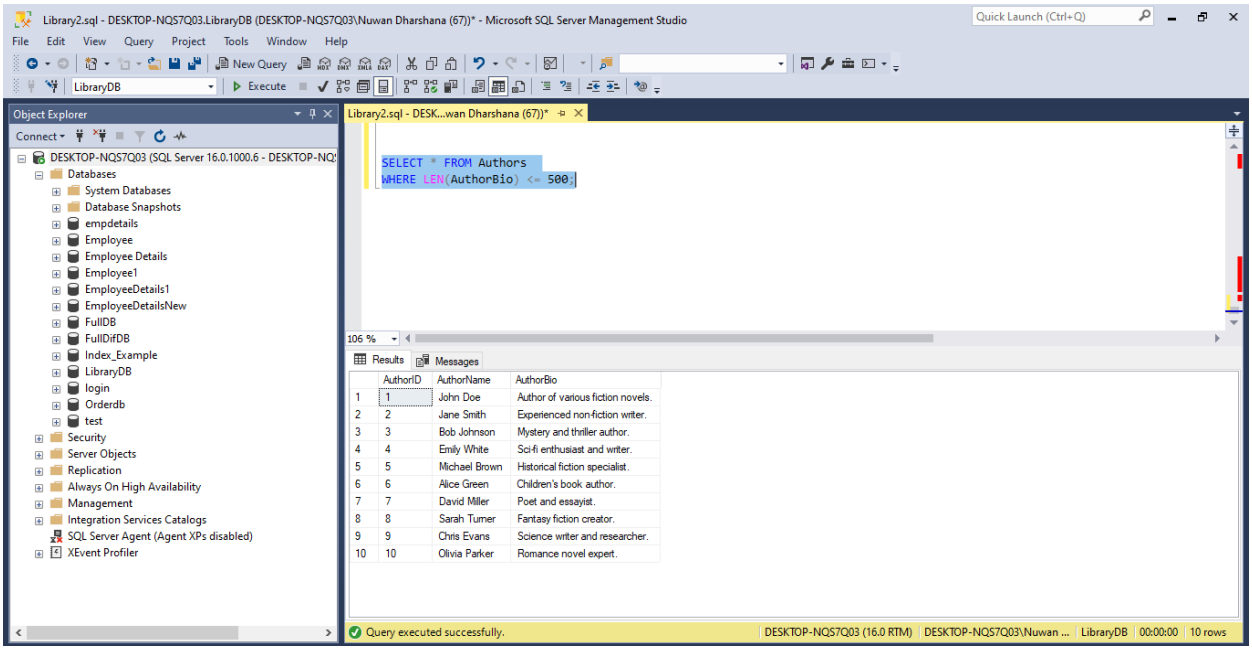


Figure 19:DQL Query

Retrieve all columns from the "Authors" table where the length of the author biography is less than or equal to 500 characters.

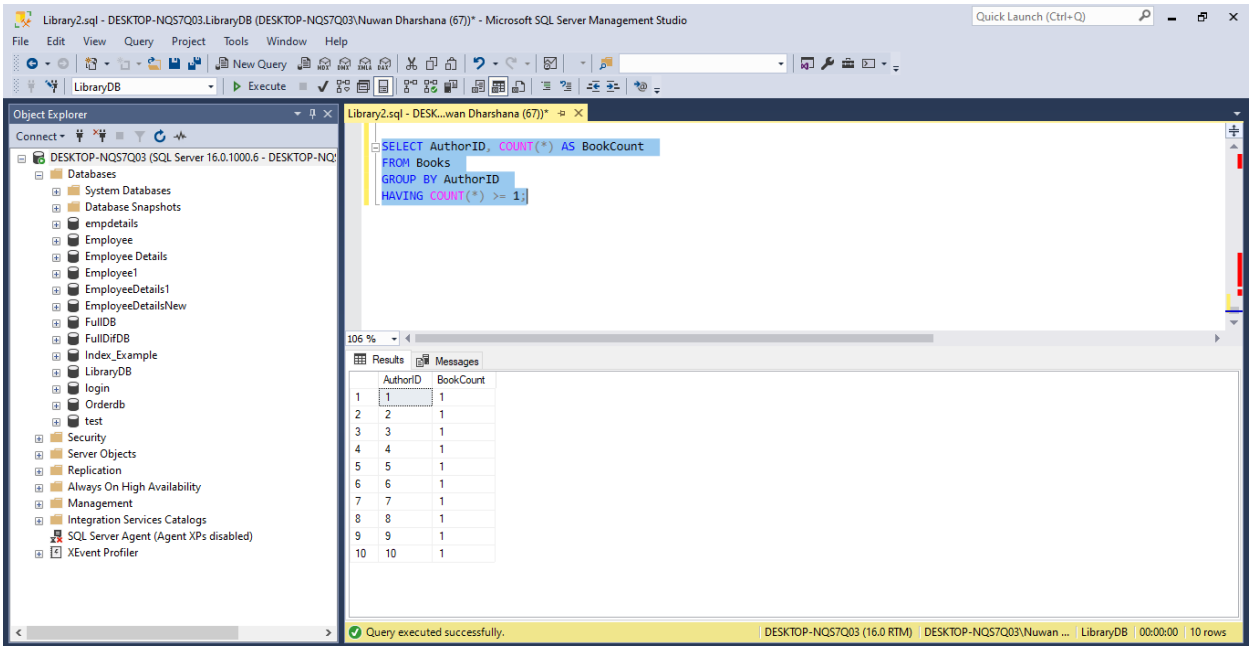


Figure 20:DQL Query

Count the number of books written by each author from the "Books" table, displaying the author ID and the count of books written for authors who have written at least one book.

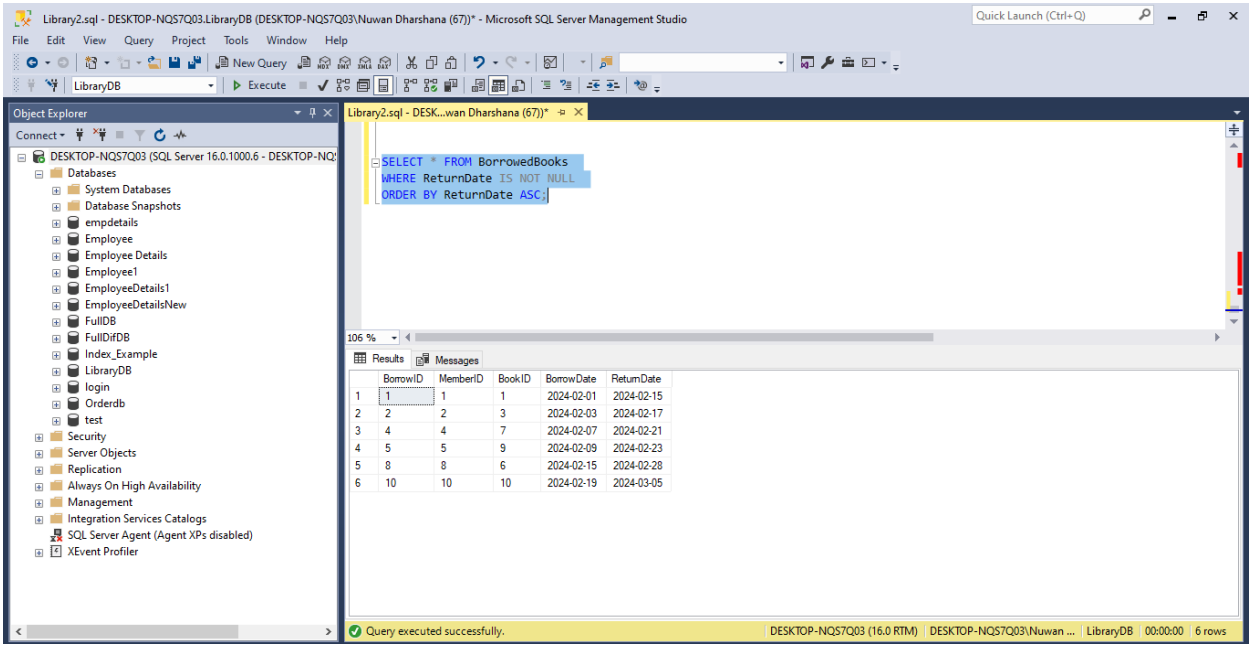


Figure 21:DQL Query

Retrieve all columns from the "BorrowedBooks" table where the return date is not null, ordering the results by return date in ascending order.

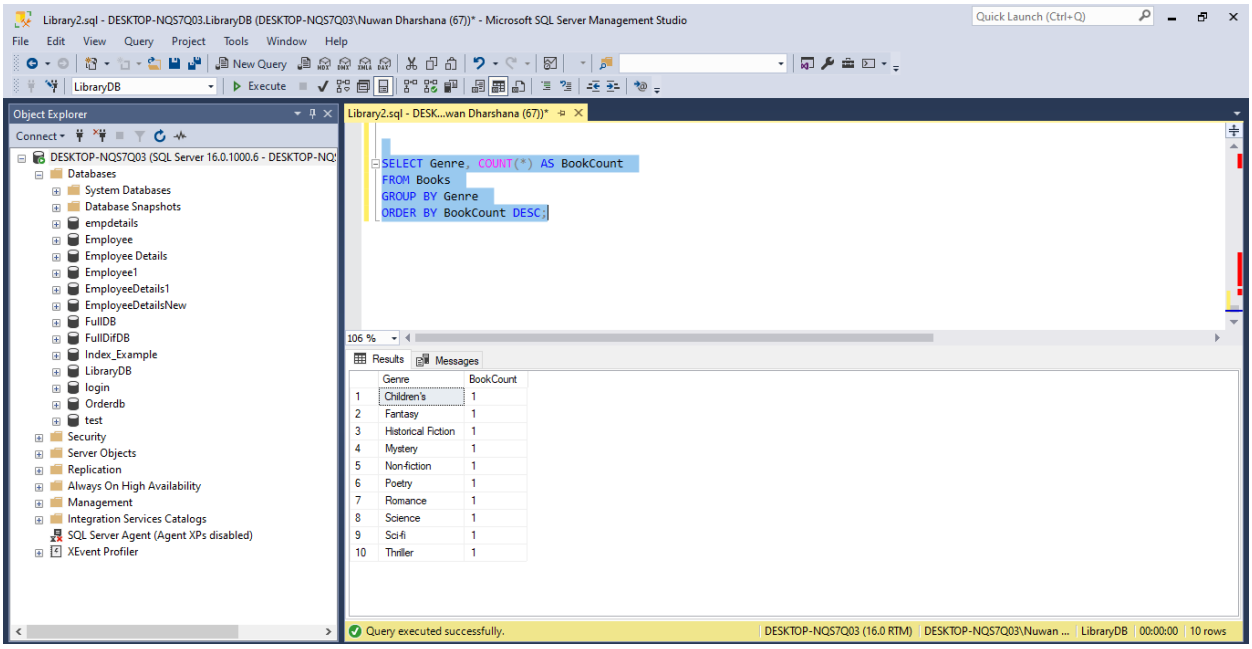


Figure 22:DQL Query

Count the number of books in each genre from the "Books" table, displaying the genre and the count of books for each genre, ordered by the count of books in descending order.

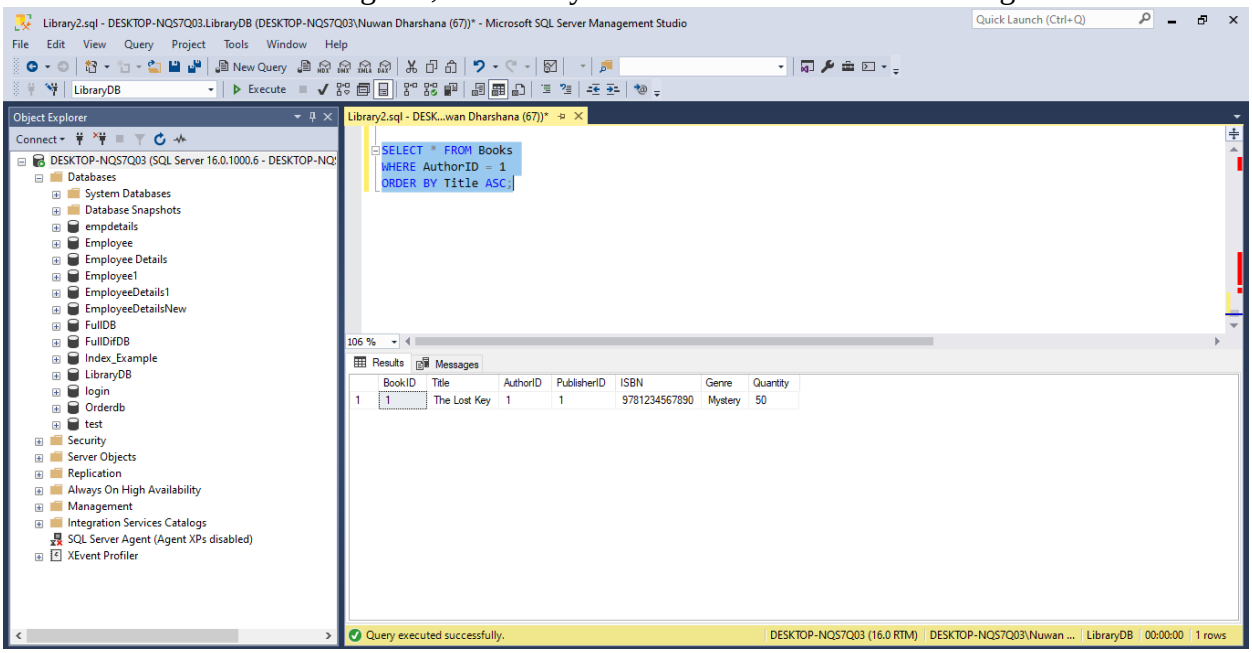


Figure 23:DQL Query

Retrieve all columns from the "Books" table where the author ID is 1, ordering the results by title in ascending order.

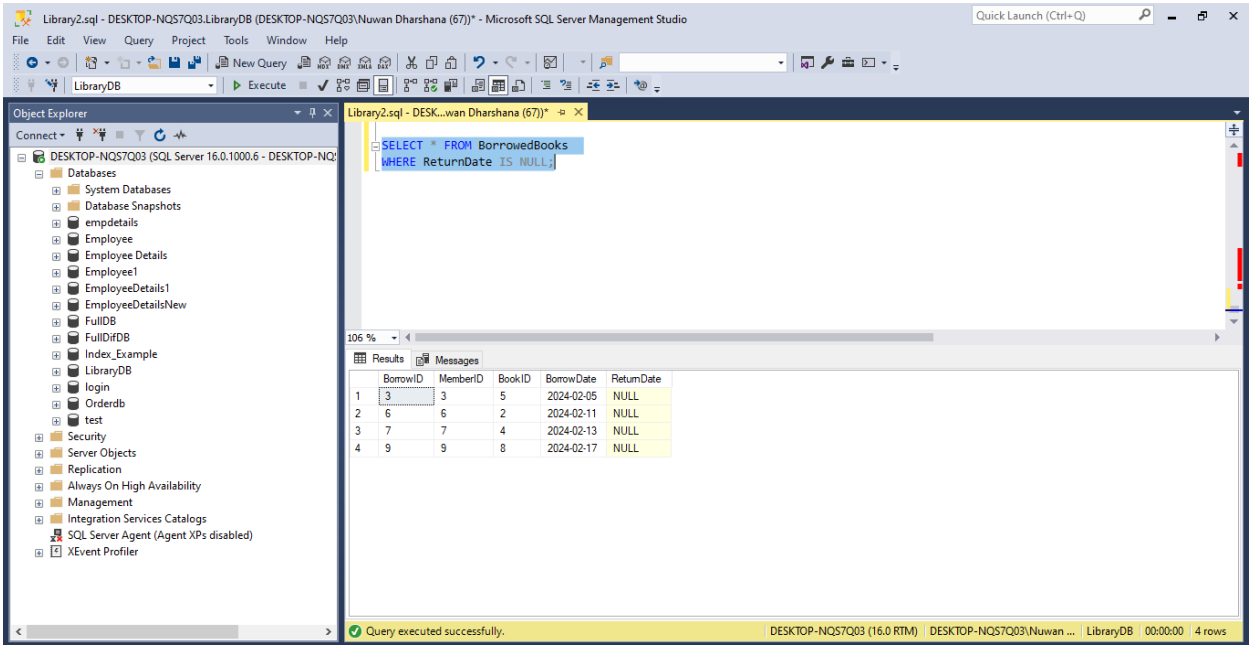


Figure 24:DQL Query

Retrieve all columns from the "BorrowedBooks" table where the return date is null.

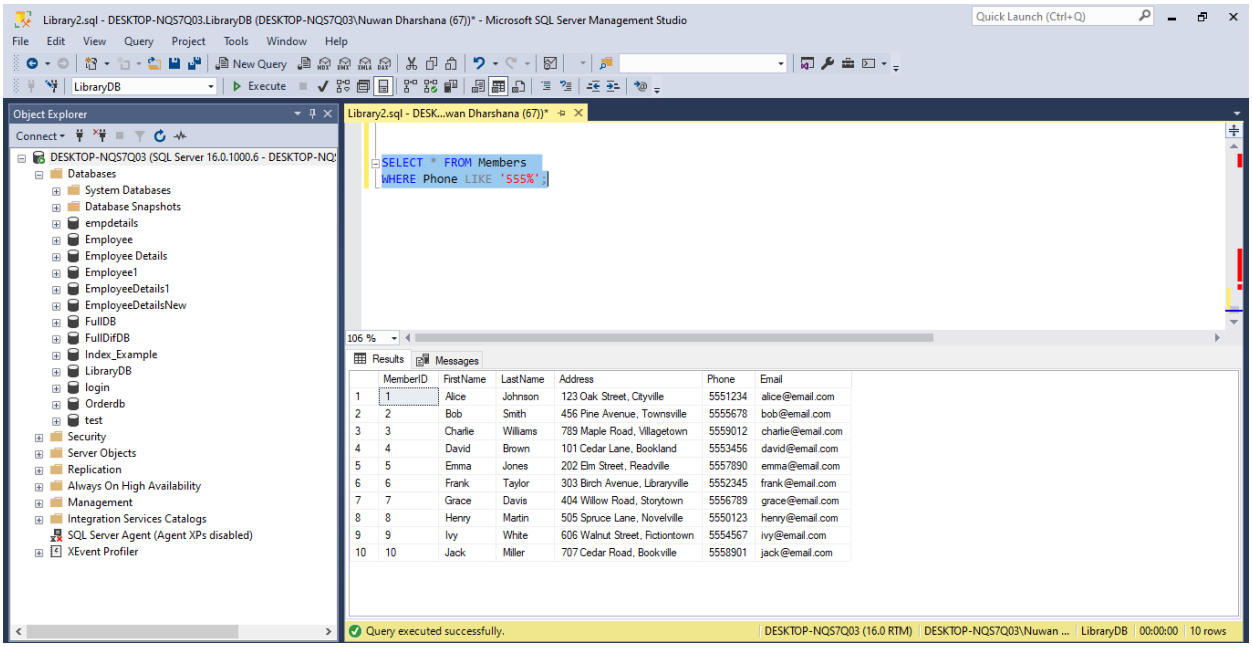


Figure 25:DQL Query

Retrieve all columns from the "Members" table where the phone number starts with '555'.

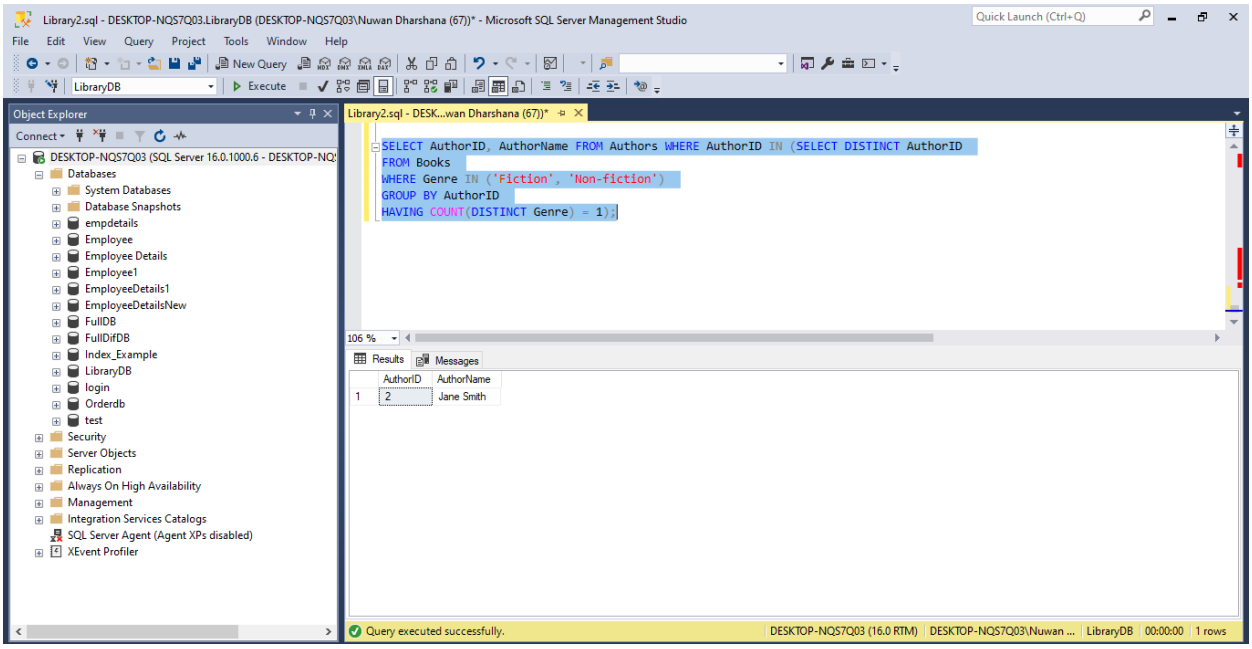


Figure 26:DQL Query

Retrieve the author ID and name from the "Authors" table for authors who have written books in only one genre, either 'Fiction' or 'Non-fiction'.

