RDBMS and SQL Assignments

Assignment 1: Analyze a given business scenario and create an ER diagram that includes entities, relationships, attributes, and cardinality. Ensure that the diagram reflects proper normalization up to the third normal form.

Identify Entities and Attributes:

Start by brainstorming the main objects or concepts that hold relevant information for your business. These become your entities.

For each entity, list the descriptive characteristics or properties you want to store. These are the attributes.

Example Scenario (Library Management System):

Entities:

Book

Author

Borrower

Attributes:

Book: ISBN, Title, Publication Year, Genre

Author: Author ID (primary key), Name, Nationality

Borrower: Borrower ID (primary key), Name, Contact Information

2. Define Relationships:

Consider how entities interact with each other. A relationship represents an association between two or more entities.

Relationships can be one-to-one (1:1), one-to-many (1:M), or many-to-many (M:N).

Example Scenario Relationships:

A Book can be written by one Author (1:M).

An Author can write many Books (M:1).

A Borrower can borrow many Books (M:N).

A Book can be borrowed by many Borrowers (M:N).

3. Normalize the ER Diagram:

Normalization is a process to minimize data redundancy and improve data integrity in a database. There are three main normal forms (1NF, 2NF, and 3NF) with increasing levels of normalization.

1NF (First Normal Form): Eliminates repeating groups within an entity.

2NF (Second Normal Form): Ensures no partial dependencies on the primary key.

3NF (Third Normal Form): Eliminates transitive dependencies on the primary key.

Normalization Steps for the Library Example:

1NF: We already have 1NF as there are no repeating groups.

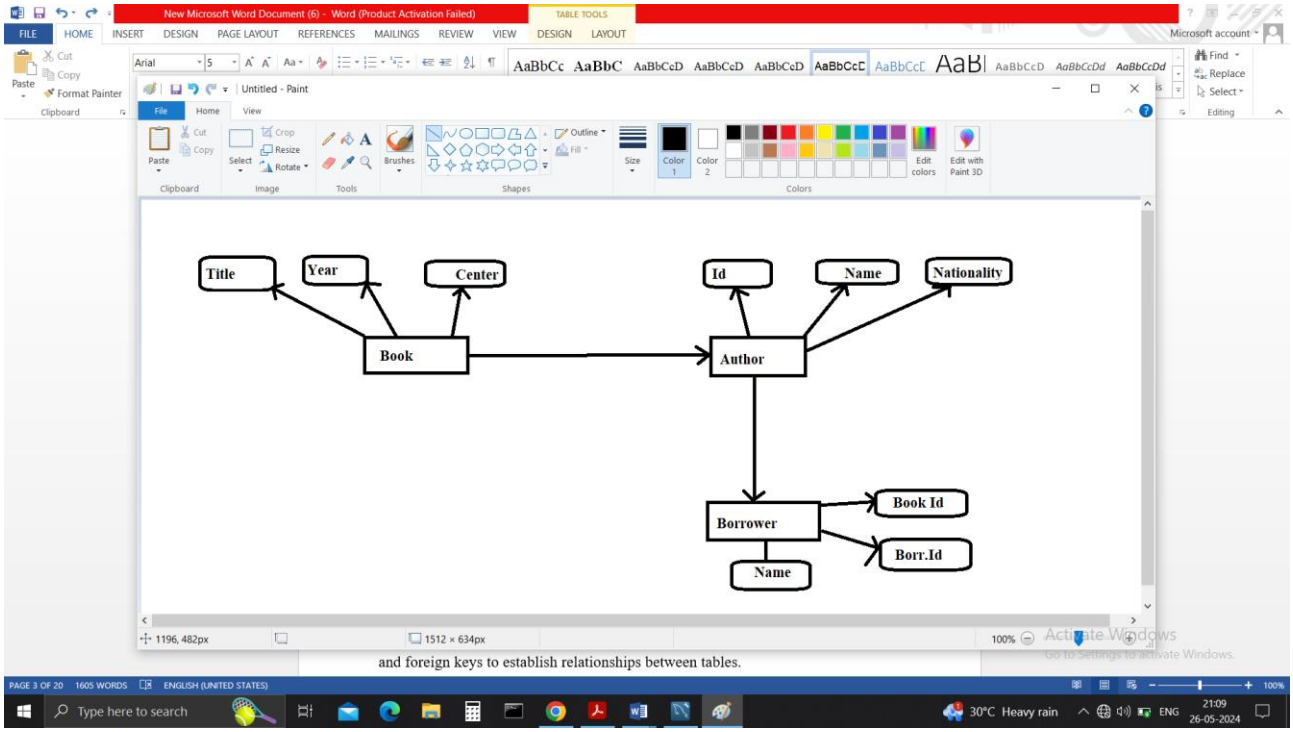
2NF: No partial dependencies exist based on primary keys (Author ID and Borrower ID).

3NF: The Borrower entity might have a transitive dependency on Book through the Author entity. To address this, we can create a separate entity Book_Borrower to link Book and Borrower with their own primary key and eliminate the dependency.

