Berat Can Satir 32531

Library and Event Management System (LEMS)

Project Description

The Library and Event Management System (LEMS) is designed to manage the daily operations of a university library and its on-site events. The system allows members to borrow and return books, and also register for various academic or social events organized in the library.

Each member has a unique ID, name, and contact information. The book entity stores essential bibliographic information such as title, author, ISBN, and the number of copies available.
The loan entity keeps track of which member borrowed which book, when it was borrowed, its due date, and whether it was returned.

In addition to book management, the library regularly hosts events such as seminars, book clubs, and guest talks. These events are recorded in the system with details such as date, time, and location. Members can register for these events through the registration entity, which connects members and events.

The system enforces several integrity constraints:

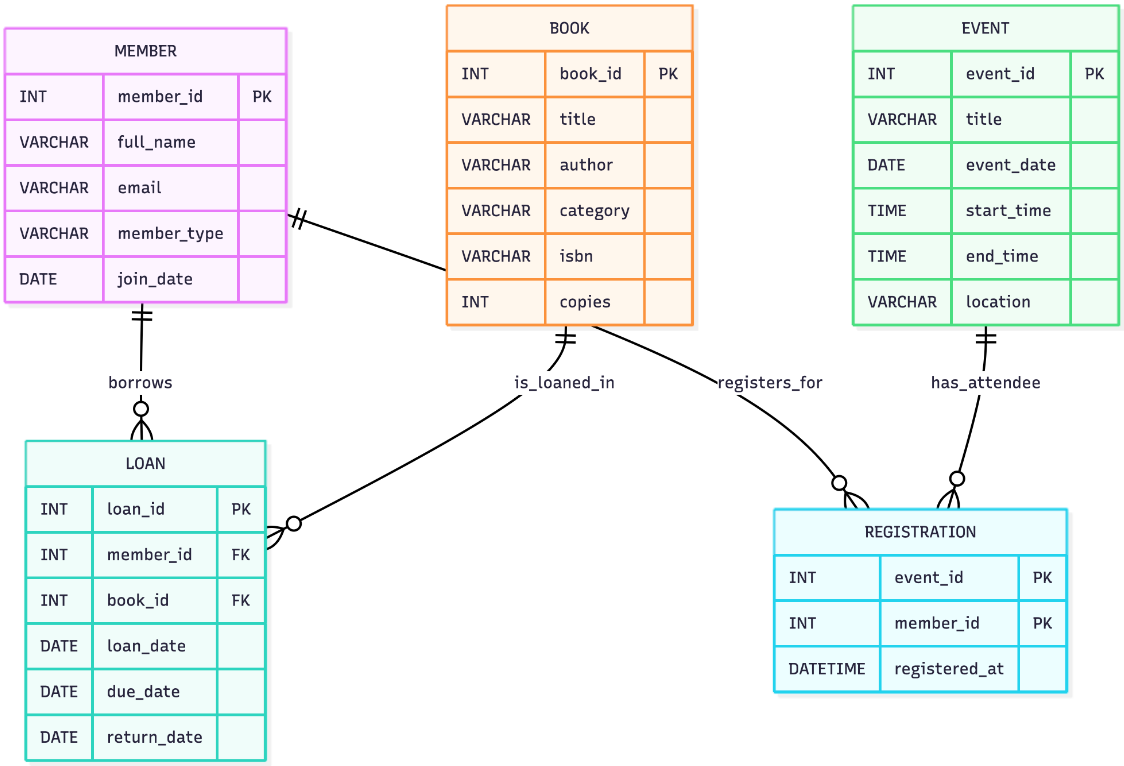
- Each book and member must have unique identifiers.
- ISBN and member email are unique.
- A member cannot register for the same event more than once.
- A loan record must refer to an existing member and book, and each loan must have a due date later than the loan date.

Entity–Relationship Diagram

The following figure represents the Entity–Relationship (ER) model of the Library and Event Management System (LEMS).

It shows all entities, attributes, primary and foreign keys, and the relationships between them.

