

# **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”

<b>Index no</b>	<b>Name</b>	<b>Signature</b>
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 .....	6
LIST OF FIGURES .....	7
ER DIAGRAM .....	9
Chapter 1 .....	10
Introduction.....	10
1.1    Background.....	10
1.2 Problem Statement.....	10
Aim and Objectives.....	11
CHAPTER 2 .....	12
Design and Result .....	12

## LIST OF TABLES

Table 1:Authors Table.....	12
Table 2:Publishers Table .....	12
Table 3:Books Table.....	12
Table 4:Members Table.....	13
Table 5:BorrowedBooks Table.....	13

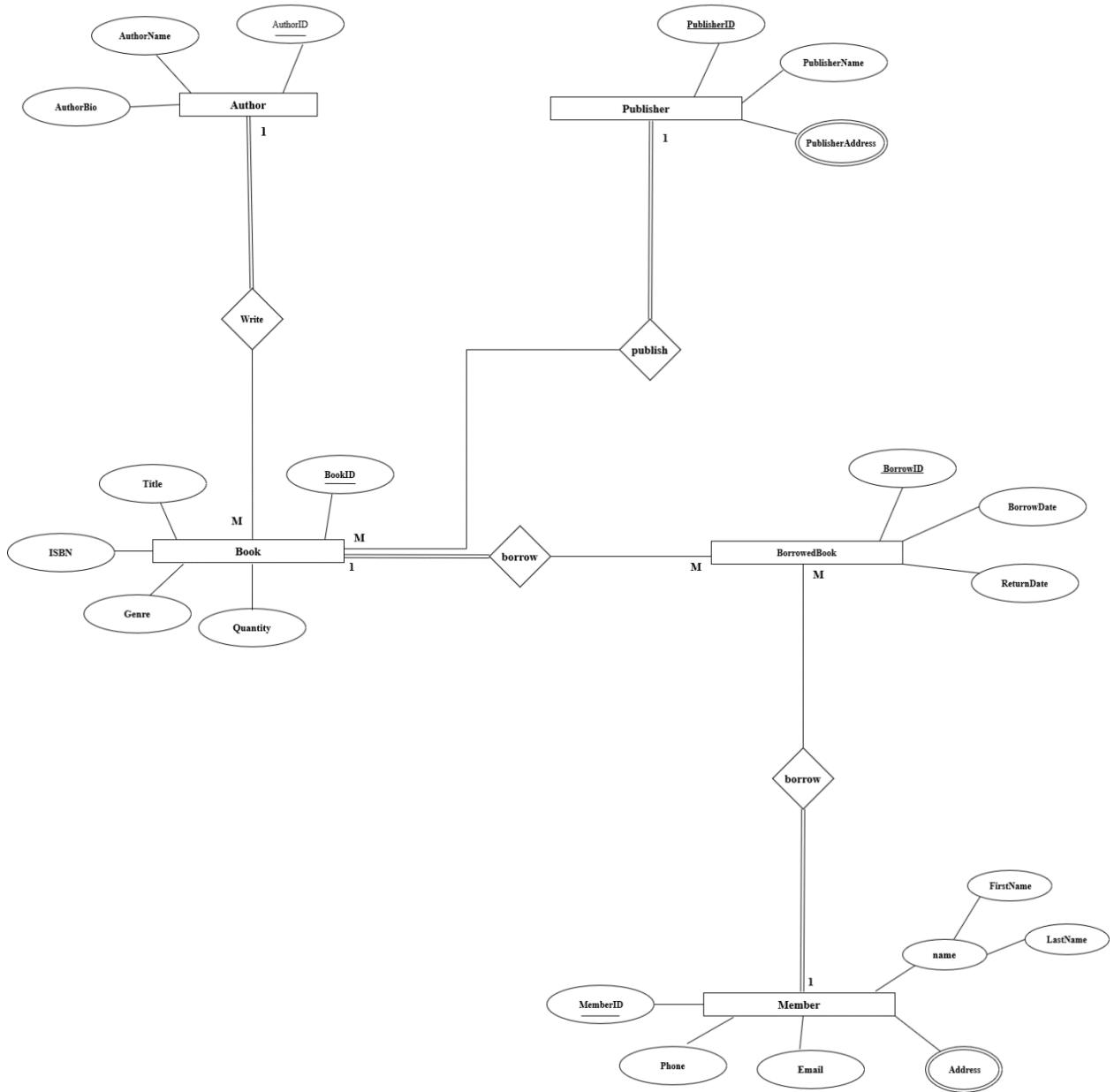
## LIST OF FIGURES

Figure 1:Create Database .....	14
Figure 2:Use Database .....	15
Figure 3:Create Tables .....	15
Figure 4:Insert Data .....	16
Figure 5:Insert Data .....	16
Figure 6:Insert Data .....	17
Figure 7:Insert Data .....	17
Figure 8:Insert Data .....	18
Figure 9:DQL Query .....	18
Figure 10:DQL Query .....	19
Figure 11:DQL Query .....	20
Figure 12:DQL Query .....	20
Figure 13:DQL Query .....	21
Figure 14:DQL Query .....	21
Figure 15:DQL Query .....	22
Figure 16:DQL Query .....	22
Figure 17:DQL Query .....	23
Figure 18:DQL Query .....	23
Figure 19:DQL Query .....	24
Figure 20:DQL Query .....	24
Figure 21:DQL Query .....	25
Figure 22:DQL Query .....	25
Figure 23:DQL Query .....	26
Figure 24:DQL Query .....	26
Figure 25:DQL Query .....	27
Figure 26:DQL Query .....	27
Figure 27:DQL Query .....	28
Figure 28:DQL Query .....	28
Figure 29:Nested Query .....	29
Figure 30:Nested Query .....	30
Figure 31:Nested Query .....	30
Figure 32:Nested Query .....	31
Figure 33:Nested Query .....	31
Figure 34:Nested Query .....	32
Figure 35:Nested Query .....	32
Figure 36:Nested Query .....	33
Figure 37:Nested Query .....	34
Figure 38:Nested Query .....	34

Figure 39:Inner Join.....	35
Figure 40:Inner Join.....	36
Figure 41:Inner Join.....	36
Figure 42:Inner Join.....	37
Figure 43:Inner Join.....	37
Figure 44:Left Join .....	38
Figure 45:Left Join.....	39
Figure 46:Left Join.....	39
Figure 47:Left Join.....	40
Figure 48:Left Join.....	41
Figure 49:Right Join .....	41
Figure 50:Right Join .....	42
Figure 51:Right Join .....	43
Figure 52:Right Join .....	43
Figure 53:Right Join .....	44
Figure 54:Full outer Join.....	45
Figure 55:Full outer Join.....	45
Figure 56:Full outer Join.....	46
Figure 57:Full outer Join.....	47
Figure 58:Full outer Join.....	47
Figure 59>Create Store Procedure .....	48
Figure 60>Create Store Procedure .....	49
Figure 61>Create Store Procedure .....	49
Figure 62>Create Store Procedure .....	50
Figure 63>Create Store Procedure .....	50
Figure 64>Create View .....	51
Figure 65>Create View .....	51
Figure 66>Create View .....	52
Figure 67>Create View .....	52
Figure 68>Create View .....	53
Figure 69>Create Two Logins .....	54
Figure 70>Create Two Logins .....	55
Figure 71>Create Two Logins .....	55
Figure 72::Create Two Logins .....	56
Figure 73:grant permission .....	57
Figure 74:grant permission .....	57
Figure 75>Create Cluster Index .....	58
Figure 76>Create Cluster Index .....	58
Figure 77>Create Noncluster Index .....	59
Figure 78>Create noncluster Index .....	59
Figure 79:Full Backup .....	61
Figure 80:Full Backup .....	61
Figure 81:differential backup.....	62
Figure 82:restoring differential backup.....	62

# ER DIAGRAM

Library Management System



# 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

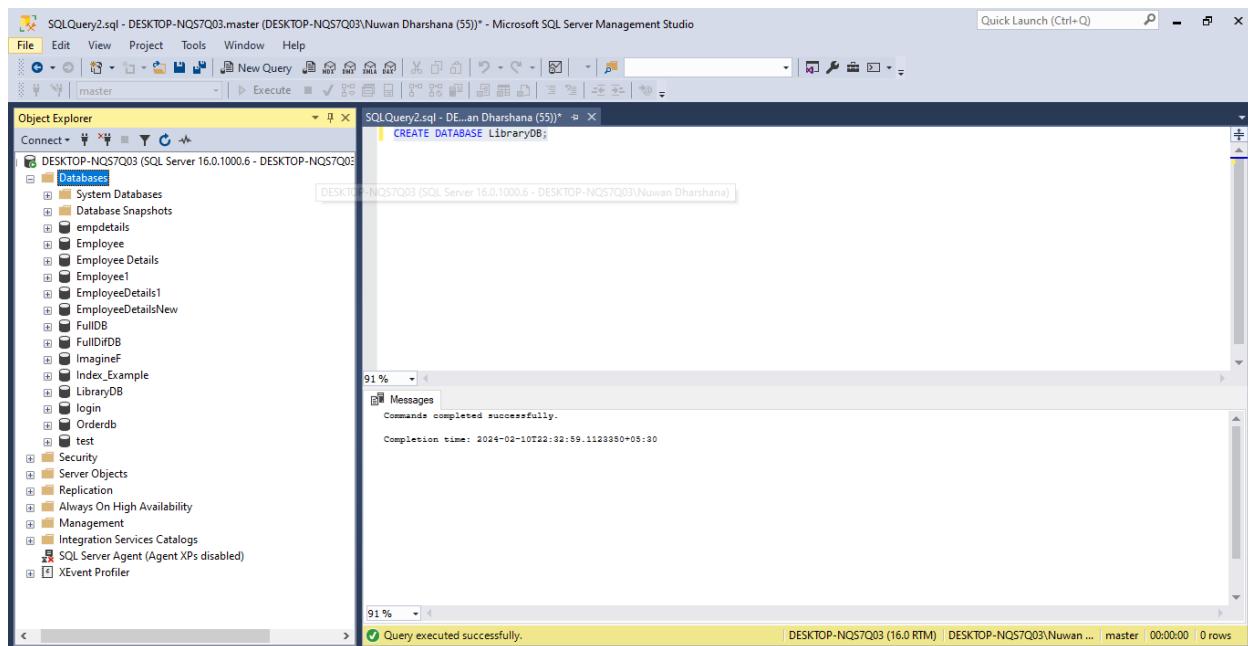
## **BorrowedBooks Table**

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



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, under the 'DESKTOP-NQS7Q03' node, the 'Databases' folder is expanded, showing various databases including 'master', 'tempdb', 'empdetails', 'Employee', 'Employee Details', 'Employee1', 'EmployeeDetails1', 'EmployeeDetailsNew', 'FullDB', 'FullDiffDB', 'Imaginf', 'Index\_Example', 'LibraryDB', 'login', 'Orderdb', and 'test'. In the center pane, a query window titled 'SQLQuery2.sql - DE...an Dharshana (55)\*' contains the command: 'CREATE DATABASE LibraryDB;'. Below the command, the 'Messages' pane displays the output: 'Commands completed successfully.' followed by the completion time: 'Completion time: 2024-02-10T22:32:59.1123350+05:30'. At the bottom of the screen, a status bar indicates 'DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | master | 00:00:00 | 0 rows'.

Figure 1:Create Database

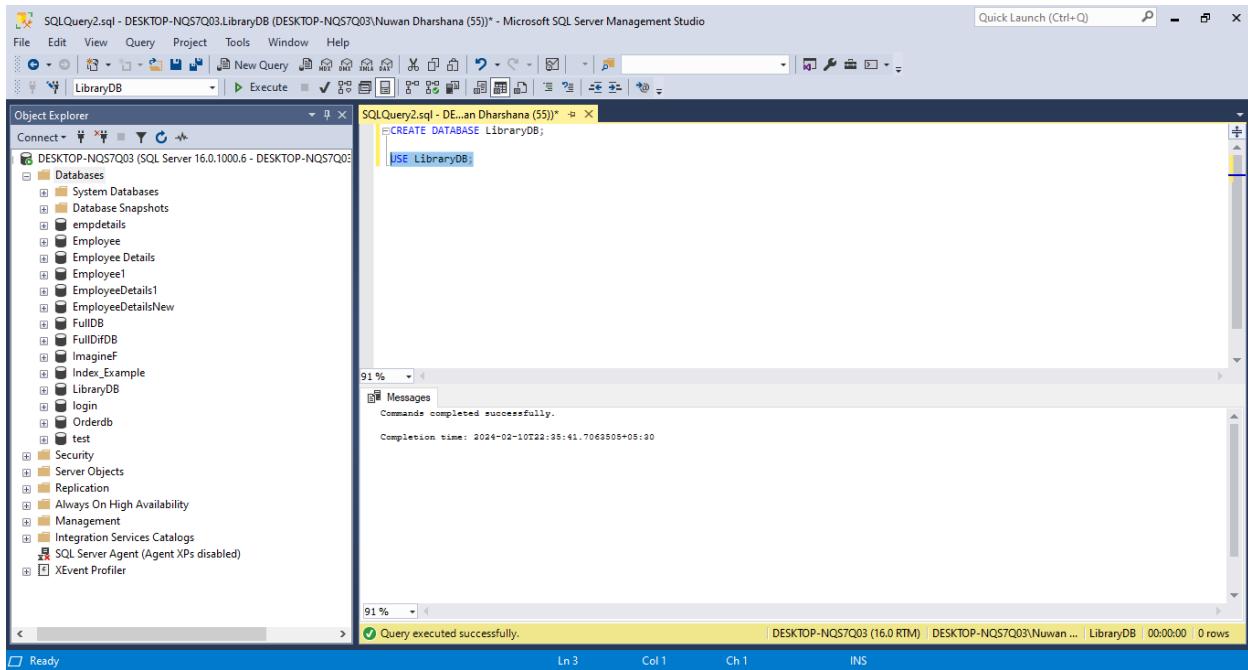


Figure 2:Use Database

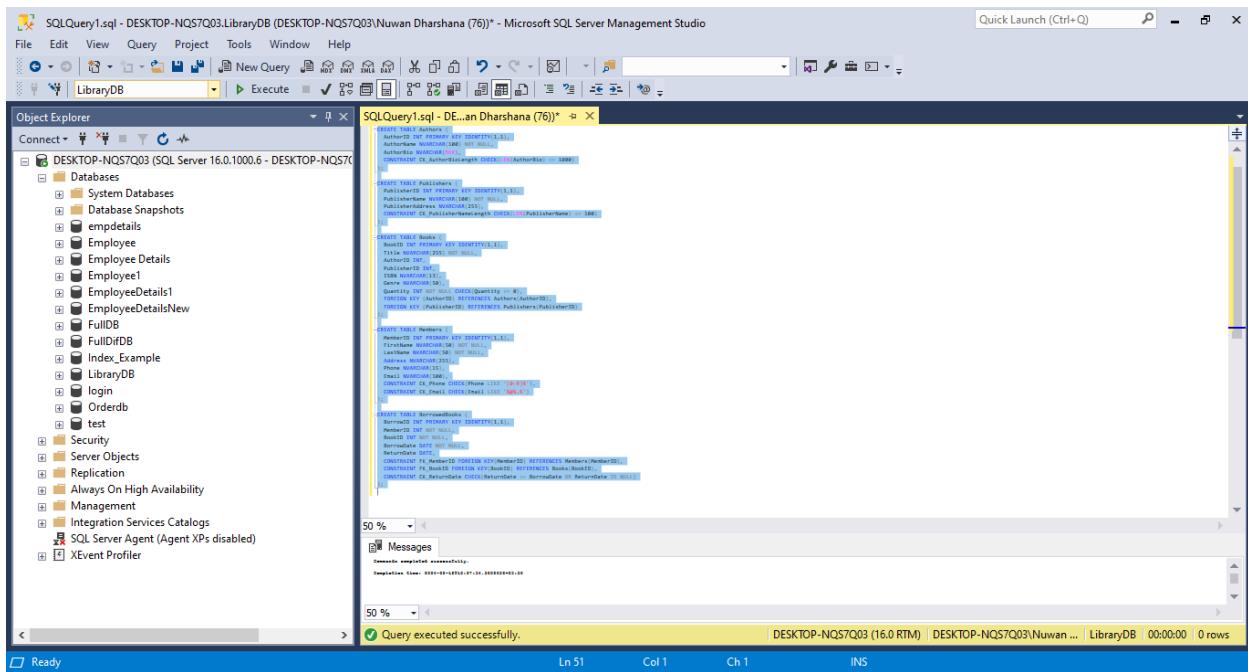
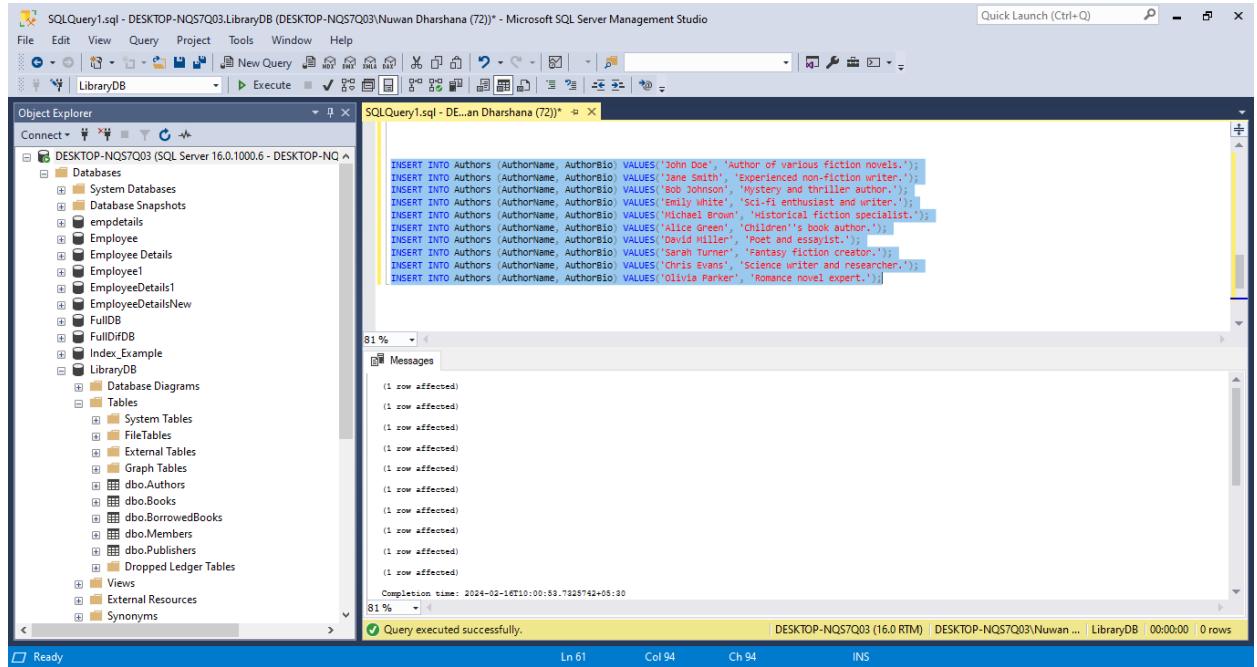


Figure 3:Create Tables

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the LibraryDB database which contains the Authors table. The SQL Query Editor window on the right contains the following SQL code:

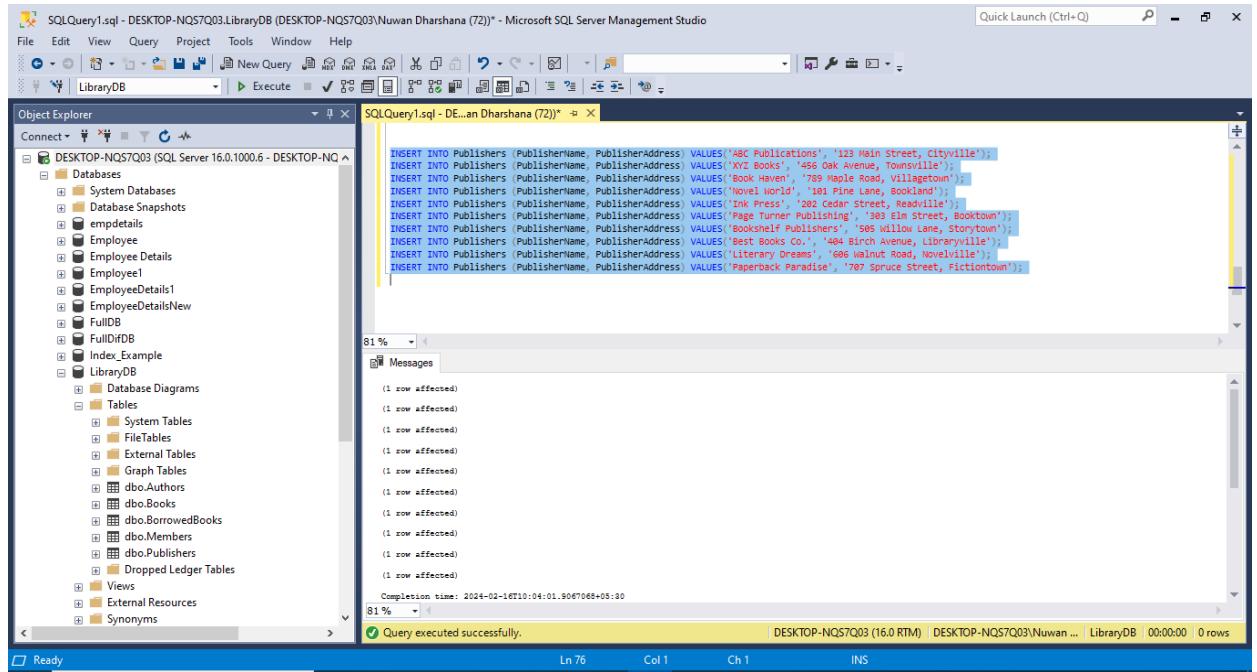
```

INSERT INTO Authors (AuthorName, AuthorBio) VALUES('John Doe', 'Author of various fiction novels.');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('Jane Smith', 'Experienced non-fiction writer.');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('Emily White', 'Young adult novelist');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('Michael Brown', 'Historical fiction specialist.');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('Alice Green', 'Children's book author.');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('David Miller', 'Poet and essayist.');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('Sarah Turner', 'Fantasy fiction creator.');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('Chris Evans', 'Science writer and researcher.');
INSERT INTO Authors (AuthorName, AuthorBio) VALUES('Olivia Parker', 'Romance novel expert.');

```

The Messages pane below the query editor shows the results of the insertions, indicating 1 row affected for each of the 10 inserted records. The status bar at the bottom right of the interface shows "Query executed successfully".