4. Create the ER Diagram:

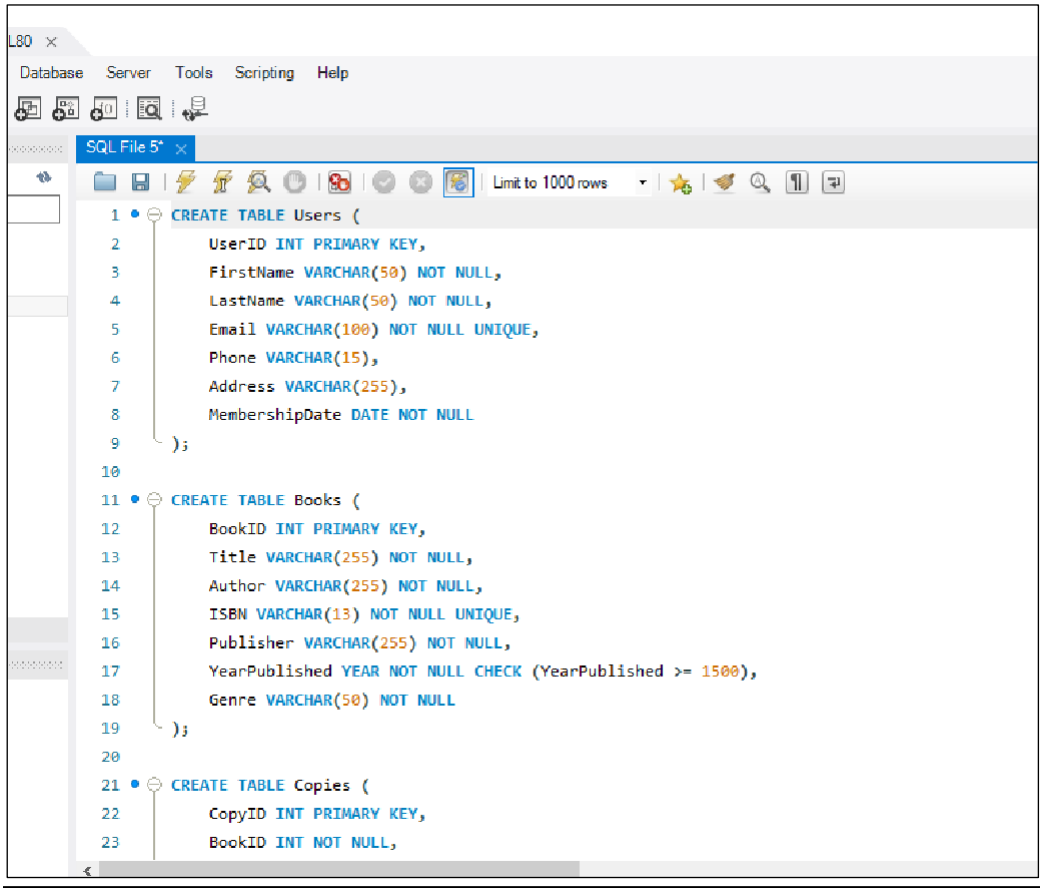
Use standard ERD symbols: Rectangles for entities, diamonds for relationships, ovals for attributes.

Label entities, attributes, and cardinalities (1:1, 1:M, M:N) on the connecting lines between entities and relationships.

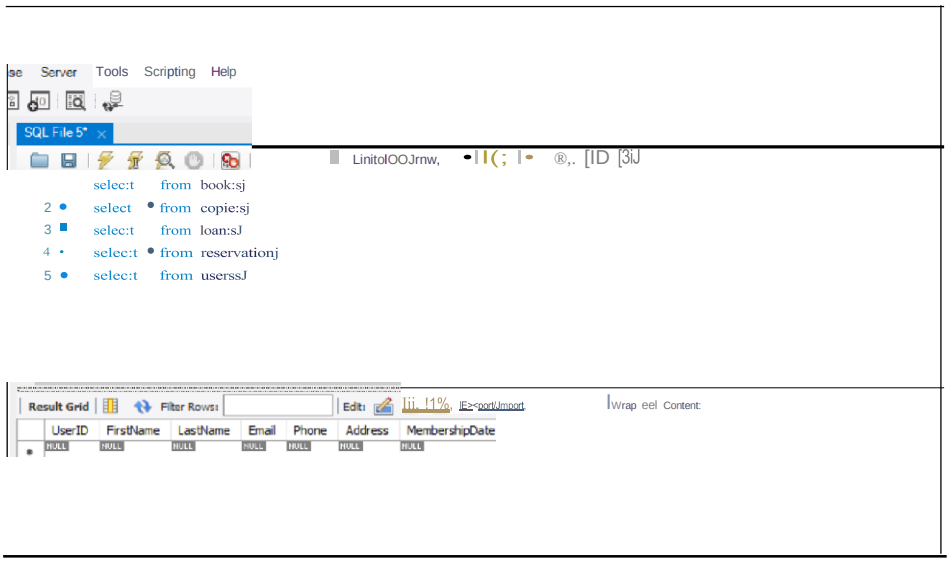


Assignment 2: Design a database schema for a library system, including tables, fields, and constraints like NOT NULL, UNIQUE, and CHECK. Include primary and foreign keys to establish relationships between tables.

Ans:




se Server Tools Seripting Help
SQL File 5*
UmItto 1000rows
26 FOREIGN KEY (BookID) REFERENCES Books (BookID) ON DELETE (ASCADE
27);
28
29 "7 CREATE TABLE loans (
30 LoanID HIT PRIAARY KEY,
31 UserID INT NOT NULL,
32 CopyID INT NOT NULL,
33 loanDatc DATE NOT NULL,
34 DueDate DATE NOT NULL,
35 ReturnDate DATE,
36 FOREIGN KEY (UserID) REFERENCES Users (User ID) ON DELETE (ASCADE,
37 FOREIGN KEY (CopyID) REFERENCES Copies (CopyID) ON DELETE (ASCADE
38);
39
40 CREATE TABLE Reservations (
41 Re:servationID INT PRIMARY KEY,
42 UserID INT NOT NULL,
43 BookID INT NOT NULL,
44 Rc:servationDatc DATE NOT NULL,
45 Status VARCHAR(20) NOT NULL CHECK (Status IN ('Pending', 'Cancelled', 'Completed')),
46 FOREIGN KEY (User ID) REFERENCES Users (User ID) ON DELETE CASCADE,
47 FOREIGN KEY (BookID) REFERENCES Books (BookID) ON DELETE (ASCADE
48);


se Server Tools Scripting Help
SQL File 5*
LinitolOOJrnw,
selec:t from book:sj
2 select from copie:sj
3 selec:t from loan:sJ
4 selec:t from reservationj
5 selec:t from userssJ
Result Grid
Filter Rows:
Edit:
Wrap eel Content:
UserID
FirstName
LastName
Email
Phone
Address
MembershipDate
NULL

Assignment 3: Explain the ACID properties of a transaction in your own words. Write SQL statements to simulate a transaction that includes locking and demonstrate different isolation levels to show concurrency control.

Ans:

ACID Properties Explained

Atomicity: This property ensures that a series of database operations within a transaction are treated as a single unit. Either all operations are successfully executed, or none are. If any part of the transaction fails, the entire transaction is rolled back.

Consistency: Consistency ensures that a transaction brings the database from one valid state to another valid state, maintaining the database's predefined rules, such as constraints, cascades, and triggers. After the transaction, all data integrity constraints are still intact.

Isolation: Isolation ensures that transactions are executed independently of each other. Intermediate states of a transaction are invisible to other transactions until the transaction is complete, preventing potential conflicts.