- Entities: Member, Book, Event, Loan, Registration
- Primary Keys: member_id, book_id, event_id, loan_id, and (event_id, member_id) for Registration
- Relationships:
    - borrows — connects Member and Loan (1–N)
    - is_loaned_in — connects Loan and Book (1–N)
    - registers_for — connects Member and Registration (1–N)
    - has_attendee — connects Event and Registration (1–N)
- Participation: Loan and Registration have total participation in their connections (every loan or registration must reference valid entities).
- Constraints:
    - email and isbn are unique.
    - copies ≥ 0
    - start_time < end_time
    - due_date > loan_date



MEMBER		
INT	member_id	PK
VARCHAR	full_name	
VARCHAR	email	
VARCHAR	member_type	
DATE	join_date	

BOOK		
INT	book_id	PK
VARCHAR	title	
VARCHAR	author	
VARCHAR	category	
VARCHAR	isbn	
INT	copies	

EVENT		
INT	event_id	PK
VARCHAR	title	
DATE	event_date	
TIME	start_time	
TIME	end_time	
VARCHAR	location	

LOAN		
INT	loan_id	PK
INT	member_id	FK
INT	book_id	FK
DATE	loan_date	
DATE	due_date	
DATE	return_date	

REGISTRATION		
INT	event_id	PK
INT	member_id	PK
DATETIME	registered_at	

Relational Schema Description

The relational model of the Library and Event Management System (LEMS) is derived from the ER diagram shown in the previous section.

It includes five main tables: Member, Book, Event, Loan, and Registration.

Each table includes the corresponding primary keys, foreign keys, and constraints that ensure data integrity.

Table Definitions

Member

Stores information about library members.

- Primary Key: member_id
- Unique Attribute: email

Book

Stores details of all books in the library.

- Primary Key: book_id
- Unique Attribute: isbn
- Constraint: copies >= 0

Event

Represents events organized by the library.

- Primary Key: event_id
- Constraint: start_time < end_time

Loan

Tracks which member borrowed which book and when.

- Primary Key: loan_id
- Foreign Keys:
  - member_id → Member(member_id)
  - book_id → Book(book_id)
- Constraint: due_date > loan_date
- Unique Combination: (member_id, book_id, loan_date)

Registration

Represents the participation of members in events.

- Composite Primary Key: (event_id, member_id)
- Foreign Keys:
    - event_id → Event(event_id)
    - member_id → Member(member_id)

SQL Implementation

Below is the SQL code implementing the relational model:

```
CREATE DATABASE IF NOT EXISTS lems;
USE lems;

CREATE TABLE Member (
  member_id   INT PRIMARY KEY,
  full_name   VARCHAR(100) NOT NULL,
  email       VARCHAR(120) NOT NULL UNIQUE,
  member_type ENUM('student','external') NOT NULL,
  join_date   DATE NOT NULL
);

CREATE TABLE Book (
  book_id  INT PRIMARY KEY,
  title    VARCHAR(150) NOT NULL,
  author   VARCHAR(100) NOT NULL,
  category VARCHAR(50)  NOT NULL,
  isbn     VARCHAR(20)  NOT NULL UNIQUE,
  copies   INT NOT NULL,
  CHECK (copies >= 0)
);

CREATE TABLE Event (
  event_id   INT PRIMARY KEY,
  title      VARCHAR(150) NOT NULL,
  event_date DATE NOT NULL,
  start_time TIME NOT NULL,
  end_time   TIME NOT NULL,
  location   VARCHAR(80) NOT NULL,
  CHECK (start_time < end_time)
);
```

```
CREATE TABLE Loan (
  loan_id     INT PRIMARY KEY,
  member_id   INT NOT NULL,
  book_id     INT NOT NULL,
  loan_date   DATE NOT NULL,
  due_date    DATE NOT NULL,
  return_date DATE,
  FOREIGN KEY (member_id) REFERENCES Member(member_id),
  FOREIGN KEY (book_id) REFERENCES Book(book_id),
  UNIQUE (member_id, book_id, loan_date),
  CHECK (due_date > loan_date)
);

CREATE TABLE Registration (
  event_id      INT NOT NULL,
  member_id     INT NOT NULL,
  registered_at DATETIME NOT NULL,
  PRIMARY KEY (event_id, member_id),
  FOREIGN KEY (event_id) REFERENCES Event(event_id),
  FOREIGN KEY (member_id) REFERENCES Member(member_id)
);
```

Sample Data Inserts (10 rows per table)