Figure 4:Insert Data



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the LibraryDB database which contains the Publishers table. The SQL Query Editor window on the right contains the following SQL code:

```

INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('ABC Publications', '123 Main Street, Cityville');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('XYZ Books', '456 Oak Avenue, Townsville');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Book Haven', '789 Maple Road, Villagetown');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Novel World', '101 Pine Lane, Bookland');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Ink Press', '202 Cedar Street, Readville');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Page Turner Publishing', '303 Elm Street, Booktown');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Bookshelf Publishers', '505 Willow Lane, Storytown');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Best Books Co.', '404 Birch Avenue, Libraryville');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Literary Dreams', '606 Walnut Road, Novelville');
INSERT INTO Publishers (PublisherName, PublisherAddress) VALUES('Paperback Paradise', '707 Spruce Street, Fictiontown');

```

The Messages pane below the query editor shows the results of the insertions, indicating 1 row affected for each of the 10 inserted records. The status bar at the bottom right of the interface shows "Query executed successfully".

Figure 5:Insert Data

```

SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Object Explorer
Connect New Query Execute
SQLQuery1.sql - DE...an Dharshana (76)*
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('The Lost Key', 1, 1, '9781234567890', 'Mystery', 50);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('The Secret Garden', 2, 2, '9781234567891', 'Romance', 30);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Whispers in the Shadows', 3, 3, '9781234567892', 'Thriller', 40);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Galactic Odyssey', 4, 4, '9781234567893', 'Sci-Fi', 25);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Echoes of Time', 5, 5, '9781234567894', 'Historical Fiction', 35);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Enchanted Forest', 6, 6, '9781234567895', 'Children's', 20);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Rhyme and Reason', 7, 7, '9781234567896', 'Poetry', 15);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Realms of Fantasy', 8, 8, '9781234567897', 'Fantasy', 30);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Science Explained', 9, 9, '9781234567898', 'Science', 40);
INSERT INTO Books (Title, AuthorID, PublisherID, ISBN, Genre, Quantity) VALUES('Love in Bloom', 10, 10, '9781234567899', 'Romance', 25);

Completion time: 2024-02-16T10:29:50.8320797+05:30
80 % ▶
Messages
(1 row affected)
Query executed successfully.

```

Figure 6:Insert Data

```

SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Object Explorer
Connect New Query Execute
SQLQuery1.sql - DE...an Dharshana (76)*
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Alice', 'Johnson', '123 Oak Street, Cityville', '5551234', 'alice@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Bob', 'Smith', '456 Pine Avenue, Townsville', '5555678', 'bob@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Charlie', 'Wilson', '789 Maple Street, Suburbia', '5550123', 'charlie@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('David', 'Brown', '101 Cedar Lane, Bookland', '5554567', 'david@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Ema', 'Jones', '202 Elm Street, Headville', '5557890', 'ema@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Frank', 'Taylor', '303 Birch Avenue, Libraryville', '5552345', 'frank@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Grace', 'Davis', '404 Willow Road, Storytown', '5556789', 'grace@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Henry', 'Martin', '505 Spruce Lane, Novelville', '5558123', 'henry@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Ivy', 'White', '606 Walnut Street, Fictionton', '5554367', 'ivy@email.com');
INSERT INTO Members (FirstName, LastName, Address, Phone, Email) VALUES('Jack', 'Miller', '707 Cedar Road, Bookville', '5558901', 'jack@email.com');

Completion time: 2024-02-16T10:30:48.3421157+05:30
80 % ▶
Messages
(1 row affected)
Query executed successfully.

```

Figure 7:Insert Data

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including 'Employee', 'Employee Details', 'Employee1', 'EmployeeDetails1', 'EmployeeDetailsNew', 'FullDB', 'FullDirDB', 'Index\_Example', 'LibraryDB', 'login', 'Orderdb', and 'test'. The 'Messages' pane at the bottom displays the results of the query execution, showing 10 rows affected. The status bar at the bottom right indicates '0 rows'.

```

INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 1, '2024-02-01', '2024-02-15');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 2, '2024-02-02', '2024-02-17');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 3, '2024-02-03', '2024-02-18');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 4, '2024-02-04', '2024-02-19');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 5, '2024-02-05', '2024-02-20');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 6, '2024-02-06', '2024-02-21');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 7, '2024-02-07', '2024-02-22');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 8, '2024-02-08', '2024-02-23');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 9, '2024-02-09', '2024-02-24');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 10, '2024-02-10', '2024-02-25');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 11, '2024-02-11', '2024-02-26');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 12, '2024-02-12', '2024-02-27');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 13, '2024-02-13', '2024-02-28');
INSERT INTO BorrowedBooks (MemberID, BookID, BorrowDate, ReturnDate) VALUES(1, 14, '2024-02-14', '2024-02-29');

```

Figure 8:Insert Data

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the same set of databases as in Figure 8. The 'Results' pane at the bottom displays the output of a SELECT query. The query is:

```

SELECT Genre, COUNT(*) AS BookCount
FROM Books
GROUP BY Genre;

```

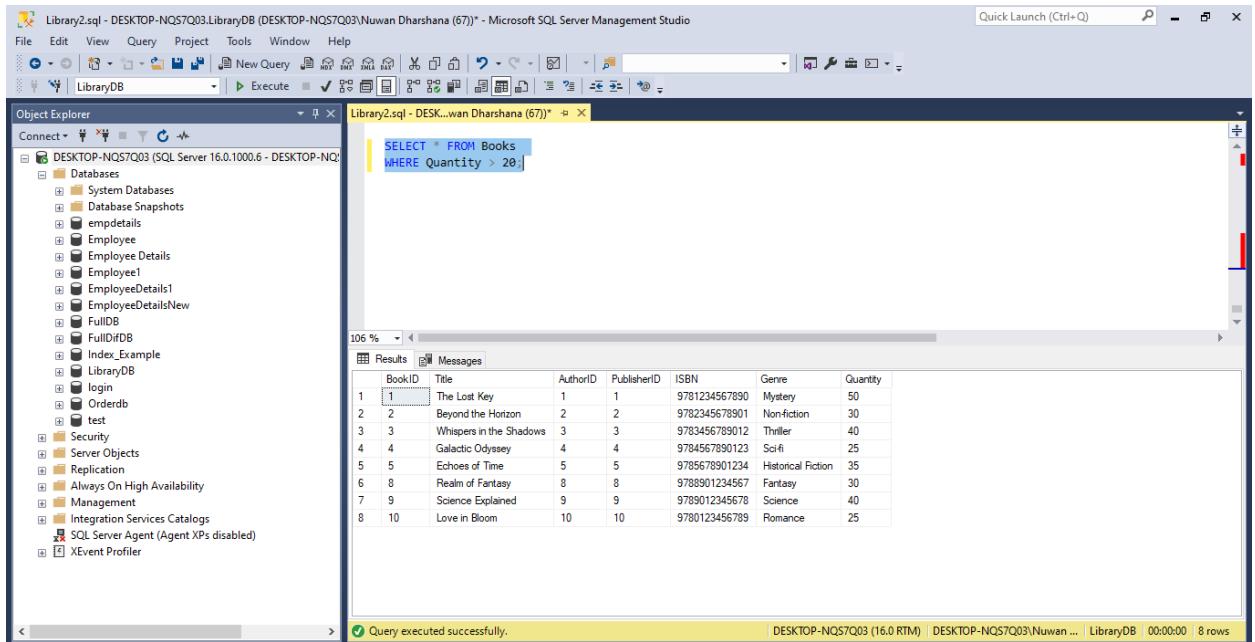
The results table shows the count of books per genre:

Genre	BookCount
Children's	1
Fantasy	1
Historical Fiction	1
Mystery	1
Non-fiction	1
Poetry	1
Romance	1
Science	1
Sci-fi	1
Thriller	1

The status bar at the bottom right indicates '10 rows'.

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays a tree view of database objects under the 'DESKTOP-NQS7Q03' connection. In the center, a query results window titled 'Library2.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (67))' contains the following SQL query:

```
SELECT * FROM Books  
WHERE Quantity > 20;
```

The results grid shows the following data:

BookID	Title	AuthorID	PublisherID	ISBN	Genre	Quantity
1	The Lost Key	1	1	9781234567890	Mystery	50
2	Beyond the Horizon	2	2	9782345678901	Non-fiction	30
3	Whispers in the Shadows	3	3	9783456789012	Thriller	40
4	Galactic Odyssey	4	4	9784567890123	Sci-fi	25
5	Echoes of Time	5	5	9785678901234	Historical Fiction	35
6	Realm of Fantasy	8	8	9788901234567	Fantasy	30
7	Science Explained	9	9	9789012345678	Science	40
8	Love in Bloom	10	10	9780123456789	Romance	25

At the bottom of the results window, a message indicates: 'Query executed successfully.'

Figure 10:DQL Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including 'LibraryDB'. The central pane displays a query window with the following SQL code:

```
SELECT * FROM Books
WHERE PublisherID = (SELECT PublisherID FROM Publishers WHERE PublisherName = 'ABC Publications');
```

The results pane shows a single row of data from the 'Books' table:

BookID	Title	AuthorID	PublisherID	ISBN	Genre	Quantity
1	The Lost Key	1	1	9781234567890	Mystery	50

At the bottom, a status bar indicates: 'Query executed successfully.' and '1 rows'.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including 'LibraryDB'. The central pane displays a query window with the following SQL code:

```
SELECT MemberID, COUNT(*) AS BorrowedCount
FROM BorrowedBooks
GROUP BY MemberID;
```

The results pane shows the count of borrowed books for each member ID:

MemberID	BorrowedCount
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1

At the bottom, a status bar indicates: 'Query executed successfully.' and '10 rows'.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'Library2.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (67))' contains the following SQL code:

```
SELECT * FROM Books
WHERE ISBN = '9783456789012';
```

The results pane shows a single row of data from the 'Books' table:

BookID	Title	AuthorID	PublisherID	ISBN	Genre	Quantity
1	Whispers in the Shadows	3	3	9783456789012	Thriller	40

At the bottom, a status bar indicates: 'Query executed successfully.' and 'DESKTOP-NQS7Q03 (16.0 RTM) DESKTOP-NQS7Q03\Nuwan ... LibraryDB 00:00:00 1 rows'.

Figure 13:DQL Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'Library2.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (67))' contains the following SQL code:

```
SELECT * FROM Books
ORDER BY Title ASC;
```

The results pane shows 10 rows of data from the 'Books' table, ordered by title ascending:

BookID	Title	AuthorID	PublisherID	ISBN	Genre	Quantity
1	Whispers in the Shadows	3	3	9783456789012	Thriller	40
2	Beyond the Horizon	2	2	9782345678901	Non-fiction	30
3	Enchanted Forest	6	6	9786789012345	Children's	20
4	Galactic Odyssey	4	4	9784567890123	Sci-fi	25
5	Echoes of Time	5	5	9785678901234	Historical Fiction	35
6	Love in Bloom	10	10	9780123456789	Romance	25
7	Realm of Fantasy	8	8	9788901234567	Fantasy	30
8	Rhyme and Reason	7	7	9787890123456	Poetry	15
9	Science Explained	9	9	9789012345678	Science	40
10	The Lost Key	1	1	9781234567890	Mystery	50

At the bottom, a status bar indicates: 'Query executed successfully.' and 'DESKTOP-NQS7Q03 (16.0 RTM) DESKTOP-NQS7Q03\Nuwan ... LibraryDB 00:00:00 10 rows'.

Figure 14:DQL Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the 'LibraryDB' database. The central pane displays a T-SQL query:

```

SELECT MemberID, COUNT(*) AS BorrowedCount
FROM BorrowedBooks
GROUP BY MemberID
HAVING COUNT(*) >= 1;

```

The results pane shows a table with two columns: 'MemberID' and 'BorrowedCount'. The data is as follows:

MemberID	BorrowedCount
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1

At the bottom, a message indicates "Query executed successfully." and provides session details.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the 'LibraryDB' database. The central pane displays a T-SQL query:

```

SELECT * FROM Books
WHERE Genre = 'Sci-Fi'
ORDER BY Quantity DESC;

```

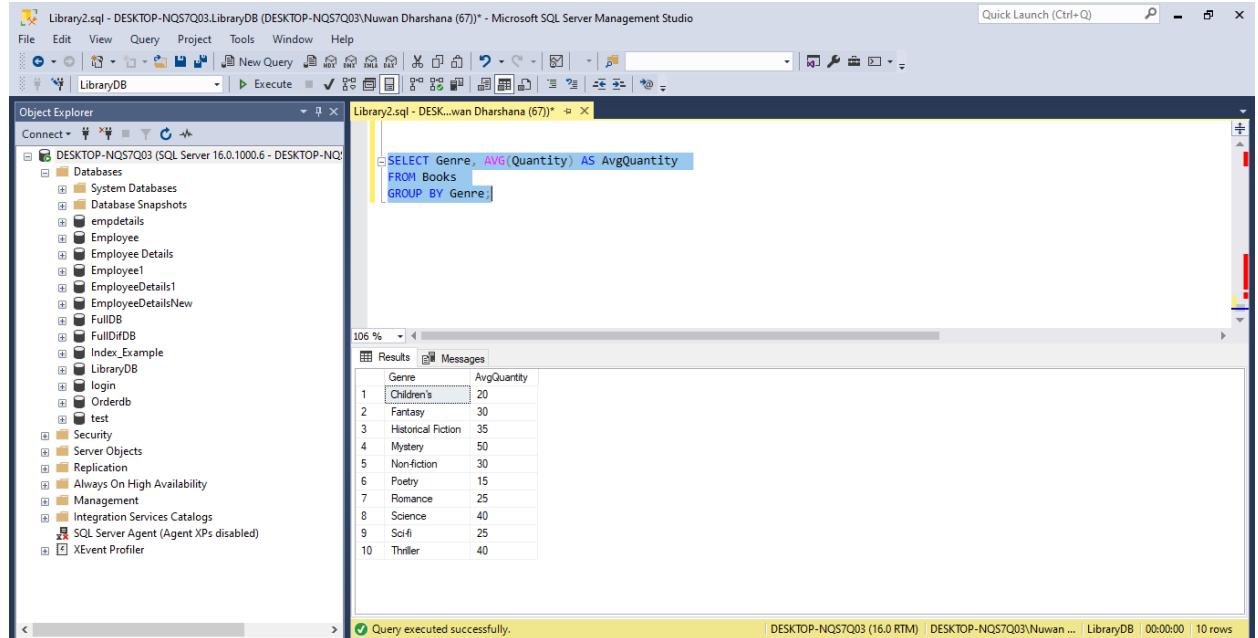
The results pane shows a table with seven columns: BookID, Title, AuthorID, PublisherID, ISBN, Genre, and Quantity. The data is as follows:

BookID	Title	AuthorID	PublisherID	ISBN	Genre	Quantity
4	Galactic Odyssey	4	4	9784567890123	Sci-Fi	25

At the bottom, a message indicates "Query executed successfully." and provides session details.

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.



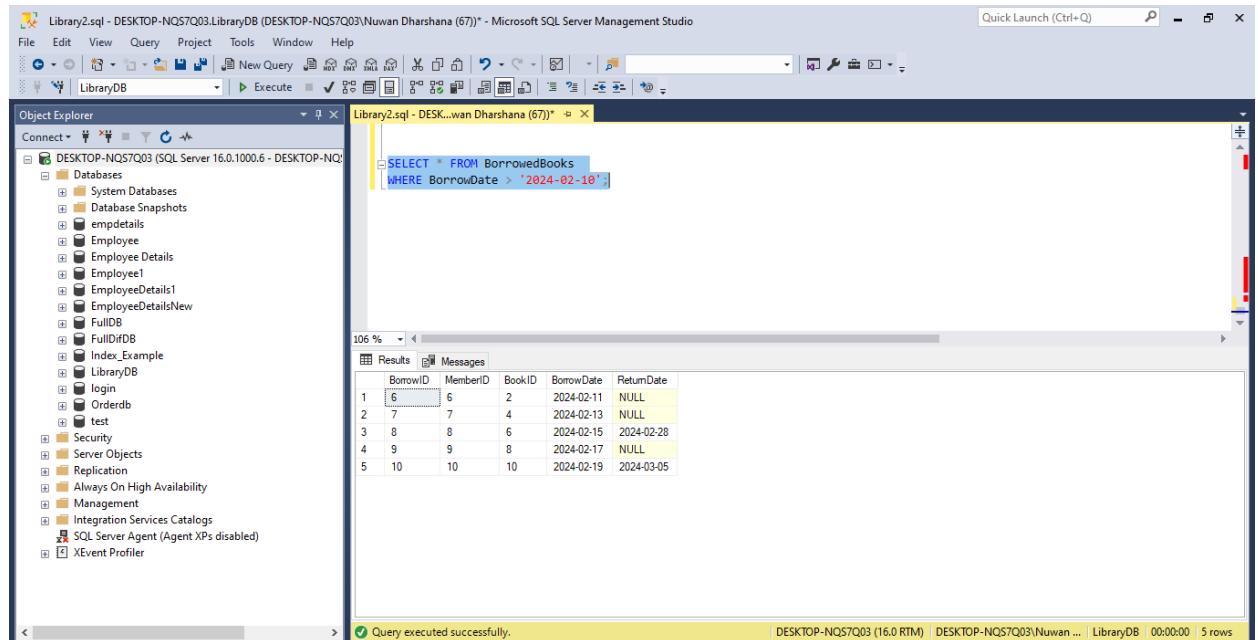
```
SELECT Genre, AVG(Quantity) AS AvgQuantity
FROM Books
GROUP BY Genre;
```

Genre	AvgQuantity
Children's	20
Fantasy	30
Historical Fiction	35
Mystery	50
Non-fiction	30
Poetry	15
Romance	25
Science	40
SciFi	25
Thriller	40

Query executed successfully.