Durability: Durability guarantees that once a transaction has been committed, it will remain in the system permanently, even in the event of a system failure. The changes are recorded in non-volatile memory.

SQL Statements for Transaction with Locking and Isolation Levels

Let's consider a library system where we want to simulate a transaction involving borrowing a book.

CREATE TABLE Users (UserID INT PRIMARY KEY, FirstName VARCHAR(50) NOT NULL);

- CREATE TABLE Books (BookID INT PRIMARY KEY, Title VARCHAR(255) NOT NULL, AvailableCopies INT NOT NULL);

- CREATE TABLE Loans (LoanID INT PRIMARY KEY, UserID INT NOT NULL,
  BookID INT NOT NULL, LoanDate DATE NOT NULL,
  FOREIGN KEY (UserID) REFERENCES Users(UserID),
  FOREIGN KEY (BookID) REFERENCES Books(BookID));

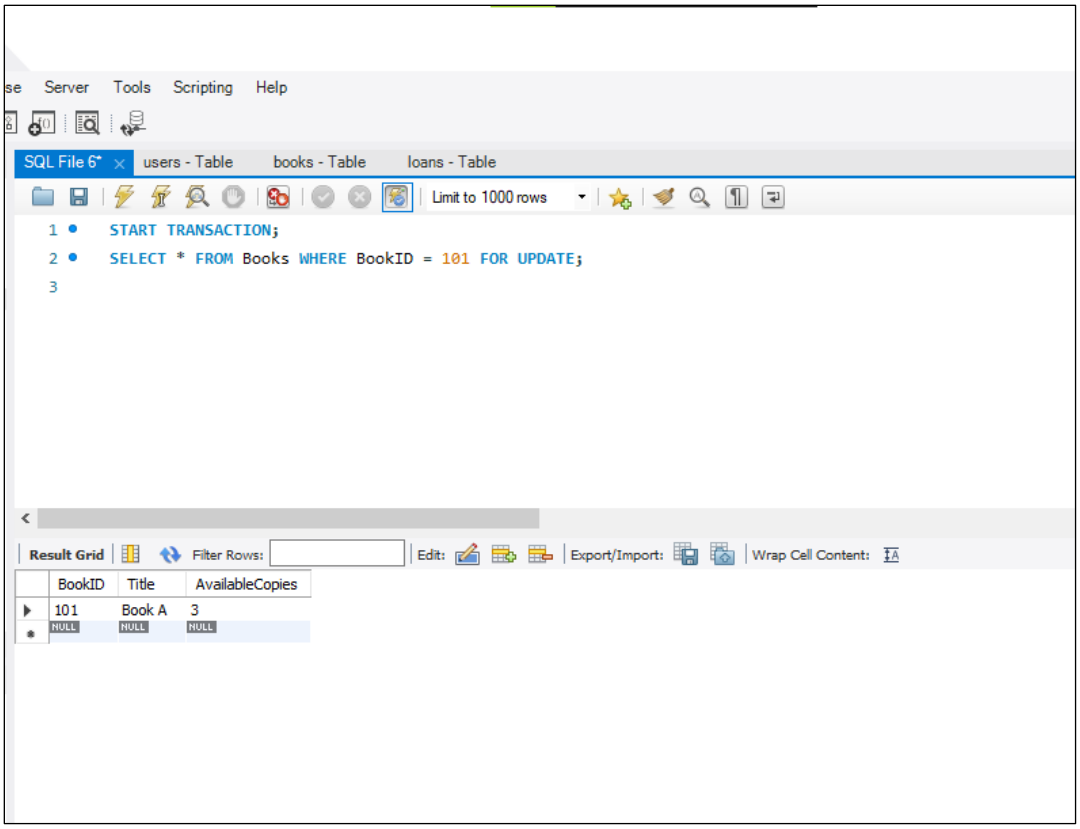
- INSERT INTO Users (UserID, FirstName) VALUES (1, 'John'), (2, 'Jane');

- INSERT INTO Books (BookID, Title, AvailableCopies) VALUES (101, 'Book A', 3), (102, 'Book B', 2);

  Transaction Example
  Borrowing a book involves decreasing the AvailableCopies and inserting a new record into the Loans table.

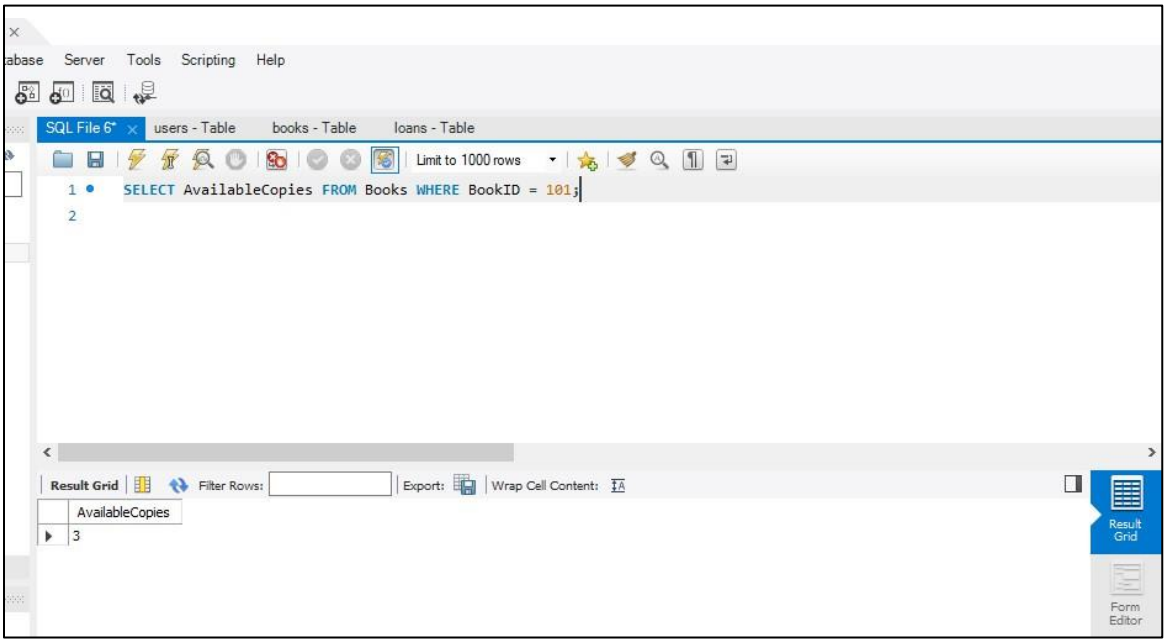
- START TRANSACTION;
- Lock the book row to prevent other transactions from modifying it simultaneously