Member data

```
INSERT INTO Member VALUES
(1001,'Alice Kaya','alice.kaya@uni.edu','student','2025-09-01'),
(1002,'Berk Yilmaz','berk.yilmaz@uni.edu','student','2025-09-03'),
(1003,'Ceren Aydin','ceren.aydin@uni.edu','student','2025-09-05'),
(1004,'Deniz Arslan','deniz.arslan@uni.edu','student','2025-09-06'),
(1005,'Efe Demir','efe.demir@uni.edu','student','2025-09-07'),
(1006,'Fatma Soylu','fatma.soylu@mail.com','external','2025-09-10'),
(1007,'Gokhan Cetin','gokhan.cetin@mail.com','external','2025-09-12'),
(1008,'Hale Ucar','hale.ucar@uni.edu','student','2025-09-15'),
(1009,'Ipek Korkmaz','ipek.korkmaz@mail.com','external','2025-09-18'),
(1010,'Kerem Oz','kerem.oz@uni.edu','student','2025-09-20');
```

Book data

```
INSERT INTO Book VALUES
(2001,'Database System Concepts','Silberschatz','CS','97812600818',5),
(2002,'Introduction to Algorithms','Cormen','CS','97802620428',3),
(2003,'Clean Code','Robert C. Martin','CS','97801323508',4),
(2004,'The Pragmatic Programmer','Andrew Hunt','CS','97802016162',2),
(2005,'Design Patterns','Gamma et al.','CS','97802016336',3),
(2006,'Deep Work','Cal Newport','Productivity','97814555866',2),
(2007,'Atomic Habits','James Clear','Productivity','97807352112',6),
(2008,'Sapiens','Yuval Noah Harari','History','97800623161',4),
(2009,'1984','George Orwell','Fiction','97804515249',5),
(2010,'To Kill a Mockingbird','Harper Lee','Fiction','97800611200',3);
```

Event data

```
INSERT INTO Event VALUES
(3001,'Author Talk: Clean Code','2025-10-25','14:00','15:30','FENS G077'),
(3002,'Research Skills Workshop','2025-10-28','10:00','12:00','FMAN G071'),
(3003,'AI Ethics Panel','2025-11-02','16:00','17:30','FASS Auditorium'),
(3004,'Data Modeling 101','2025-11-05','11:00','12:30','FENS G077'),
(3005,'Career Talk: Tech CVs','2025-11-08','09:30','10:45','UC Cinema'),
(3006,'Book Club: Sapiens','2025-11-12','13:00','14:30','Library Room 1'),
(3007,'Query Optimization Clinic','2025-11-15','10:00','11:30','FENS G050'),
(3008,'Time Management Tips','2025-11-18','15:00','16:00','FMAN G071'),
(3009,'Open Source Meetup','2025-11-20','18:00','19:30','FASS Hall'),
(3010,'Cybersecurity Basics','2025-11-22','10:00','11:30','FENS G077');
```

Loan data

```
INSERT INTO Loan VALUES
(4001,1001,2001,'2025-10-01','2025-10-15',NULL),
(4002,1002,2003,'2025-10-02','2025-10-16','2025-10-14'),
(4003,1003,2002,'2025-10-03','2025-10-17',NULL),
(4004,1004,2007,'2025-10-04','2025-10-18','2025-10-10'),
(4005,1005,2005,'2025-10-05','2025-10-19',NULL),
(4006,1006,2008,'2025-10-06','2025-10-20',NULL),
(4007,1007,2009,'2025-10-07','2025-10-21',NULL),
(4008,1008,2010,'2025-10-08','2025-10-22','2025-10-20'),
(4009,1009,2004,'2025-10-09','2025-10-23',NULL),
(4010,1010,2006,'2025-10-10','2025-10-24',NULL);
```

Registration data

```
INSERT INTO Registration VALUES
(3001,1001,'2025-10-20 09:10:00'),
(3002,1002,'2025-10-21 14:05:00'),
(3003,1003,'2025-10-22 10:30:00'),
(3004,1004,'2025-10-22 11:15:00'),
(3005,1005,'2025-10-22 12:20:00'),
(3006,1006,'2025-10-22 12:45:00'),
(3007,1007,'2025-10-22 13:05:00'),
(3008,1008,'2025-10-22 13:25:00'),
(3009,1009,'2025-10-22 13:40:00'),
(3010,1010,'2025-10-22 14:00:00');
```