Figure 17:DQL Query

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



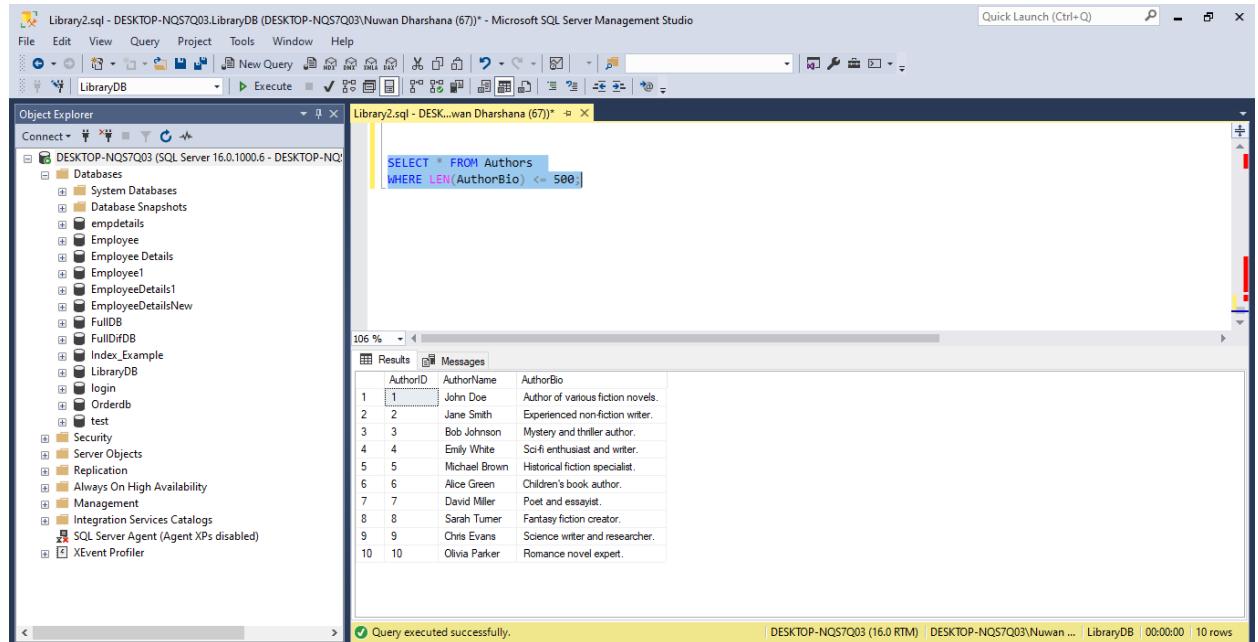
```
SELECT * FROM BorrowedBooks
WHERE BorrowDate > '2024-02-10';
```

BorrowID	MemberID	BookID	BorrowDate	ReturnDate
1	6	2	2024-02-11	NULL
2	7	4	2024-02-13	NULL
3	8	6	2024-02-15	2024-02-28
4	9	8	2024-02-17	NULL
5	10	10	2024-02-19	2024-03-05

Query executed successfully.

Figure 18:DQL Query

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a database named 'LibraryDB' containing various tables like 'Employee', 'Employee Details', etc. The central pane displays a query window with the following SQL code:

```
SELECT * FROM Authors
WHERE LEN(AuthorBio) <= 500;
```

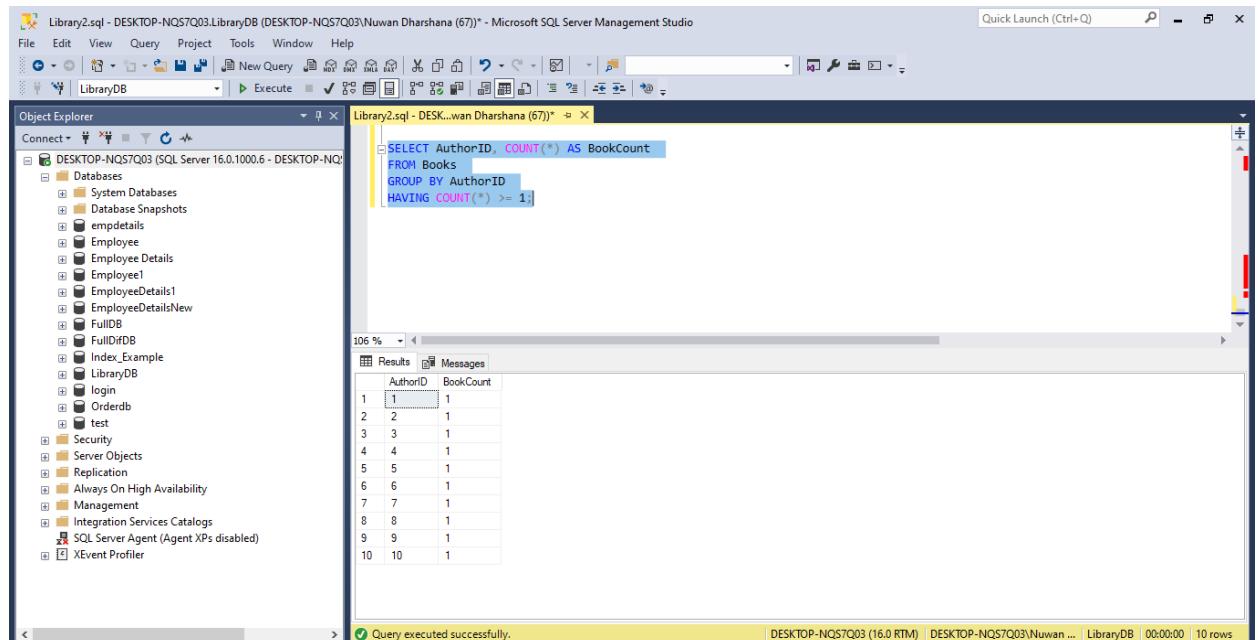
The results pane shows a table with columns 'AuthorID', 'AuthorName', and 'AuthorBio'. The data is as follows:

AuthorID	AuthorName	AuthorBio
1	John Doe	Author of various fiction novels.
2	Jane Smith	Experienced non-fiction writer.
3	Bob Johnson	Mystery and thriller author.
4	Emily White	Sci-fi enthusiast and writer.
5	Michael Brown	Historical fiction specialist.
6	Alice Green	Children's book author.
7	David Miller	Poet and essayist.
8	Sarah Turner	Fantasy fiction creator.
9	Chris Evans	Science writer and researcher.
10	Olivia Parker	Romance novel expert.

At the bottom, a message indicates 'Query executed successfully.'

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a database named 'LibraryDB' containing various tables like 'Employee', 'Employee Details', etc. The central pane displays a query window with the following SQL code:

```
SELECT AuthorID, COUNT(*) AS BookCount
FROM Books
GROUP BY AuthorID
HAVING COUNT(*) >= 1;
```

The results pane shows a table with columns 'AuthorID' and 'BookCount'. The data is as follows:

AuthorID	BookCount
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1

At the bottom, a message indicates 'Query executed successfully.'

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.

```
SELECT * FROM BorrowedBooks
WHERE ReturnDate IS NOT NULL
ORDER BY ReturnDate ASC;
```

BorrowID	MemberID	BookID	BorrowDate	ReturnDate
1	1	1	2024-02-01	2024-02-15
2	2	3	2024-02-03	2024-02-17
3	4	7	2024-02-07	2024-02-21
4	5	9	2024-02-09	2024-02-23
5	8	6	2024-02-15	2024-02-28
6	10	10	2024-02-19	2024-03-05

Query executed successfully.

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.

```
SELECT Genre, COUNT(*) AS BookCount
FROM Books
GROUP BY Genre
ORDER BY BookCount DESC;
```

Genre	BookCount
Children's	1
Fantasy	1
Historical Fiction	1
Mystery	1
Nonfiction	1
Poetry	1
Romance	1
Science	1
Sci-fi	1
Thriller	1

Query executed successfully.

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.

```
SELECT * FROM Books
WHERE AuthorID = 1
ORDER BY Title ASC;
```

BookID	Title	AuthorID	PublisherID	ISBN	Genre	Quantity
1	The Lost Key	1	1	9781234567890	Mystery	50

Query executed successfully.

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.

```
SELECT * FROM BorrowedBooks
WHERE ReturnDate IS NULL;
```

BorrowID	MemberID	BookID	BorrowDate	ReturnDate
1	3	5	2024-02-05	NULL
2	6	2	2024-02-11	NULL
3	7	4	2024-02-13	NULL
4	9	8	2024-02-17	NULL

Query executed successfully.

Figure 24:DQL Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including tables like 'Members', 'Books', and 'BorrowedBooks'. The central pane displays a query window with the following SQL code:

```
SELECT * FROM Members
WHERE Phone LIKE '555%'
```

The results pane shows a table with 10 rows of member data, each with MemberID, FirstName, LastName, Address, Phone, and Email columns. The data includes entries for Alice Johnson, Bob Smith, Charlie Williams, David Brown, Emma Jones, Frank Taylor, Grace Davis, Henry Martin, Ivy White, and Jack Miller.

MemberID	FirstName	LastName	Address	Phone	Email
1	Alice	Johnson	123 Oak Street, Cityville	5551234	alice@email.com
2	Bob	Smith	456 Pine Avenue, Townsville	5559678	bob@email.com
3	Charlie	Williams	789 Maple Road, Villagetown	5559012	charlie@email.com
4	David	Brown	101 Cedar Lane, Bookland	5553456	david@email.com
5	Emma	Jones	202 Elm Street, Readville	5557890	emma@email.com
6	Frank	Taylor	303 Birch Avenue, Libraryville	5552345	frank@email.com
7	Grace	Davis	404 Willow Road, Storytown	5556789	grace@email.com
8	Henry	Martin	505 Spruce Lane, Novelville	5550123	henry@email.com
9	Ivy	White	606 Walnut Street, Fictontown	5554557	ivy@email.com
10	Jack	Miller	707 Cedar Road, Bookville	5558901	jack@email.com

At the bottom, a message indicates "Query executed successfully."

Figure 25:DQL Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including tables like 'Authors', 'Books', and 'BorrowedBooks'. The central pane displays a query window with the following SQL code:

```
SELECT AuthorID, AuthorName
FROM Authors
WHERE AuthorID IN (SELECT DISTINCT AuthorID
FROM Books
WHERE Genre IN ('Fiction', 'Non-fiction'))
GROUP BY AuthorID
HAVING COUNT(DISTINCT Genre) = 1;
```

The results pane shows a table with 1 row of author data, each with AuthorID and AuthorName columns. The data includes an entry for Jane Smith.

AuthorID	AuthorName
1	Jane Smith

At the bottom, a message indicates "Query executed successfully."

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'.

```
SELECT b.* , a.AuthorName
FROM Books b INNER JOIN Authors a ON b.AuthorID = a.AuthorID
ORDER BY a.AuthorName ASC;
```

BookID	Title	AuthorID	PublisherID	ISBN	Genre	Quantity	AuthorName
1	Enchanted Forest	6	6	9786789012345	Children's	20	Alice Green
2	Whispers in the Shadows	3	3	9783456789012	Thriller	40	Bob Johnson
3	Science Explained	9	9	9789012345678	Science	40	Chris Evans
4	Rhyme and Reason	7	7	978790123456	Poetry	15	David Miller
5	Galactic Odyssey	4	4	9784567890123	Sci-fi	25	Emily White
6	Beyond the Horizon	2	2	9782345678901	Non-fiction	30	Jane Smith
7	The Lost Key	1	1	9781234567890	Mystery	50	John Doe
8	Echoes of Time	5	5	97895678901234	Historical Fiction	35	Michael Brown
9	Love in Bloom	10	10	9780123456789	Romance	25	Olivia Parker
10	Realm of Fantasy	8	8	9788901234567	Fantasy	30	Sarah Turner

Query executed successfully.

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.

```
SELECT p.PublisherID, p.PublisherName, COUNT(b.BookID) AS PublishedBooks
FROM Publishers p
JOIN Books b ON p.PublisherID = b.PublisherID
GROUP BY p.PublisherID, p.PublisherName
HAVING COUNT(b.BookID) >= 1;
```

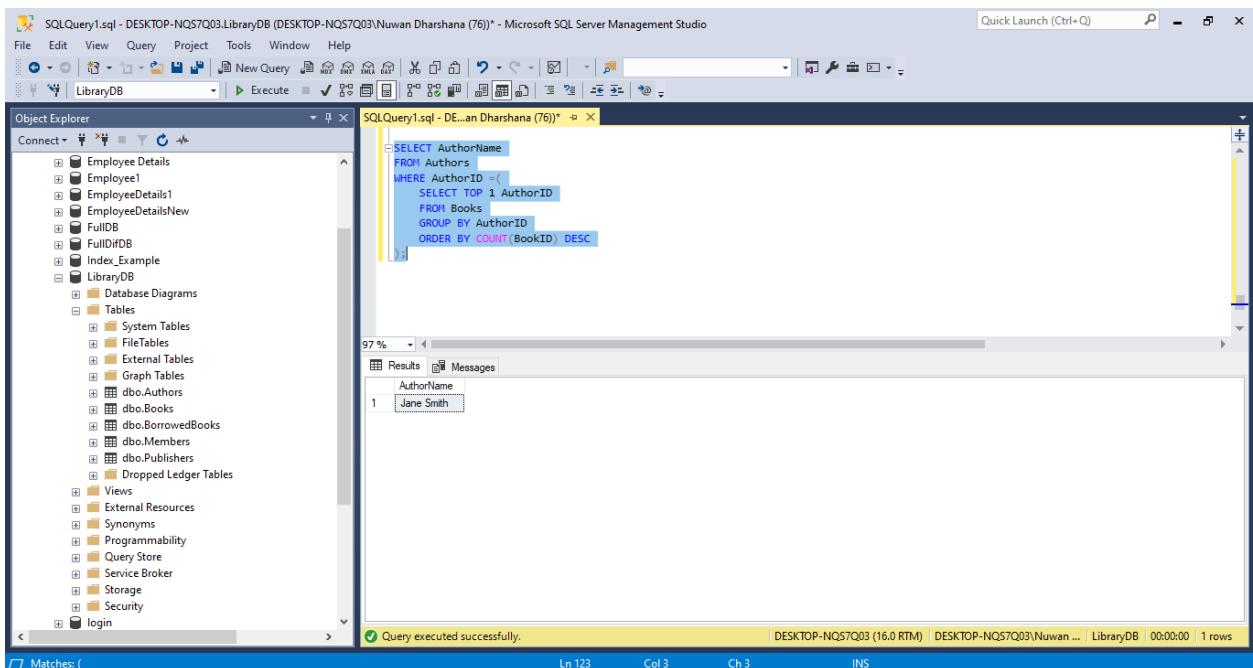
PublisherID	PublisherName	PublishedBooks
1	ABC Publications	1
2	XYZ Books	1
3	Book Haven	1
4	Novel World	1
5	Ink Press	1
6	Page Turner Publishing	1
7	Bookshelf Publishers	1
8	Best Books Co	1
9	Literary Dreams	1
10	Paperback Paradise	1

Query executed successfully.

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects including Employee Details, LibraryDB, and Tables. The Query Editor window on the right contains a nested query:

```
SELECT AuthorName
FROM Authors
WHERE AuthorID =(
    SELECT TOP 1 AuthorID
    FROM Books
    GROUP BY AuthorID
    ORDER BY COUNT(BookID) DESC
);
```

The Results tab shows the output of the query:

AuthorName
Jane Smith

At the bottom, a status bar indicates "Query executed successfully." and other details like the session ID and row count.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left is the Object Explorer pane, which lists various database objects like Employee Details, LibraryDB, and Tables. The main area contains a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The query is:

```

SELECT MemberID, FirstName, LastName, Address, Phone, Email
FROM Members
WHERE MemberID IN(
    SELECT DISTINCT MemberID
    From BorrowedBooks
);

```

The results pane shows a table with 10 rows of member details. The columns are MemberID, FirstName, LastName, Address, Phone, and Email. The data is as follows:

MemberID	FirstName	LastName	Address	Phone	Email
1	Alice	Johnson	123 Oak Street, Cityville	5551234	alice@email.com
2	Bob	Smith	456 Pine Avenue, Townsville	5555678	bob@email.com
3	Charlie	Williams	789 Maple Road, Villagetown	5559012	charlie@email.com
4	David	Brown	101 Cedar Lane, Bookland	5553456	david@email.com
5	Emma	Jones	202 Elm Street, Readville	5557890	emma@email.com
6	Frank	Taylor	303 Birch Avenue, Libraville	5552345	frank@email.com
7	Grace	Davis	404 Willow Road, Stonytown	5556789	grace@email.com
8	Henry	Martin	505 Spruce Lane, Noveltown	5550123	henry@email.com
9	Ivy	White	606 Walnut Street, Fictiontown	5554567	ivy@email.com
10	Jack	Miller	707 Cedar Road, Bookville	5558901	jack@email.com

At the bottom of the results pane, it says 'Query executed successfully.' and shows the session information: DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 10 rows.

Figure 30:Nested Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left is the Object Explorer pane. The main area contains a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The query is:

```

SELECT BookID, MemberID, BorrowDate, ReturnDate
FROM BorrowedBooks
WHERE MemberID IN(
    SELECT DISTINCT MemberID
    FROM BorrowedBooks
    WHERE ReturnDate IS NULL
);

```

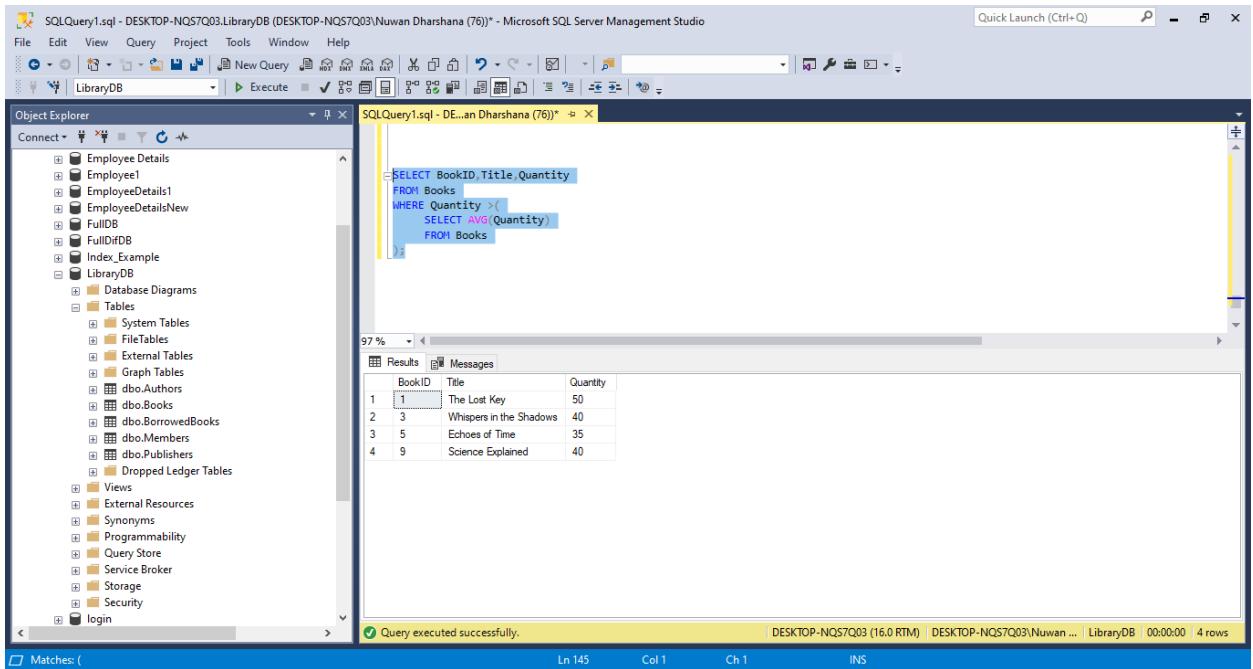
The results pane shows a table with 4 rows of book borrowing information. The columns are BookID, MemberID, BorrowDate, and ReturnDate. The data is as follows:

BookID	MemberID	BorrowDate	ReturnDate
1	5	2024-02-05	NULL
2	2	2024-02-11	NULL
3	4	2024-02-13	NULL
4	8	2024-02-17	NULL

At the bottom of the results pane, it says 'Query executed successfully.' and shows the session information: DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 4 rows.

