# RDBMS FUNDAMENTALS ASSIGNMENT

## 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.

**Assignment 1: ER Diagram for a Business Scenario**

**Business Scenario: Library Management System**

**Entities and Attributes:**

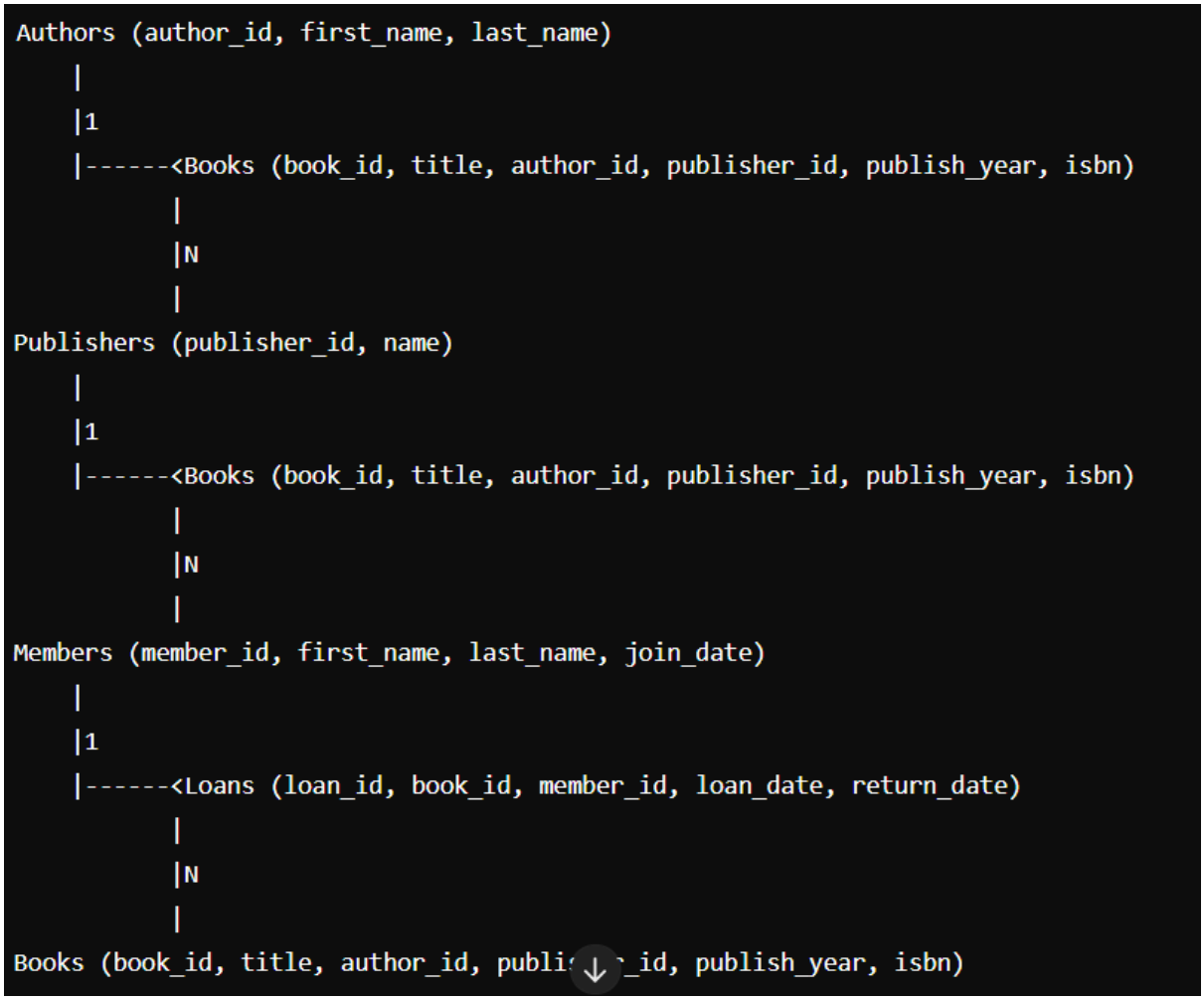
1. **Books**
    - book_id (PK)
    - title
    - author_id (FK)
    - publisher_id (FK)
    - publish_year
    - isbn (UNIQUE)
2. **Authors**
    - author_id (PK)
    - first_name
    - last_name
3. **Publishers**
    - publisher_id (PK)
    - name
4. **Members**
    - member_id (PK)
    - first_name
    - last_name
    - join_date
5. **Loans**
    - loan_id (PK)

- book_id (FK)
- member_id (FK)
- loan_date
- return_date

**Relationships and Cardinality:**

- One-to-Many between Authors and Books (one author can write many books).
- One-to-Many between Publishers and Books (one publisher can publish many books).
- One-to-Many between Members and Loans (one member can have many loans).
- One-to-Many between Books and Loans (one book can be loaned many times).