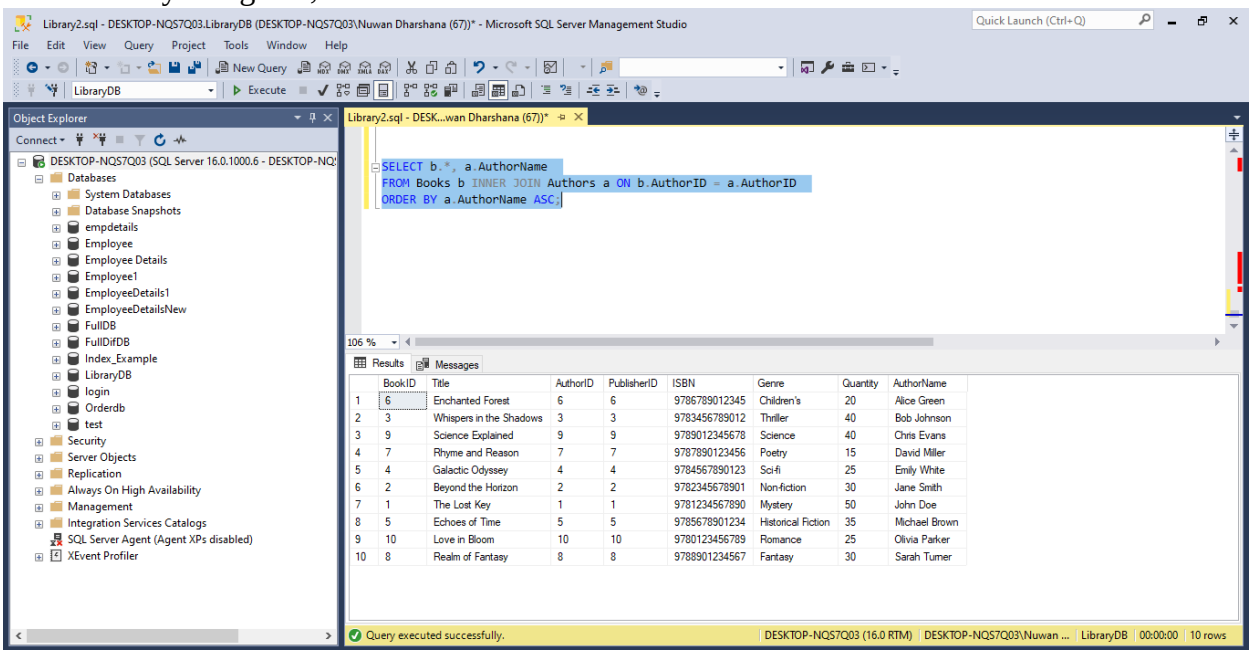


Figure 27:DQL Query

Retrieve all columns from the "Books" table, joining with the "Authors" table on author ID, ordering the results by author name in ascending order.

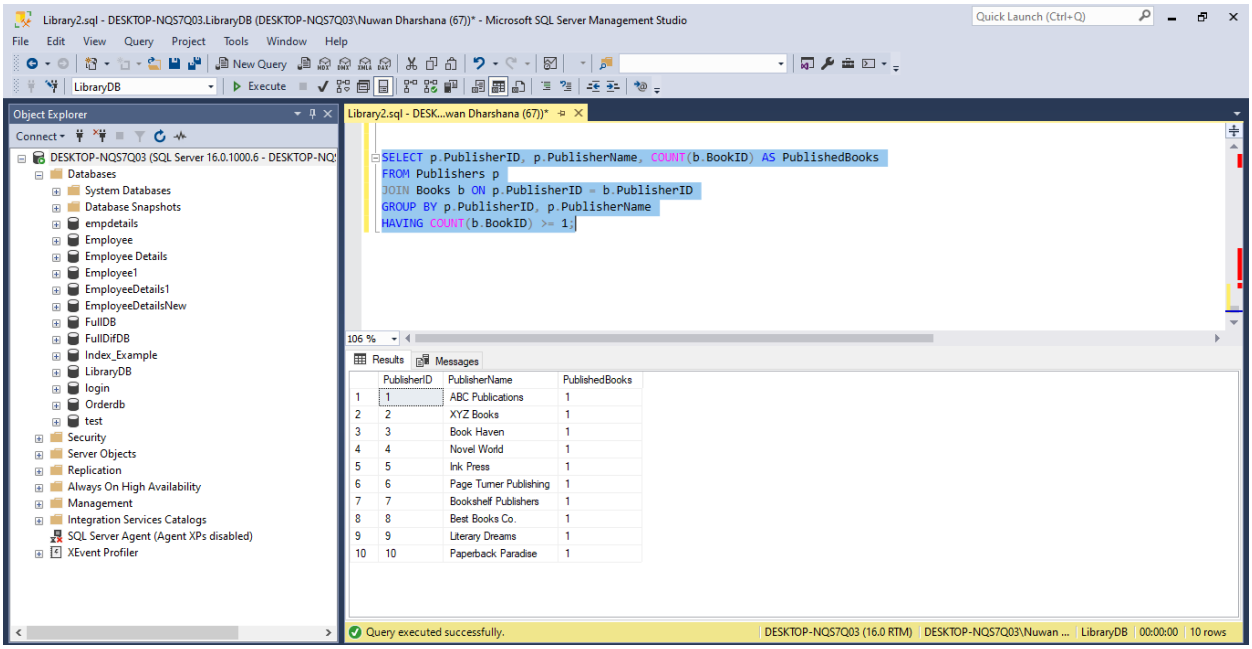


Figure 28:DQL Query

Count the number of books published by each publisher from the "Publishers" table, displaying the publisher ID, publisher name, and the count of books published by each publisher, for publishers who have published at least one book.

4) Write ten nested queries using the above tables

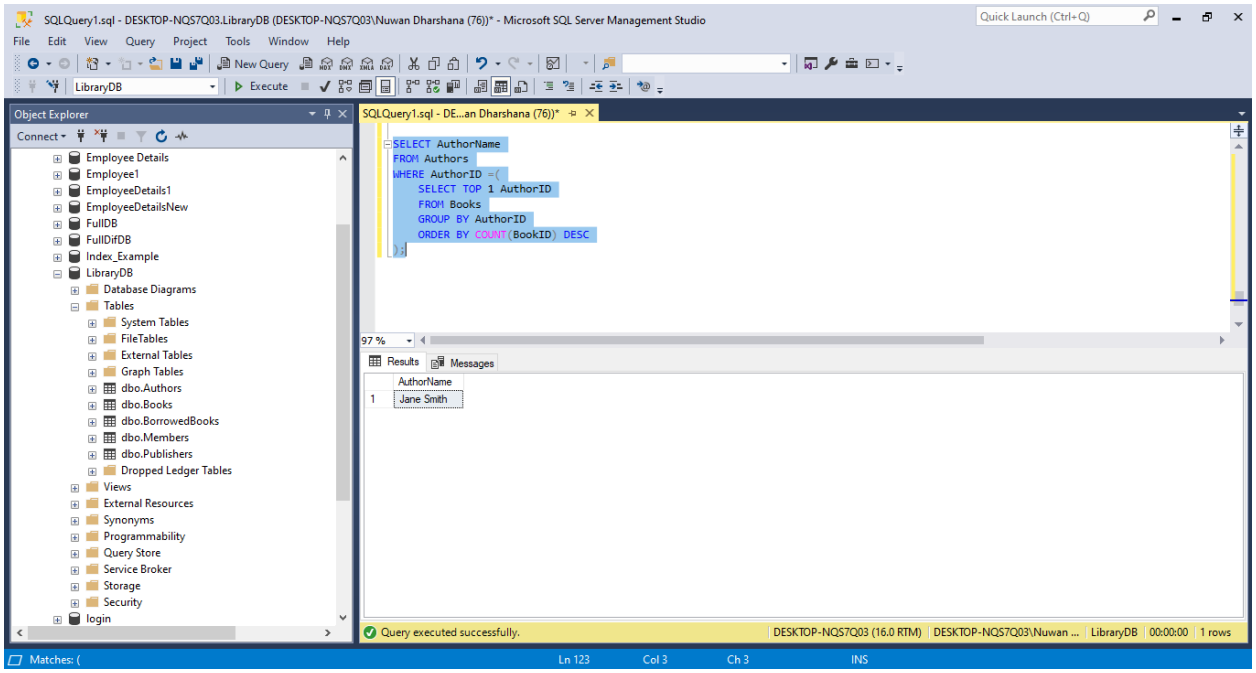


Figure 29:Nested Query

Find the author with the highest count of books written by counting books for each author and selecting the top one.

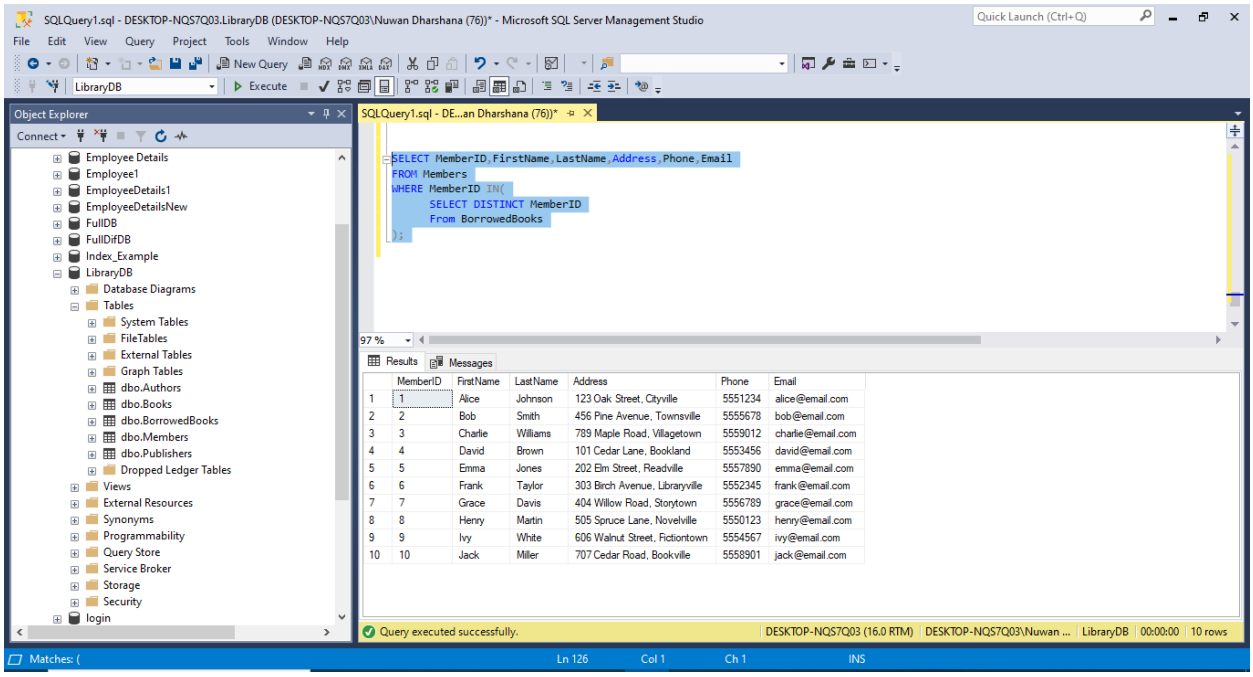


Figure 30:Nested Query

Select member details for members who have borrowed books by finding their MemberID in the BorrowedBooks table.

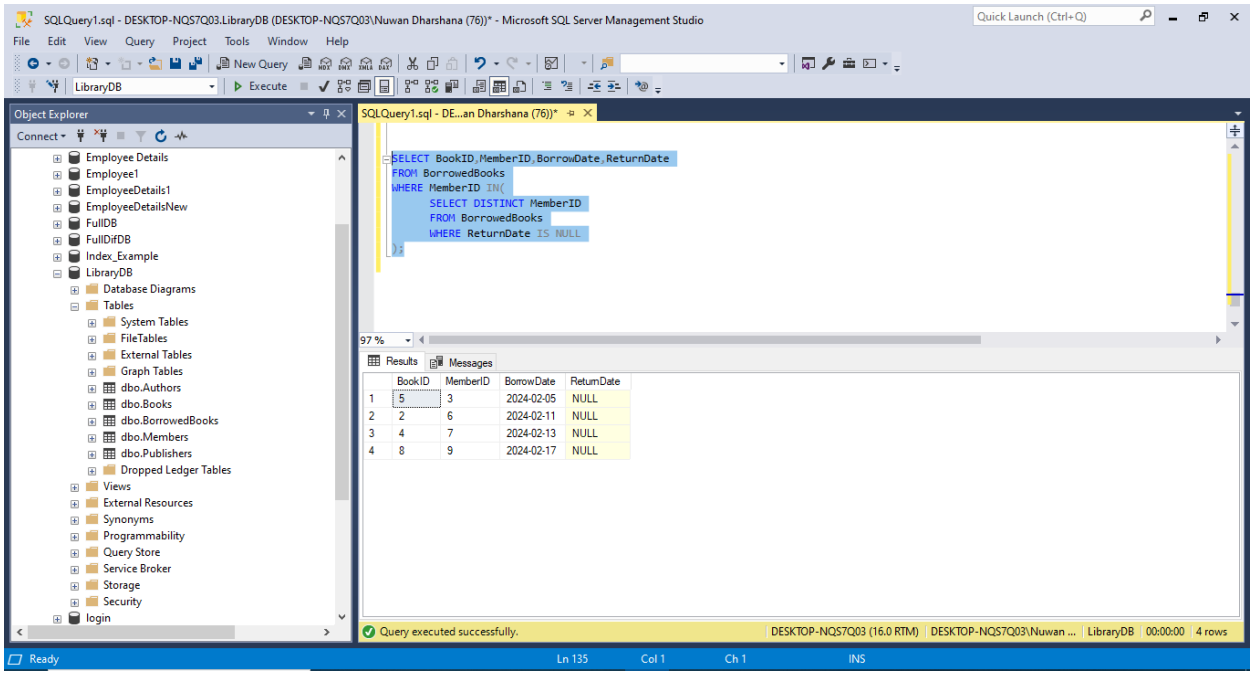


Figure 31:Nested Query

Retrieve details of borrowed books for members who haven't returned their borrowed books yet.

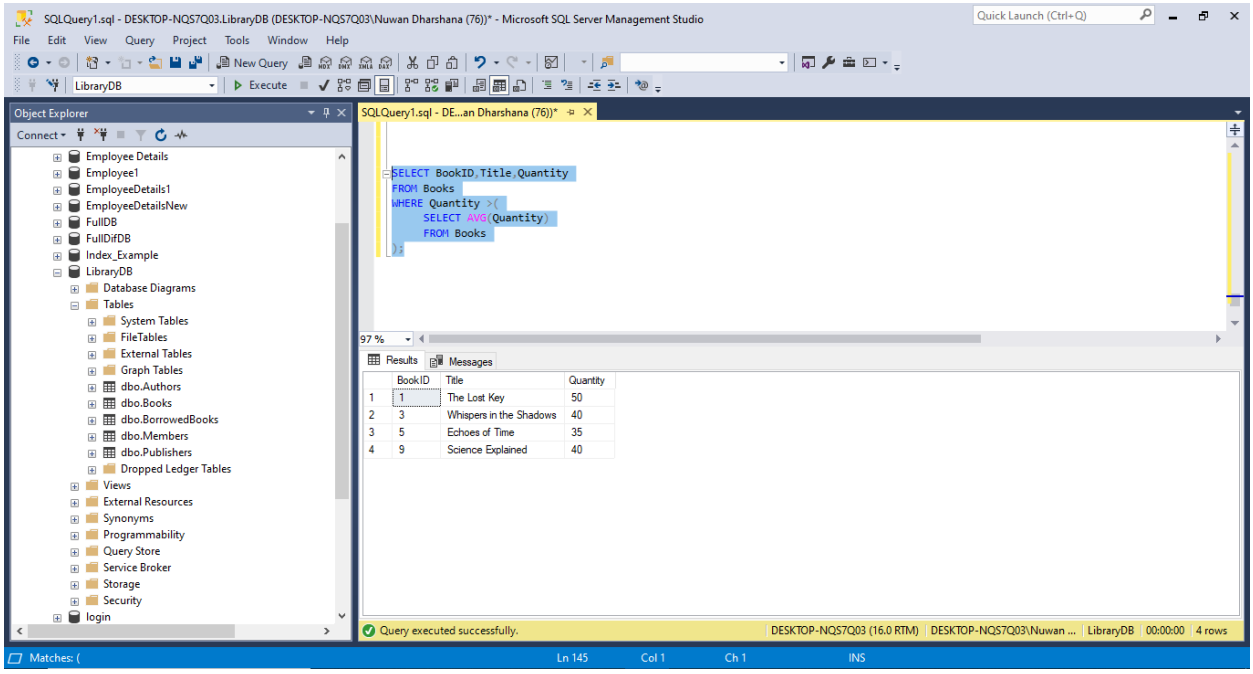


Figure 32:Nested Query

Select books with quantities greater than the average quantity of all books in the Books table.

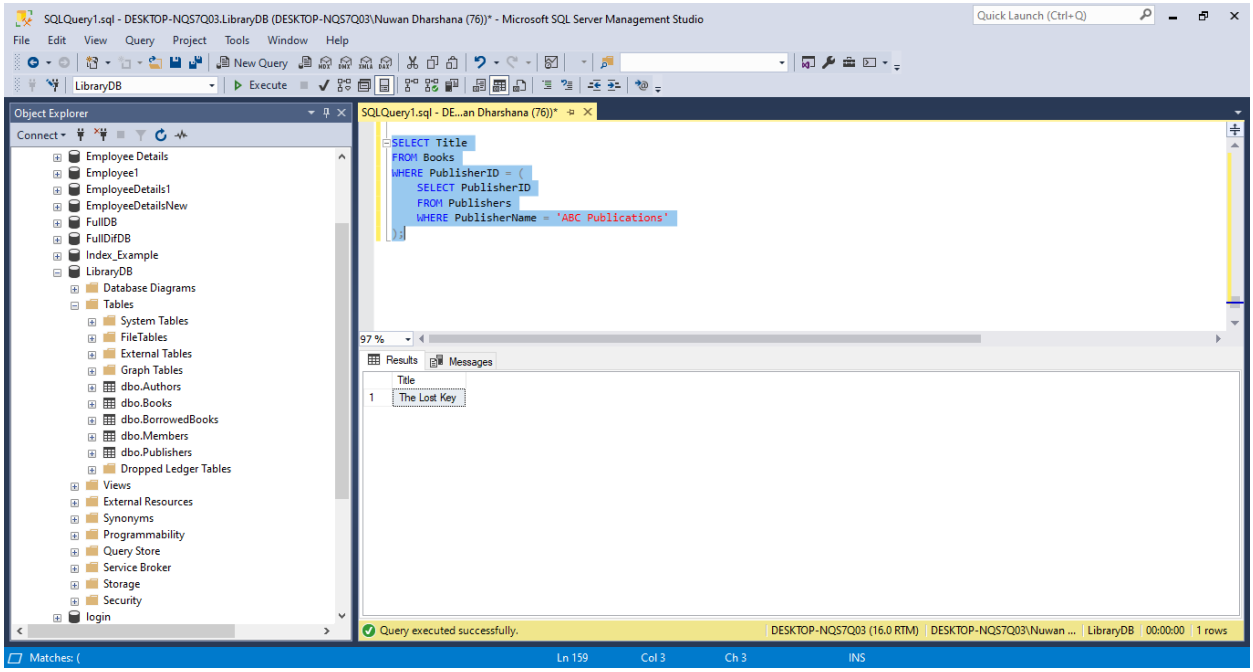


Figure 33:Nested Query

Select titles of books published by 'ABC Publications' by finding the corresponding PublisherID in the Publishers table.

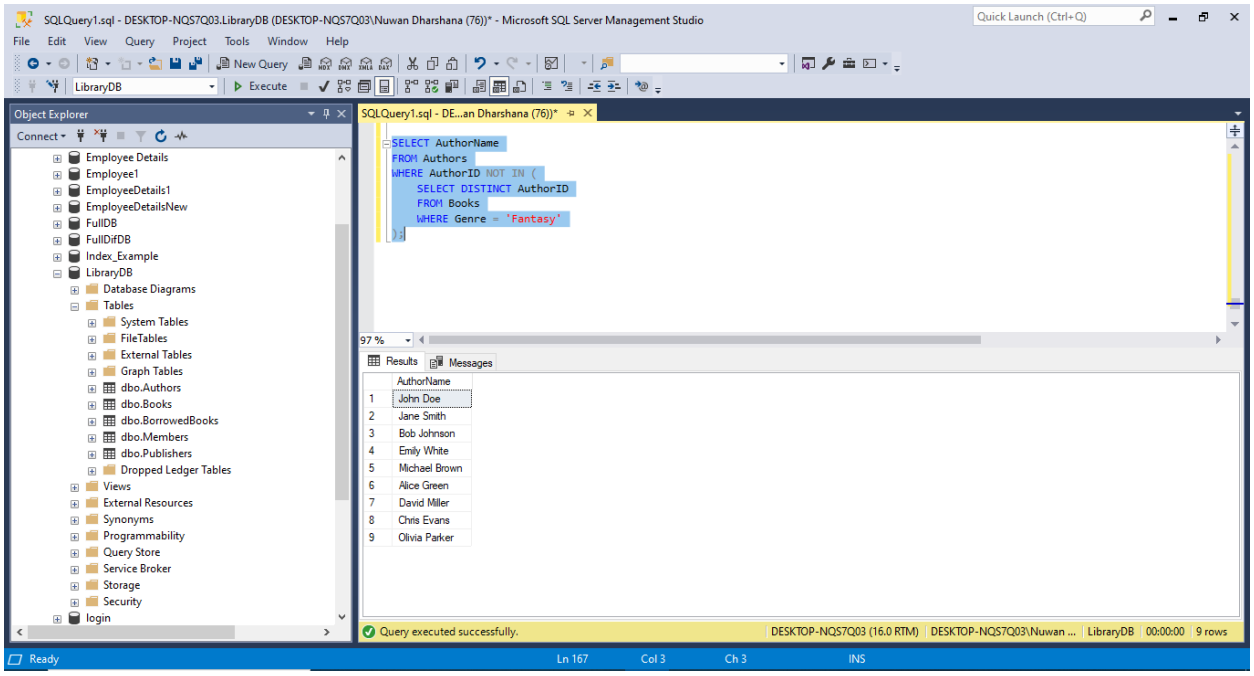


Figure 34:Nested Query

Select authors who don't have any books listed under the 'Fantasy' genre.