Figure 31:Nested Query

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like Employee Details, Books, and Publishers. The central pane displays a query window titled 'SQLQuery1.sql'. The query is:`SELECT BookID, Title, Quantity
FROM Books
WHERE Quantity > (
 SELECT AVG(Quantity)
 FROM Books
);`

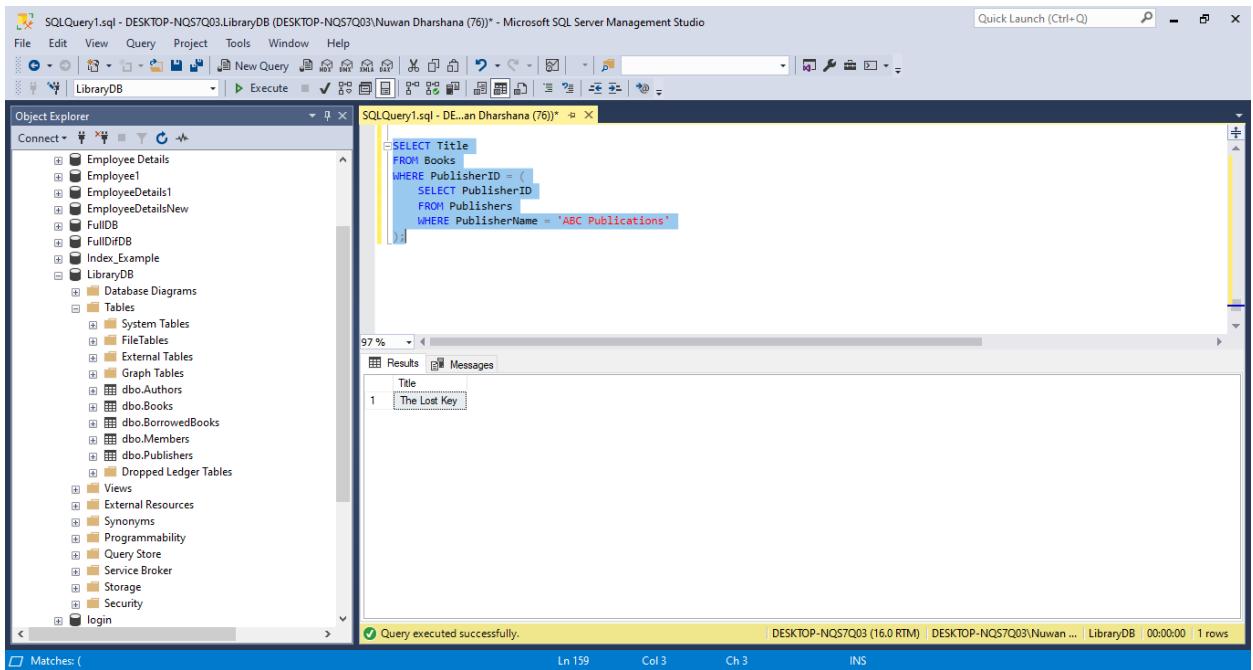
The results pane shows a table with four rows:

BookID	Title	Quantity
1	The Lost Key	50
2	Whispers in the Shadows	40
3	Echoes of Time	35
4	Science Explained	40

At the bottom, a message indicates 'Query executed successfully.'

Figure 32:Nested Query

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects. The central pane displays a query window titled 'SQLQuery1.sql'. The query is:`SELECT Title
FROM Books
WHERE PublisherID = (
 SELECT PublisherID
 FROM Publishers
 WHERE PublisherName = 'ABC Publications'
);`

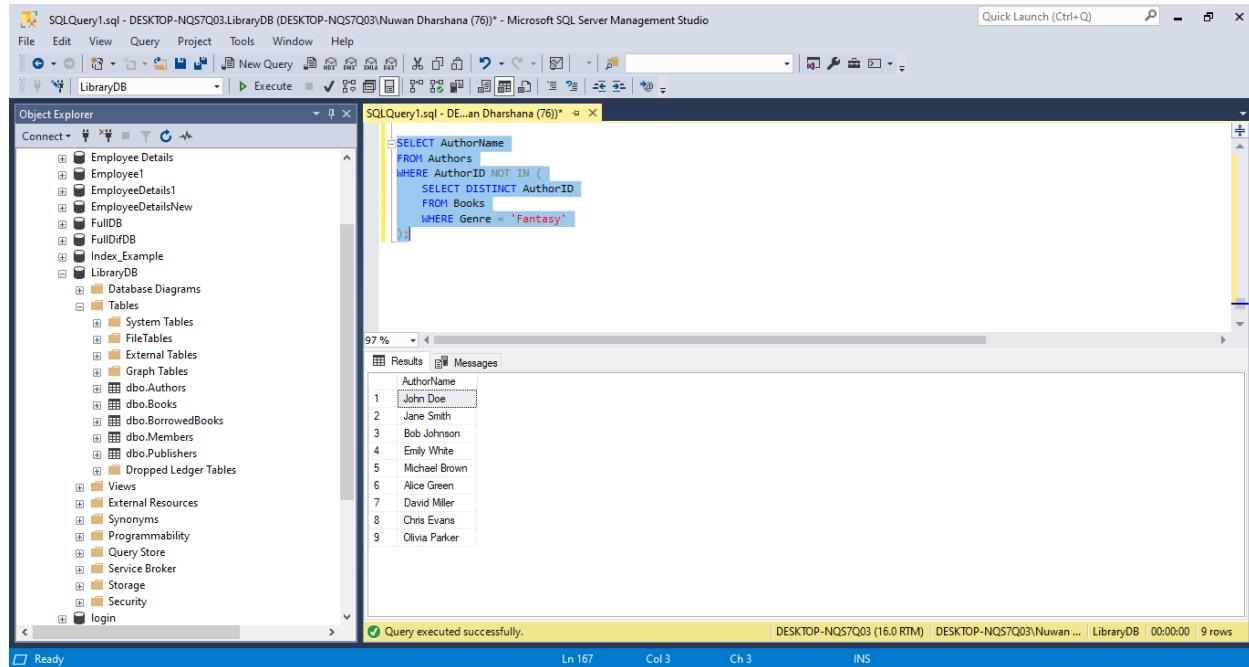
The results pane shows a table with one row:

Title
The Lost Key

At the bottom, a message indicates 'Query executed successfully.'

Figure 33:Nested Query

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like Employee Details, Books, and Publishers. The main window contains a query editor with the following SQL code:

```

SELECT AuthorName
FROM Authors
WHERE AuthorID NOT IN (
    SELECT DISTINCT AuthorID
    FROM Books
    WHERE Genre = 'Fantasy'
)

```

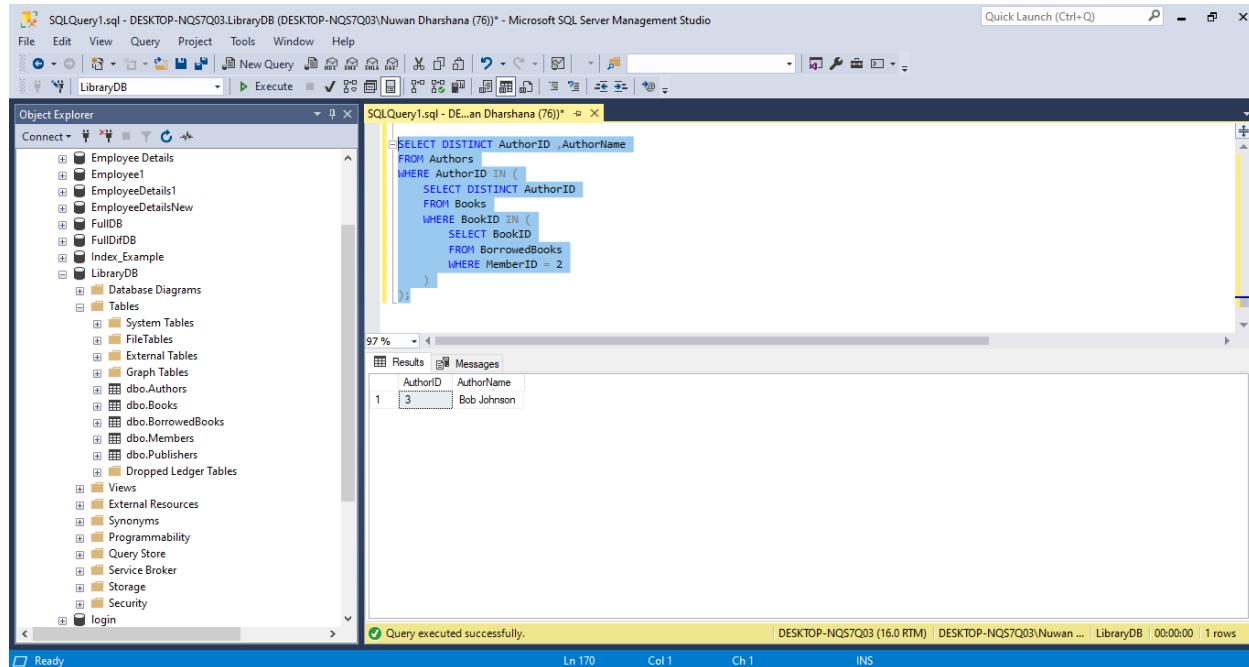
The results pane below shows a list of authors whose books are not in the Fantasy genre. The output is:

AuthorName
John Doe
Jane Smith
Bob Johnson
Emily White
Michael Brown
Alice Green
David Miller
Chris Evans
Olivia Parker

At the bottom, a message indicates the query was executed successfully.

Figure 34:Nested Query

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects. The main window contains a query editor with the following SQL code:

```

SELECT DISTINCT AuthorID , AuthorName
FROM Authors
WHERE AuthorID IN (
    SELECT DISTINCT AuthorID
    FROM Books
    WHERE BookID IN (
        SELECT BookID
        FROM BorrowedBooks
        WHERE MemberID = 2
    )
)

```

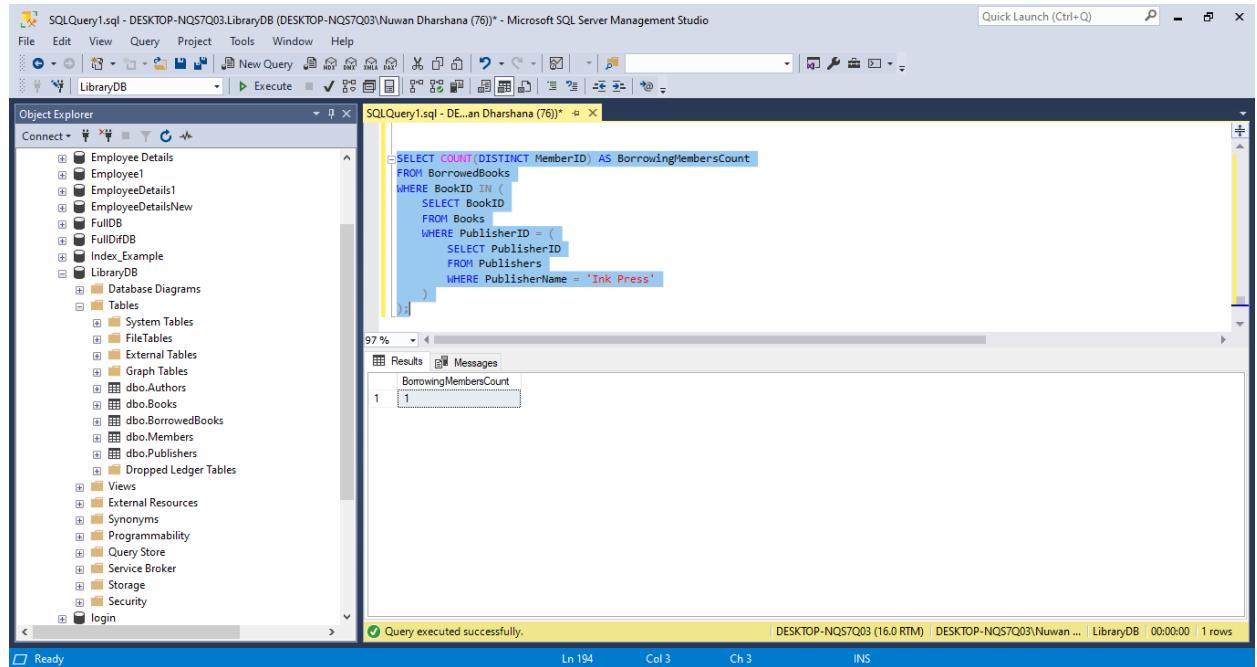
The results pane shows one author who has books listed under MemberID 2. The output is:

AuthorID	AuthorName
3	Bob Johnson

At the bottom, a message indicates the query was executed successfully.

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like Employee Details, Employee1, and LibraryDB. The main window contains a query editor with the following SQL code:

```
SELECT COUNT(DISTINCT MemberID) AS BorrowingMembersCount
FROM BorrowedBooks
WHERE BookID IN (
    SELECT BookID
    FROM Books
    WHERE PublisherID = (
        SELECT PublisherID
        FROM Publishers
        WHERE PublisherName = 'Ink Press'
    )
)
```

The results pane shows a single row with a value of 1 under the column 'BorrowingMembersCount'. A status bar at the bottom indicates the query was executed successfully.

Figure 36:Nested Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'SQLQuery1.sql - DESKTOP-NQ57Q03.LibraryDB (DESKTOP-NQ57Q03)\Nuwan Dharshana (76)\*' contains the following nested query:

```

SELECT Title
FROM Books
WHERE AuthorID = (
    SELECT AuthorID
    FROM Authors
    WHERE AuthorName = 'John Doe'
)

```

The results pane shows the output:

Title
The Lost Key

Below the results, a message indicates: 'Query executed successfully.'

Figure 37:Nested Query

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

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'SQLQuery1.sql - DESKTOP-NQ57Q03.LibraryDB (DESKTOP-NQ57Q03)\Nuwan Dharshana (76)\*' contains the following nested query:

```

SELECT FirstName, LastName, Address, Phone, Email
FROM Members
WHERE MemberID IN (
    SELECT MemberID
    FROM BorrowedBooks
    WHERE BookID IN (
        SELECT BookID
        FROM Books
        WHERE Quantity > 30
    )
)

```

The results pane shows the output:

FirstName	LastName	Address	Phone	Email
Alice	Johnson	123 Oak Street, Cityville	5551234	alice@email.com
Bob	Smith	456 Pine Avenue, Townsville	5555678	bob@email.com
Charlie	Williams	789 Maple Road, Villagetown	5559012	charlie@email.com
Emma	Jones	202 Elm Street, Readville	5557890	emma@email.com

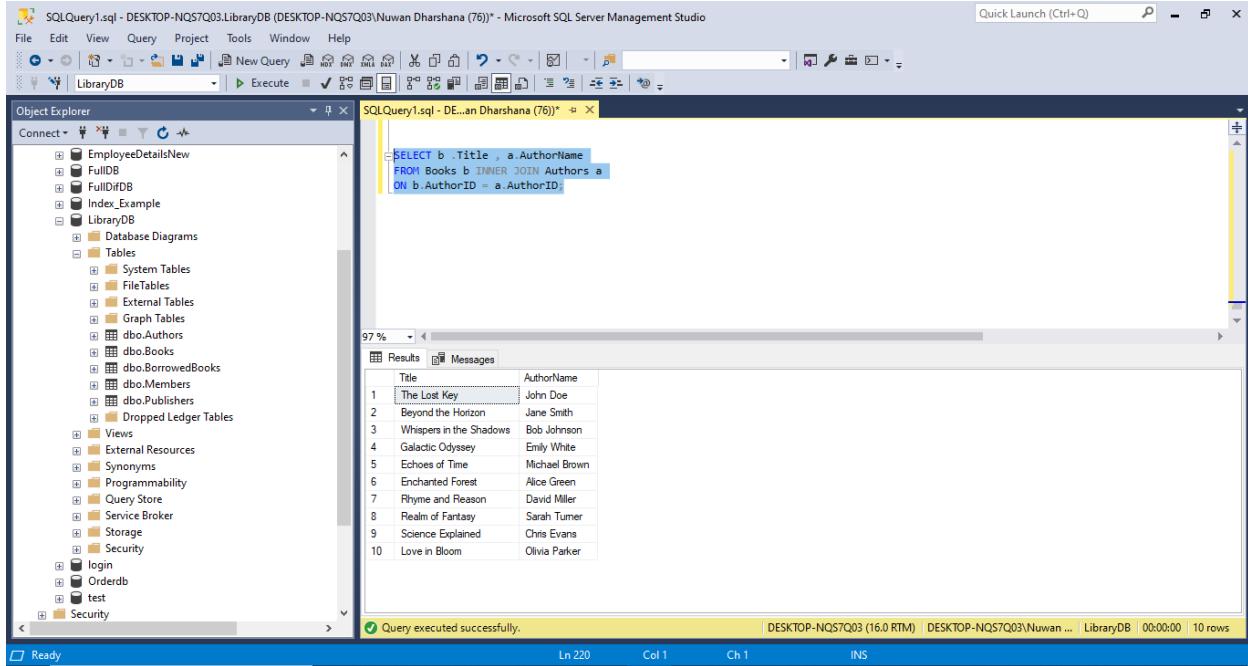
Below the results, a message indicates: 'Query executed successfully.'

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows a database named 'LibraryDB' with various objects like EmployeeDetailsNew, FullDB, FullDiffDB, Index\_Example, LibraryDB, dbo.Books, dbo.BorrowedBooks, dbo.Members, dbo.Publishers, dbo.Views, External Resources, Synonyms, Programmability, Query Store, Service Broker, Storage, Security, login, Orderdb, and test. The Query Editor window in the center contains the following SQL code:

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

The Results pane below shows the output of the query:

	Title	AuthorName
1	The Lost Key	John Doe
2	Beyond the Horizon	Jane Smith
3	Whispers in the Shadows	Bob Johnson
4	Galactic Odyssey	Emily White
5	Echoes of Time	Michael Brown
6	Enchanted Forest	Alice Green
7	Rhyme and Reason	David Miller
8	Realm of Fantasy	Sarah Turner
9	Science Explained	Chris Evans
10	Love in Bloom	Olivia Parker

At the bottom of the Results pane, it says "Query executed successfully." The status bar at the bottom of the screen shows "Ready", "Ln 220", "Col 1", "Ch 1", and "INS".

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases and their objects. The Query Editor window contains the following SQL code:

```

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

```

The Results pane displays the output of the query:

	Title	AuthorName
1	The Lost Key	John Doe

Below the results, a message indicates "Query executed successfully."

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases and their objects. The Query Editor window contains the following SQL code:

```

SELECT b.Title, P.PublisherName
FROM Books b INNER JOIN Publishers P
ON b.PublisherID = P.PublisherID;

```

The Results pane displays the output of the query:

	Title	PublisherName
1	The Lost Key	ABC Publications
2	Beyond the Horizon	XYZ Books
3	Whispers in the Shadows	Book Haven
4	Galactic Odyssey	Novel World
5	Echoes of Time	Ink Press
6	Enchanted Forest	Page Turner Publishing
7	Rhyme and Reason	Bookshelf Publishers
8	Realm of Fantasy	Best Books Co.
9	Science Explained	Literary Dreams
10	Love in Bloom	Paperback Paradise

Below the results, a message indicates "Query executed successfully."