- SELECT * FROM Books WHERE BookID = 101 FOR UPDATE;



Check if the book is available

SELECT AvailableCopies FROM Books WHERE BookID = 101;



Decrease the number of available copies

UPDATE Books SET AvailableCopies = AvailableCopies – 1 WHERE BookID = 101;

Insert a new loan record

INSERT INTO Loans (LoanID, UserID, BookID, LoanDate) VALUES (1, 1, 101, CURDATE());

COMMIT;

Isolation Levels and Concurrency Control

Different isolation levels can be set to demonstrate concurrency control. Here's how you can set and demonstrate each isolation level:

Read Uncommitted: Allows dirty reads, where one transaction can see uncommitted changes made by another transaction.

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

START TRANSACTION;

SELECT * FROM Books WHERE BookID = 101;

Changes from other transactions are visible even if not committed

Read Committed: Prevents dirty reads. Only committed changes are visible.

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

START TRANSACTION;

SELECT * FROM Books WHERE BookID = 101;

Changes from other transactions are visible only if committed

Repeatable Read: Ensures that if a transaction reads a row, it will see the same data if it reads it again within the same transaction, preventing non-repeatable reads.

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

START TRANSACTION;

SELECT * FROM Books WHERE BookID = 101;

Subsequent reads will see the same data, even if other transactions modify it

Serializable: The highest isolation level, ensuring complete isolation from other transactions. It prevents phantom reads and guarantees that the transaction operates in a serializable manner.

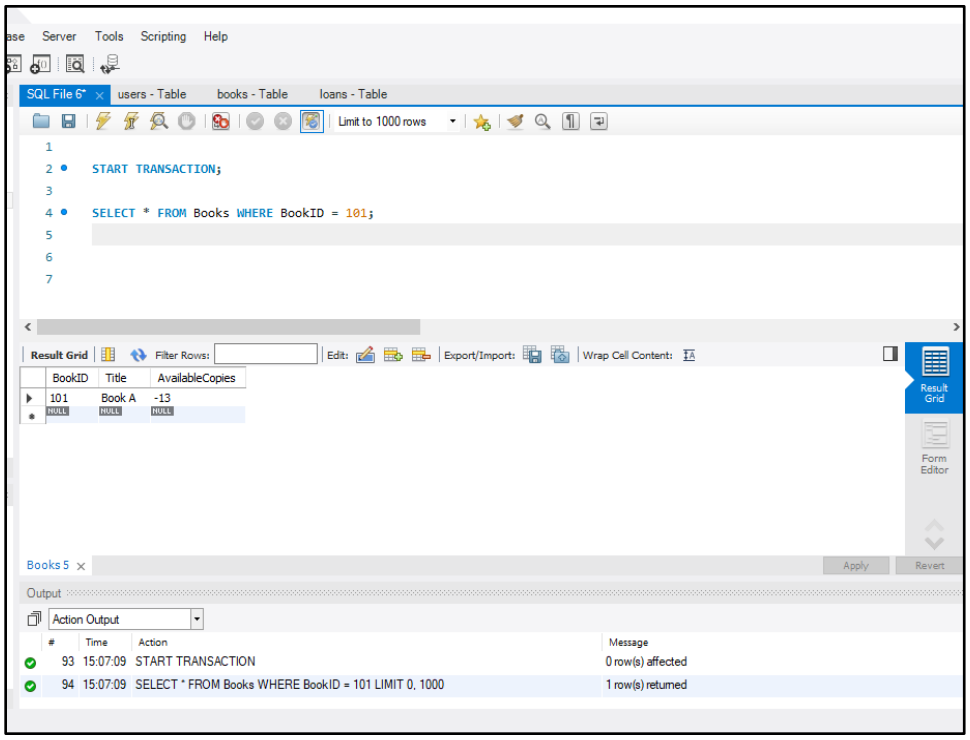
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

START TRANSACTION;

SELECT * FROM Books WHERE BookID = 101;

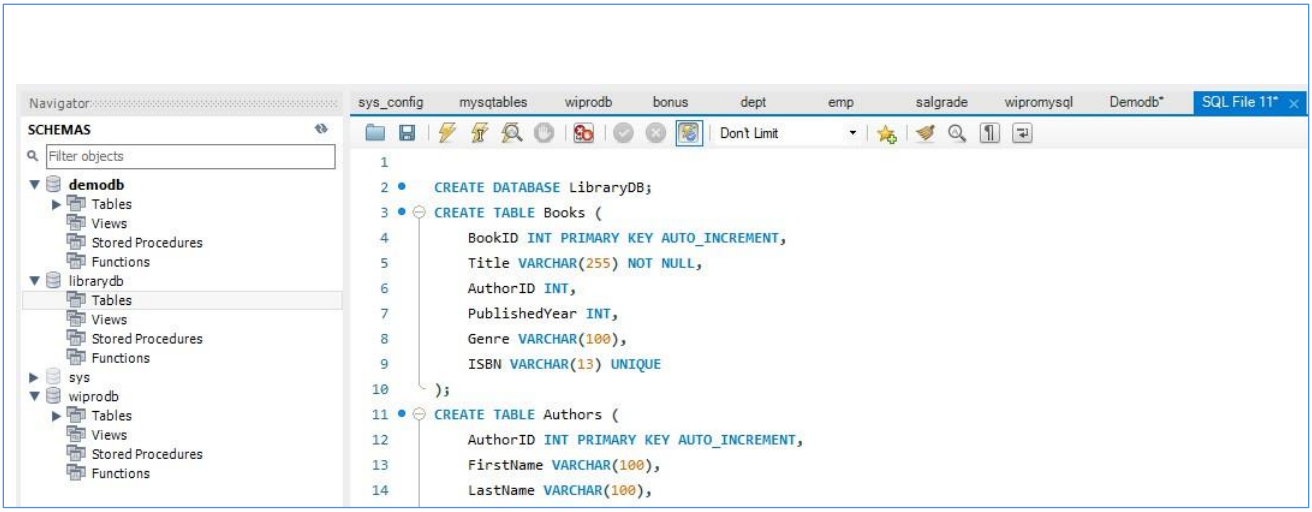
No other transactions can insert, update, or delete rows that would affect the result

By setting different isolation levels, you can control the level of concurrency and consistency in your transactions. This is crucial for ensuring that your transactions meet the ACID properties and maintain the integrity and reliability of the database.