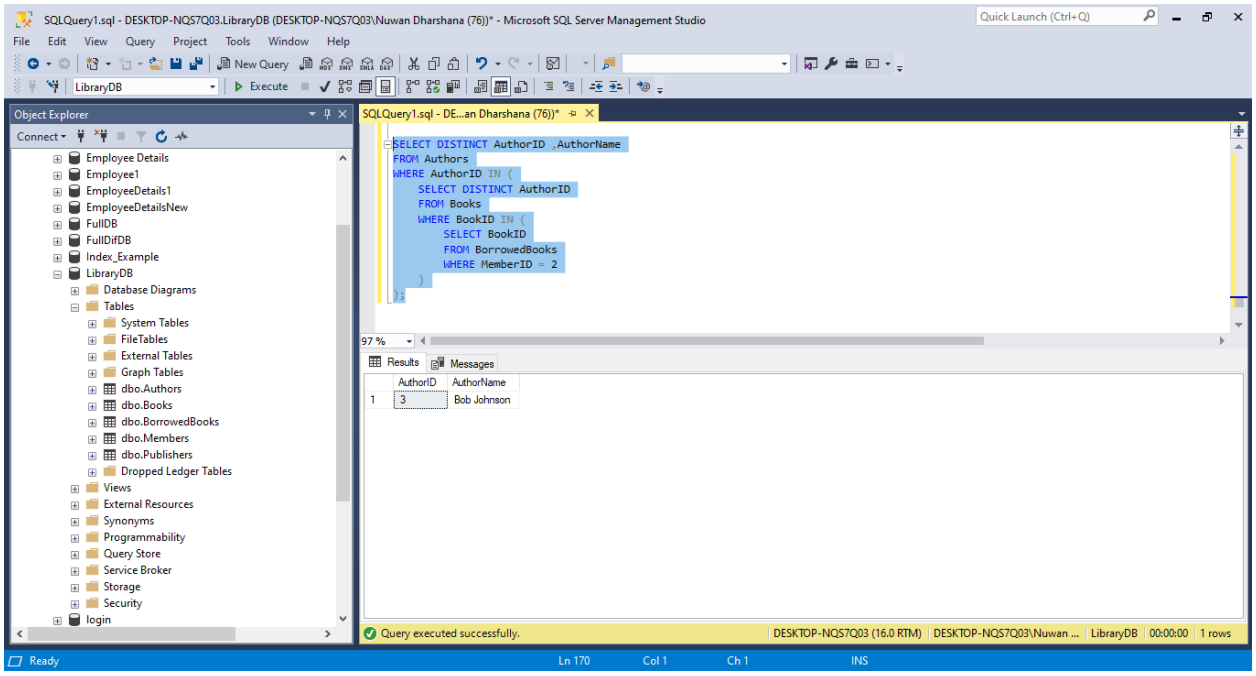


Figure 35:Nested Query

Find authors whose books have been borrowed by a particular member by first finding the BookIDs borrowed by that member and then finding the corresponding authors.

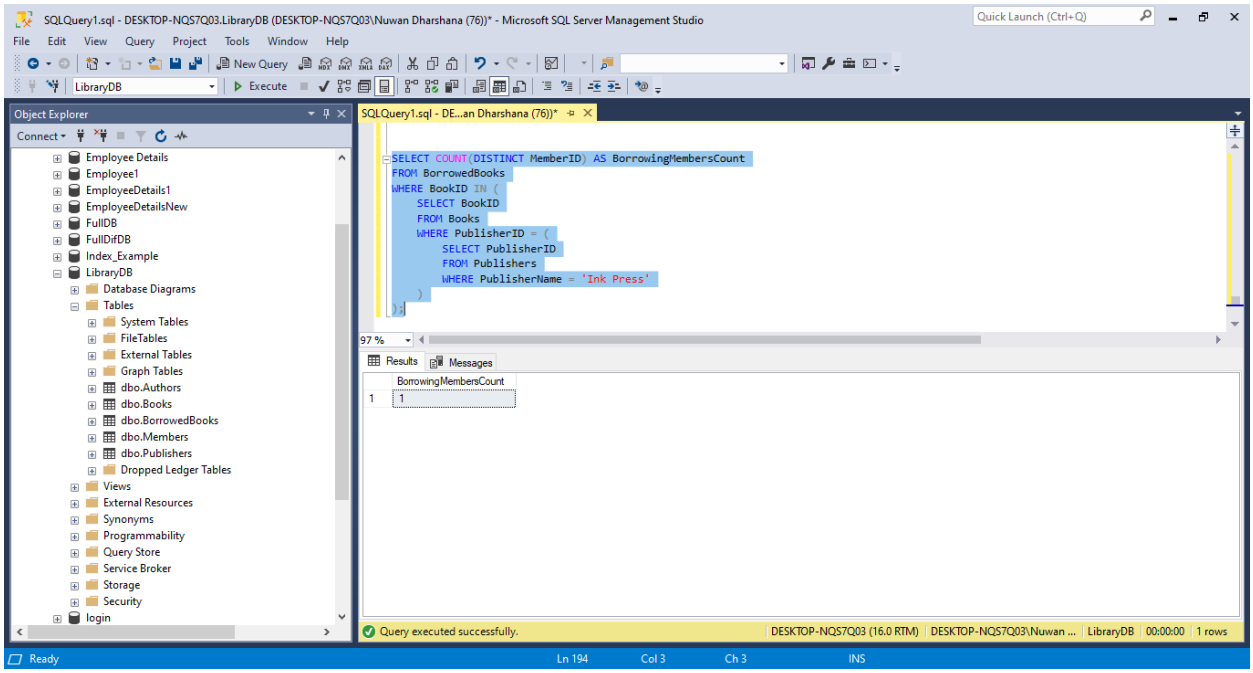


Figure 36:Nested Query

Count the distinct number of members who have borrowed books from the publisher 'Ink Press'.

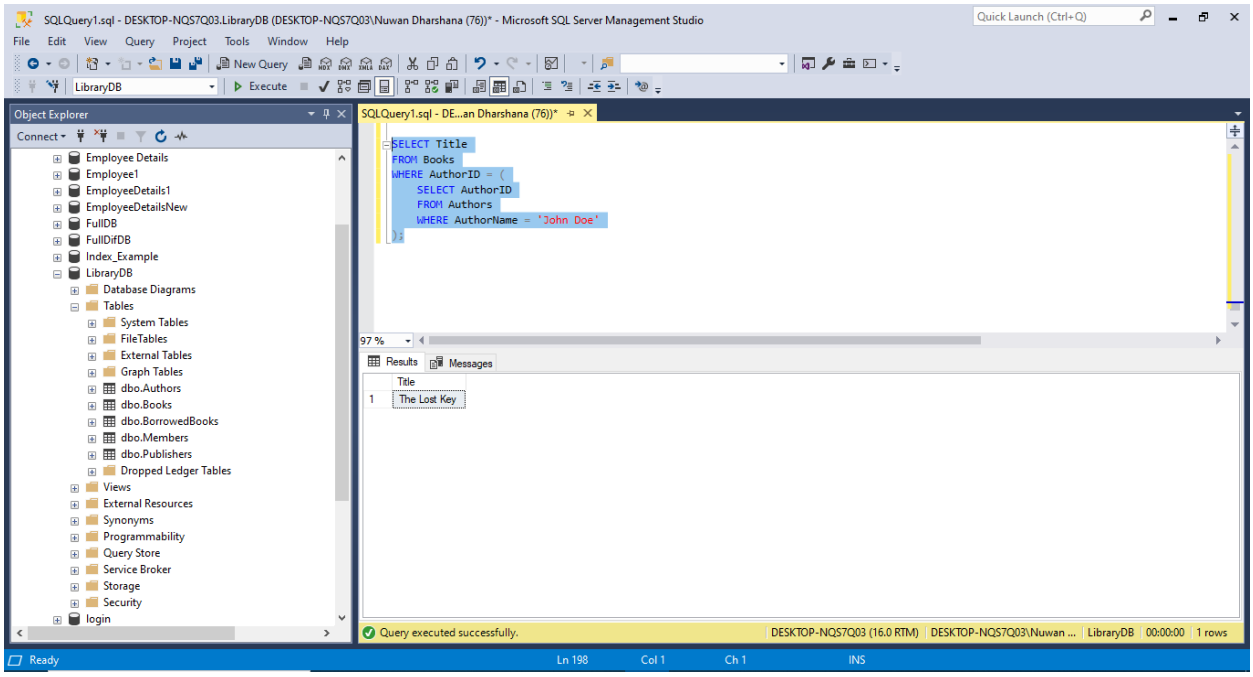


Figure 37:Nested Query

Find the titles of books written by the author 'John Doe' by first identifying his AuthorID.

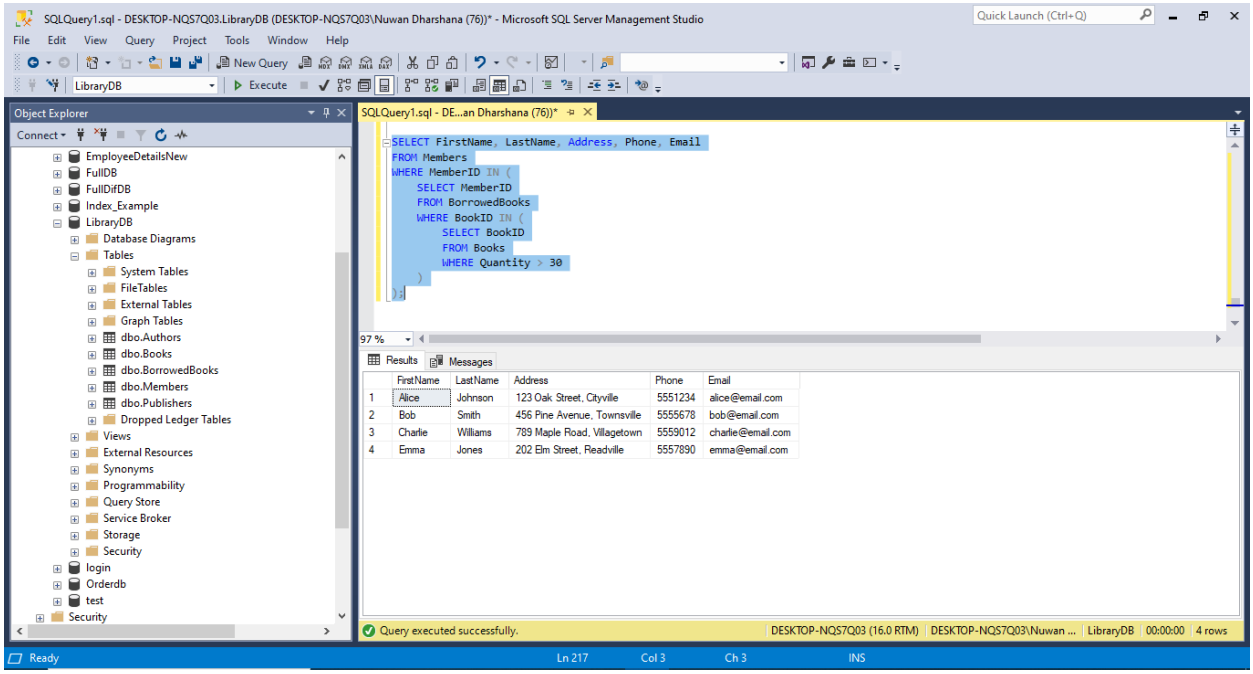


Figure 38:Nested Query

Retrieve details of members who borrowed books with quantities greater than 30 by first finding those books in the Books table and then finding the corresponding members in the BorrowedBooks table.

5) Write at least five queries considering following joining categories. (In order to obtain the desired results, appropriate records must be inserted into tables in database). • Inner Join • Left Join • Right Join • Full outer Join

• Inner Join

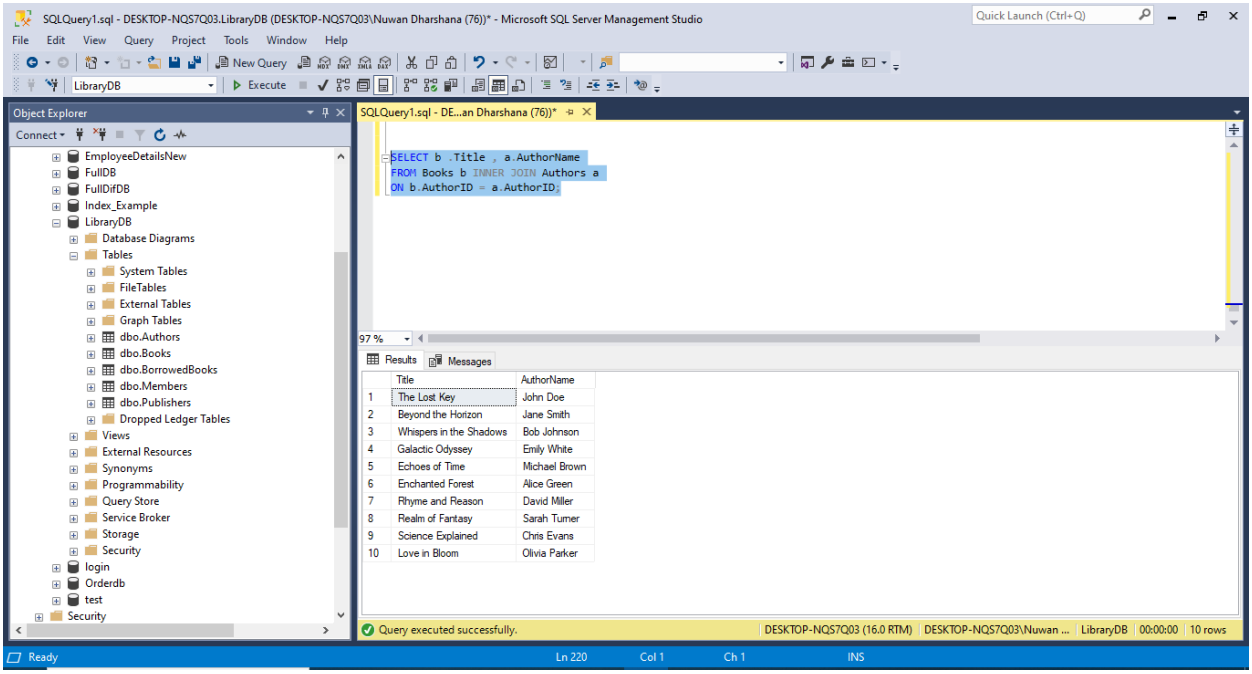


Figure 39:Inner Join

This query retrieves the title of books along with the names of their authors by joining the Books table with the Authors table based on the AuthorID column.

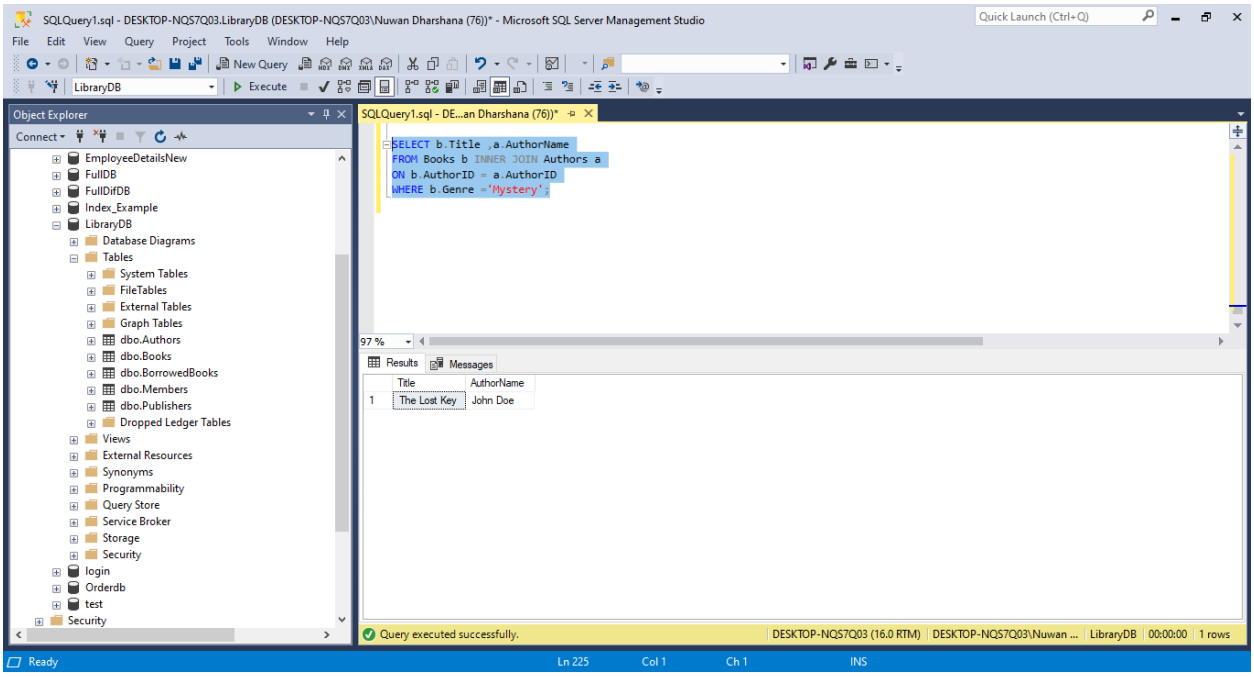


Figure 40:Inner Join

This query retrieves the title of books and their corresponding author names for books specifically in the 'Mystery' genre by joining the Books and Authors tables and filtering for books with the 'Mystery' genre.

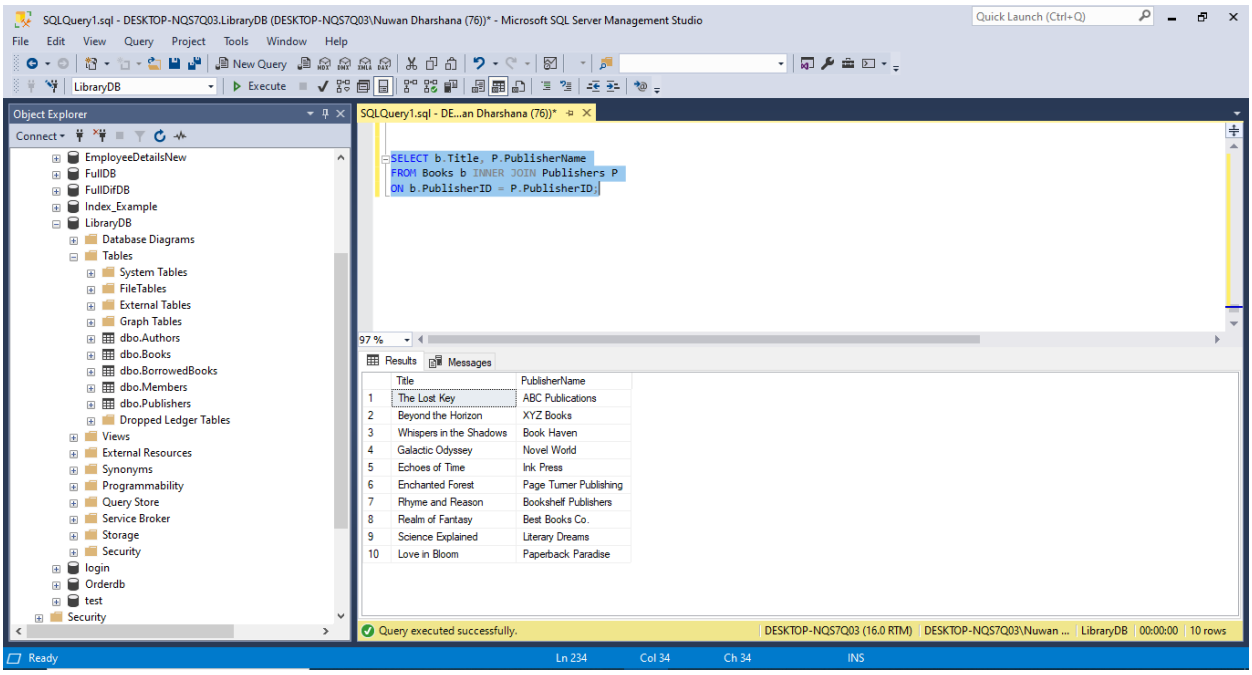


Figure 41:Inner Join

This query retrieves the title of books along with the names of their publishers by joining the Books table with the Publishers table based on the PublisherID column.

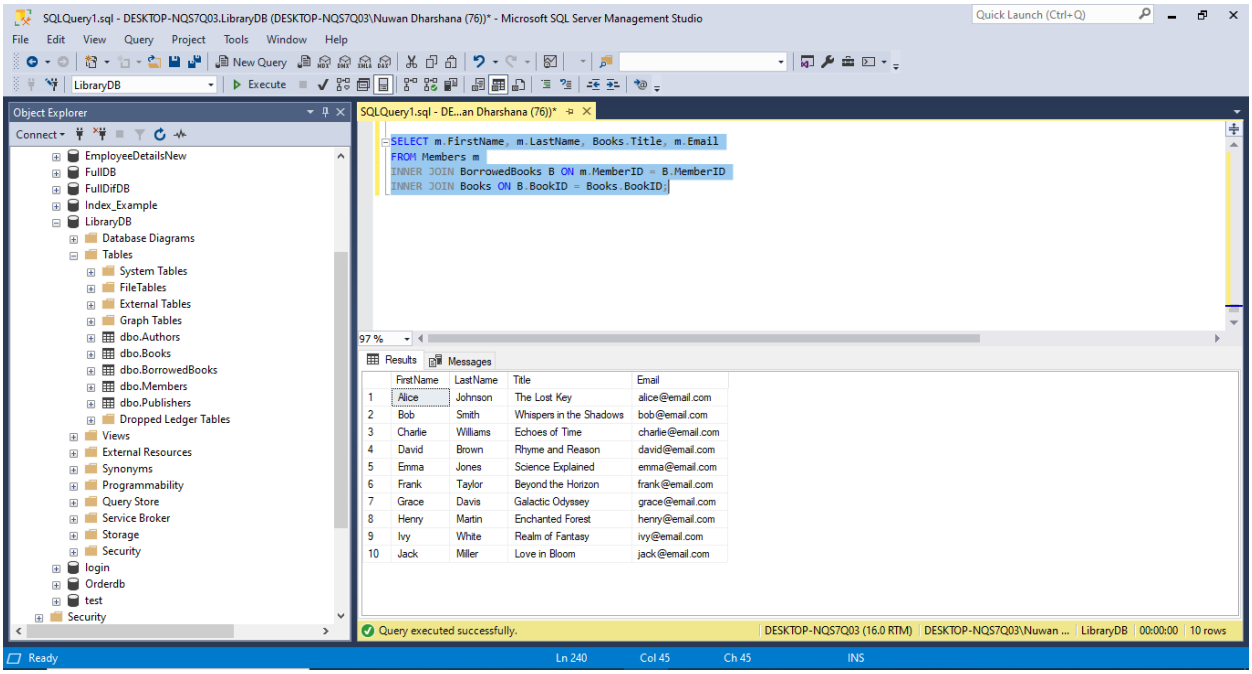


Figure 42:Inner Join

This query retrieves the first name, last name, email, and the title of books borrowed by members by joining the Members table with the BorrowedBooks table and then joining with the Books table to get the title of the borrowed books.