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'SQLQuery1.sql - DE...an Dharshana (76)\*' displays the following T-SQL code:

```

SELECT m.FirstName, m.LastName, Books.Title, m.Email
FROM Members m
INNER JOIN BorrowedBooks B ON m.MemberID = B.MemberID
INNER JOIN Books ON B.BookID = Books.BookID;

```

The results pane shows a table with 10 rows of data, mapping members to their borrowed books. The columns are FirstName, LastName, Title, and Email. The data is as follows:

	FirstName	LastName	Title	Email
1	Alice	Johnson	The Lost Key	alice@email.com
2	Bob	Smith	Whispers in the Shadows	bob@email.com
3	Charlie	Williams	Echoes of Time	charlie@email.com
4	David	Brown	Rhyme and Reason	david@email.com
5	Emma	Jones	Science Explained	emma@email.com
6	Frank	Taylor	Beyond the Horizon	frank@email.com
7	Grace	Davis	Galactic Odyssey	grace@email.com
8	Henry	Martin	Enchanted Forest	henry@email.com
9	Ivy	White	Realm of Fantasy	ivy@email.com
10	Jack	Miller	Love in Bloom	jack@email.com

At the bottom of the results pane, a message indicates 'Query executed successfully.'

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'SQLQuery1.sql - DE...an Dharshana (76)\*' displays the following T-SQL code:

```

SELECT Books.Title, m.FirstName, m.LastName
FROM Books Books
INNER JOIN BorrowedBooks B ON Books.BookID = B.BookID
INNER JOIN Members m ON B.MemberID = m.MemberID;

```

The results pane shows a table with 10 rows of data, mapping book titles to member names. The columns are Title, FirstName, and LastName. The data is as follows:

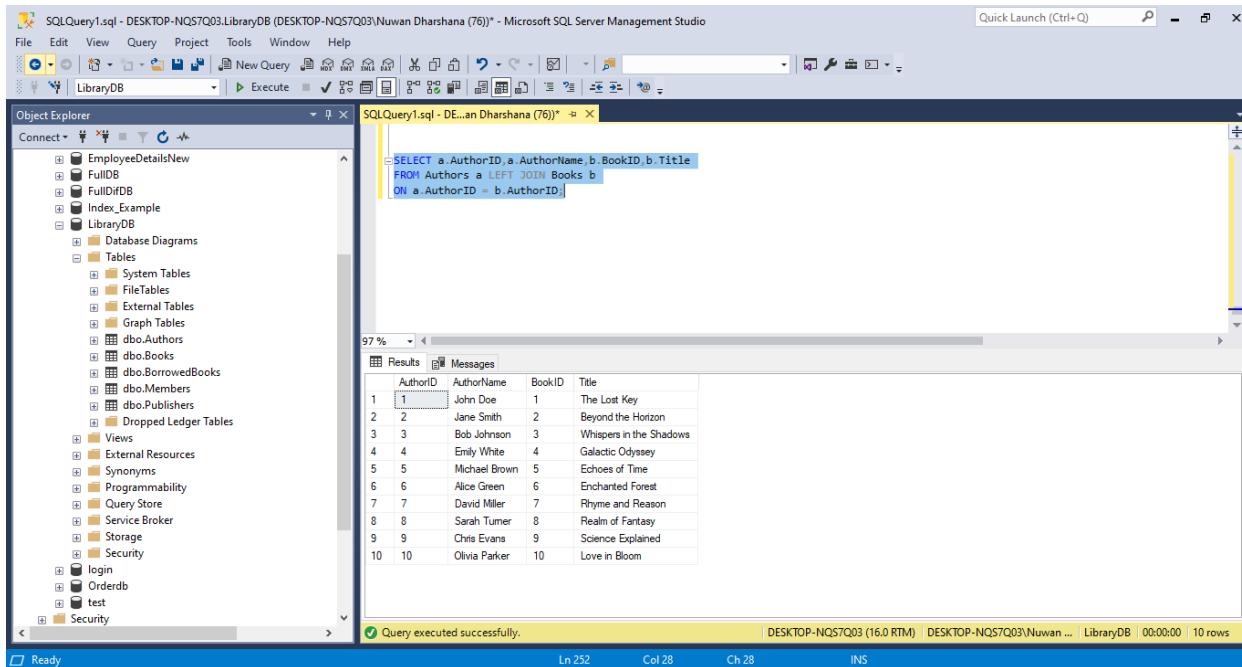
	Title	FirstName	LastName
1	The Lost Key	Alice	Johnson
2	Whispers in the Shadows	Bob	Smith
3	Echoes of Time	Charlie	Williams
4	Rhyme and Reason	David	Brown
5	Science Explained	Emma	Jones
6	Beyond the Horizon	Frank	Taylor
7	Galactic Odyssey	Grace	Davis
8	Enchanted Forest	Henry	Martin
9	Realm of Fantasy	Ivy	White
10	Love in Bloom	Jack	Miller

At the bottom of the results pane, a message indicates 'Query executed successfully.'

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects including tables like Authors, Books, and BorrowedBooks. The central pane displays a query window with the following SQL code:

```
SELECT a.AuthorID, a.AuthorName, b.BookID, b.Title
FROM Authors a LEFT JOIN Books b
ON a.AuthorID = b.AuthorID;
```

The results pane below shows the output of the query, which includes 10 rows of data:

	AuthorID	AuthorName	BookID	Title
1	1	John Doe	1	The Lost Key
2	2	Jane Smith	2	Beyond the Horizon
3	3	Bob Johnson	3	Whispers in the Shadows
4	4	Emily White	4	Galactic Odyssey
5	5	Michael Brown	5	Echoes of Time
6	6	Alice Green	6	Enchanted Forest
7	7	David Miller	7	Rhyme and Reason
8	8	Sarah Turner	8	Realm of Fantasy
9	9	Chris Evans	9	Science Explained
10	10	Olivia Parker	10	Love in Bloom

At the bottom of the results pane, a message indicates "Query executed successfully." The status bar at the bottom right shows "INS".

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'LibraryDB' is selected. In the center pane, a query window titled 'SQLQuery1.sql' displays the following SQL code:

```

SELECT a.AuthorID, a.AuthorName, b.BookID, b.Title
FROM Authors a LEFT JOIN Books b
ON a.AuthorID = b.AuthorID AND b.Genre = 'Fantasy';

```

The results pane shows a table with four columns: AuthorID, AuthorName, BookID, and Title. The data is as follows:

AuthorID	AuthorName	BookID	Title
1	John Doe	NULL	NULL
2	Jane Smith	NULL	NULL
3	Bob Johnson	NULL	NULL
4	Emily White	NULL	NULL
5	Michael Brown	NULL	NULL
6	Alice Green	NULL	NULL
7	David Miller	NULL	NULL
8	Sarah Turner	8	Realm of Fantasy
9	Chris Evans	NULL	NULL
10	Olivia Parker	NULL	NULL

At the bottom of the results pane, it says 'Query executed successfully.' and shows statistics: Ln 256, Col 1, Ch 1, INS.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'LibraryDB' is selected. In the center pane, a query window titled 'SQLQuery1.sql' displays the following SQL code:

```

SELECT P.PublisherName, B.Title
FROM Publishers P
LEFT JOIN Books B
ON P.PublisherID = B.PublisherID;

```

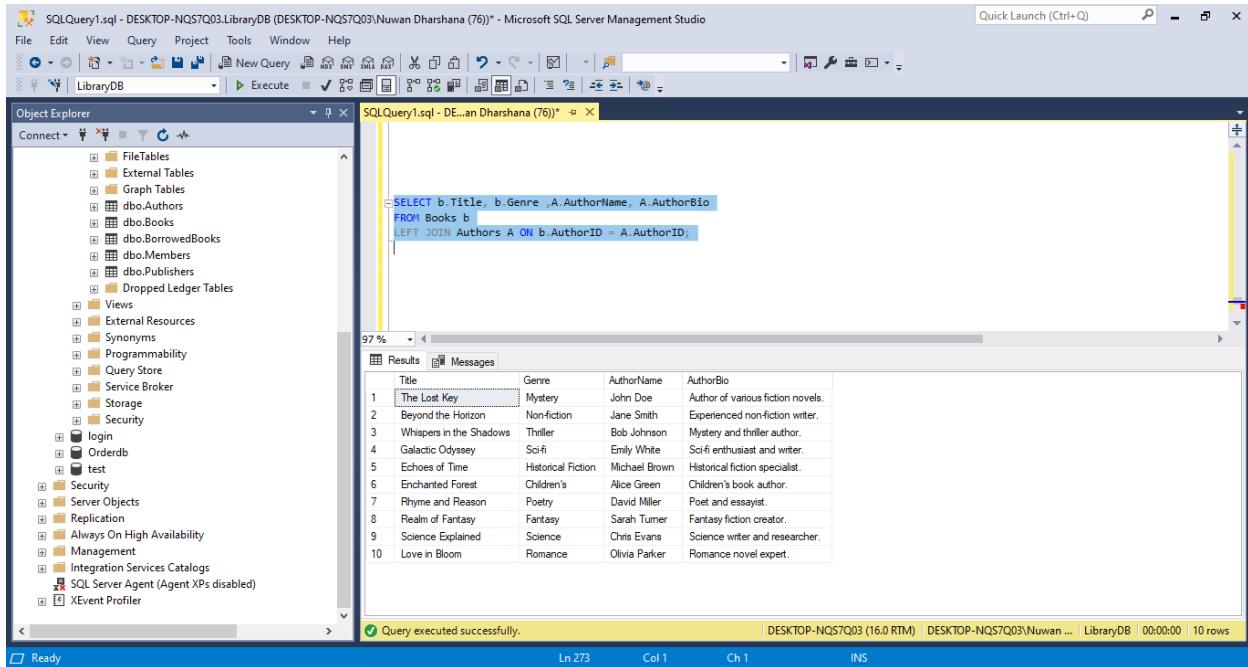
The results pane shows a table with two columns: PublisherName and Title. The data is as follows:

PublisherName	Title
ABC Publications	The Lost Key
XYZ Books	Beyond the Horizon
Book Haven	Whispers in the Shadows
Novel World	Galactic Odyssey
Ink Press	Echoes of Time
Page Turner Publishing	Enchanted Forest
Bookshelf Publishers	Rhyme and Reason
Best Books Co.	Realm of Fantasy
Literary Dreams	Science Explained
Paperback Paradise	Love in Bloom

At the bottom of the results pane, it says 'Query executed successfully.' and shows statistics: Ln 264, Col 34, Ch 34, INS.

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like FileTables, External Tables, Graph Tables, and several dbo.\* tables including Authors, Books, BorrowedBooks, Members, Publishers, Views, and Security-related objects. The central pane displays a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwam Dharshana (76))'. The query itself is:

```
SELECT b.Title, b.Genre, A.AuthorName, A.AuthorBio
FROM Books b
LEFT JOIN Authors A ON b.AuthorID = A.AuthorID;
```

The results pane shows a table with 10 rows of data, corresponding to the 10 books listed in the Books table. The columns are Title, Genre, AuthorName, and AuthorBio. The data is as follows:

	Title	Genre	AuthorName	AuthorBio
1	The Lost Key	Mystery	John Doe	Author of various fiction novels.
2	Beyond the Horizon	Non-fiction	Jane Smith	Experienced non-fiction writer.
3	Whispers in the Shadows	Thriller	Bob Johnson	Mystery and thriller author.
4	Galactic Odyssey	Sci-fi	Emily White	Sci-fi enthusiast and writer.
5	Echoes of Time	Historical Fiction	Michael Brown	Historical fiction specialist.
6	Enchanted Forest	Children's	Alice Green	Children's book author.
7	Rhyme and Reason	Poetry	David Miller	Poet and essayist.
8	Realm of Fantasy	Fantasy	Sarah Turner	Fantasy fiction creator.
9	Science Explained	Science	Chris Evans	Science writer and researcher.
10	Love in Bloom	Romance	Olivia Parker	Romance novel expert.

At the bottom of the results pane, a message says 'Query executed successfully.' and provides details about the execution: DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwam Dharshana (76) | LibraryDB | 00:00:00 | 10 rows.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like FileTables, External Tables, Graph Tables, and tables such as Authors, Books, BorrowedBooks, Members, Publishers, and so on. The central pane displays a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The query is:

```

SELECT m.FirstName, m.LastName, Books.Title, m.Email
FROM Members m
LEFT JOIN BorrowedBooks B ON m.MemberID = B.MemberID
LEFT JOIN Books ON B.BookID = Books.BookID;

```

The results pane shows a table with 10 rows, each containing a member's first name, last name, the title of the book they borrowed, and their email address. The columns are FirstName, LastName, Title, and Email. The data is as follows:

	FirstName	LastName	Title	Email
1	Alice	Johnson	The Lost Key	alice@email.com
2	Bob	Smith	Whispers in the Shadows	bob@email.com
3	Charlie	Williams	Echoes of Time	charlie@email.com
4	David	Brown	Rhyme and Reason	david@email.com
5	Emma	Jones	Science Explained	emma@email.com
6	Frank	Taylor	Beyond the Horizon	frank@email.com
7	Grace	Davis	Galactic Odyssey	grace@email.com
8	Henry	Martin	Enchanted Forest	henry@email.com
9	Ivy	White	Realm of Fantasy	ivy@email.com
10	Jack	Miller	Love in Bloom	jack@email.com

At the bottom, a message indicates 'Query executed successfully.' and shows the session details: DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 10 rows.

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects. The central pane displays a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The query is:

```

SELECT a.AuthorID, a.AuthorName, b.BookID, b.Title
FROM Authors a RIGHT JOIN Books b
ON a.AuthorID = b.AuthorID AND b.Genre = 'Mystery';

```

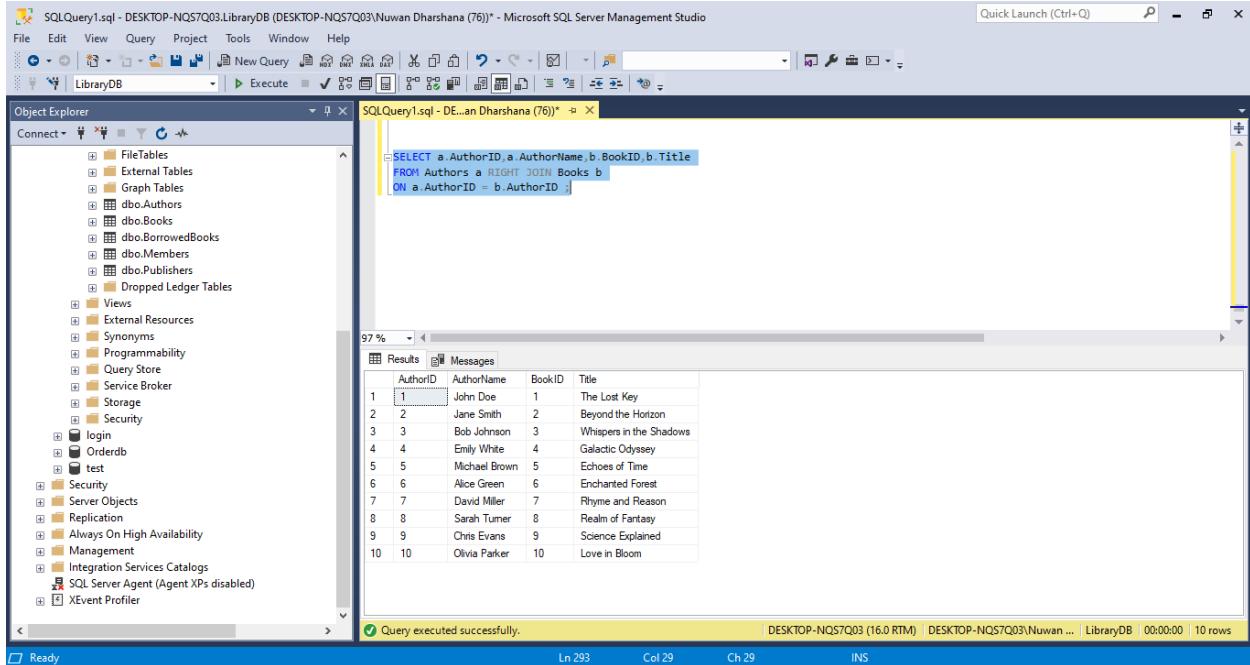
The results pane shows a table with 10 rows, each containing an author's ID, name, the book's ID, and its title. The columns are AuthorID, AuthorName, BookID, and Title. The data is as follows:

	AuthorID	AuthorName	BookID	Title
1	1	John Doe	1	The Lost Key
2	NULL	NULL	2	Beyond the Horizon
3	NULL	NULL	3	Whispers in the Shadows
4	NULL	NULL	4	Galactic Odyssey
5	NULL	NULL	5	Echoes of Time
6	NULL	NULL	6	Enchanted Forest
7	NULL	NULL	7	Rhyme and Reason
8	NULL	NULL	8	Realm of Fantasy
9	NULL	NULL	9	Science Explained
10	NULL	NULL	10	Love in Bloom

At the bottom, a message indicates 'Query executed successfully.' and shows the session details: DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 10 rows.

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like FileTables, External Tables, Graph Tables, etc., under the LibraryDB database. The central pane displays a query window with the following SQL code:

```
SELECT a.AuthorID, a.AuthorName, b.BookID, b.Title
FROM Authors a RIGHT JOIN Books b
ON a.AuthorID = b.AuthorID;
```

The Results pane below shows the output of the query, which includes 10 rows of data:

	AuthorID	AuthorName	BookID	Title
1	1	John Doe	1	The Lost Key
2	2	Jane Smith	2	Beyond the Horizon
3	3	Bob Johnson	3	Whispers in the Shadows
4	4	Emily White	4	Galactic Odyssey
5	5	Michael Brown	5	Echoes of Time
6	6	Alice Green	6	Enchanted Forest
7	7	David Miller	7	Rhyme and Reason
8	8	Sarah Turner	8	Realm of Fantasy
9	9	Chris Evans	9	Science Explained
10	10	Olivia Parker	10	Love in Bloom

At the bottom of the Results pane, a message states "Query executed successfully." and provides details about the execution: DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 10 rows.

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.

SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76)) - Microsoft SQL Server Management Studio

```
SELECT P.PublisherName, B.Title
FROM Publishers P
RIGHT JOIN Books B
ON P.PublisherID = B.PublisherID;
```

Results

PublisherName	Title
ABC Publications	The Lost Key
XYZ Books	Beyond the Horizon
Book Haven	Whispers in the Shadows
Novel World	Galactic Odyssey
Ink Press	Echoes of Time
Page Turner Publishing	Enchanted Forest
Bookshelf Publishers	Rhyme and Reason
Best Books Co.	Realm of Fantasy
Literary Dreams	Science Explained
Paperback Paradise	Love in Bloom

Query executed successfully.

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.

SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76)) - Microsoft SQL Server Management Studio

```
SELECT b.Title, b.Genre, A.AuthorName, A.AuthorBio
FROM Books b
RIGHT JOIN Authors A ON b.AuthorID = A.AuthorID;
```

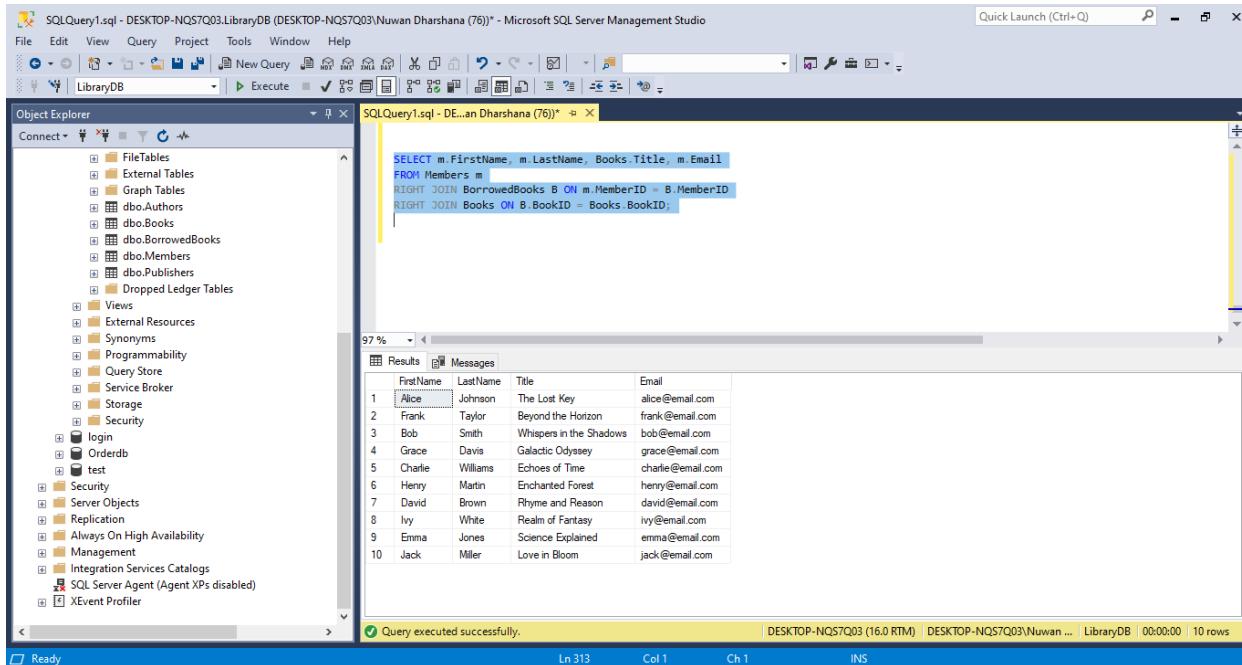
Results

Title	Genre	AuthorName	AuthorBio
The Lost Key	Mystery	John Doe	Author of various fiction novels.
Beyond the Horizon	Non-fiction	Jane Smith	Experienced non-fiction writer.
Whispers in the Shadows	Thriller	Bob Johnson	Mystery and thriller author.
Galactic Odyssey	Sci-fi	Emily White	Sci-fi enthusiast and writer.
Echoes of Time	Historical Fiction	Michael Brown	Historical fiction specialist.
Enchanted Forest	Children's	Alice Green	Children's book author.
Rhyme and Reason	Poetry	David Miller	Poet and essayist.
Realm of Fantasy	Fantasy	Sarah Turner	Fantasy fiction creator.
Science Explained	Science	Chris Evans	Science writer and researcher.
Love in Bloom	Romance	Olivia Parker	Romance novel expert.

Query executed successfully.

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the LibraryDB database selected. The right pane contains a query window with the following T-SQL code:

```
SELECT m.FirstName, m.LastName, Books.Title, m.Email
FROM Members m
RIGHT JOIN BorrowedBooks B ON m.MemberID = B.MemberID
RIGHT JOIN Books ON B.BookID = Books.BookID;
```

Below the code, the Results tab shows a table with 10 rows of data:

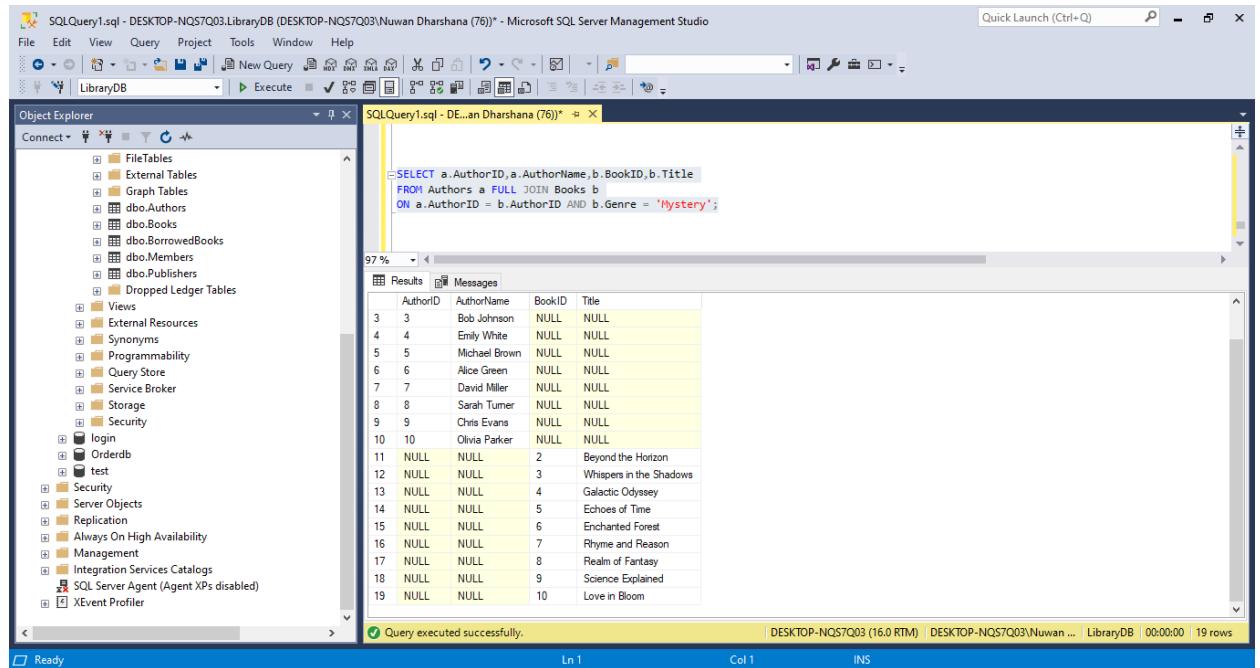
	FirstName	LastName	Title	Email
1	Alice	Johnson	The Lost Key	alice@email.com
2	Frank	Taylor	Beyond the Horizon	frank@email.com
3	Bob	Smith	Whispers in the Shadows	bob@email.com
4	Grace	Davis	Galactic Odyssey	grace@email.com
5	Charlie	Williams	Echoes of Time	charlie@email.com
6	Henry	Martin	Enchanted Forest	henry@email.com
7	David	Brown	Rhyme and Reason	david@email.com
8	Ivy	White	Realm of Fantasy	ivy@email.com
9	Emma	Jones	Science Explained	emma@email.com
10	Jack	Miller	Love in Bloom	jack@email.com

At the bottom of the results window, a message indicates "Query executed successfully." The status bar at the bottom of the screen shows "Ready", "Ln 313", "Col 1", "Ch 1", and "INS".

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like FileTables, External Tables, Graph Tables, and tables such as Authors, Books, BorrowedBooks, Members, Publishers, and Views. The central Results pane displays the output of a SQL query:

```
SELECT a.AuthorID, a.AuthorName, b.BookID, b.Title
FROM Authors a FULL JOIN Books b
ON a.AuthorID = b.AuthorID AND b.Genre = 'Mystery';
```

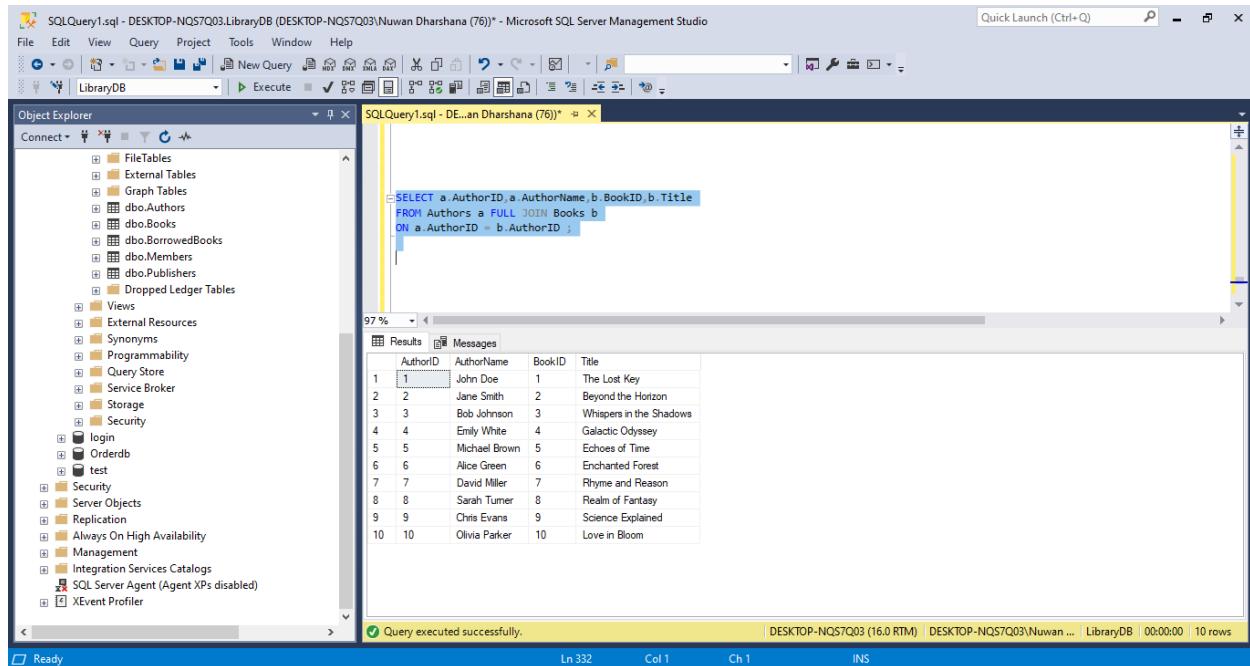
The results show 19 rows of data, where authors without matching books have NULL values in the BookID and Title columns.

AuthorID	AuthorName	BookID	Title
3	Bob Johnson	NULL	NULL
4	Emily White	NULL	NULL
5	Michael Brown	NULL	NULL
6	Alice Green	NULL	NULL
7	David Miller	NULL	NULL
8	Sarah Turner	NULL	NULL
9	Chris Evans	NULL	NULL
10	Olivia Parker	NULL	NULL
11	NULL	2	Beyond the Horizon
12	NULL	3	Whispers in the Shadows
13	NULL	4	Galactic Odyssey
14	NULL	5	Echoes of Time
15	NULL	6	Enchanted Forest
16	NULL	7	Rhyme and Reason
17	NULL	8	Realm of Fantasy
18	NULL	9	Science Explained
19	NULL	10	Love in Bloom

Query executed successfully. | DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 19 rows

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects. The central Results pane displays the output of a SQL query:

```
SELECT a.AuthorID, a.AuthorName, b.BookID, b.Title
FROM Authors a FULL JOIN Books b
ON a.AuthorID = b.AuthorID ;
```

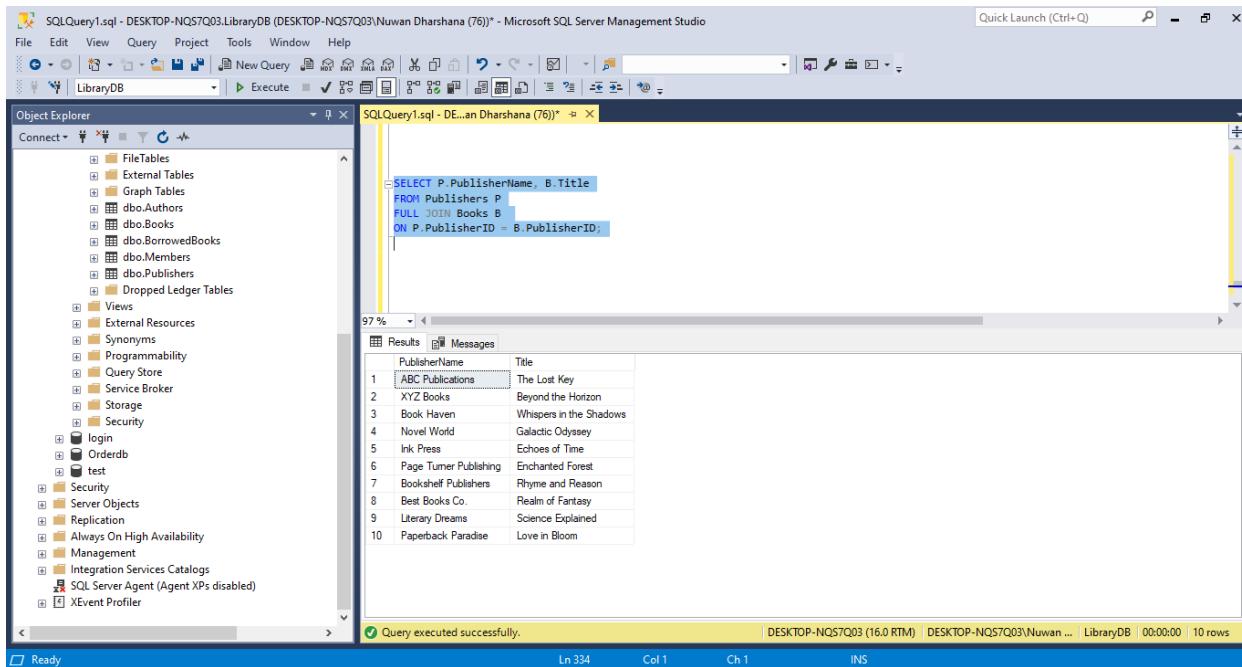
The results show 10 rows of data, where authors without matching books have NULL values in the BookID and Title columns.

AuthorID	AuthorName	BookID	Title
1	John Doe	1	The Lost Key
2	Jane Smith	2	Beyond the Horizon
3	Bob Johnson	3	Whispers in the Shadows
4	Emily White	4	Galactic Odyssey
5	Michael Brown	5	Echoes of Time
6	Alice Green	6	Enchanted Forest
7	David Miller	7	Rhyme and Reason
8	Sarah Turner	8	Realm of Fantasy
9	Chris Evans	9	Science Explained
10	Olivia Parker	10	Love in Bloom

Query executed successfully. | DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 10 rows

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.



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like FileTables, External Tables, Graph Tables, and tables such as Authors, Books, and Publishers. The central pane displays a query window titled 'SQLQuery1.sql'. The query is:

```
SELECT P.PublisherName, B.Title
FROM Publishers P
FULL JOIN Books B
ON P.PublisherID = B.PublisherID;
```

The results pane shows a table with two columns: 'PublisherName' and 'Title'. There are 10 rows of data, each containing a publisher name and its corresponding book title. The data is as follows:

PublisherName	Title
ABC Publications	The Lost Key
XYZ Books	Beyond the Horizon
Book Haven	Whispers in the Shadows
Novel World	Galactic Odyssey
Ink Press	Echoes of Time
Page Turner Publishing	Enchanted Forest
Bookshelf Publishers	Rhyme and Reason
Best Books Co.	Realm of Fantasy
Literary Dreams	Science Explained
Paperback Paradise	Love in Bloom

At the bottom of the results pane, a message says 'Query executed successfully.' and shows statistics: DESKTOP-NQS7Q03 (16.0 RTM) | DESKTOP-NQS7Q03\Nuwan ... | LibraryDB | 00:00:00 | 10 rows.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))' displays the following SQL code:

```

SELECT b.Title, b.Genre, A.AuthorName, A.AuthorBio
FROM Books b
FULL JOIN Authors A ON b.AuthorID = A.AuthorID;

```

The results pane shows the output of the query, which includes 10 rows of data. The columns are Title, Genre, AuthorName, and AuthorBio. The data is as follows:

	Title	Genre	AuthorName	AuthorBio
1	The Lost Key	Mystery	John Doe	Author of various fiction novels.
2	Beyond the Horizon	Non-fiction	Jane Smith	Experienced non-fiction writer.
3	Whispers in the Shadows	Thriller	Bob Johnson	Mystery and thriller author.
4	Galactic Odyssey	Sci-fi	Emily White	Sci-fi enthusiast and writer.
5	Echoes of Time	Historical Fiction	Michael Brown	Historical fiction specialist.
6	Enchanted Forest	Children's	Alice Green	Children's book author.
7	Rhyme and Reason	Poetry	David Miller	Poet and essayist.
8	Realm of Fantasy	Fantasy	Sarah Turner	Fantasy fiction creator.
9	Science Explained	Science	Chris Evans	Science writer and researcher.
10	Love in Bloom	Romance	Olivia Parker	Romance novel expert.

At the bottom of the results pane, a message says 'Query executed successfully.' and the status bar indicates 'LN 337 Col 1 Ch 1 INS'.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'LibraryDB' database is selected. In the center pane, a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))' displays the following SQL code:

```

SELECT m.FirstName, m.LastName, Books.Title, m.Email
FROM Members m
FULL JOIN BorrowedBooks B ON m.MemberID = B.MemberID
FULL JOIN Books ON B.BookID = Books.BookID;

```

The results pane shows the output of the query, which includes 10 rows of data. The columns are FirstName, LastName, Title, and Email. The data is as follows:

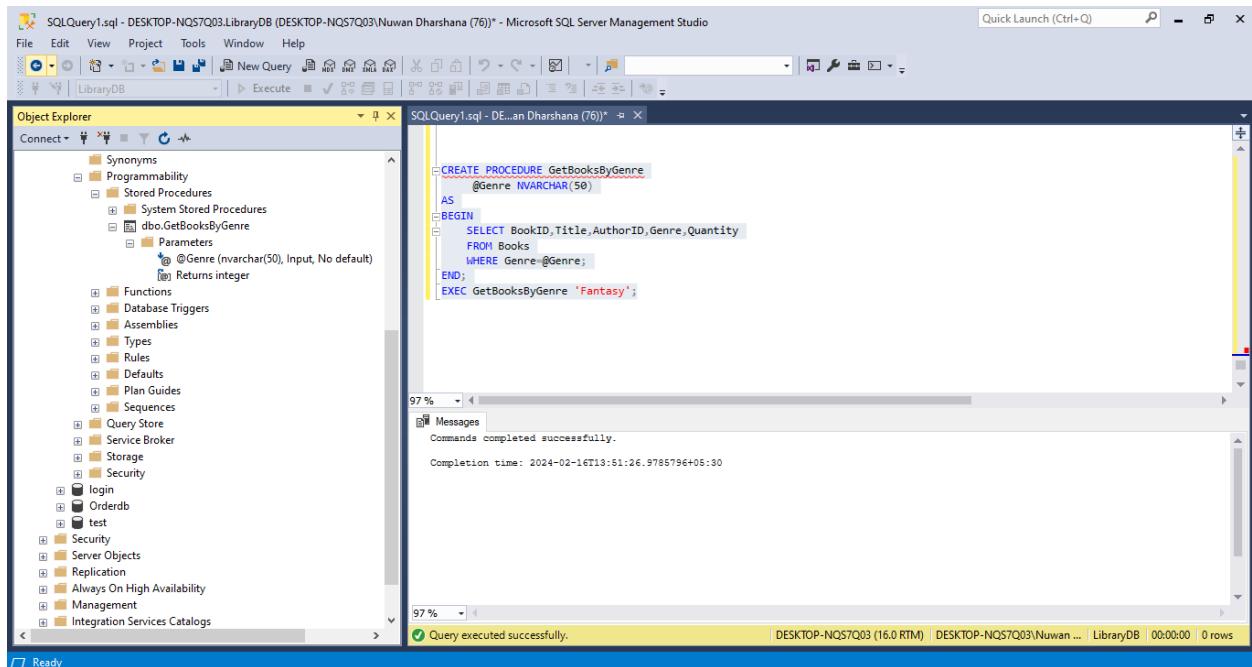
	FirstName	LastName	Title	Email
1	Alice	Johnson	The Lost Key	alice@email.com
2	Bob	Smith	Whispers in the Shadows	bob@email.com
3	Charlie	Williams	Echoes of Time	charlie@email.com
4	David	Brown	Rhyme and Reason	david@email.com
5	Emma	Jones	Science Explained	emma@email.com
6	Frank	Taylor	Beyond the Horizon	frank@email.com
7	Grace	Davis	Galactic Odyssey	grace@email.com
8	Henry	Martin	Enchanted Forest	henry@email.com
9	Ivy	White	Realm of Fantasy	ivy@email.com
10	Jack	Miller	Love in Bloom	jack@email.com

At the bottom of the results pane, a message says 'Query executed successfully.' and the status bar indicates 'LN 341 Col 1 Ch 1 INS'.

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



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including Synonyms, Programmability (Stored Procedures), System Stored Procedures, and Parameters. A new stored procedure, `dbo.GetBooksByGenre`, has been created under Parameters. The script pane contains the following T-SQL code:

```
CREATE PROCEDURE GetBooksByGenre
    @Genre NVARCHAR(50)
AS
BEGIN
    SELECT BookID,Title,AuthorID,Genre,Quantity
    FROM Books
    WHERE Genre = @Genre;
END;
EXEC GetBooksByGenre 'Fantasy';
```

The Messages pane at the bottom right indicates that the command was completed successfully with a completion time of 2024-02-16T13:51:26.9785796+05:30. The status bar at the bottom shows "Query executed successfully." and other system information.