Assignment 4: Write SQL statements to CREATE a new database and tables that reflect the library schema you designed earlier. Use ALTER statements to modify the table structures and DROP statements to remove a redundant table.

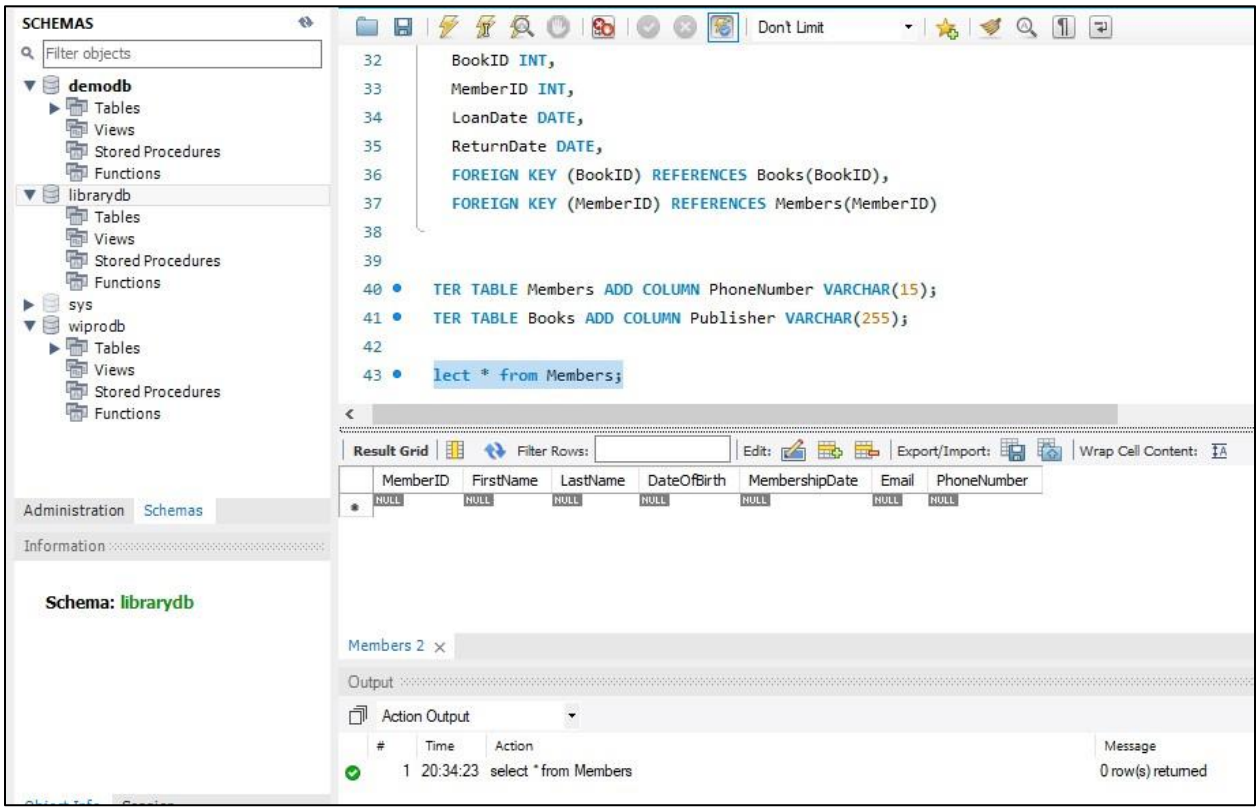
Ans: 1: Create a New Database CREATE DATABASE LibraryDB;