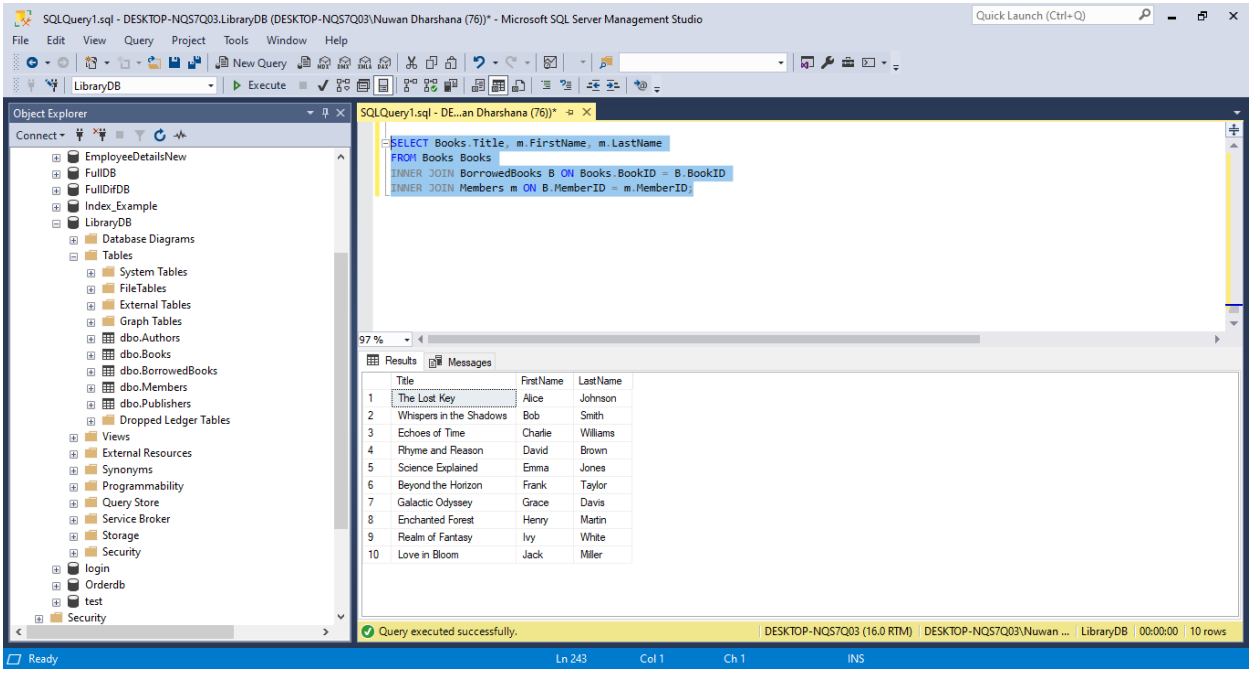


Figure 43:Inner Join

This query retrieves the title of books along with the first name and last name of members who borrowed them by joining the Books table with the BorrowedBooks table based on the BookID column and then joining with the Members table to get the member names.

• Left Join

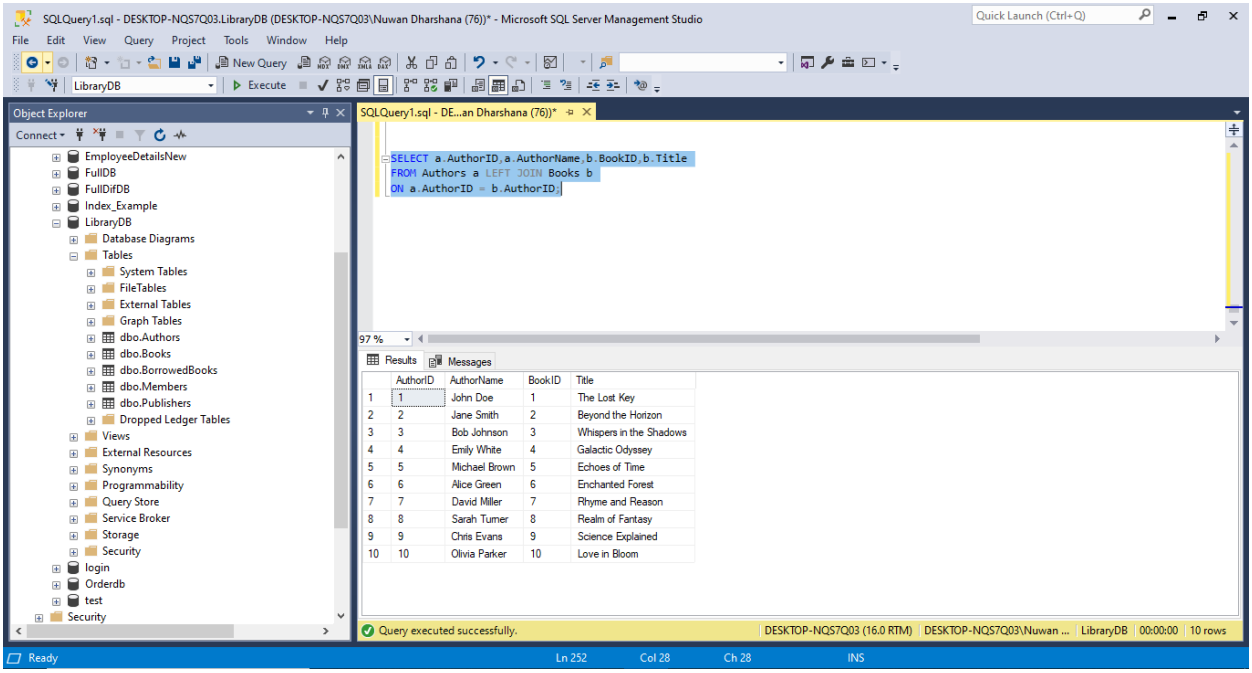


Figure 44:Left Join

This query retrieves AuthorID, AuthorName, BookID, and Title from the Authors table along with the corresponding books from the Books table, ensuring that all authors are included, even if they haven't authored any books.

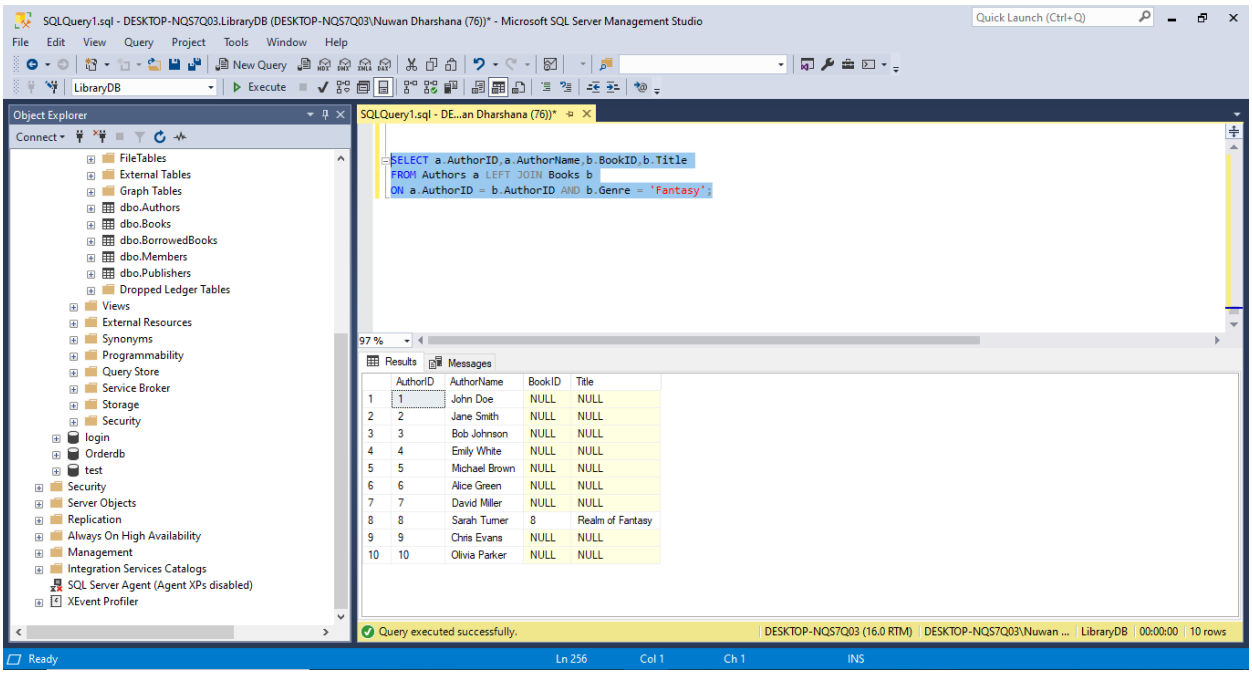


Figure 45:Left Join

it only includes books with the genre 'Fantasy'. All authors are still included, and for those who haven't authored any 'Fantasy' books, NULL values will appear in the BookID and Title columns.

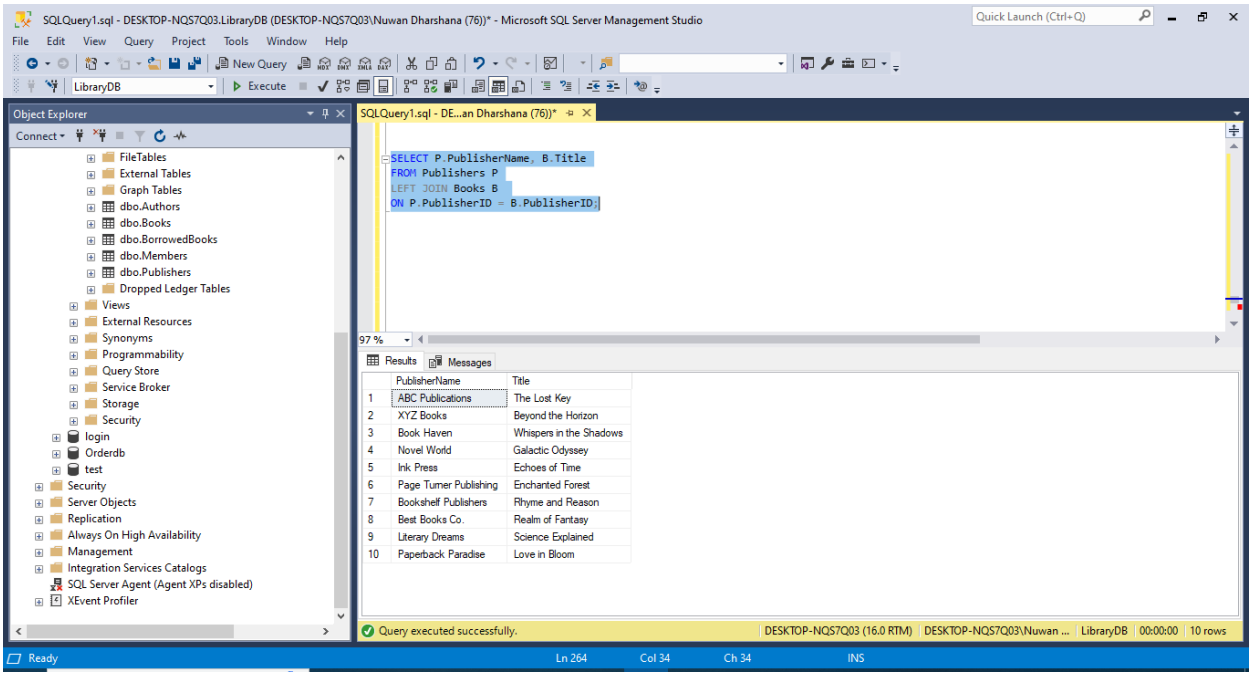


Figure 46:Left Join

This query retrieves PublisherName along with the titles of books published by each publisher. All publishers are included, even if they haven't published any books.

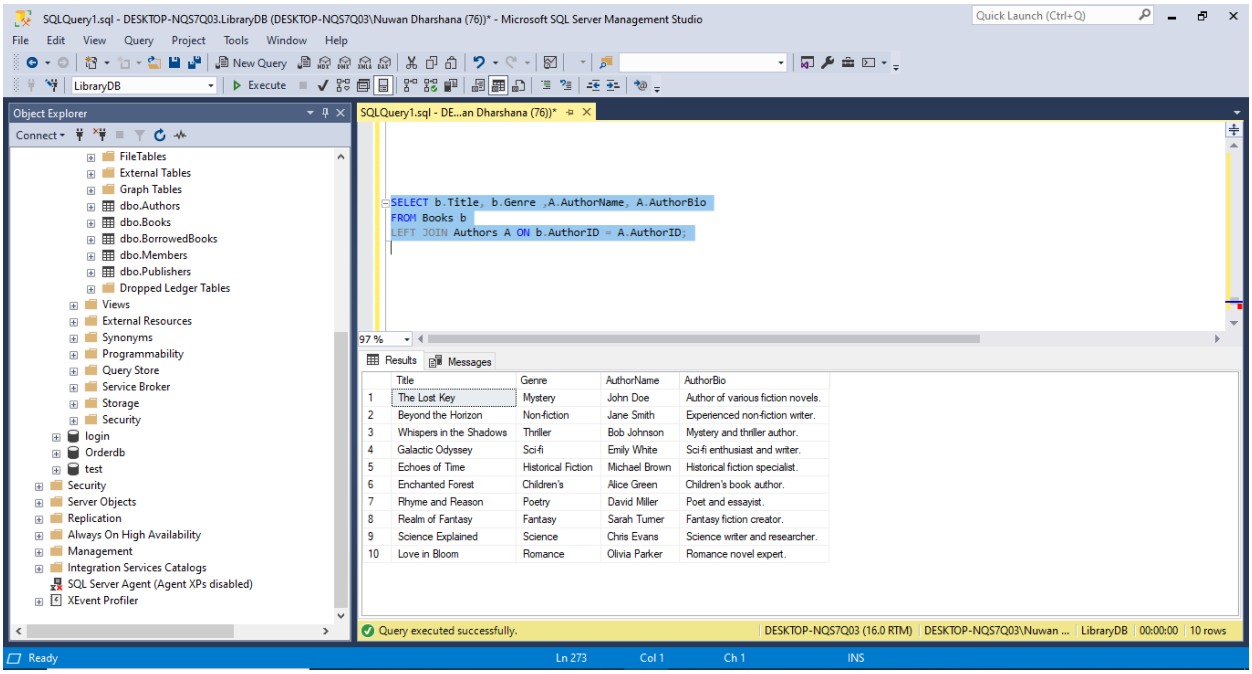


Figure 47:Left Join

This query fetches the title, genre of books along with the corresponding AuthorName and AuthorBio. All books are included, and for those without corresponding authors, NULL values will appear in the AuthorName and AuthorBio columns.

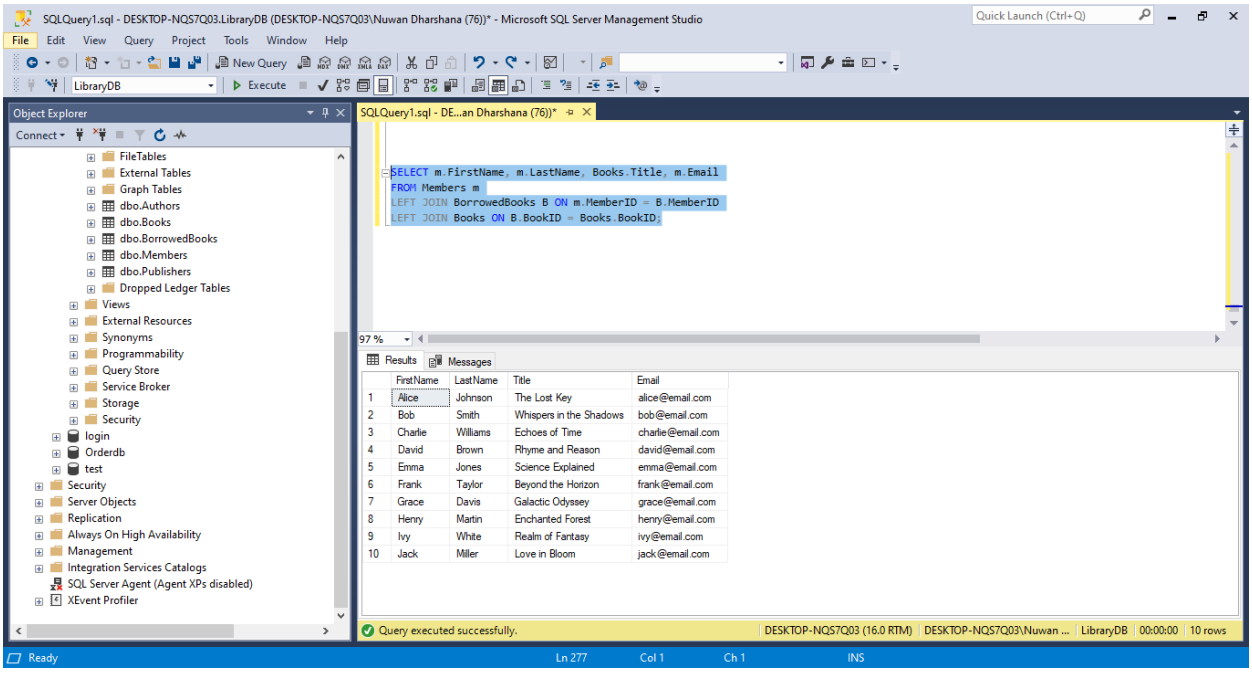


Figure 48:Left Join

This query retrieves the first name, last name, email of members along with the titles of books they have borrowed. All members are included, and for those who haven't borrowed any books, NULL values will appear in the Title column.

• Right Join

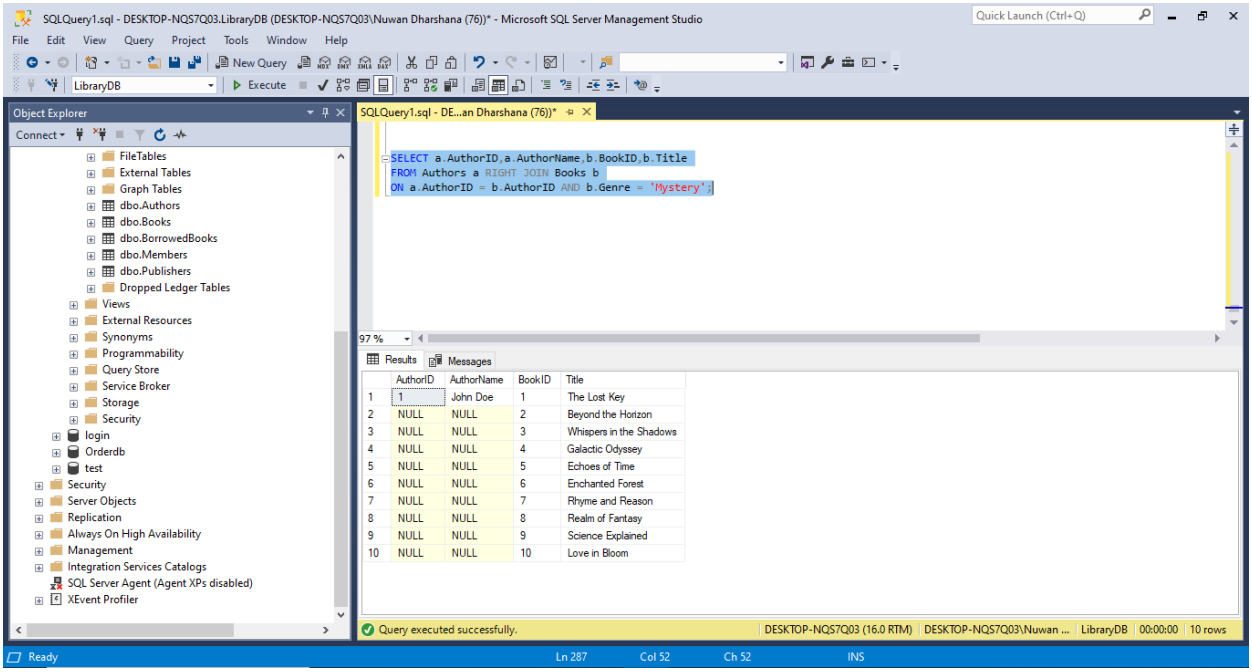


Figure 49:Right Join

This query retrieves AuthorID, AuthorName, BookID, and Title from the Authors table along with the corresponding books from the Books table, specifically for books in the 'Mystery' genre. All 'Mystery' books are included, and for those without authors, NULL values will appear in the AuthorID and AuthorName columns.

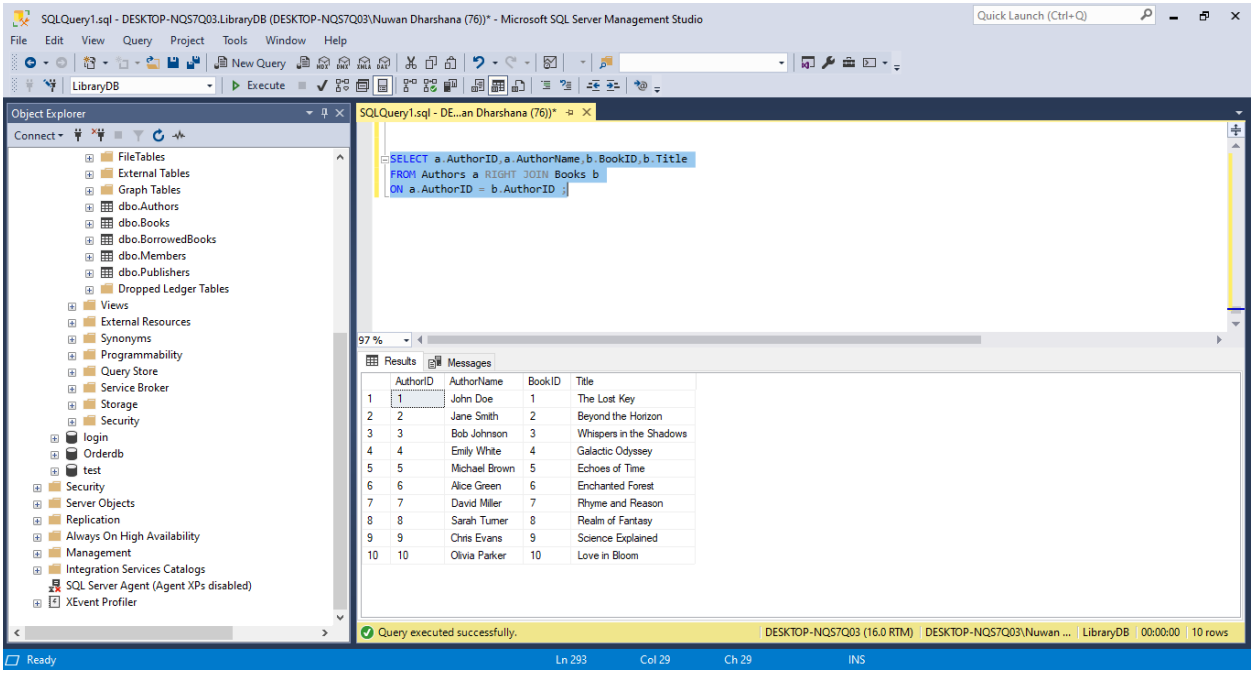


Figure 50:Right Join

it includes all books from the Books table, ensuring that all books are included, even if they don't have corresponding authors. NULL values will appear in the AuthorID and AuthorName columns for books without authors.