Figure 59:Create Store Procedure

Gets books from the database based on a specific genre.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several database objects including Employee1, EmployeeDetails1, EmployeeDetailsNew, FullIDB, FullDiffDB, Index\_Example, LibraryDB, and others. The central pane displays a T-SQL script for creating a stored procedure:

```

CREATE PROCEDURE InsertNewAuthor
    @AuthorName NVARCHAR(100),
    @AuthorBio NVARCHAR(50)
AS
BEGIN
    INSERT INTO Authors(AuthorName, AuthorBio)
    VALUES(@AuthorName , @AuthorBio);
END;
EXEC InsertNewAuthor 'A.N Martha', 'Love';

```

The status bar at the bottom indicates "Query executed successfully." and "0 rows".

Figure 60:Create Store Procedure

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects. The central pane displays a T-SQL script for creating a stored procedure:

```

CREATE PROCEDURE GetBorrowedBooksByMemberID
    @MemberID INT
AS
BEGIN
    SELECT BookID, BorrowDate, ReturnDate
    FROM BorrowedBooks
    WHERE MemberID = @MemberID;
END;
EXEC GetBorrowedBooksByMemberID 1;

```

The status bar at the bottom indicates "Query executed successfully." and "0 rows".

Figure 61:Create Store Procedure

Gets the borrowed books information for a given member ID.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several database objects including Employee1, EmployeeDetails1, EmployeeDetailsNew, FullDB, FullDiffDB, Index\_Example, LibraryDB, Database Diagrams, Tables, Views, External Resources, Synonyms, Programmability, and Stored Procedures. The right pane displays a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The code in the window is:

```

CREATE PROCEDURE UpdateAddress
    @MemberID INT,
    @NewAddress NVARCHAR(50)
AS
BEGIN
    UPDATE Members
    SET Address = @NewAddress
    WHERE MemberID = @MemberID
END;
EXEC UpdateAddress 1, '103 Main st,Town';

```

The status bar at the bottom indicates 'Query executed successfully.' and 'Completion time: 2024-02-16T14:31:07.3430417+05:30'.

Figure 62:Create Store Procedure

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects including LibraryDB, System Stored Procedures, Functions, and Programmability. The right pane displays a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The code in the window is:

```

CREATE PROCEDURE UpdatePublisherAddress
    @PublisherID INT,
    @NewName NVARCHAR(100)
AS
BEGIN
    UPDATE Publishers
    SET PublisherAddress = @NewName
    WHERE PublisherID = @PublisherID
END;
EXEC UpdatePublisherAddress 1, 'sadun';

```

The status bar at the bottom indicates 'Query executed successfully.' and 'Completion time: 2024-02-16T14:41:15.1794415+05:30'.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure of 'LibraryDB' with various objects like Employee1, EmployeeDetails1, FullDB, FullDiffDB, Index\_Example, and LibraryDB. The main window contains a query editor with the following SQL code:

```
CREATE VIEW AuthorView AS
SELECT AuthorName , AuthorBio
FROM Authors;
```

Below the query editor, the 'Messages' pane shows the output: "Commands completed successfully." and "Completion time: 2024-02-16T14:48:46.0713758+06:30". At the bottom, a status bar indicates "Query executed successfully." and other session details.

Figure 64:Create View

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

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure of 'LibraryDB'. The main window contains a query editor with the following SQL code:

```
CREATE VIEW BookView AS
SELECT Title,AuthorID,ISBN,Genre,Quantity
FROM Books;
```

Below the query editor, the 'Messages' pane shows the output: "Commands completed successfully." and "Completion time: 2024-02-16T14:50:33.8642484+06:30". At the bottom, a status bar indicates "Query executed successfully." and other session details.

Figure 65:Create View

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects like Employee1, EmployeeDetails1, and LibraryDB. The central query window contains the following SQL code:

```
CREATE VIEW MemberView AS  
SELECT FirstName, LastName, Address, Phone, Email  
FROM Members;
```

The status bar at the bottom indicates "Query executed successfully." and provides completion details: "Completion time: 2024-02-16T14:52:12.5171034+05:30".

Figure 66:Create View

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

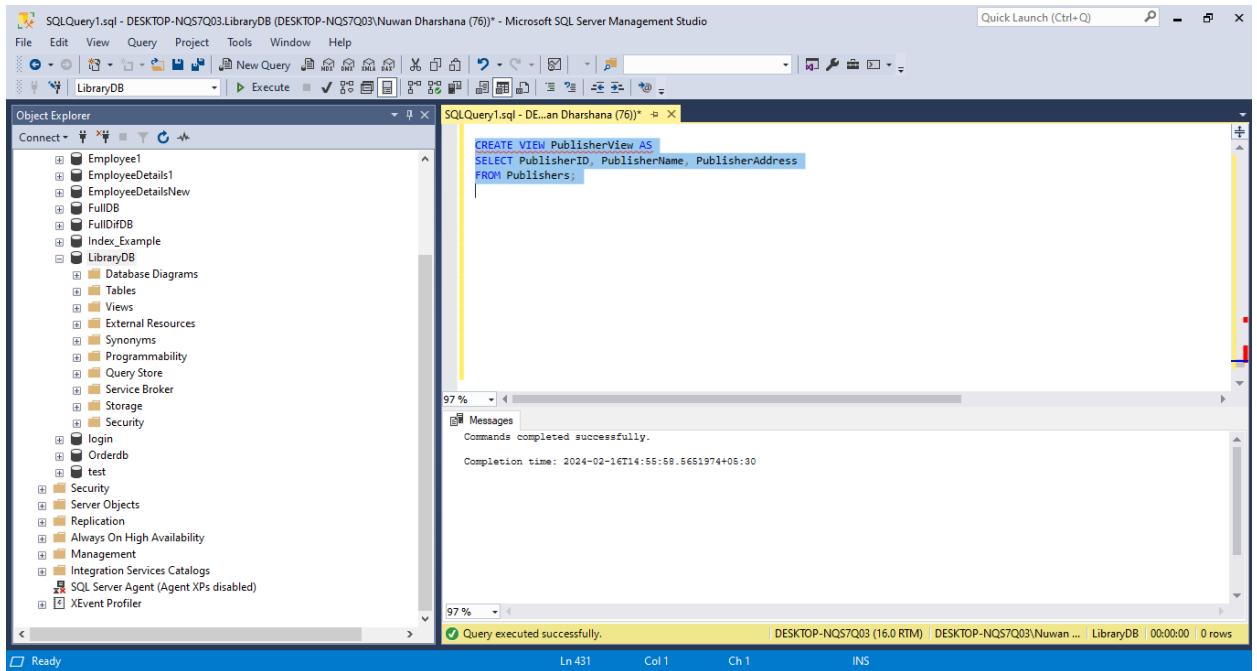
The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various database objects. The central query window contains the following SQL code:

```
CREATE VIEW BorrowedBooksView AS  
SELECT MemberID, BookID, BorrowDate, ReturnDate  
FROM BorrowedBooks;
```

The status bar at the bottom indicates "Query executed successfully." and provides completion details: "Completion time: 2024-02-16T14:54:08.0161300+05:30".

Figure 67:Create View

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



The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays various database objects like Employee1, EmployeeDetails1, EmployeeDetailsNew, FullDB, FullDiffDB, Index\_Example, LibraryDB, login, Orderdb, test, Security, Server Objects, Replication, Always On High Availability, Management, Integration Services Catalogs, and SQL Server Agent. The main central area contains a query window titled 'SQLQuery1.sql - DESKTOP-NQ57Q03.LibraryDB (DESKTOP-NQ57Q03\Nuwan Dharshana (76)) - Microsoft SQL Server Management Studio'. The query is:

```
CREATE VIEW PublisherView AS  
SELECT PublisherID, PublisherName, PublisherAddress  
FROM Publishers;
```

The status bar at the bottom indicates 'Query executed successfully.' and shows the completion time as 2024-02-16T14:55:58.5661974+05:30. The bottom right corner of the status bar also shows '0 rows'.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer pane on the left displays the database structure, including the Security folder which contains Logins and Server Roles. The Logins folder is expanded, showing various system logins like sa, sa, and several Windows logins. Two new logins have been created: 'Admin' and 'Seller'. The 'Admin' login is highlighted with a blue selection bar. The SQL Query Editor pane on the right contains the T-SQL commands used to create these logins:

```
CREATE LOGIN Admin WITH PASSWORD = 'admin123';
CREATE LOGIN Seller WITH PASSWORD = 'seller123';
```

The status bar at the bottom indicates the command completed successfully with a completion time of 2024-02-16T15:38:25.6318169+05:30.

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'Security' node under 'LibraryDB' is expanded, showing 'Users', 'Roles', and other security components. In the center pane, a query window titled 'SQLQuery1.sql' contains the following T-SQL code:

```
CREATE LOGIN Admin WITH PASSWORD = 'admin123';
CREATE LOGIN Seller WITH PASSWORD = 'seller123';

--create users and Associate with Logins
CREATE USER adminUser FOR LOGIN Admin;
CREATE USER sellerUser FOR LOGIN Seller;
```

The 'Messages' pane at the bottom displays the output: "Commands completed successfully." and "Completion time: 2024-02-16T16:03:44.8292032+05:30". The status bar at the bottom right shows "Query executed successfully.", "DESKTOP-NQS7Q03 (16.0 RTM)", "DESKTOP-NQS7Q03\Nuwan ... LibraryDB", "00:00:00", and "0 rows".

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the 'Security' node under 'LibraryDB' is expanded, showing 'Users', 'Roles', and other security components. In the center pane, a query window titled 'SQLQuery1.sql' contains the following T-SQL code:

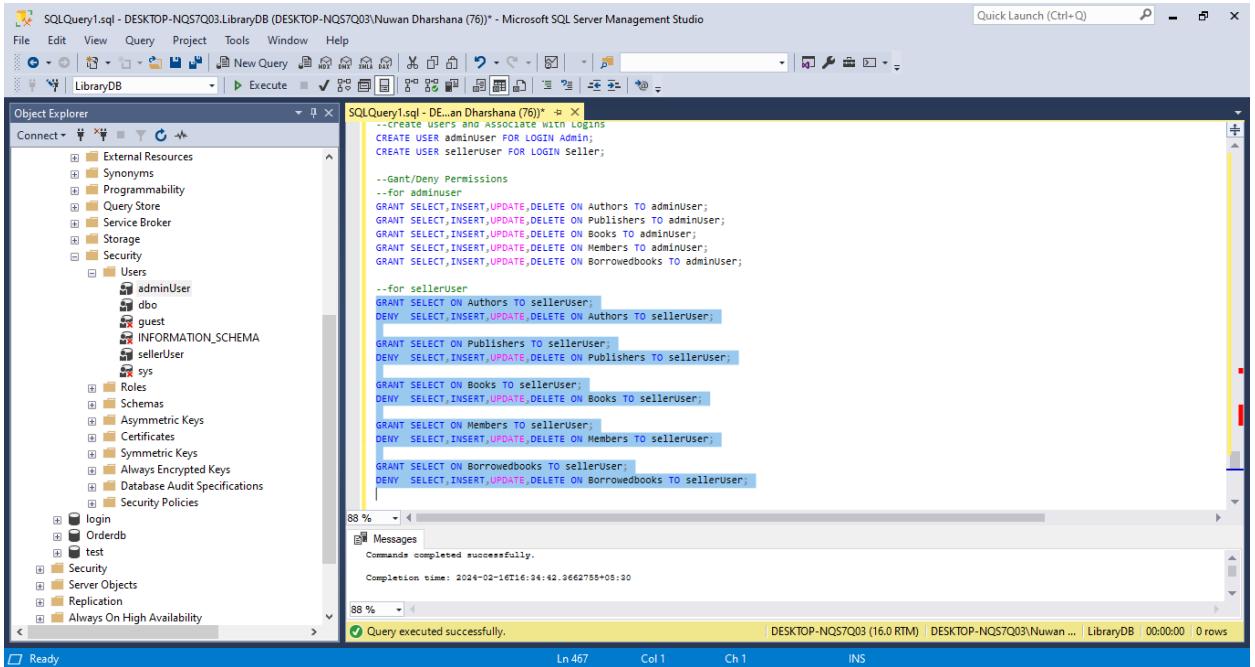
```
--create users and Associate with Logins
CREATE USER adminUser FOR LOGIN Admin;
CREATE USER sellerUser FOR LOGIN Seller;

--Grant/Deny Permissions
--for adminUser
GRANT SELECT, INSERT, UPDATE, DELETE ON Authors TO adminUser;
GRANT SELECT, INSERT, UPDATE, DELETE ON Publishers TO adminUser;
GRANT SELECT, INSERT, UPDATE, DELETE ON Books TO adminUser;
GRANT SELECT, INSERT, UPDATE, DELETE ON Members TO adminUser;
GRANT SELECT, INSERT, UPDATE, DELETE ON Borrowedbooks TO adminUser;
```

The 'Messages' pane at the bottom displays the output: "Commands completed successfully." and "Completion time: 2024-02-16T16:28:09.7350083+05:30". The status bar at the bottom right shows "Query executed successfully.", "DESKTOP-NQS7Q03 (16.0 RTM)", "DESKTOP-NQS7Q03\Nuwan ... LibraryDB", "00:00:00", and "0 rows".

Figure 71:Create Two Logins

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



```
--create users and Associate with Logins
CREATE USER adminUser FOR LOGIN Admin;
CREATE USER sellerUser FOR LOGIN Seller;

--Grant/Deny Permissions
--for adminUser
GRANT SELECT,INSERT,UPDATE,DELETE ON Authors TO adminUser;
GRANT SELECT,INSERT,UPDATE,DELETE ON Publishers TO adminUser;
GRANT SELECT,INSERT,UPDATE,DELETE ON Books TO adminUser;
GRANT SELECT,INSERT,UPDATE,DELETE ON Members TO adminUser;
GRANT SELECT,INSERT,UPDATE,DELETE ON Borrowedbooks TO adminUser;

--for sellerUser
GRANT SELECT ON Authors TO sellerUser;
DENY SELECT,INSERT,UPDATE,DELETE ON Authors TO sellerUser;

GRANT SELECT ON Publishers TO sellerUser;
DENY SELECT,INSERT,UPDATE,DELETE ON Publishers TO sellerUser;

GRANT SELECT ON Books TO sellerUser;
DENY SELECT,INSERT,UPDATE,DELETE ON Books TO sellerUser;

GRANT SELECT ON Members TO sellerUser;
DENY SELECT,INSERT,UPDATE,DELETE ON Members TO sellerUser;

GRANT SELECT ON Borrowedbooks TO sellerUser;
DENY SELECT,INSERT,UPDATE,DELETE ON Borrowedbooks TO sellerUser;
```

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.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including Security, Roles, and Database Roles. The Database Roles node is expanded, showing various system and application roles. The central pane displays a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The query contains the following T-SQL code:

```
-->--  
--Create Roles  
CREATE ROLE AdminRole;  
CREATE ROLE SellerRole;
```

The status bar at the bottom right indicates 'Query executed successfully.' and 'Completion time: 2024-02-16T16:42:26.5871287+05:30'.

Figure 73:grant permission

Creates a role named "AdminRole" in the database.

Creates a role named "SellerRole" in the database.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including Security, Roles, and Database Roles. The Database Roles node is expanded, showing various system and application roles. The central pane displays a query window titled 'SQLQuery1.sql - DESKTOP-NQS7Q03.LibraryDB (DESKTOP-NQS7Q03\Nuwan Dharshana (76))'. The query contains the following T-SQL code:

```
-->--  
--Create Roles  
CREATE ROLE AdminRole;  
CREATE ROLE SellerRole;
```

Below this, there is a section titled '--Grant/Deny Permissions' with the following code:

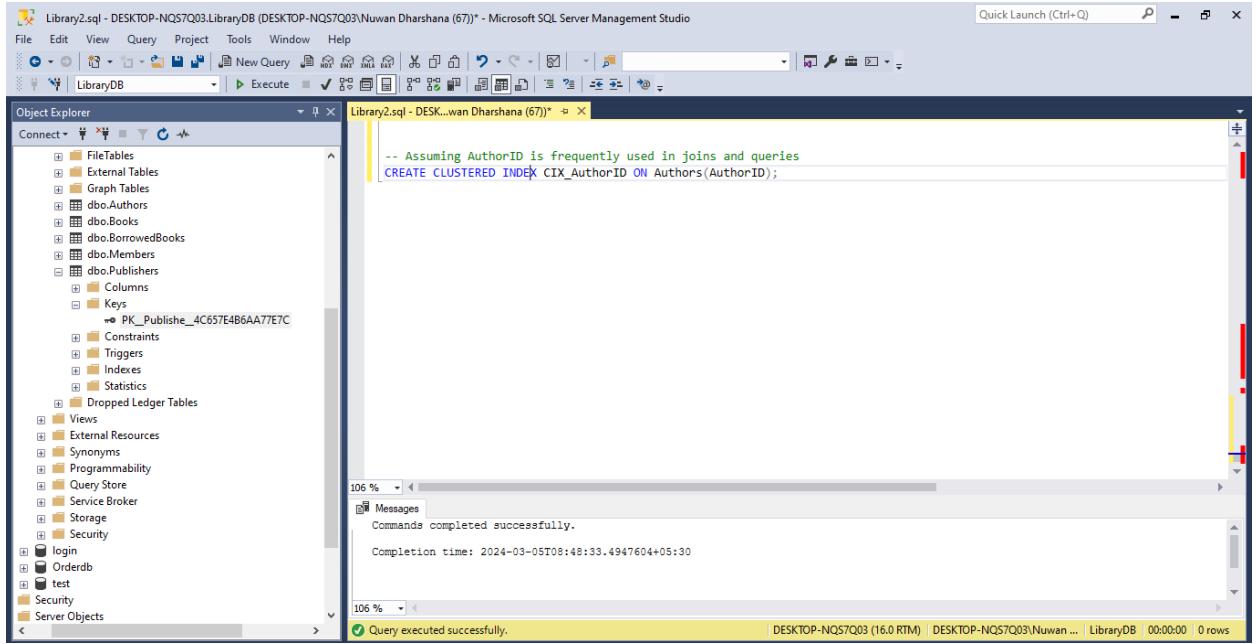
```
-->--  
--Grant/Deny Permissions  
--for adminRole  
GRANT SELECT,INSERT,UPDATE,DELETE ON Authors TO adminRole;  
GRANT SELECT,INSERT,UPDATE,DELETE ON Publishers TO adminRole;  
GRANT SELECT,INSERT,UPDATE,DELETE ON Books TO adminRole;  
GRANT SELECT,INSERT,UPDATE,DELETE ON Members TO adminRole;  
GRANT SELECT,INSERT,UPDATE,DELETE ON Borrowedbooks TO adminRole;
```

The status bar at the bottom right indicates 'Query executed successfully.' and 'Completion time: 2024-02-16T16:45:17.0013067+05:30'.

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.

