Task 1: Inserting Initial Data with Conditions

a) Populate Initial Data:

1. Insert a predefined list of books and authors into the Books and Authors tables. Ensure each book is linked to an author through the AuthorID.

-- Inserting authors

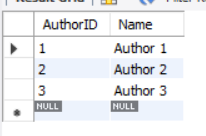
INSERT INTO Authors (Name)

VALUES

('Author 1'),

('Author 2'),

('Author 3');



-- Inserting books

INSERT INTO Books (Title, AuthorID, PublicationDate, Price, StockQuantity)

VALUES

-- High Sales

('Popular Book 1', 1, '2022-05-15', 14.99, 100),

('Popular Book 2', 2, '2022-03-10', 12.99, 150),

('New Release 1', 1, '2023-01-15', 14.99, 100),

('New Release 2', 2, '2023-01-25', 12.99, 80),

-- Recent Releases

('New Release 1', 1, '2023-01-15', 11.99, 50),

('New Release 2', 3, '2023-01-25', 10.99, 40),

-- Low Sales

('Not Popular 1', 2, '2022-11-01', 9.99, 20),

('Not Popular 2', 1, '2022-09-15', 8.99, 25),

-- Other Books

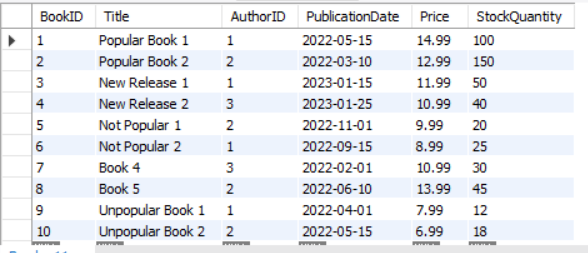
('Book 4', 3, '2022-02-01', 10.99, 30),

('Book 5', 2, '2022-06-10', 13.99, 45);

--no sales in past 3 months

('Unpopular Book 1', 1, '2022-04-01', 7.99, 12),

('Unpopular Book 2', 2, '2022-05-15', 6.99, 18);



2. Insert initial sales records into the Sales table.

-- Inserting initial sales records

INSERT INTO Sales (BookID, SaleDate, Quantity)

VALUES

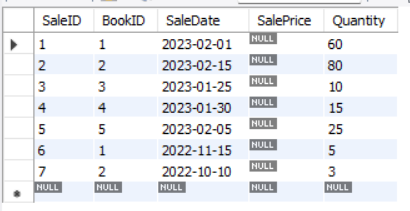
(1, '2023-02-01', 60),

(2, '2023-02-15', 80),

(3, '2023-01-25', 10),

(4, '2023-01-30', 15),

(5, '2023-02-05', 25);



b) Conditional Insertion for Promotions:

If a book's publication date is within the last month, insert it into a new table named NewReleases to highlight recent additions to the bookstore.

-- Insert new releases

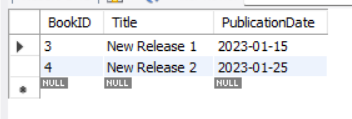
INSERT INTO NewReleases

SELECT b.BookID, b.Title, b.PublicationDate

FROM Books b

WHERE b.PublicationDate >= '2023-01-01'

AND b.PublicationDate < '2023-02-01';



Task 2: Updating Inventory and Pricing Based on Sales Performance

a) Dynamic Pricing Update:

1. Write an SQL script to increase the price of books by 10% that have sold more than 50 copies in the past month, indicating high demand.

-- Increase price by 10% for books sold more than 50 copies in the past month

UPDATE Books

SET Price = Price \* 1.1

WHERE BookID IN (

SELECT BookID

FROM Sales

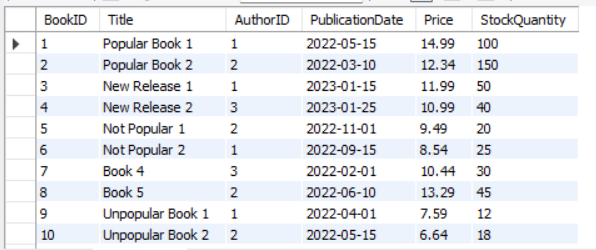
WHERE SaleDate >= '2023-01-01' AND SaleDate < '2023-02-01'

GROUP BY BookID

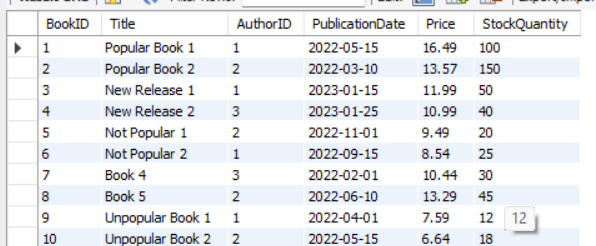
HAVING SUM(Quantity) > 50

);

Before :



After :



2. Conversely, reduce the price by 5% for books that have not sold any copies in the past three months, indicating low demand.