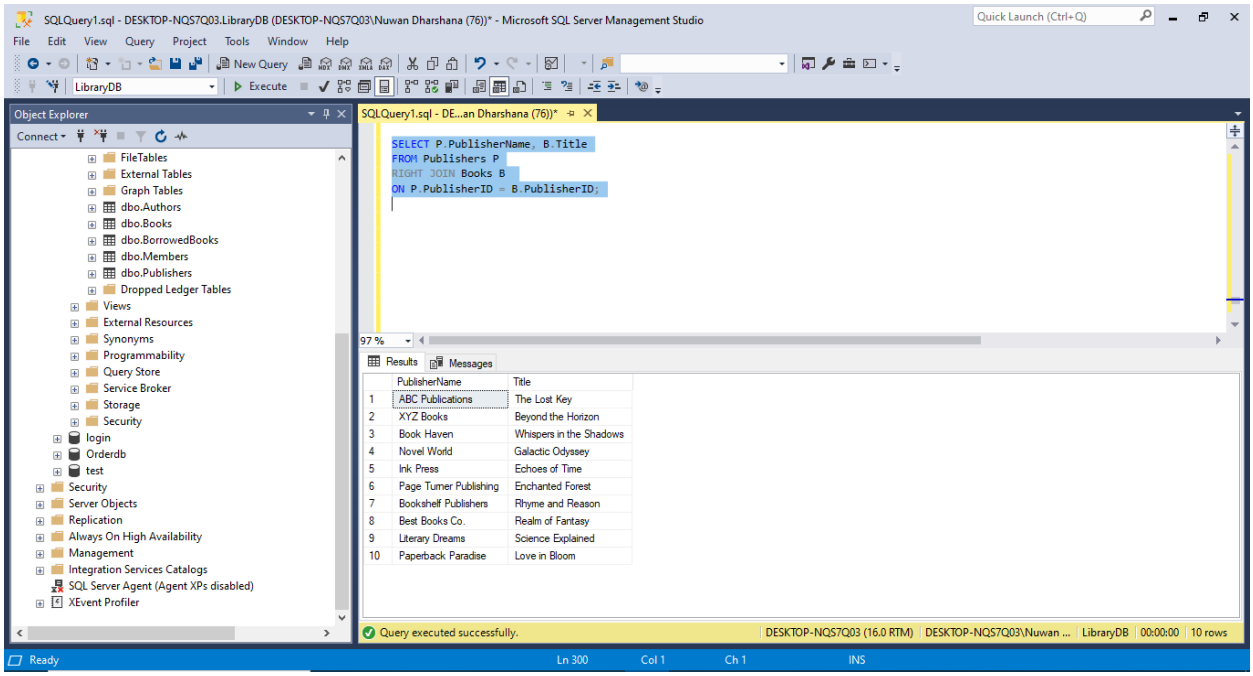


Figure 51:Right Join

This query retrieves PublisherName along with the titles of books published by each publisher. All books are included, and for those without publishers, NULL values will appear in the PublisherName column.

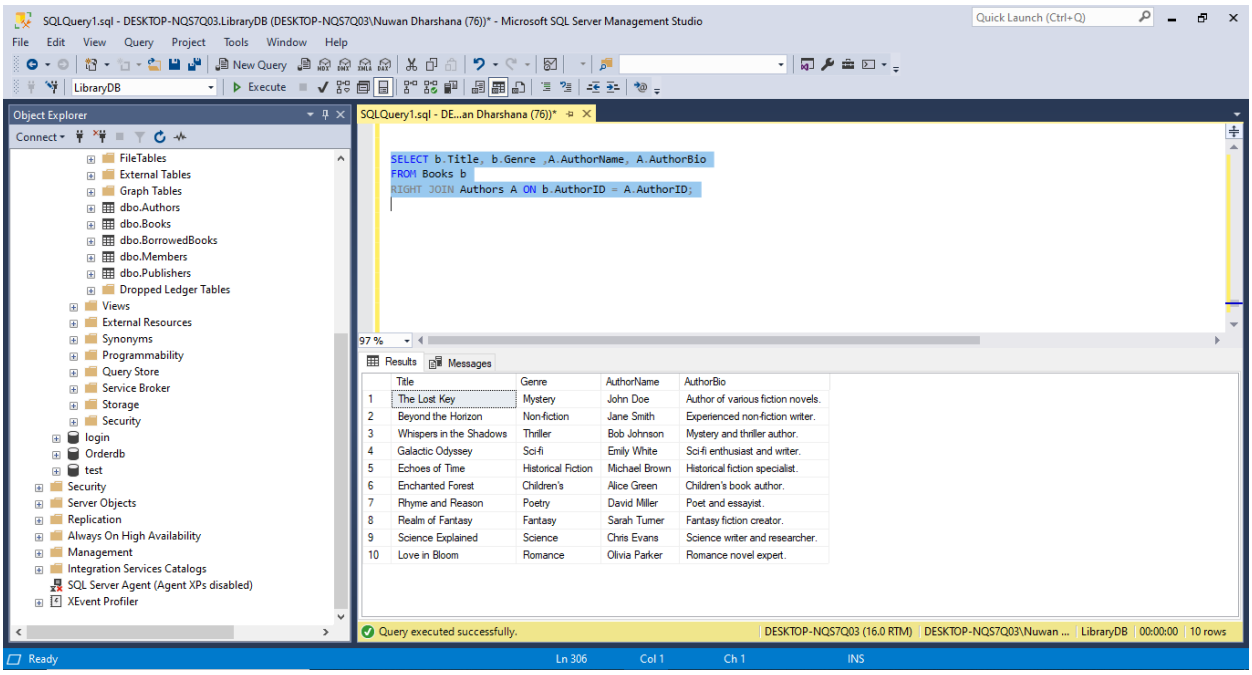


Figure 52:Right Join

This query fetches the title, genre of books along with the corresponding AuthorName and AuthorBio. All authors are included, and for those who haven't authored any books, NULL values will appear in the Title and Genre columns.

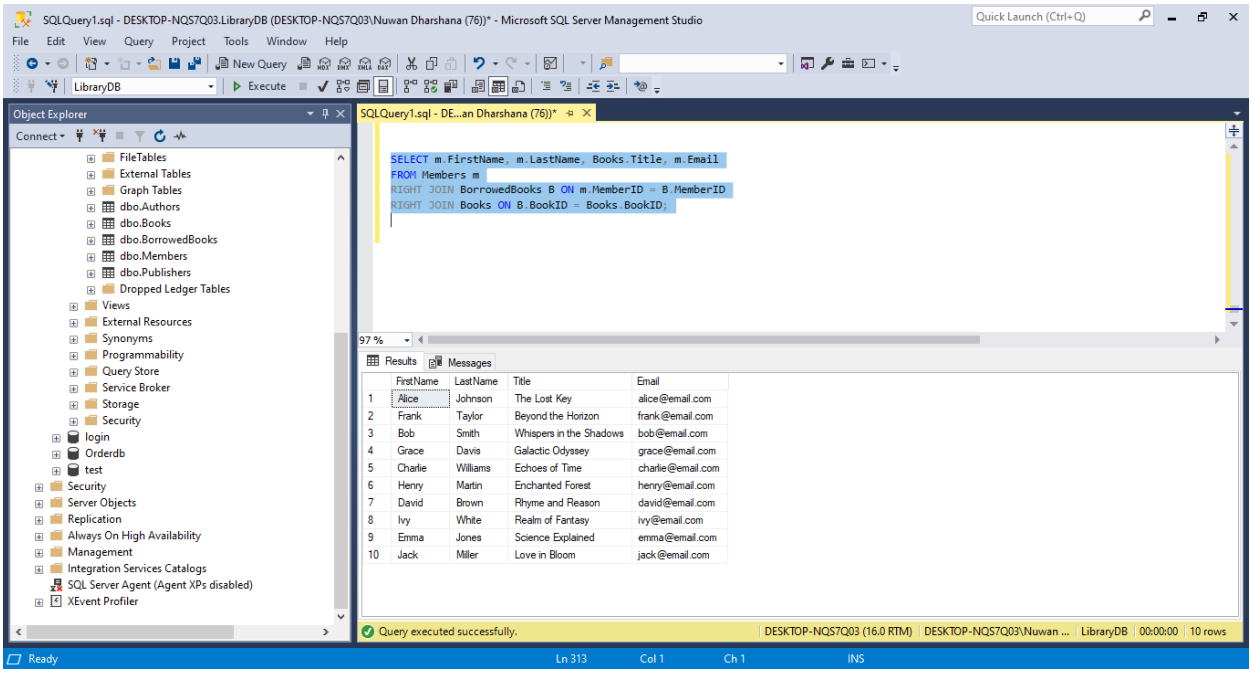


Figure 53:Right Join

This query retrieves the first name, last name, and email of members along with the titles of books they have borrowed. All books are included, and for those without corresponding borrowers, NULL values will appear in the FirstName, LastName, and Email columns.

1) • Full outer Join

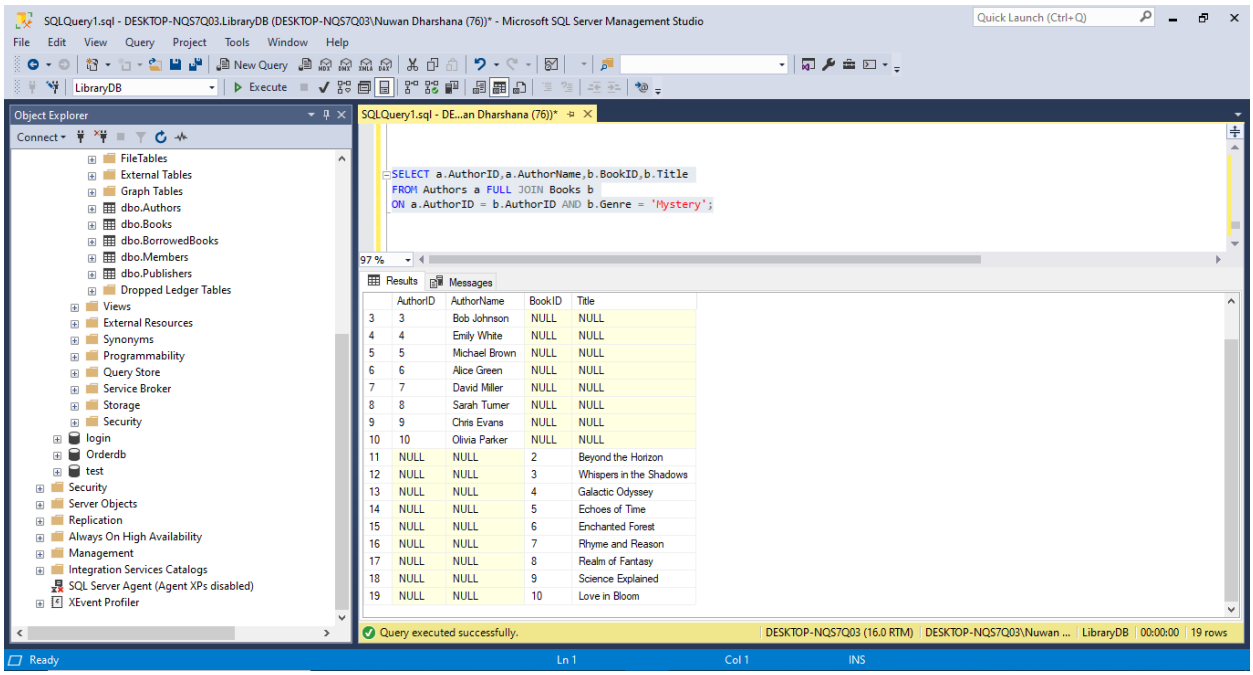


Figure 54:Full outer Join

This query retrieves AuthorID, AuthorName, BookID, and Title from the Authors table along with the corresponding books from the Books table, specifically for books in the 'Mystery' genre. All authors and books are included, and for those without matches, NULL values will appear in the AuthorID, AuthorName, BookID, and Title columns.

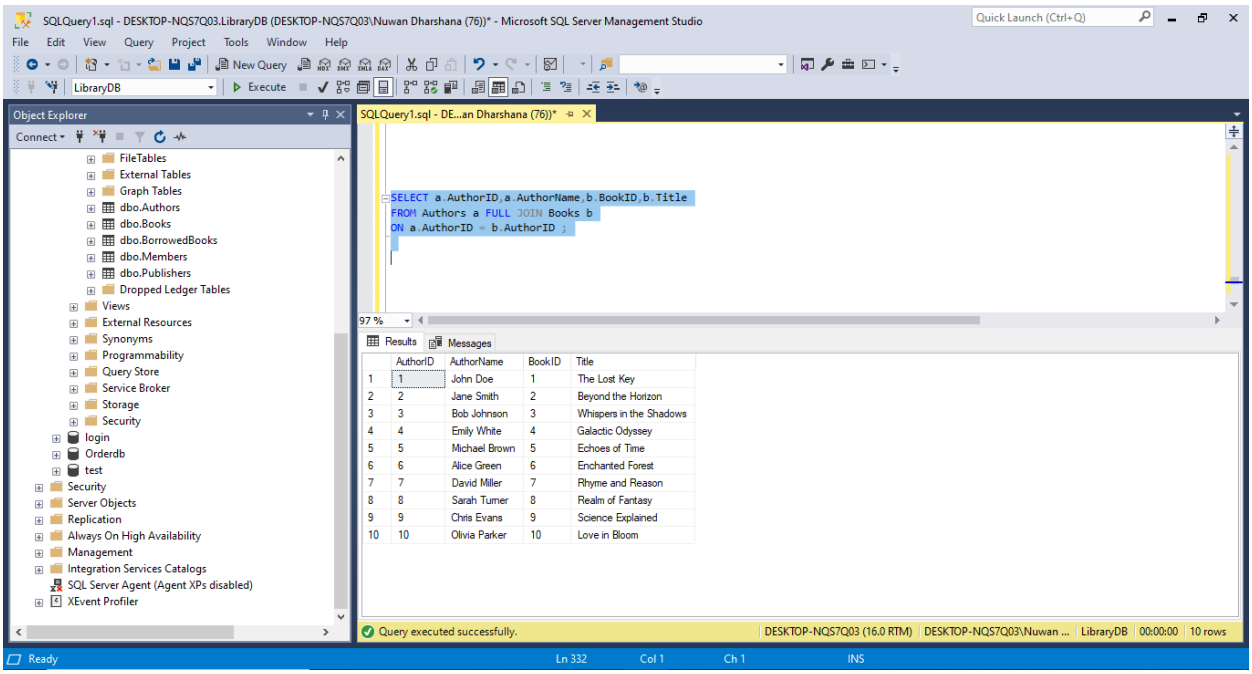


Figure 55:Full outer Join

books from the Books table and all authors from the Authors table. All authors and books are included, and for those without matches, NULL values will appear in the AuthorID, AuthorName, BookID, and Title columns.

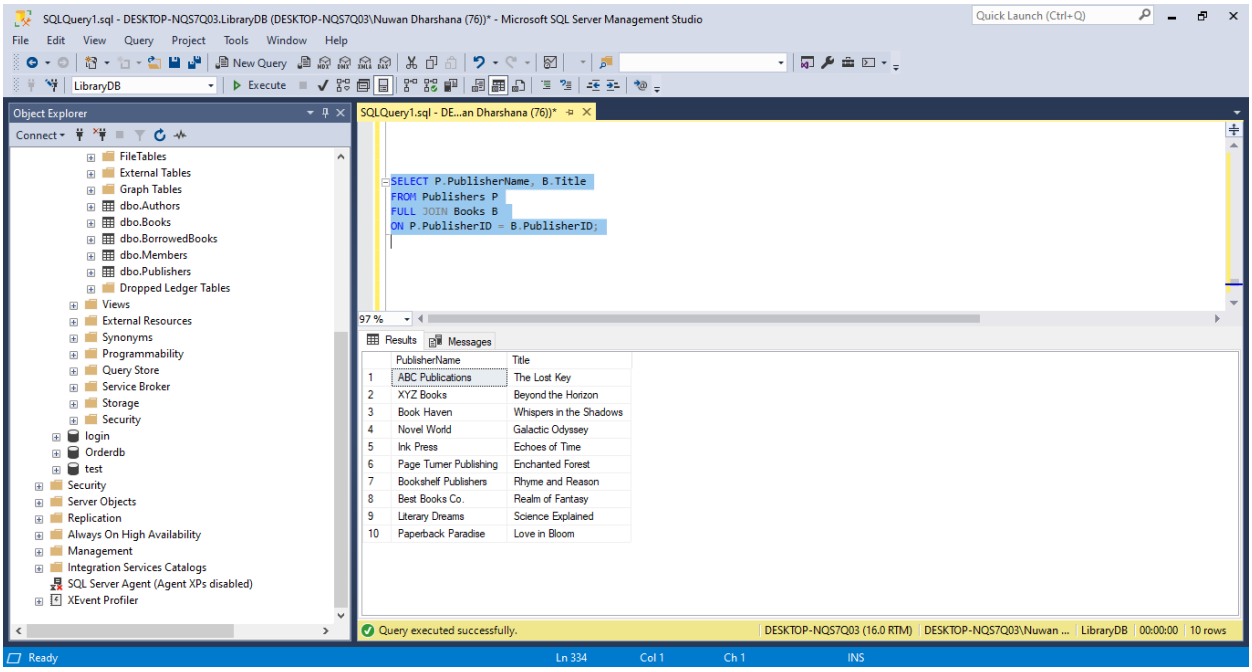


Figure 56:Full outer Join

This query retrieves PublisherName along with the titles of books published by each publisher. All publishers and books are included, and for those without matches, NULL values will appear in the PublisherName and Title columns.

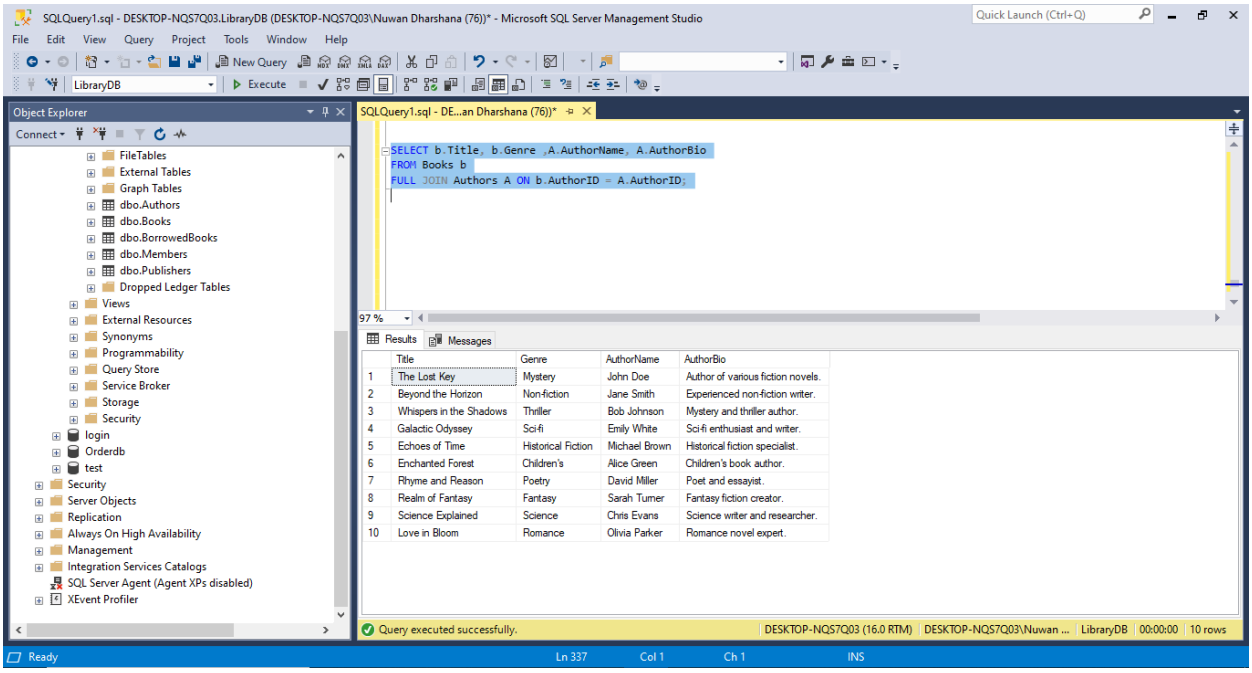


Figure 57:Full outer Join

This query fetches the title, genre of books along with the corresponding AuthorName and AuthorBio. All books and authors are included, and for those without matches, NULL values will appear in the Title, Genre, AuthorName, and AuthorBio columns.

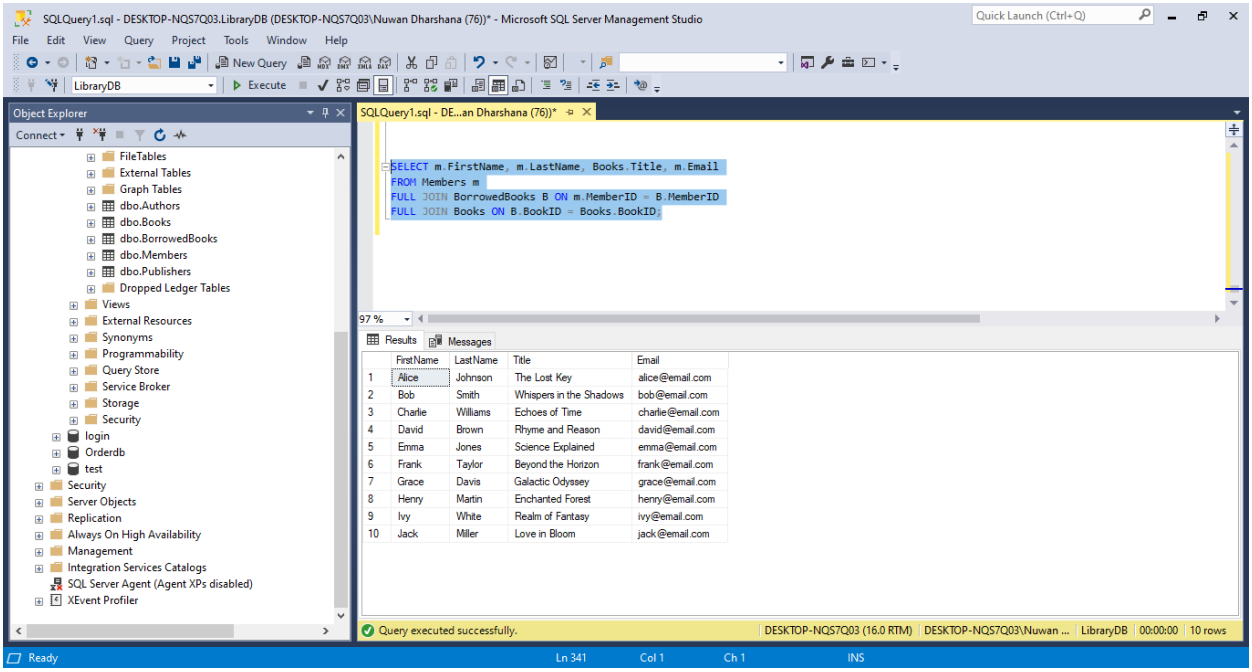


Figure 58:Full outer Join

This query retrieves the first name, last name, and email of members along with the titles of books they have borrowed. All members and books are included, and for those without matches, NULL values will appear in the FirstName, LastName, Title, and Email columns.