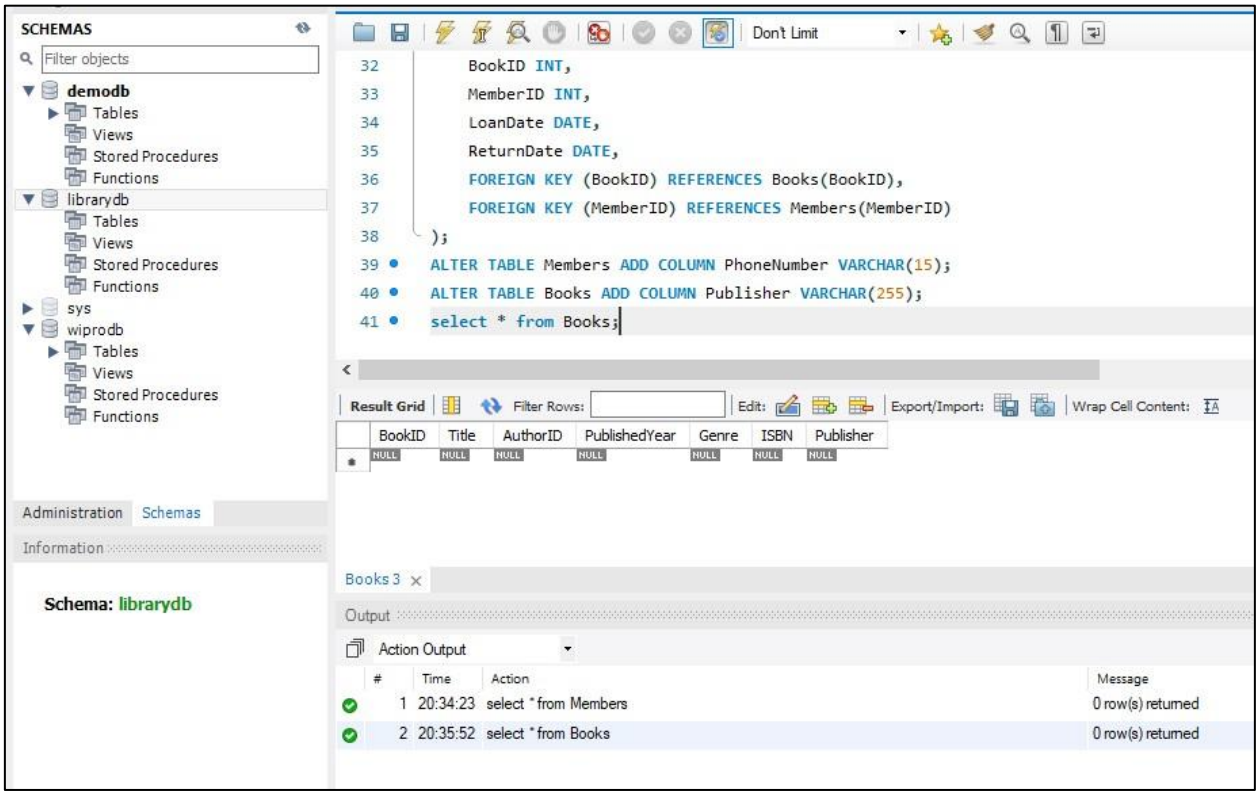


Use ALTER Statements to Modify Table Structures

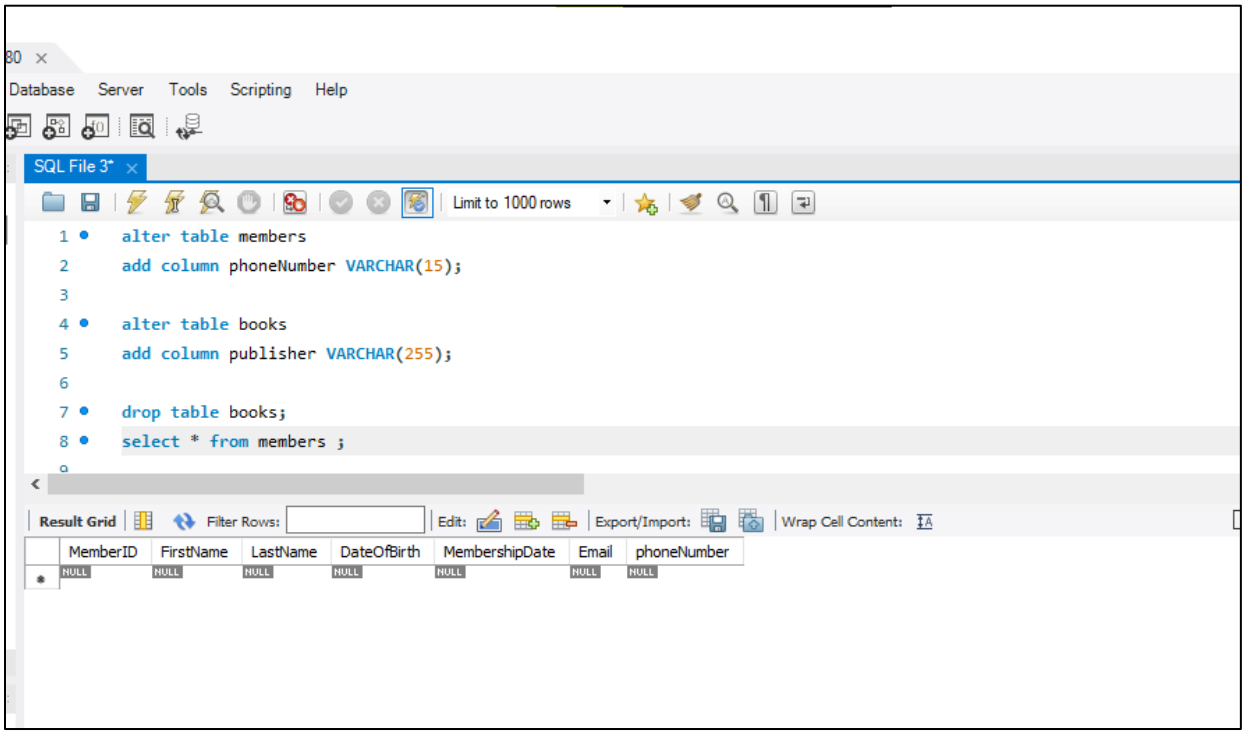
ALTER TABLE Members ADD COLUMN PhoneNumber VARCHAR(15);

ALTER TABLE Books ADD COLUMN Publisher VARCHAR(255);





DROP a Redundant Table DROP TABLE Books;

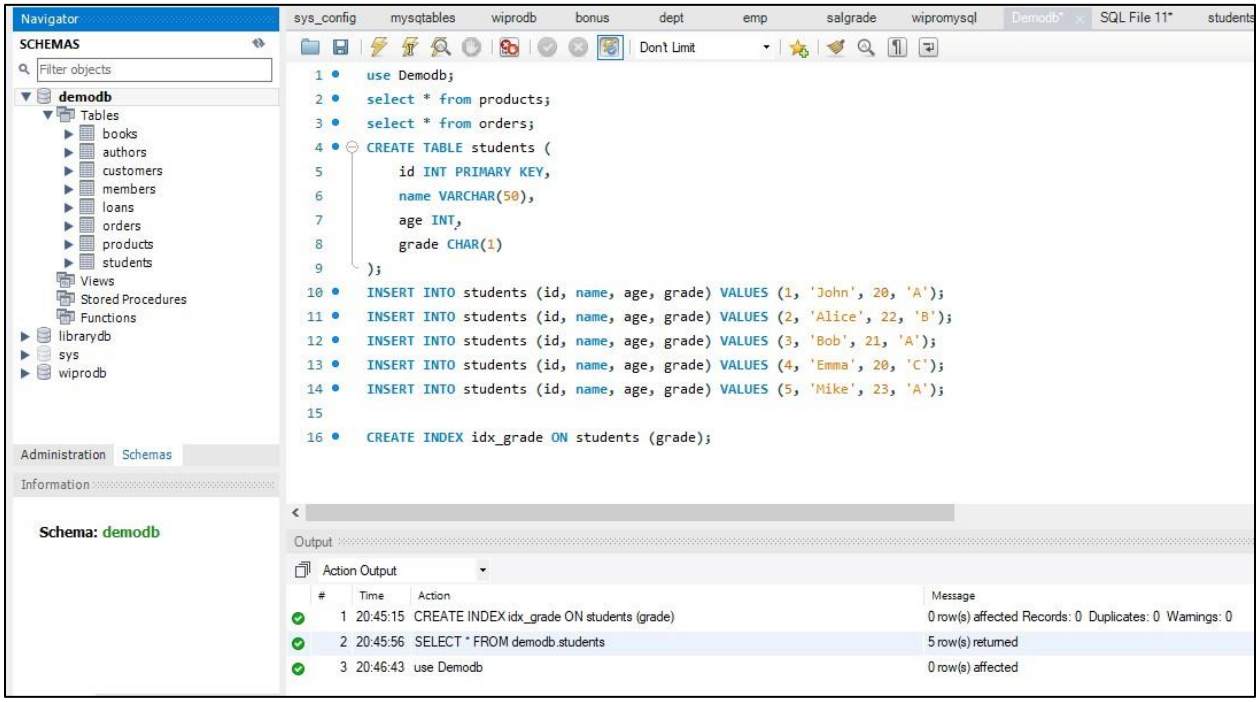


Assignment 5: Demonstrate the creation of an index on a table and discuss how it improves query performance. Use a DROP INDEX statement to remove the index and analyze the impact on query execution.