**ER Diagram:**



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.

```sql
CREATE TABLE Books (
    book_id INT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    author_id INT NOT NULL,
    publisher_id INT NOT NULL,
    publish_year YEAR,
    isbn VARCHAR(13) UNIQUE NOT NULL,
    FOREIGN KEY (author_id) REFERENCES Authors(author_id),
    FOREIGN KEY (publisher_id) REFERENCES Publishers(publisher_id)
);
-- Authors table
CREATE TABLE Authors (
    author_id INT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL
);
-- Publishers table
CREATE TABLE Publishers (
    publisher_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);
-- Members table
CREATE TABLE Members (
    member_id INT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    join_date DATE NOT NULL
);
```

```sql
-- Loans table
CREATE TABLE Loans (
    loan_id INT PRIMARY KEY,
    book_id INT NOT NULL,
    member_id INT NOT NULL,
    loan_date DATE NOT NULL,
    return_date DATE,
    FOREIGN KEY (book_id) REFERENCES Books(book_id),
    FOREIGN KEY (member_id) REFERENCES Members(member_id)
);
```

## 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.

**ACID Properties:**

- **Atomicity**: Ensures that all operations within a transaction are completed successfully. If one part of the transaction fails, the entire transaction fails and the database state is left unchanged.
- **Consistency**: Ensures that a transaction brings the database from one valid state to another valid state, maintaining database rules.
- **Isolation**: Ensures that concurrently executing transactions do not affect each other's operations.
- **Durability**: Ensures that once a transaction is committed, it remains in the database even in case of a system failure.

**SQL Statements:**

```sql
-- Simulate a transaction with locking and different isolation levels

-- Begin transaction
start TRANSACTION;

-- Set isolation level to Serializable (highest isolation level)
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

-- Insert a new book
INSERT INTO Books (book_id, title, author_id, publisher_id, publish_year, isbn)
VALUES (1, 'Book Title', 1, 1, 2024, '1234567890123');

-- Update an existing book
UPDATE Books
SET title = 'Updated Book Title'
WHERE book_id = 1;

-- Delete a book
DELETE FROM Books
WHERE book_id = 1;

-- Commit the transaction
COMMIT;
```

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.

```sql
-- Create a new database
CREATE DATABASE LibraryDB;

-- Use the newly created database
USE LibraryDB;

-- Create tables
CREATE TABLE Books (
    book_id INT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    author_id INT NOT NULL,
    publisher_id INT NOT NULL,
    publish_year YEAR,
    isbn VARCHAR(13) UNIQUE NOT NULL,
    FOREIGN KEY (author_id) REFERENCES Authors(author_id),
    FOREIGN KEY (publisher_id) REFERENCES Publishers(publisher_id)
);

CREATE TABLE Authors (
    author_id INT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL
);
```

```sql
CREATE TABLE Members (
    member_id INT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    join_date DATE NOT NULL
);

CREATE TABLE Loans (
    loan_id INT PRIMARY KEY,
    book_id INT NOT NULL,
    member_id INT NOT NULL,
    loan_date DATE NOT NULL,
    return_date DATE,
    FOREIGN KEY (book_id) REFERENCES Books(book_id),
    FOREIGN KEY (member_id) REFERENCES Members(member_id)
);

-- Alter table to add a new column
ALTER TABLE Books
ADD genre VARCHAR(50);

-- Drop a redundant table (example table: OldPublishers)
DROP TABLE OldPublishers;
```

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.

```sql
-- Create an index on the title column of the Books table
CREATE INDEX idx_title ON Books(title);

-- Analyze query performance before dropping the index
EXPLAIN SELECT * FROM Books WHERE title = 'Book Title';

-- Drop the index
DROP INDEX idx_title ON Books;

-- Analyze query performance after dropping the index
EXPLAIN SELECT * FROM Books WHERE title = 'Book Title';
```

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.

```sql
-- Create a new database user
CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'password';

-- Grant specific privileges to the user
GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost';

-- Revoke certain privileges from the user
REVOKE DELETE ON LibraryDB.* FROM 'library_user'@'localhost';

-- Drop the user
DROP USER 'library_user'@'localhost';
```

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.

```sql
-- Insert new records into the Books table
INSERT INTO Books (book_id, title, author_id, publisher_id, publish_year, isbn)
VALUES (2, 'New Book', 2, 2, 2024, '9876543210987');

-- Update existing records
UPDATE Books
SET title = 'Updated Book Title'
WHERE book_id = 2;

-- Delete records based on specific criteria
DELETE FROM Books
WHERE book_id = 2;

-- Bulk insert operations (example with a hypothetical external source)
-- This assumes you have a CSV file named 'books.csv' with appropriate data
LOAD DATA INFILE '/path/to/books.csv'
INTO TABLE Books
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
(book_id, title, author_id, publisher_id, publish_year, isbn);
```