6) Create at least one store procedure with passing parameters per table created

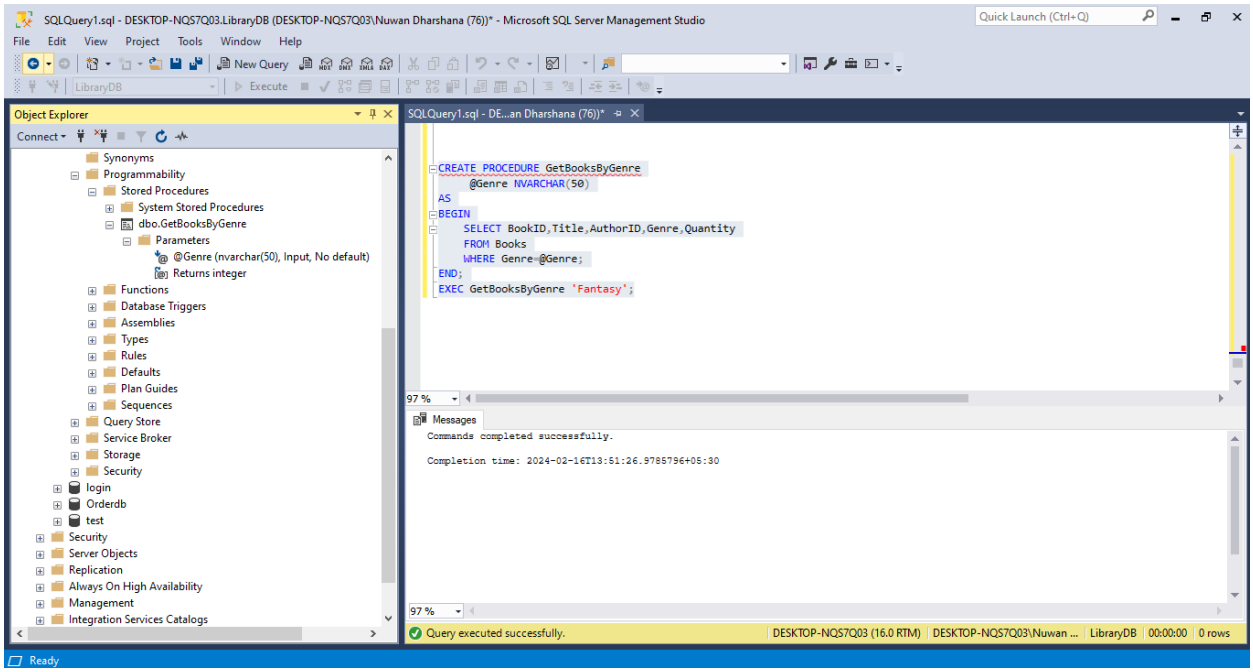


Figure 59:Create Store Procedure

Gets books from the database based on a specific genre.

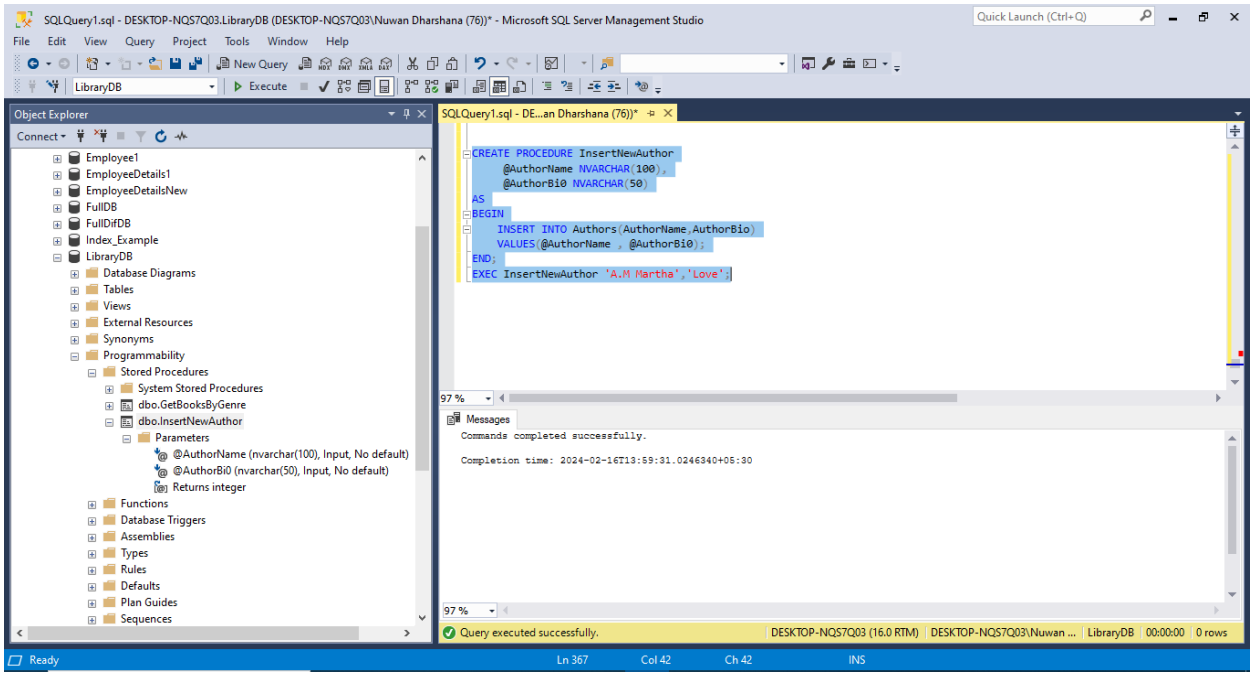


Figure 60:Create Store Procedure

Inserts a new author into the Author table with the provided name and bio.

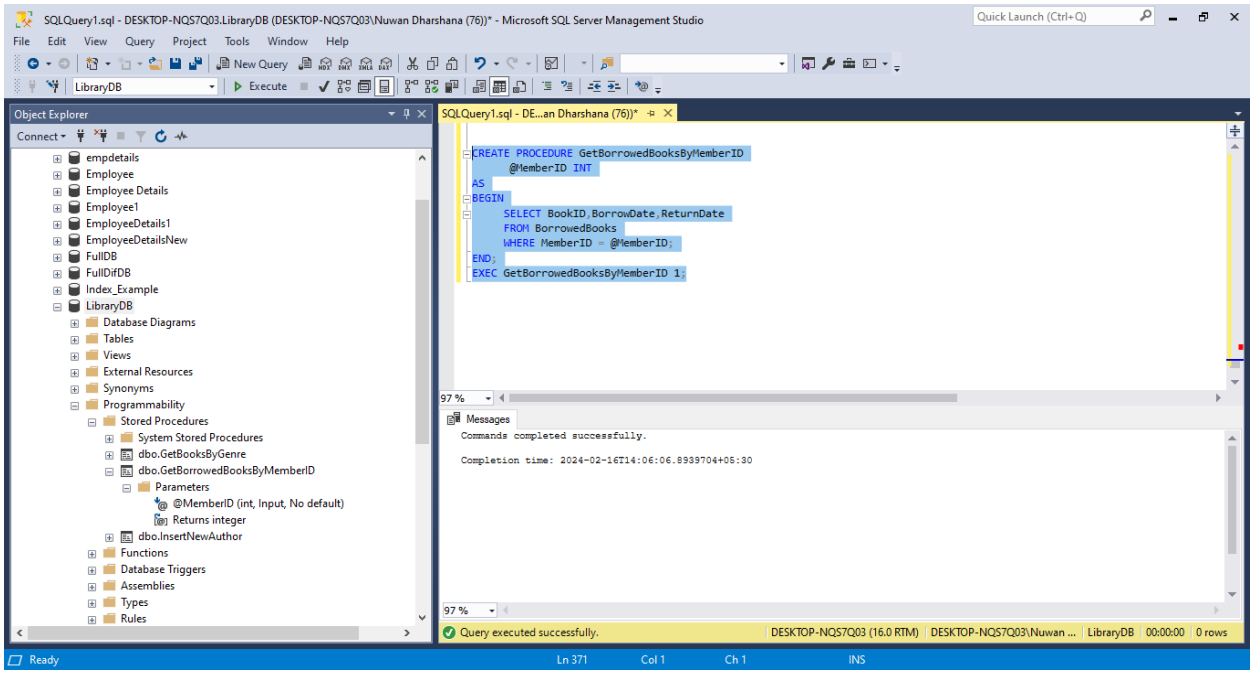


Figure 61:Create Store Procedure

Gets the borrowed books information for a given member ID.

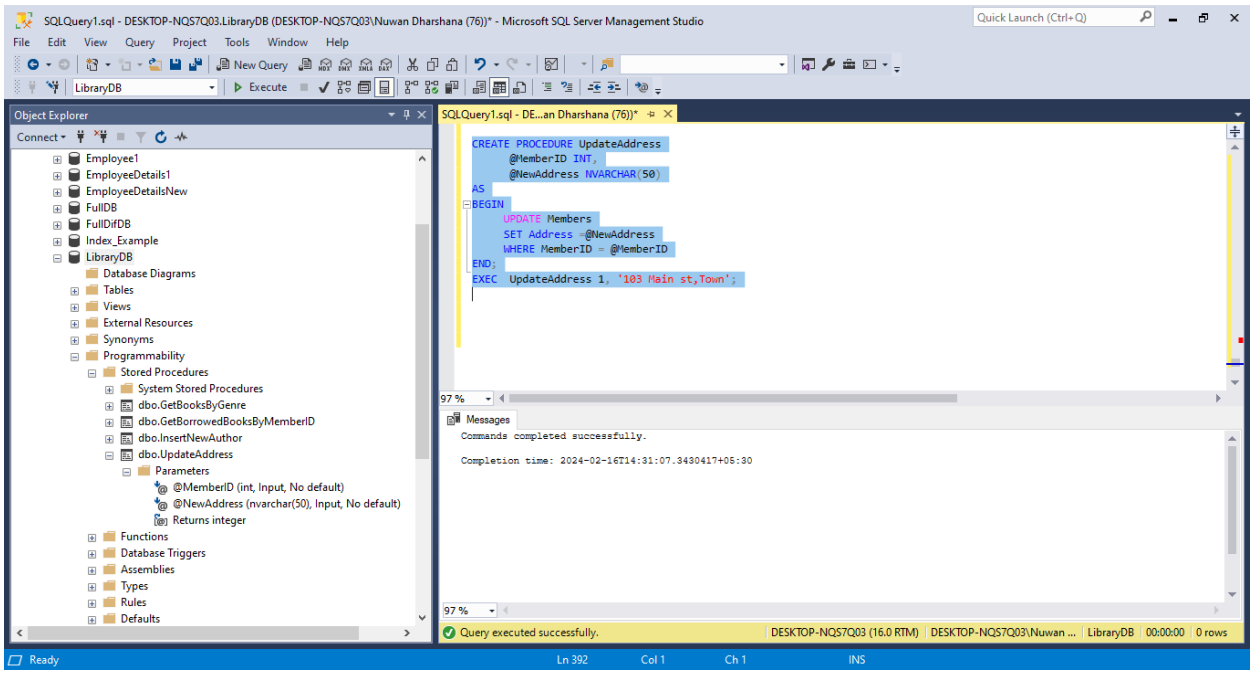


Figure 62:Create Store Procedure

Updates a member's address in the members table with a new address.

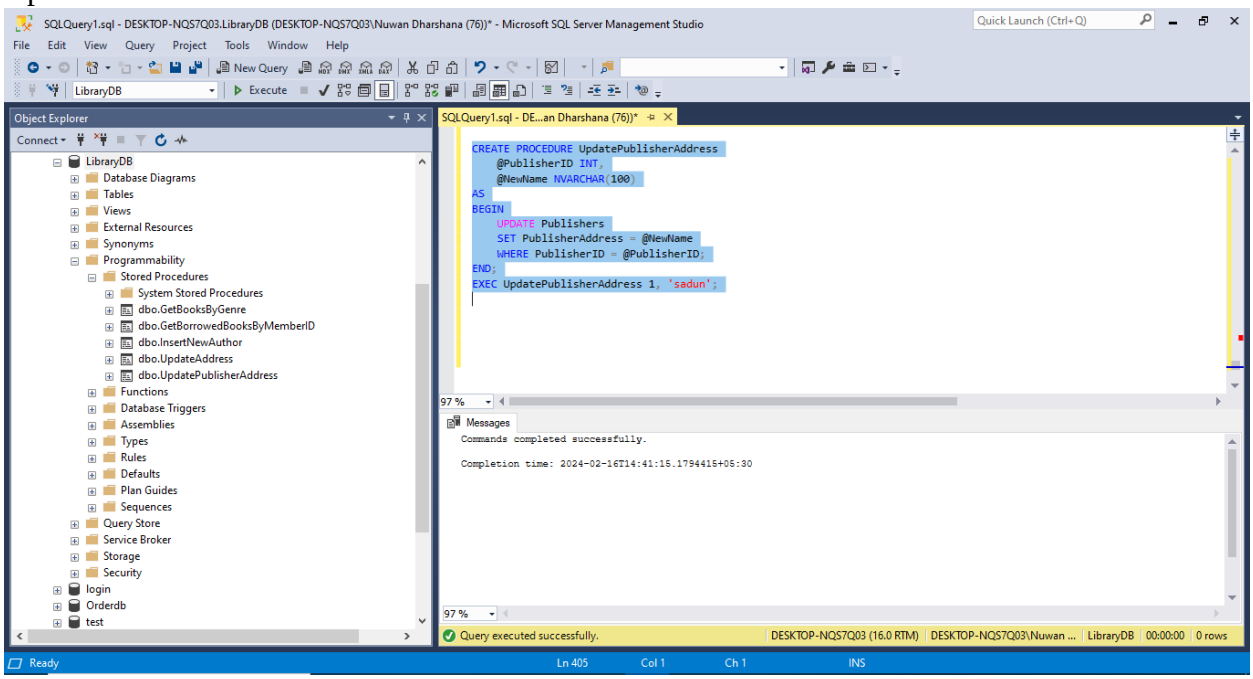


Figure 63:Create Store Procedure

Updates a publisher's address in the publishers table with a new address.

7) Create at least one view per table created.

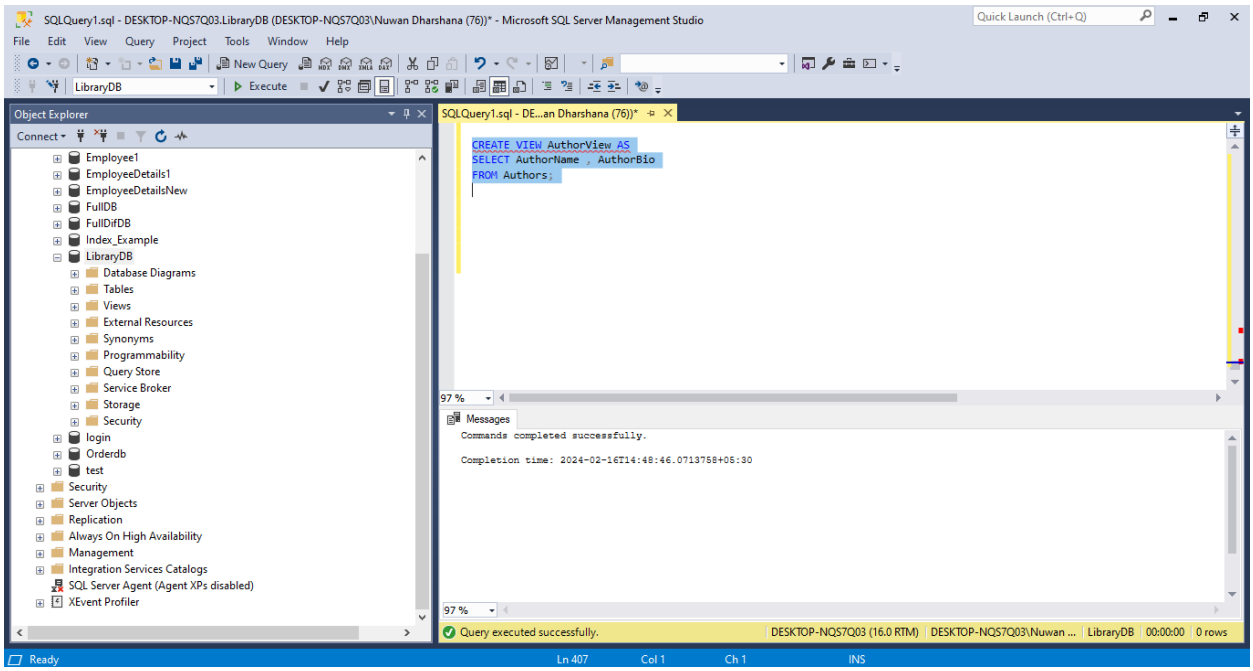


Figure 64:Create View

The Author Table presents a view of the authors' names and biographies.

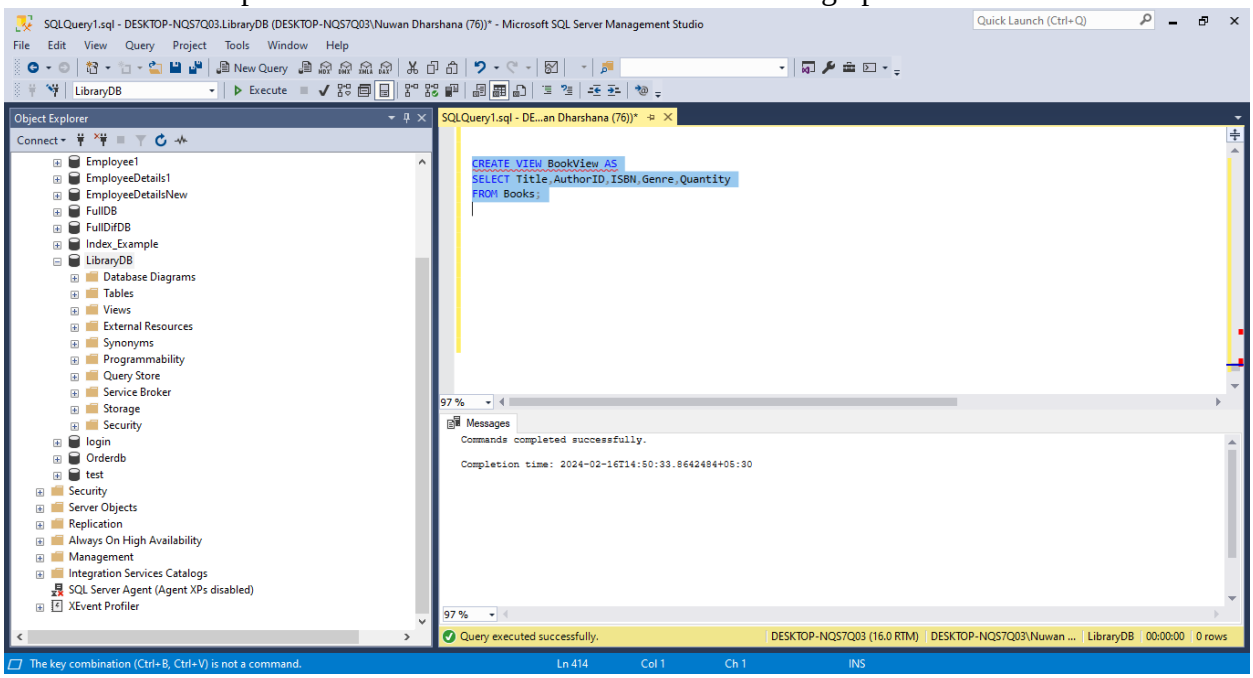


Figure 65:Create View

The book table provides a view of book titles, author IDs, ISBNs, genres and sizes.

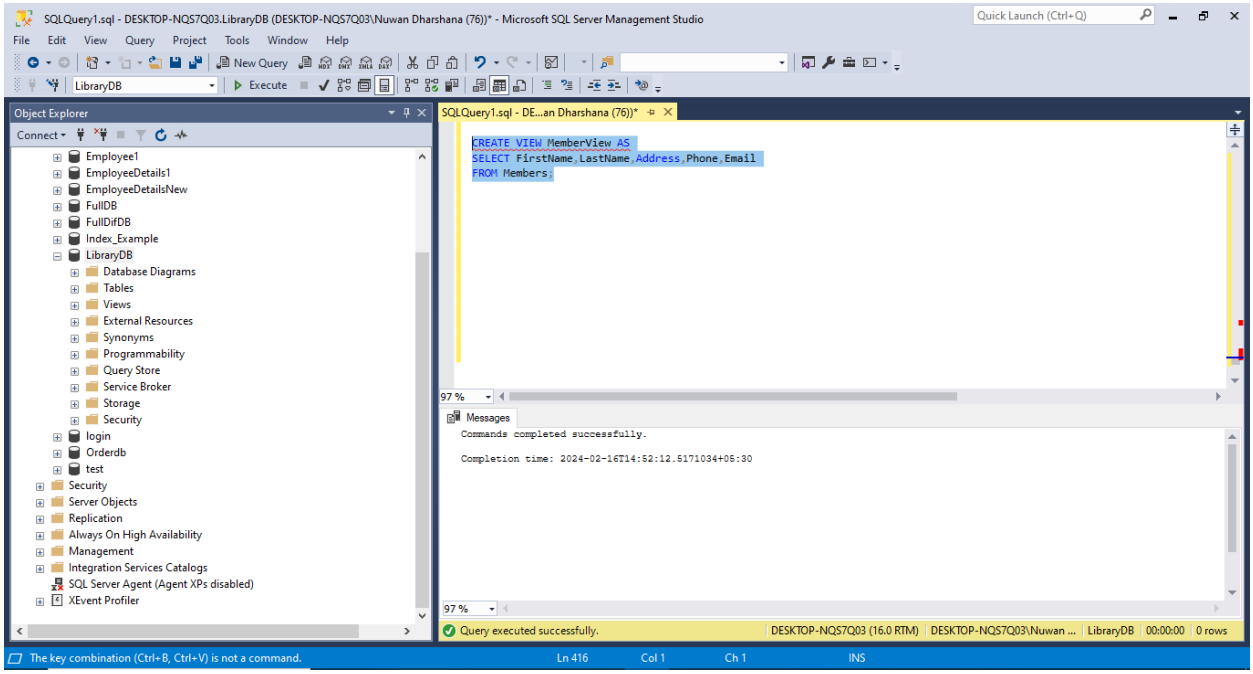


Figure 66:Create View

Displays members' first names, last names, addresses, phone numbers, and email addresses from the Members table.