Ans:

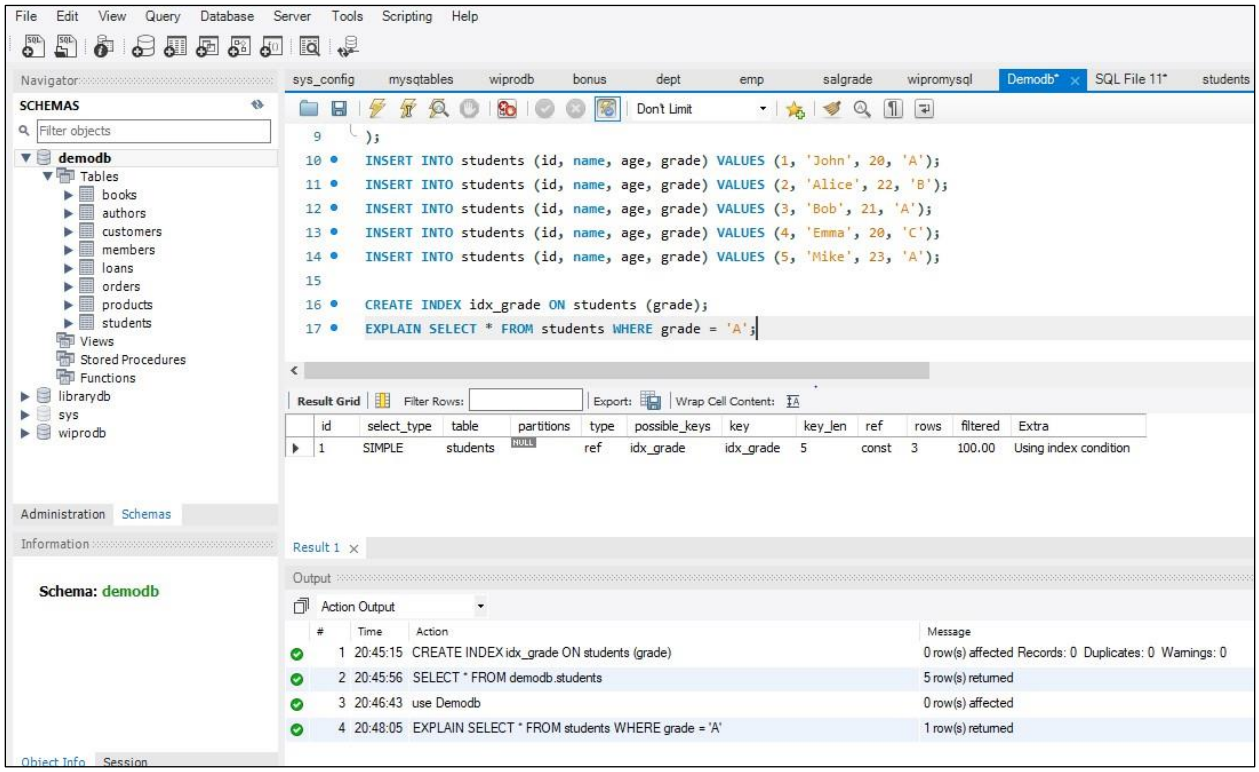
Create an index on the 'grade' column

CREATE INDEX idx_grade ON students (grade);