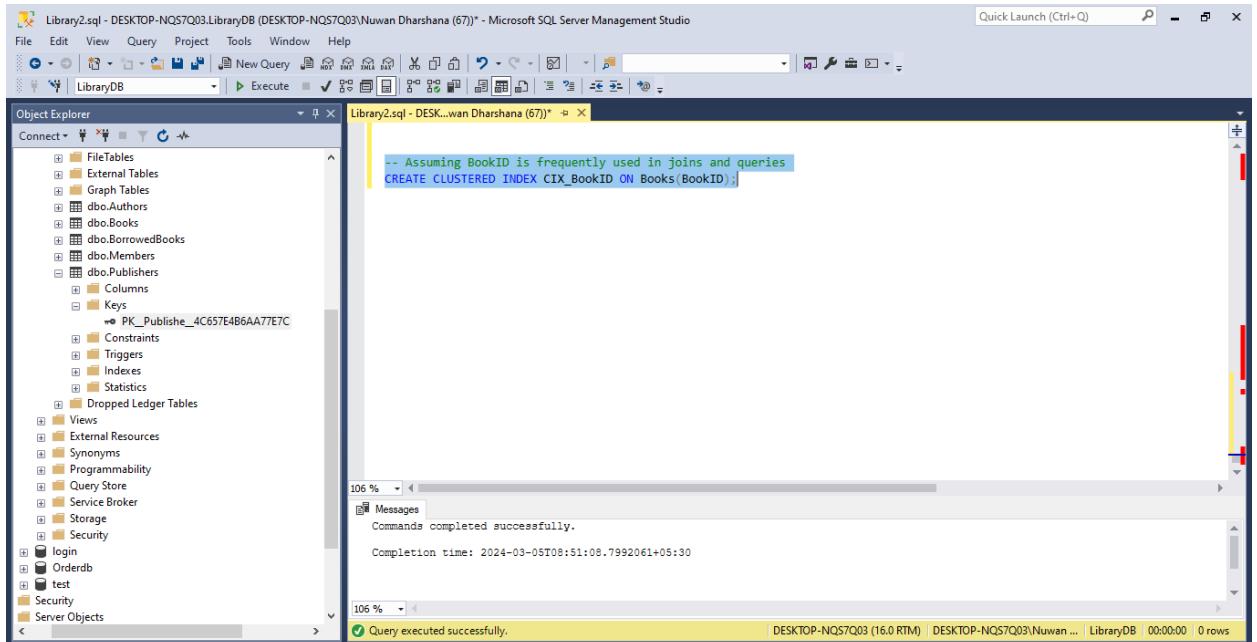


The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database schema, including tables like Authors, Books, and Publishers. The central pane displays a SQL query window with the following code:

```
-- Assuming AuthorID is frequently used in joins and queries
CREATE CLUSTERED INDEX CX_AuthorID ON Authors(AuthorID);
```

The Messages pane at the bottom shows the command completed successfully with a completion time of 2024-03-05T08:48:33.4947604+05:30. The status bar at the bottom right indicates the query executed successfully with 0 rows affected.

Figure 75:Create Cluster Index



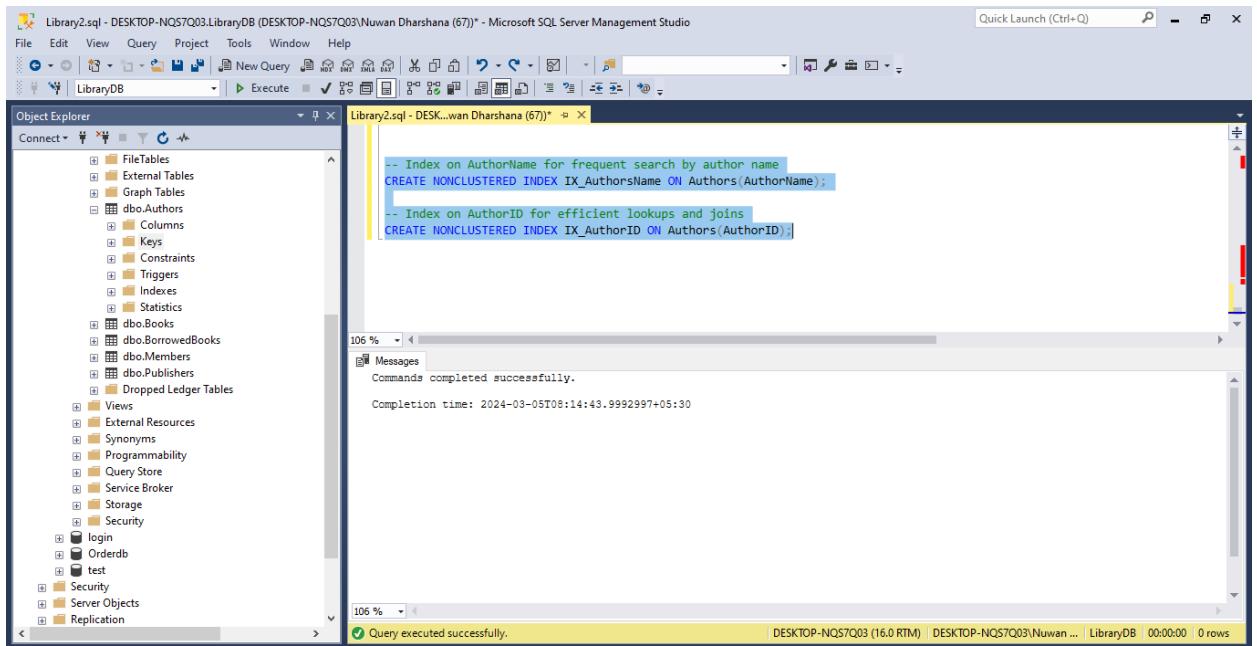
The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database schema, including tables like Authors, Books, and Publishers. The central pane displays a SQL query window with the following code:

```
-- Assuming BookID is frequently used in joins and queries
CREATE CLUSTERED INDEX CX_BookID ON Books(BookID);
```

The Messages pane at the bottom shows the command completed successfully with a completion time of 2024-03-05T08:51:08.7992061+05:30. The status bar at the bottom right indicates the query executed successfully with 0 rows affected.

Figure 76:Create Cluster Index

12) Create two non-cluster indexes on each table



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure for 'LibraryDB'. The 'Authors' table is selected. The 'Script Selection' dropdown at the top right is set to 'CREATE'. The 'Script' tab in the center is active, displaying the following T-SQL code:

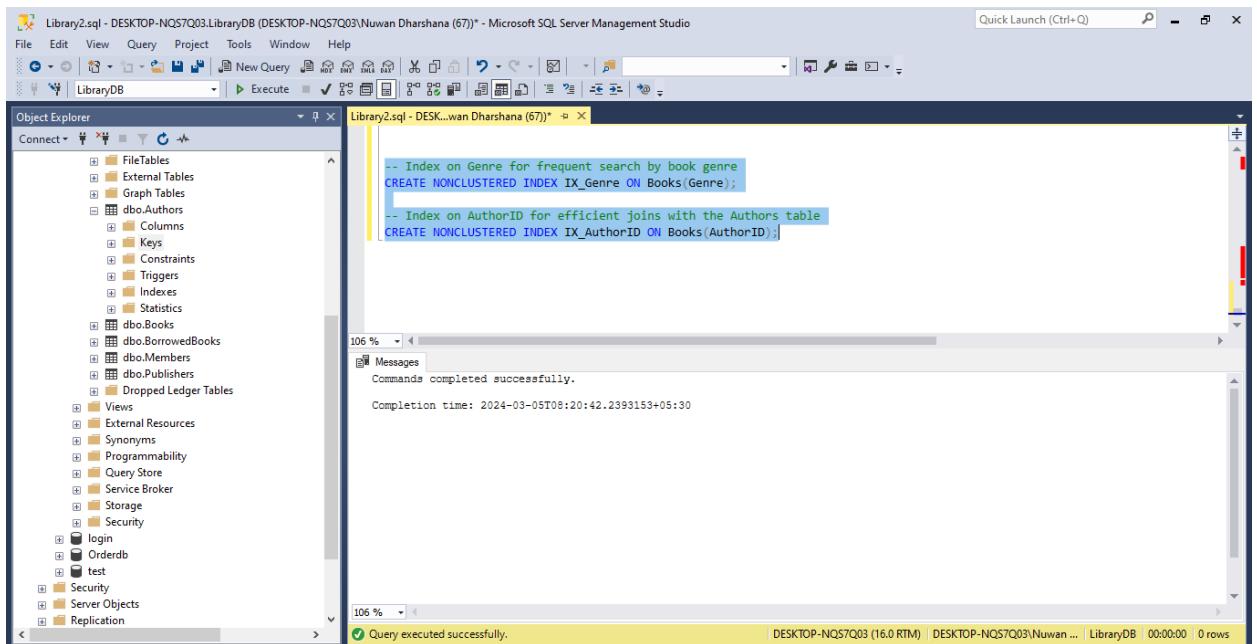
```
-- Index on AuthorName for frequent search by author name  
CREATE NONCLUSTERED INDEX IX_AuthorsName ON Authors(AuthorName);  
  
-- Index on AuthorID for efficient lookups and joins  
CREATE NONCLUSTERED INDEX IX_AuthorID ON Authors(AuthorID);
```

The 'Messages' pane at the bottom shows the execution results:

- Commands completed successfully.
- Completion time: 2024-03-05T08:14:43.9992997+05:30

A yellow status bar at the bottom indicates: Query executed successfully.

Figure 77:Create Noncluster Index



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure for 'LibraryDB'. The 'Books' table is selected. The 'Script Selection' dropdown at the top right is set to 'CREATE'. The 'Script' tab in the center is active, displaying the following T-SQL code:

```
-- Index on Genre for frequent search by book genre  
CREATE NONCLUSTERED INDEX IX_Genre ON Books(Genre);  
  
-- Index on AuthorID for efficient joins with the Authors table  
CREATE NONCLUSTERED INDEX IX_AuthorID ON Books(AuthorID);
```

The 'Messages' pane at the bottom shows the execution results:

- Commands completed successfully.
- Completion time: 2024-03-05T08:20:42.2393153+05:30

A yellow status bar at the bottom indicates: Query executed successfully.

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.

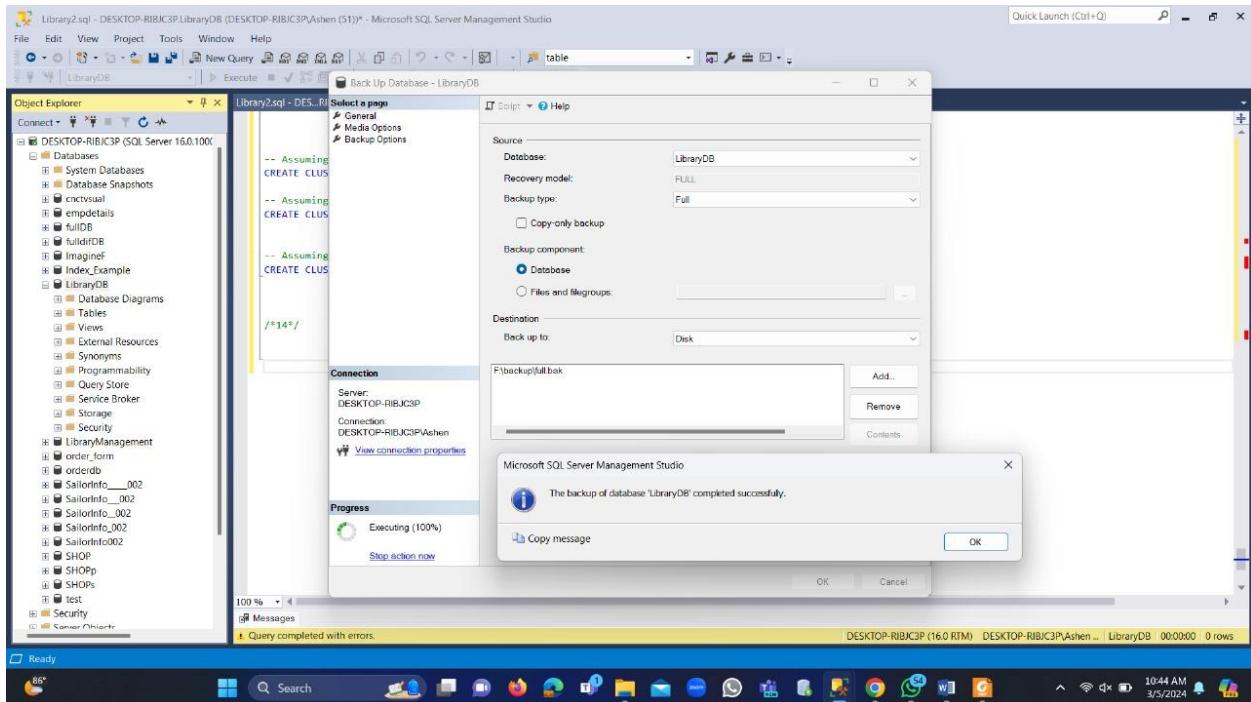


Figure 79:Full Backup

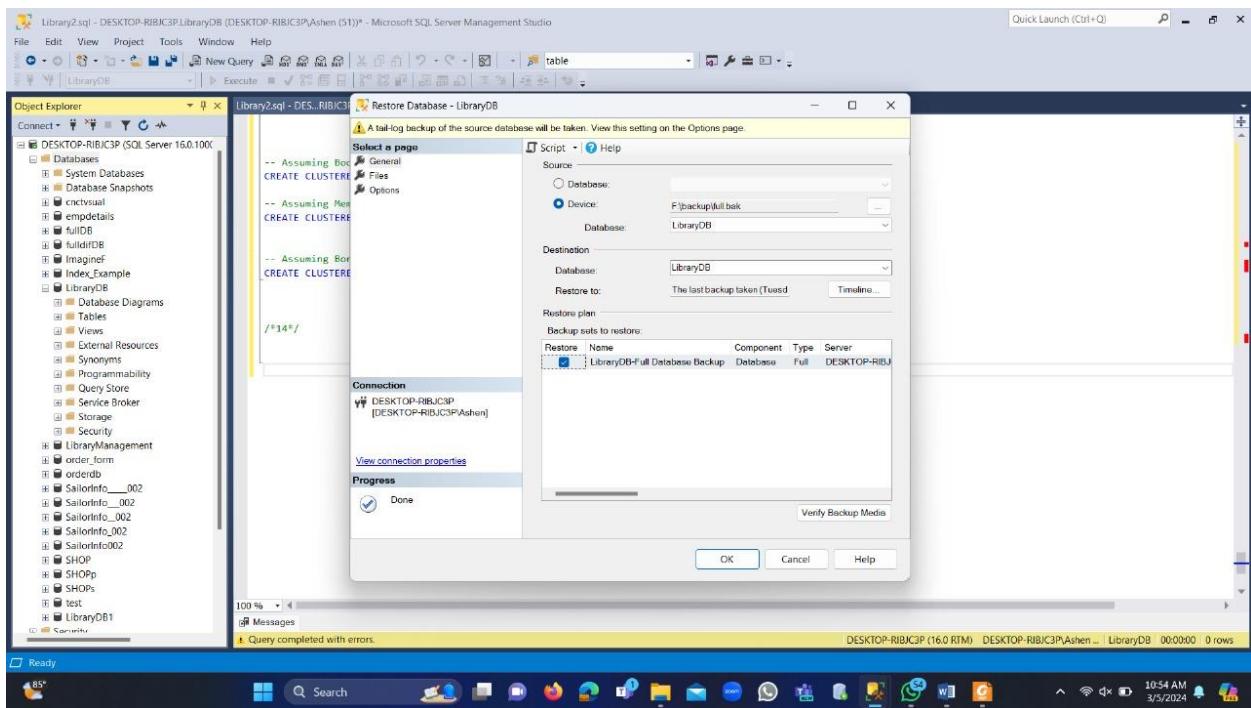


Figure 80:Full Backup

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

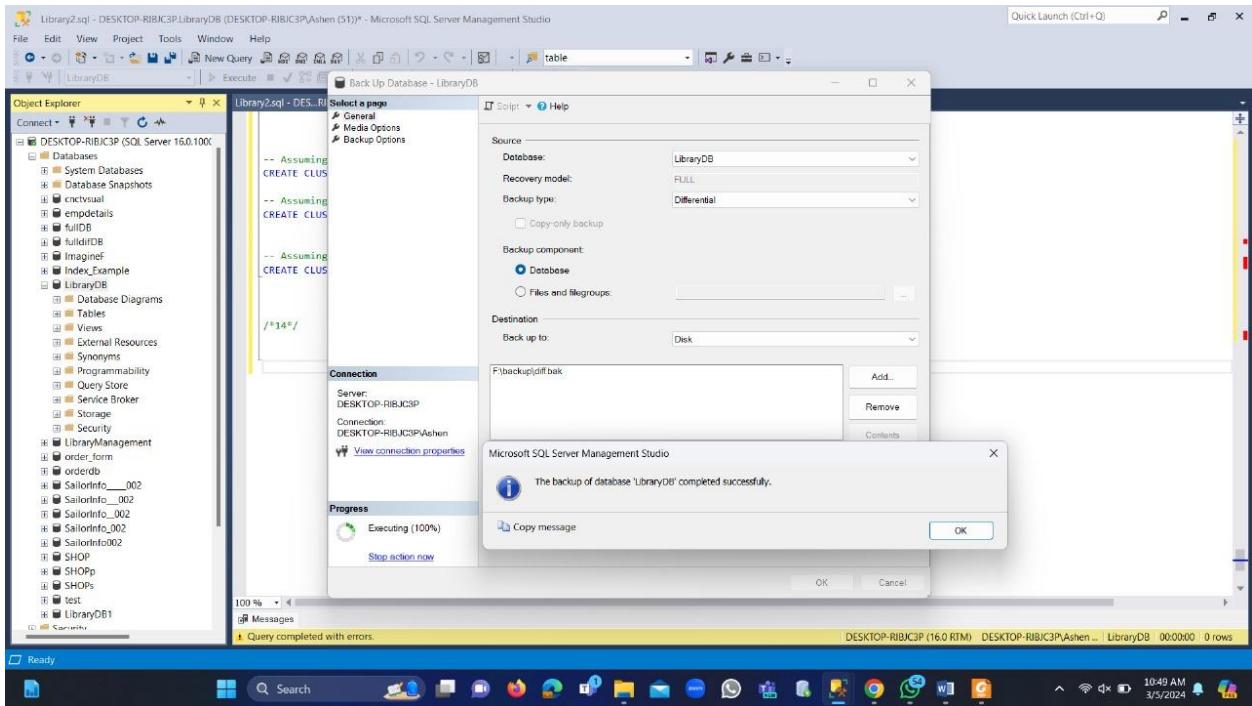


Figure 81:differential backup

**16) Create another new database with restoring differential backup.**

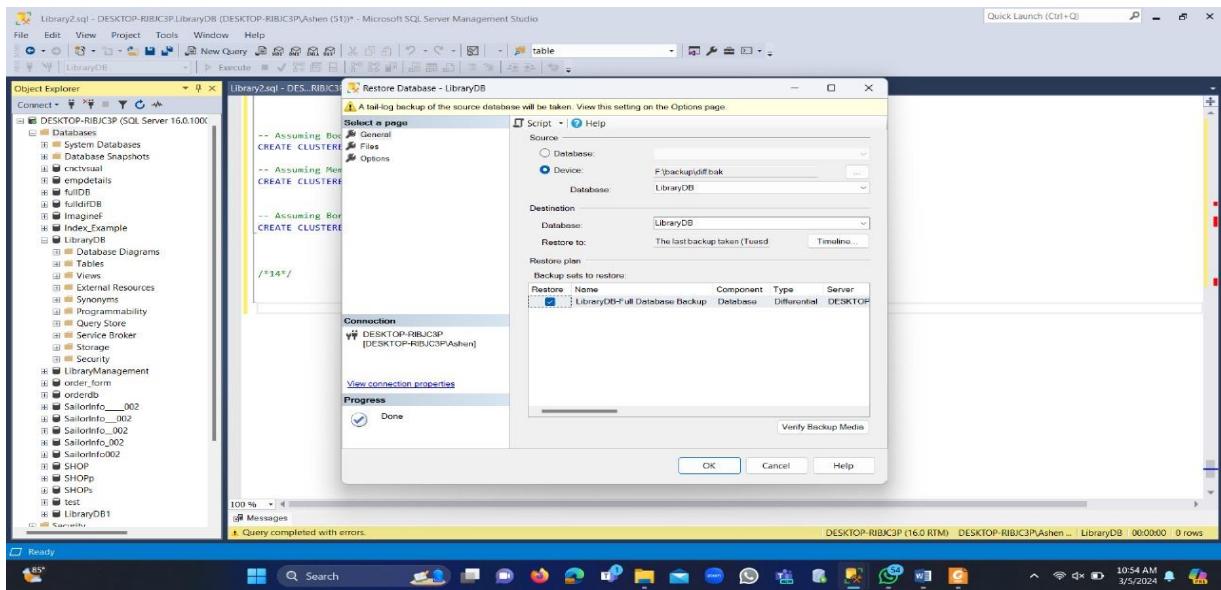


Figure 82:restoring differential backup