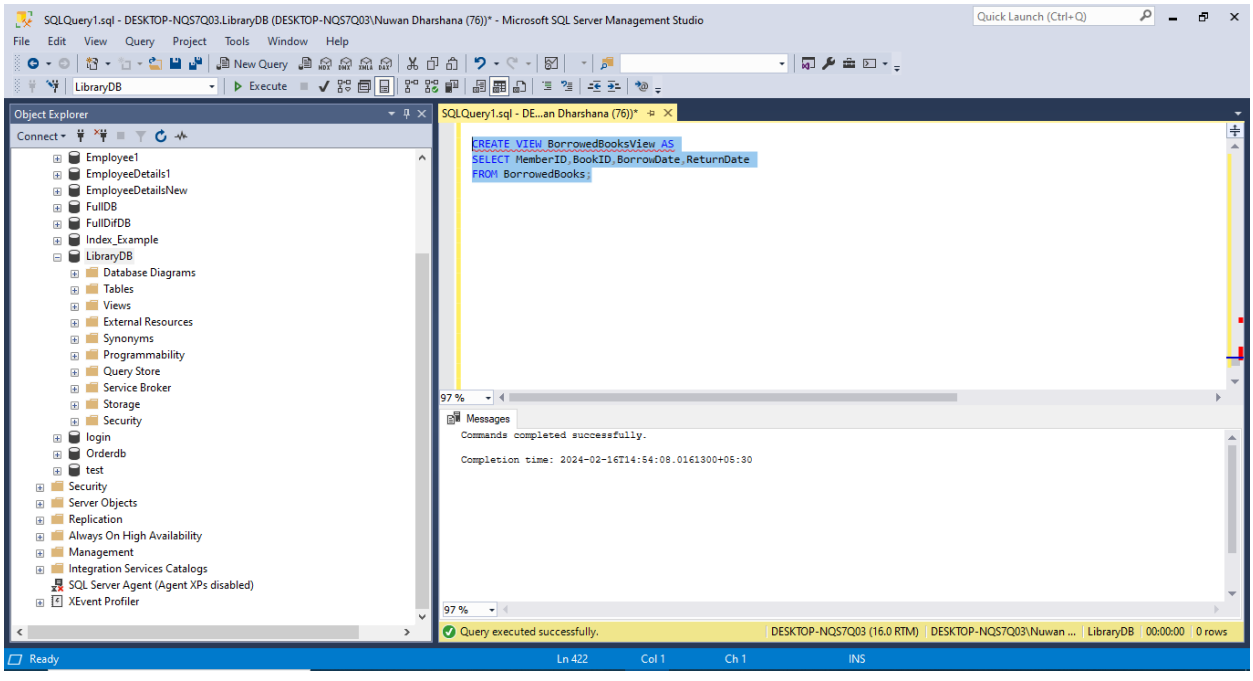


Figure 67:Create View

BorrowedBooks table displays a view of Member IDs, Book IDs, Borrowed Dates and Returned Dates.

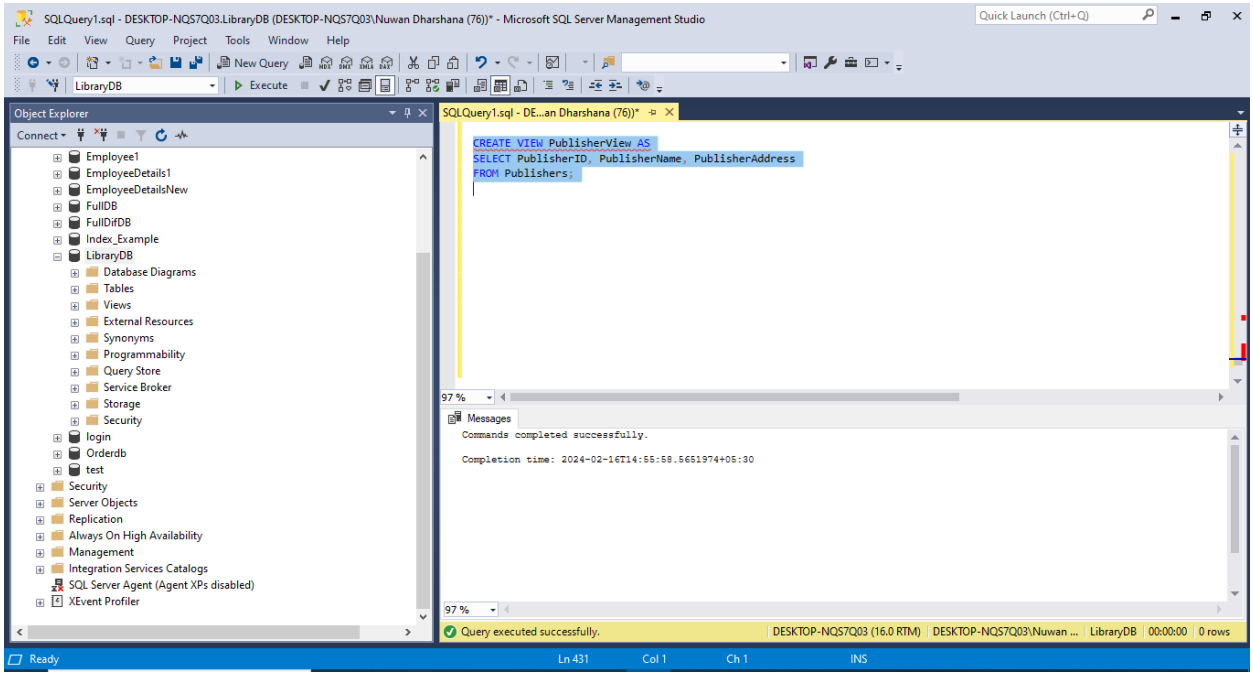


Figure 68:Create View

Offers a view of Publisher IDs, Publisher Names and Publisher Addresses from the Publishers table.

8) Create two logins to connect to a SQL Server Instance.

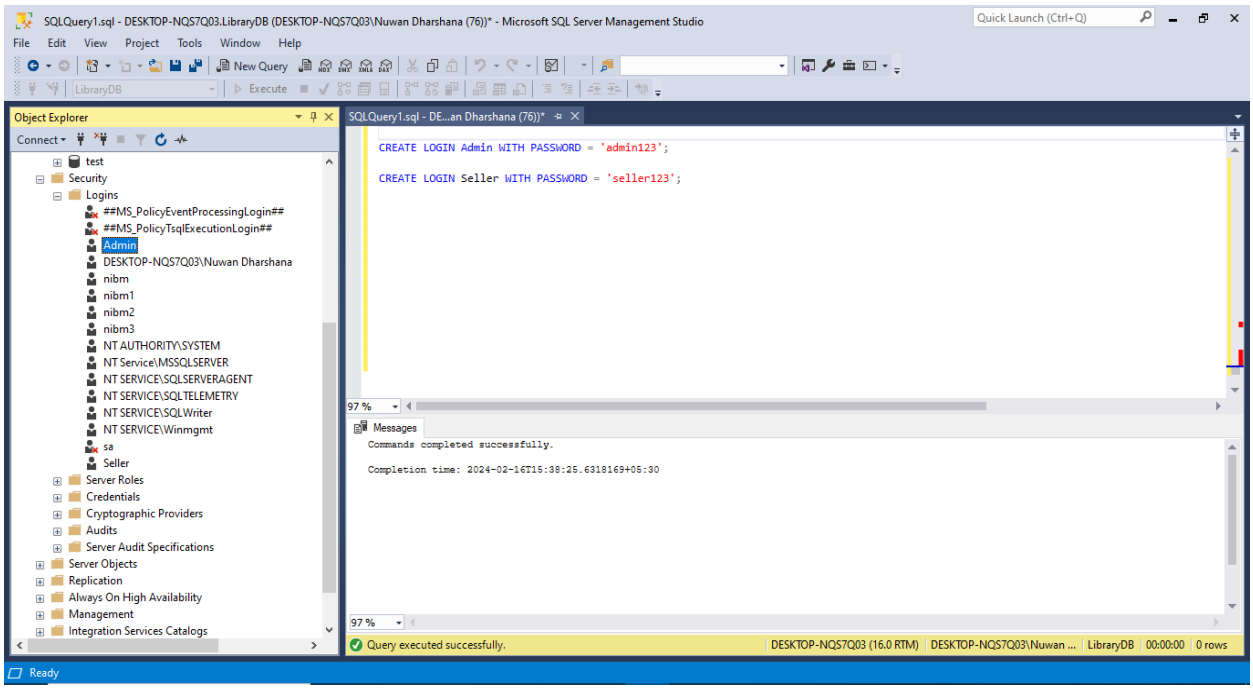


Figure 69:Create Two Logins

'admin123'- This command creates a login named "Admin" with password "admin123". This login can be used to authenticate and access the database system with administrator privileges.

'seller123'- This command creates a login named "Seller" with password "seller123". This login can be used to authenticate and access the database system with vendor-specific privileges.

9) Create two users for each login and allowed and deny some permission for each user category.

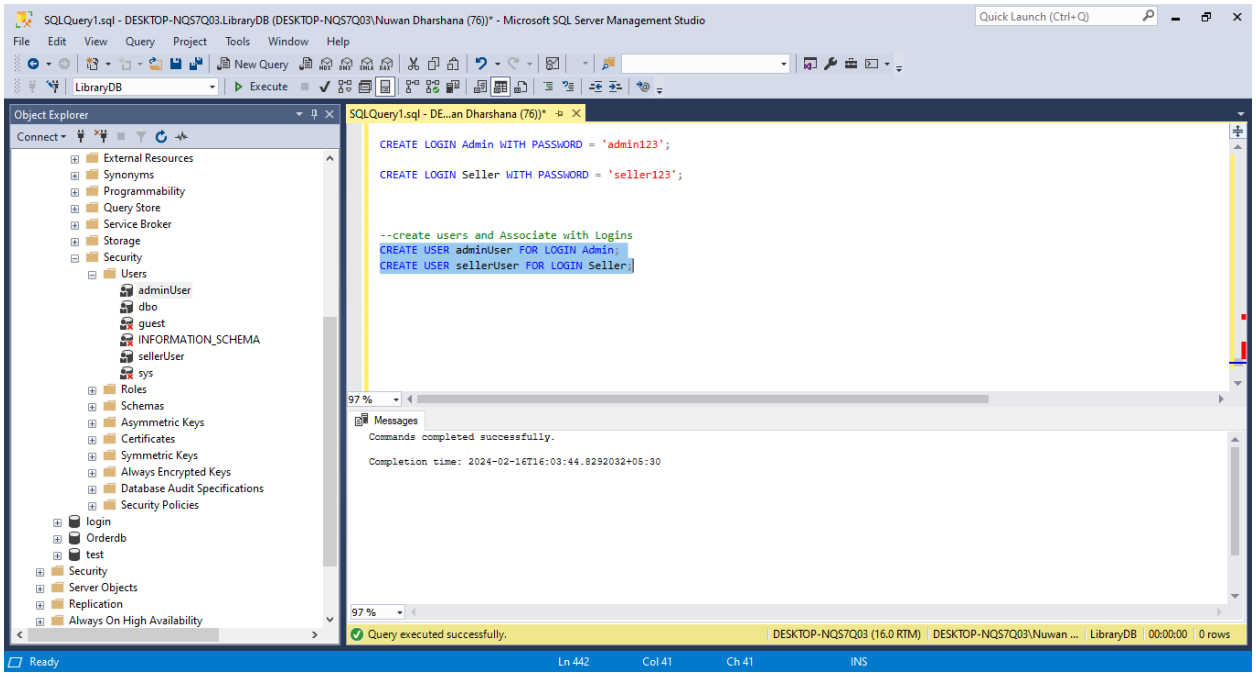


Figure 70:Create Two Logins

Creates a user named "adminUser" associated with the "admin" login.

Creates a user named "sellerUser" associated with the "Seller" login.

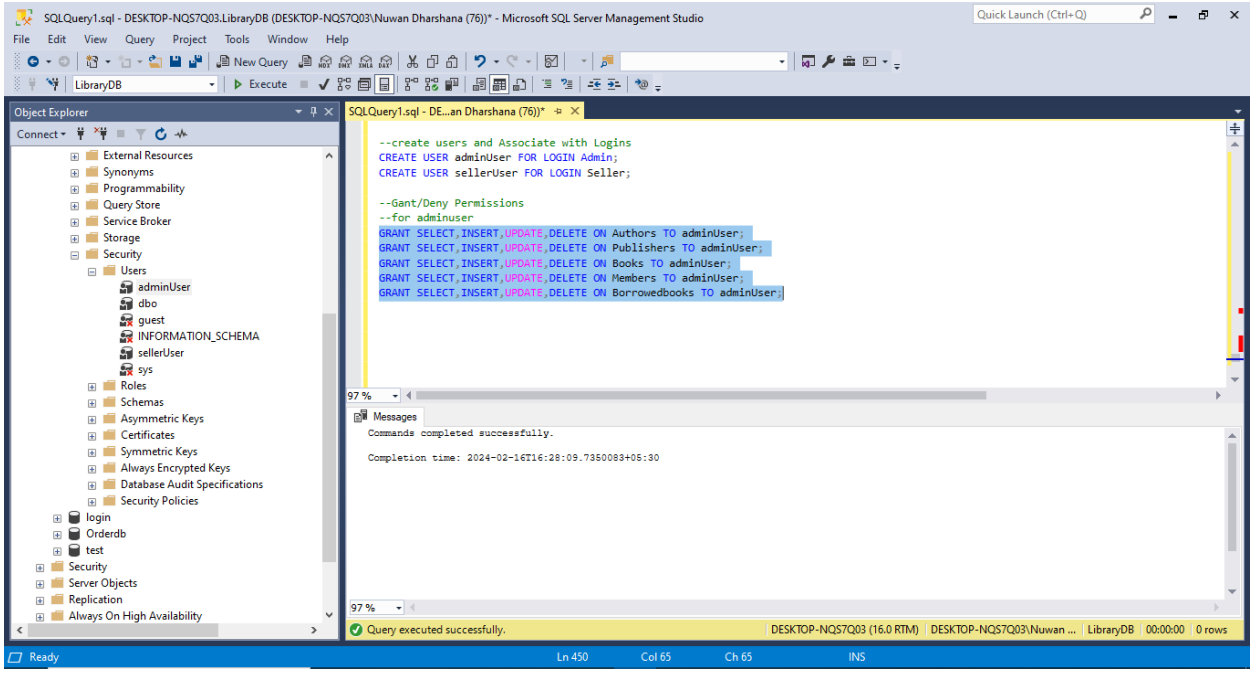


Figure 71:Create Two Logins

Granting Select, Insert, Update, Delete permissions on various tables (Authors, Publishers, Books, Members, Borrowed Books) to admin.

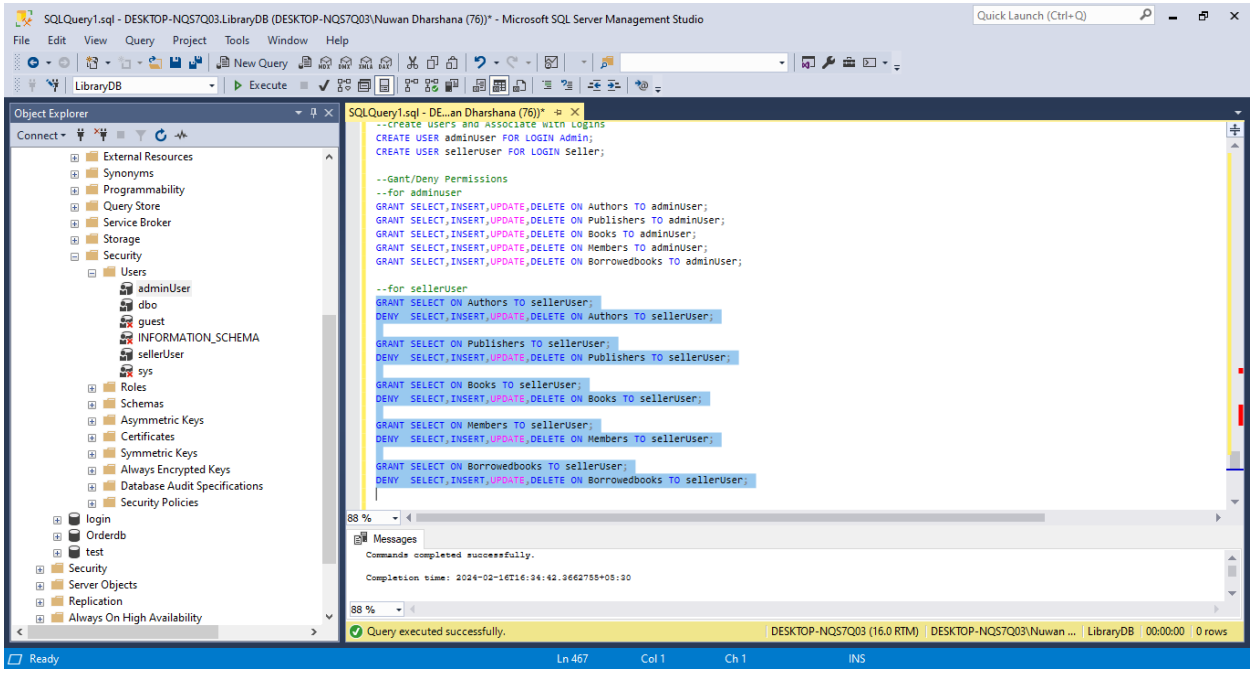


Figure 72::Create Two Logins

The vendor allows the user to select on various tables (Authors, Publishers, Books, Members, Borrowed Books) but denies any insert, update, delete operations on these tables for the vendor user.

10) Create two database level roles and grant permission for the users using created roles.

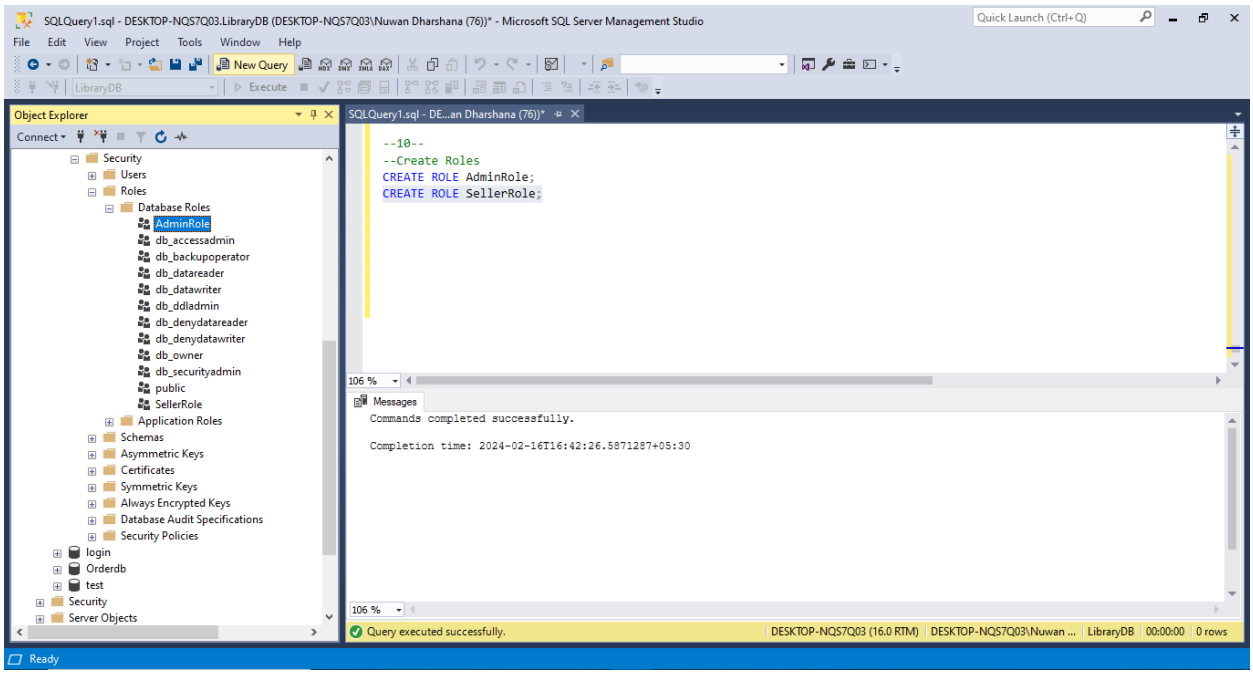


Figure 73:grant permission

Creates a role named "AdminRole" in the database.

Creates a role named "SellerRole" in the database.

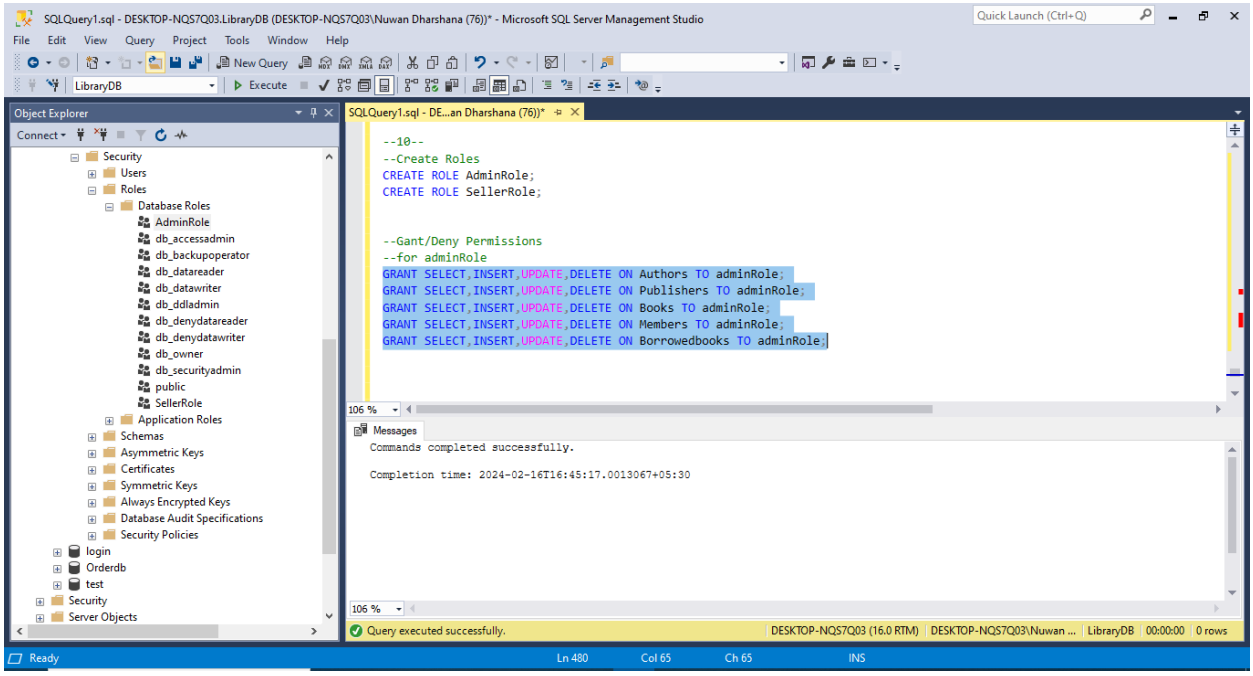


Figure 74:grant permission

Grant the admin role Select, Insert, Update, Delete permissions on various tables (Authors, Publishers, Books, Members, Borrowed Books). This allows users assigned to this role to perform these operations on specific tables.