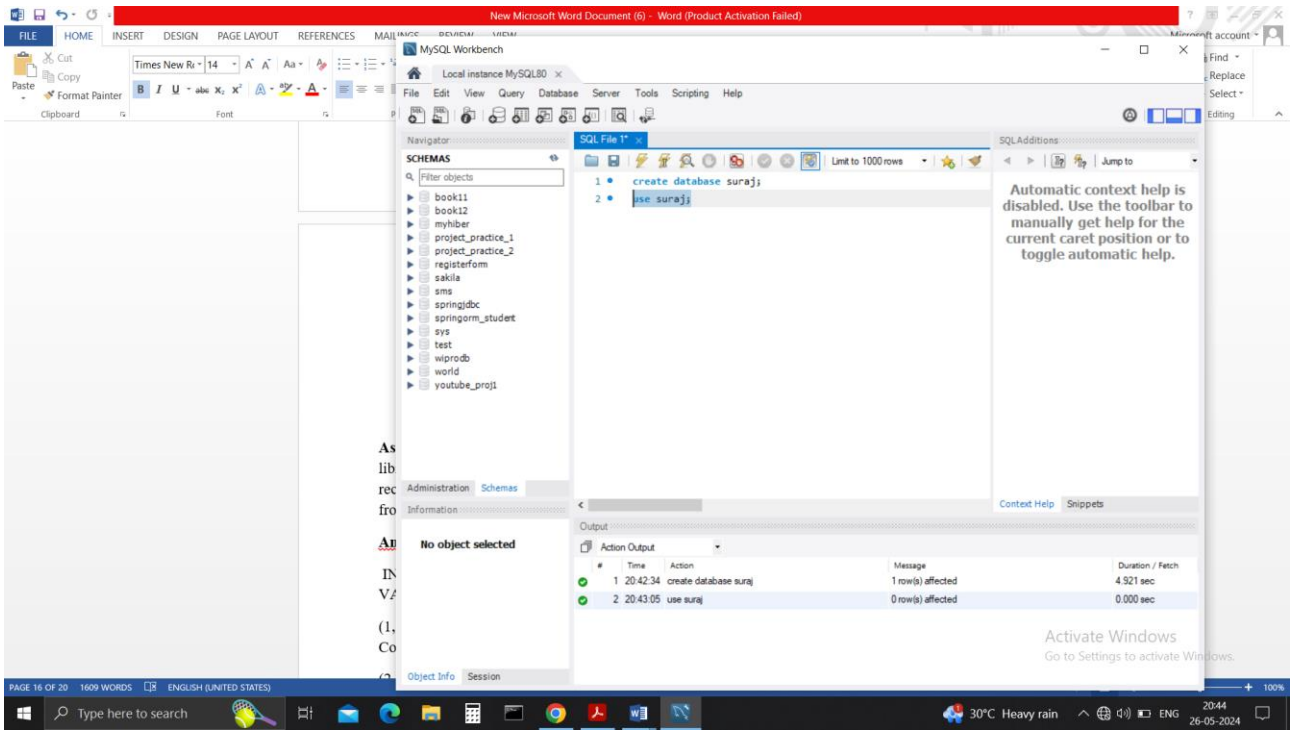
Query without index

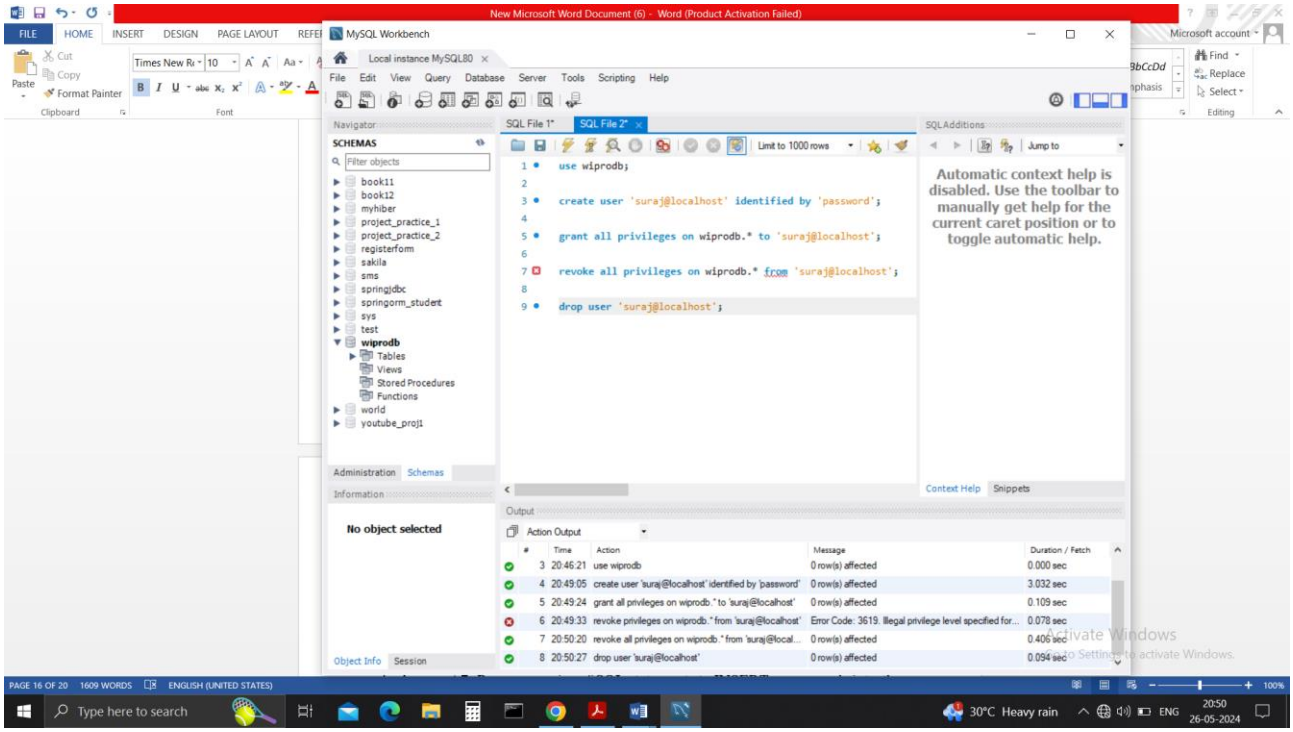
EXPLAIN SELECT * FROM students WHERE grade = 'A';



Assignment 6: Create a new database user with specific privileges using the CREATE USER and GRANT commands. Then, write a script to REVOKE certain privileges and DROP the user.

Ans:





Assignment 7: Prepare a series of SQL statements to INSERT new records into the library tables, UPDATE existing records with new information, and DELETE records based on specific criteria. Include BULK INSERT operations to load data from an external source.

Ans:

INSERT INTO books (bookid, title, authorid, publishedyear, isbn, publisher) VALUES

(1, 'To Kill a Mockingbird', 101, 1960, '978-0-06-112008-4', 'J.B. Lippincott & Co.'),

(2, '1984', 102, 1949, '978-0-452-28423-4', 'Secker & Warburg'),

(3, 'The Great Gatsby', 103, 1925, '978-0-7432-7356-5', 'Charles Scribner\'s Sons');

Update book information

UPDATE books

SET title = 'To Kill a Mockingbird (Updated)', authorid = 104, publishedyear = 1961, isbn = '978-0-06-112008-5', publisher = 'J.B. Lippincott & Co. (Updated)'

WHERE bookid = 1;

UPDATE books

SET title = '1984 (Updated)', authorid = 105, publishedyear = 1950, isbn = '978-0-452-28423-5', publisher = 'Secker & Warburg (Updated)'

WHERE bookid = 2;

UPDATE books

SET title = 'The Great Gatsby (Updated)', authorid = 106, publishedyear = 1926, isbn = '978-0-7432-7356-6', publisher = 'Charles Scribner\'s Sons (Updated)'

WHERE bookid = 3;

BULK INSERT Books

```
FROM 'C:\path\to\books.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2 -- If the first row contains headers
);
```

Bulk insert data into the Members table from a CSV file

```
BULK INSERT Members
FROM
'C:\path\to\members.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2 -- If the first row contains headers
);
```

Bulk insert data into the Loans table from a CSV file

```
BULK INSERT Loans
FROM
'C:\path\to\loans.csv'
WITH (
    FIELDTERMINATOR = ',',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2 -- If the first row contains headers
);
```