**Conceptual ERD Model:**

- Correctly identified entities and relationships.

- Consider adding an entity for "Category" to further normalize the data.

**Logical ERD Model:**

- Well-defined attributes for each entity.

- Consider adding a "Status" attribute to the Enrollment entity (e.g., "Enrolled", "Completed", "InProgress").

**Physical ERD Model (Database Schema):**

- Proper use of primary keys, foreign keys, and data types.

- Consider indexing columns used in WHERE, JOIN, and ORDER BY clauses for improved performance.

**Additional Suggestions:**

1. Consider adding a "Rating" attribute to the Course entity to store user ratings.

2. Add a "Timestamp" attribute to the Completion entity to track completion time.

3. Create a separate table for "Questions" to store individual quiz questions.

4. Use ENUM data type for "Category" and "Status" attributes.

5. Use VARCHAR(255) for "Password" to accommodate hashed passwords.

**SQL Enhancements:**

1. Use CONSTRAINTS to enforce data integrity.

2. Create INDEXES on columns used in queries.

3. Consider using VIEWs to simplify complex queries.

**Next Steps:**

1. Implement the database schema.

2. Create CRUD (Create, Read, Update, Delete) operations for each entity.

3. Develop queries for reporting and analytics.

Would you like me to:

1. Review and refine the database schema?

2. Assist with CRUD operation implementation?

3. Develop queries for reporting and analytics?

4. Discuss database security and backup strategies?

**Conceptual ER Diagram:**

**Books:**

Book\_ID

Title

Author\_ID

Publisher

Publication\_Date

**Authors:**

Author\_ID

Name

Bio

**Members:**

Member\_ID

Name

Email

Phone

Address

**Staff :**

Staff\_ID

Name

Email

Phone

Role

**Borrow\_Records:**

Borrow\_ID

Book\_ID

Member\_ID

Borrow\_Date

Return\_Date

Fine

**Logical ER Diagram:**

CREATE TABLE Books (

Book\_ID INT PRIMARY KEY,

Title VARCHAR(100),

Author\_ID INT,

Publisher VARCHAR(50),

Publication\_Date DATE,

FOREIGN KEY (Author\_ID) REFERENCES Authors(Author\_ID)

);

CREATE TABLE Authors (

Author\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Bio VARCHAR(200)

);

CREATE TABLE Members (

Member\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Email VARCHAR(50),

Phone VARCHAR(20),

Address VARCHAR(100)

);

CREATE TABLE Staff (

Staff\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Email VARCHAR(50),

Phone VARCHAR(20),

Role VARCHAR(20)

);

CREATE TABLE Borrow\_Records (

Borrow\_ID INT PRIMARY KEY,

Book\_ID INT,

Member\_ID INT,

Borrow\_Date DATE,

Return\_Date DATE,

Fine DECIMAL(5, 2),

FOREIGN KEY (Book\_ID) REFERENCES Books(Book\_ID),

FOREIGN KEY (Member\_ID) REFERENCES Members(Member\_ID)

);

**Physical ER Diagram (SQL):**

CREATE TABLE Books (

Book\_ID INT PRIMARY KEY,

Title VARCHAR(100),

Author\_ID INT,

Publisher VARCHAR(50),

Publication\_Date DATE,

FOREIGN KEY (Author\_ID) REFERENCES Authors(Author\_ID)

);

CREATE TABLE Authors (

Author\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Bio VARCHAR(200)

);

CREATE TABLE Members (

Member\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Email VARCHAR(50),

Phone VARCHAR(20),

Address VARCHAR(100)

);

CREATE TABLE Staff (

Staff\_ID INT PRIMARY KEY,

Name VARCHAR(50),

Email VARCHAR(50),

Phone VARCHAR(20),

Role VARCHAR(20)

);

CREATE TABLE Borrow\_Records (

Borrow\_ID INT PRIMARY KEY,

Book\_ID INT,

Member\_ID INT,

Borrow\_Date DATE,

Return\_Date DATE,

Fine DECIMAL(5, 2),

FOREIGN KEY (Book\_ID) REFERENCES Books(Book\_ID),

FOREIGN KEY (Member\_ID) REFERENCES Members(Member\_ID)

);

Stored Procedures

CREATE PROCEDURE sp\_checkout\_book

@Book\_ID INT,

@Member\_ID INT,

@Borrow\_Date DATE,

@Return\_Date DATE

AS

BEGIN

INSERT INTO Borrow\_Records (Book\_ID, Member\_ID, Borrow\_Date, Return\_Date)

VALUES (@Book\_ID, @Member\_ID, @Borrow\_Date, @Return\_Date);

END;

CREATE PROCEDURE sp\_return\_book

@Borrow\_ID INT

AS

BEGIN

UPDATE Borrow\_Records

SET Return\_Date = GETDATE()

WHERE Borrow\_ID = @Borrow\_ID;

END;

CREATE PROCEDURE sp\_reserve\_book

@Book\_ID INT,

@Member\_ID INT

AS

BEGIN

INSERT INTO Borrow\_Records (Book\_ID, Member\_ID, Borrow\_Date, Return\_Date)

VALUES (@Book\_ID, @Member\_ID, NULL, NULL);

END;

Triggers

CREATE TRIGGER tr\_update\_book\_availability

ON Borrow\_Records

AFTER INSERT, UPDATE

AS

BEGIN

UPDATE Books

SET Availability = 'Not Available'

WHERE Book\_ID = INSERTED.Book\_ID;

END;

CREATE TRIGGER tr\_update\_late\_fees

ON Borrow\_Records

AFTER UPDATE

AS

BEGIN

UPDATE Borrow\_Records

SET Fine = (Return\_Date - Borrow\_Date) \* 0.10

WHERE Borrow\_ID = INSERTED.Borrow\_ID;

END;