11) Create new cluster index on each table with considering appropriate field.

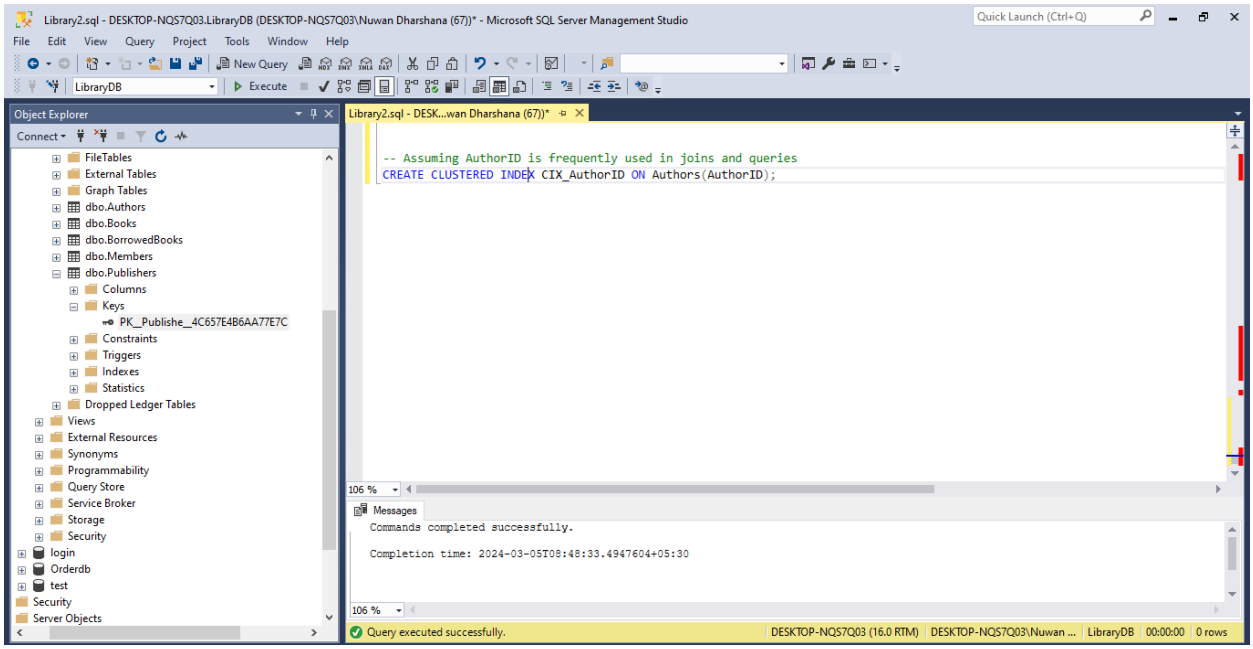


Figure 75:Create Cluster Index

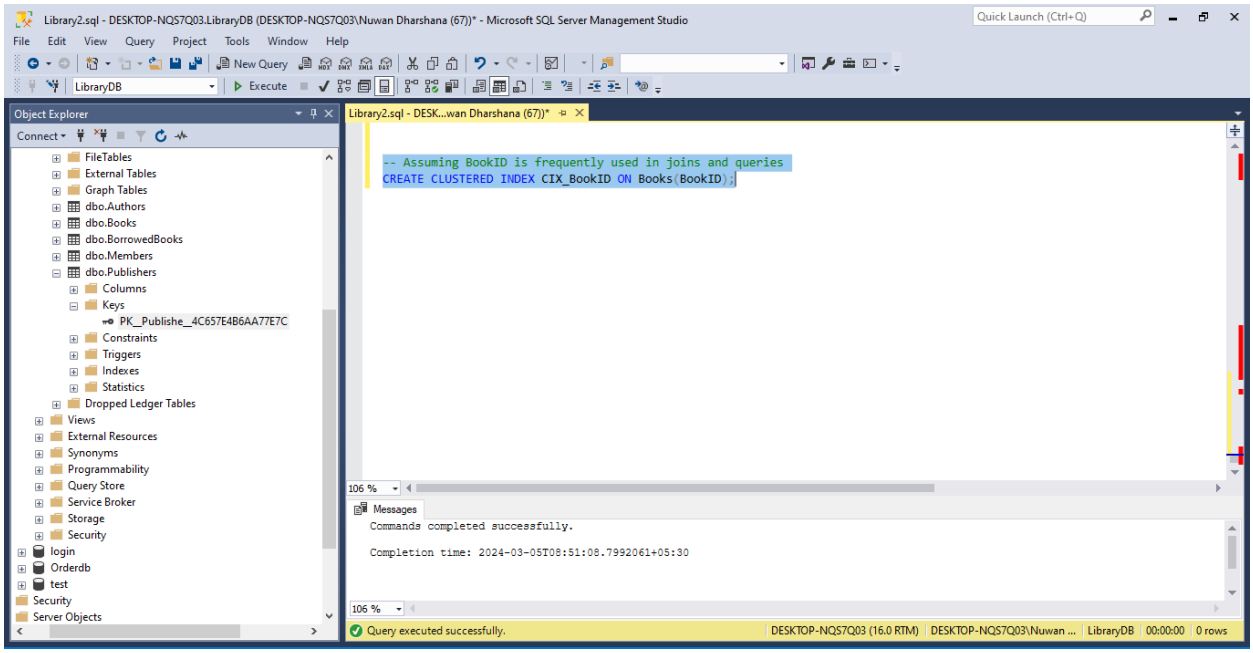


Figure 76:Create Cluster Index

12) Create two non-cluster indexes on each table

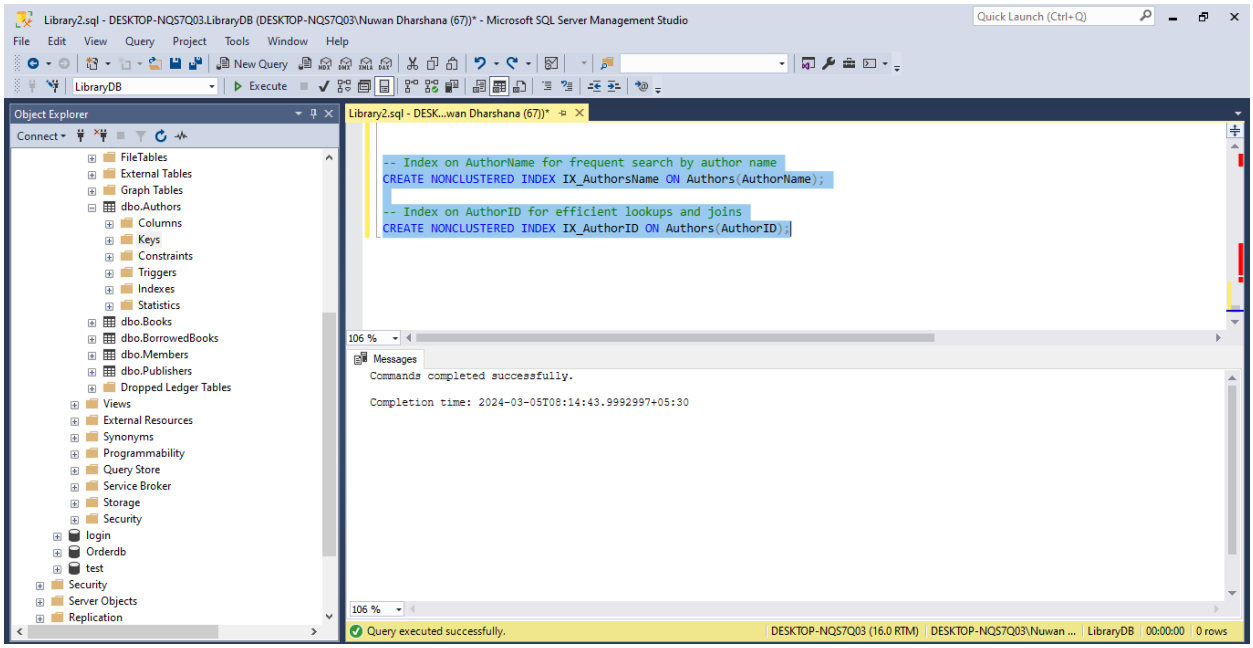


Figure 77:Create Noncluster Index

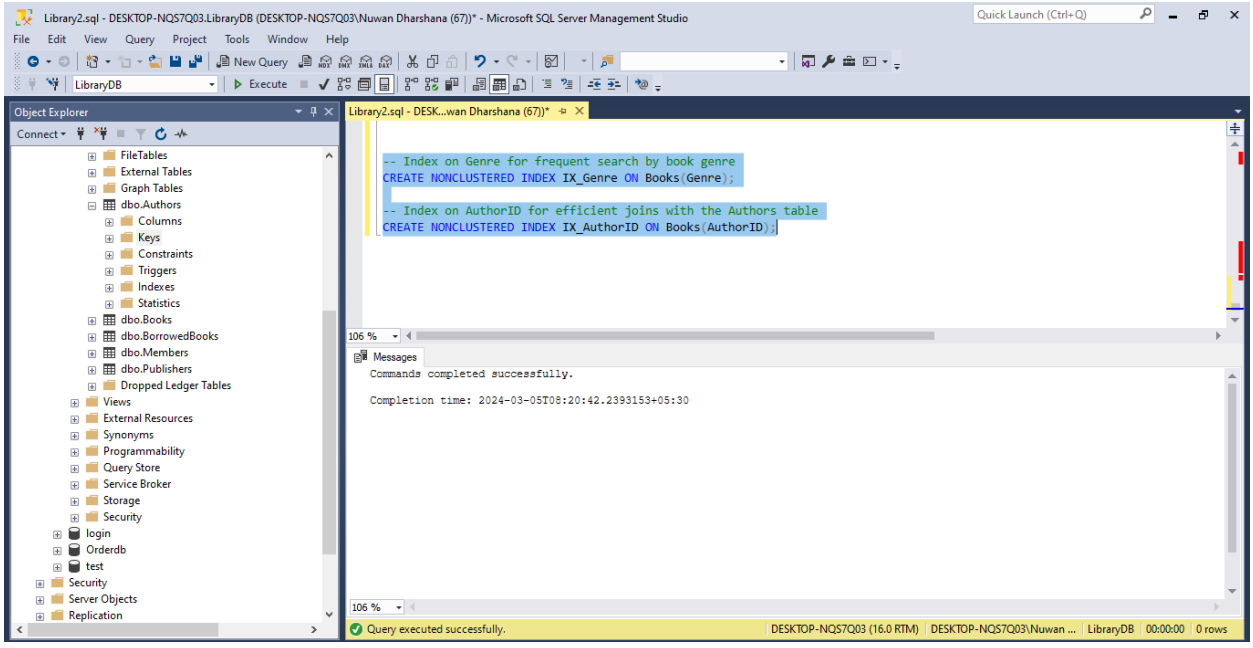


Figure 78:Create noncluster Index

13) Explain execution plan with considering two example queries.

An execution plan in the context of a relational database is a set of operations and steps that the database management system (DBMS) uses to retrieve or manipulate data based on a given SQL query.
Example Query 1: Simple SELECT Query
Let's say have a simple SELECT query that retrieves information from the Books table:

```
SELECT Title, Genre
FROM Books
WHERE AuthorID = 1;
```

Execution Plan Explanation:

Table Scan/Full Table Scan: The DBMS may perform a table scan, which means it will read all rows of the Books table.
It applies a filter to select rows where AuthorID is equal to 1. This narrows down the results.
The DBMS retrieves only the columns specified in the SELECT clause (Title and Genre).

Result Set: The final result set is returned to the user. Example Query 2: JOIN Query
Now, let's consider a more complex query involving a JOIN operation between the Books and Authors tables:

```
SELECT b.Title, a.AuthorName
FROM Books b
INNER JOIN Authors a ON b.AuthorID = a.AuthorID
WHERE b.Genre = 'Fantasy';
```

14) Create Full backup and restore it into another database.

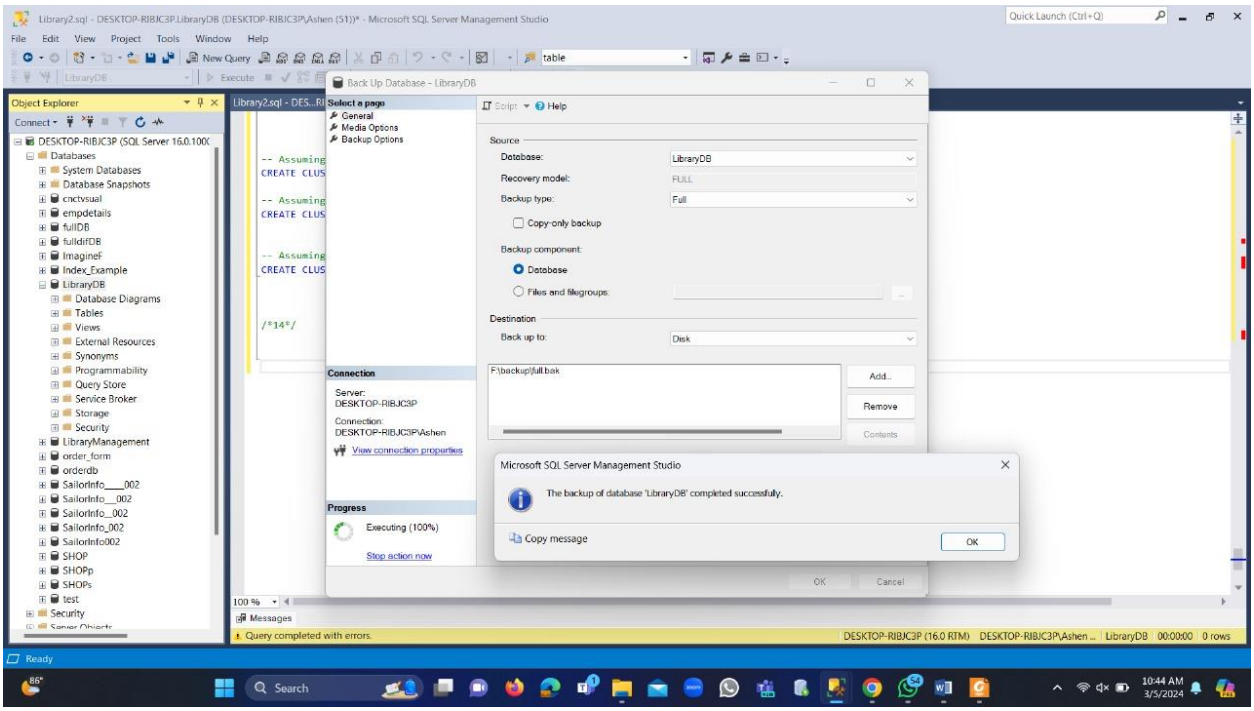


Figure 79:Full Backup

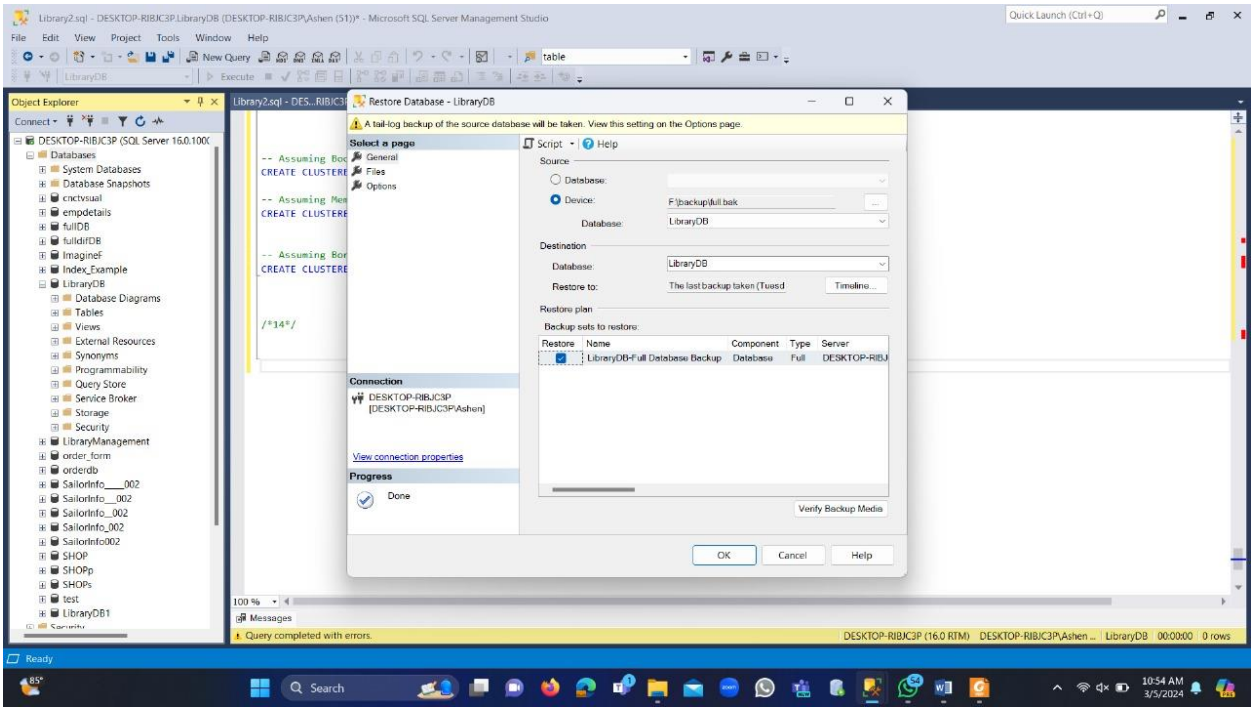


Figure 80:Full Backup

15) Add some changes on existing database and create a differential backup

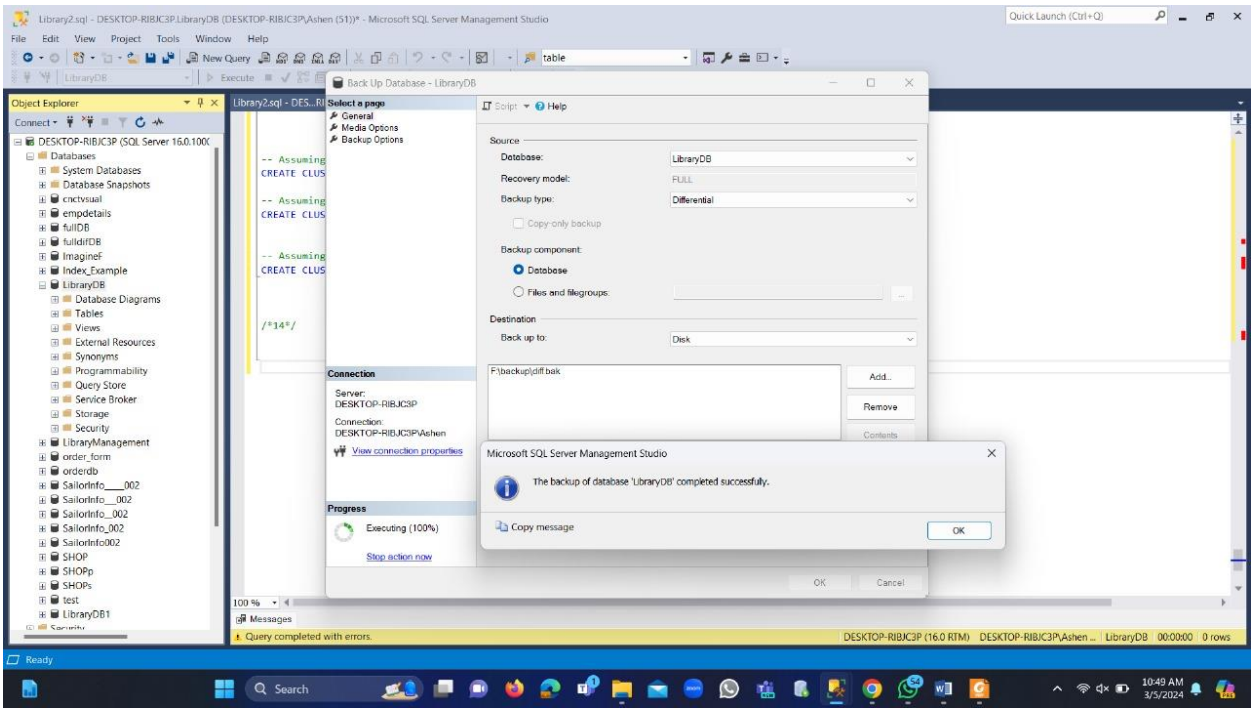


Figure 81:differential backup

16) Create another new database with restoring differential backup.

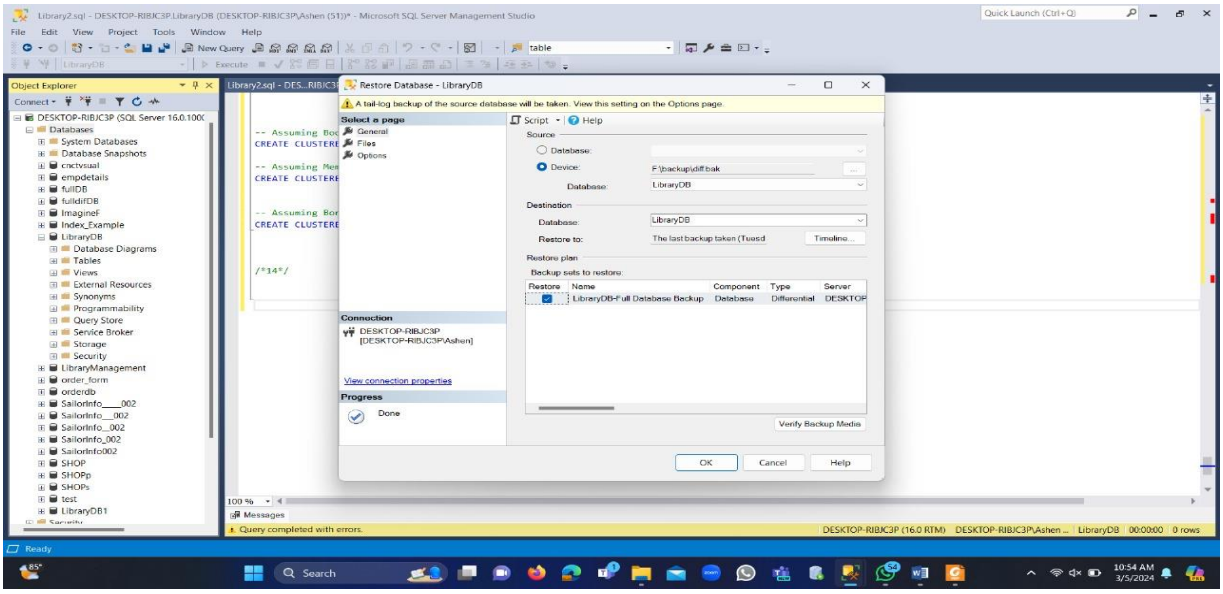


Figure 82:restoring differential backup