-- Decrease price by 5% for books that haven't sold any copies in the past three months

UPDATE Books

SET Price = Price \* 0.95

WHERE BookID NOT IN (

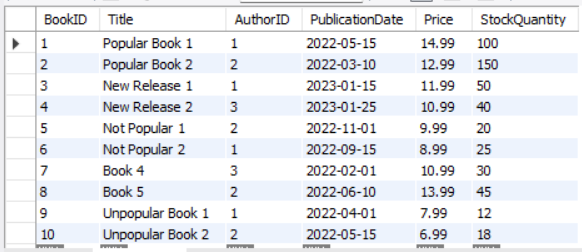
SELECT BookID

FROM Sales

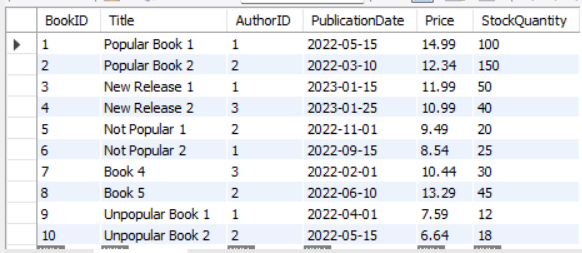
WHERE SaleDate >= '2022-11-01' AND SaleDate < '2023-02-01'

);

Before:



After:



Task 3: Managing Stock and Sales Records

a)Stock Adjustment:

1. After inserting sales records, update the Books table to reduce the stock quantity by the number of sales made. Assume an additional column StockQuantity exists in the Books table.

-- Update stock quantity after sales

UPDATE Books b

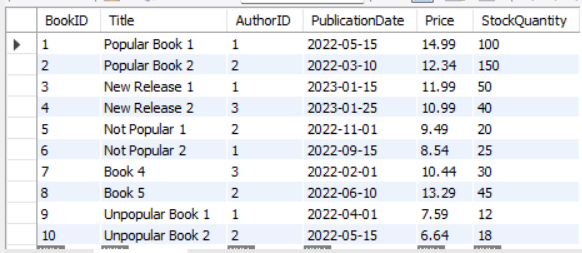
JOIN Sales s

ON b.BookID = s.BookID

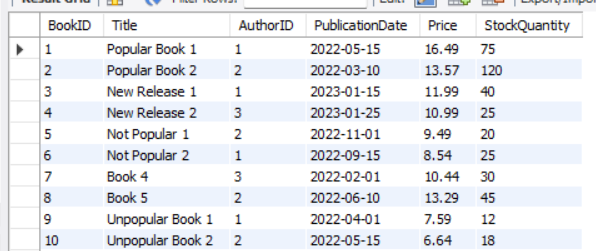
AND s.SaleDate >= '2023-01-01' AND s.SaleDate < '2023-02-01'

SET b.StockQuantity = b.StockQuantity - s.Quantity;

Before:



After:



b) Sales Record Cleanup:

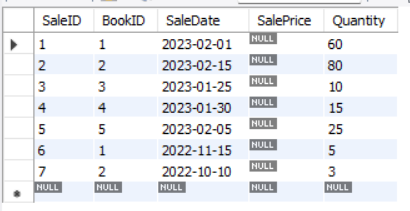
1. Delete sales records older than one year, as these are archived externally and no longer need to be stored in the database.

-- Delete sales records older than one year

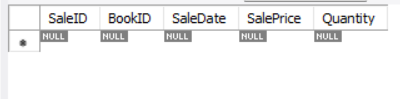
DELETE FROM Sales

WHERE SaleDate < DATE\_SUB(CURDATE(), INTERVAL 1 YEAR);

Before:



After:

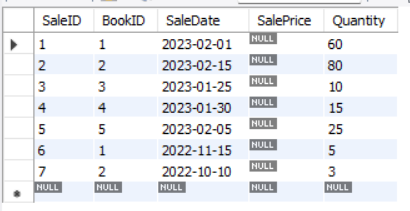


2. Use TRUNCATE TABLE to reset the Sales table at the beginning of a new fiscal year, ensuring to explain the implications of this action compared to using DELETE.

-- Reset Sales table at the beginning of a new fiscal year

TRUNCATE TABLE Sales;

Before:



After:

