**Assignment#1**

**Advanced Library Management System**

**1. Extended ER Diagram (EERD)**

**Entities and Attributes:**

**🔷BOOK**

* **BookID** *(PK)*
* Title
* ISBN
* Edition (Multivalued Attribute)
* Authors (Multivalued Attribute)
* Publisher
* YearPublished

**🔷MEMBER**

* **MemberID** *(PK)*
* Name
* Email
* Phone
* Address
* LoginID *(FK to LOGIN)*

**🔷STAFF**

* **StaffID** *(PK)*
* Name
* Role *(e.g., Admin, Librarian)*
* LoginID *(FK to LOGIN)*

**🔷LOGIN**

* **LoginID** *(PK)*
* Username
* Password
* Role *(Enum: Admin, Librarian, Member)*

**🔷BORROWING (Weak Entity)**

* **BorrowID** *(PK)*
* **BookID** *(FK)*
* **MemberID** *(FK)*
* BorrowDate
* DueDate
* ReturnDate
* FineAmount
* ManagedBy *(FK to StaffID)*

**🔷RESERVATION**

* **ReservationID** *(PK)*
* **BookID** *(FK)*
* **MemberID** *(FK)*
* ReservationDate
* Status *(Waiting, Fulfilled, Cancelled)*

**🔷BOOK\_AUTHOR (Associative Entity)**

* **BookID** *(FK)*
* **AuthorName**

**🔷BOOK\_EDITION (Associative Entity)**

* **BookID** *(FK)*
* EditionNumber
* Year

**2. Normalized Relational Schema (Upto 3NF)**

**BOOK (BookID, Title, ISBN, Publisher, YearPublished)**

* PK: BookID

**BOOK\_AUTHOR (BookID, AuthorName)**

* PK: (BookID, AuthorName)
* FK: BookID → BOOK(BookID)

**BOOK\_EDITION (BookID, EditionNumber, Year)**

* PK: (BookID, EditionNumber)
* FK: BookID → BOOK(BookID)

**MEMBER (MemberID, Name, Email, Phone, Address, LoginID)**

* PK: MemberID
* FK: LoginID → LOGIN(LoginID)

**STAFF (StaffID, Name, Role, LoginID)**

* PK: StaffID
* FK: LoginID → LOGIN(LoginID)

**LOGIN (LoginID, Username, Password, Role)**

* PK: LoginID
* Role can be ENUM('Admin', 'Librarian', 'Member')

**BORROWING (BorrowID, BookID, MemberID, BorrowDate, DueDate, ReturnDate, FineAmount, ManagedBy)**

* PK: BorrowID
* FKs: BookID → BOOK(BookID), MemberID → MEMBER(MemberID), ManagedBy → STAFF(StaffID)

**RESERVATION (ReservationID, BookID, MemberID, ReservationDate, Status)**

* PK: ReservationID
* FKs: BookID → BOOK(BookID), MemberID → MEMBER(MemberID)
* Status: ENUM('Waiting', 'Fulfilled', 'Cancelled')

**3. Candidate Keys and Foreign Keys**

**🔹 Candidate Keys:**

| **Table** | **Candidate Key(s)** |
| --- | --- |
| BOOK | BookID |
| BOOK\_AUTHOR | (BookID, AuthorName) |
| BOOK\_EDITION | (BookID, EditionNumber) |
| MEMBER | MemberID |
| STAFF | StaffID |
| LOGIN | LoginID |
| BORROWING | BorrowID |
| RESERVATION | ReservationID |

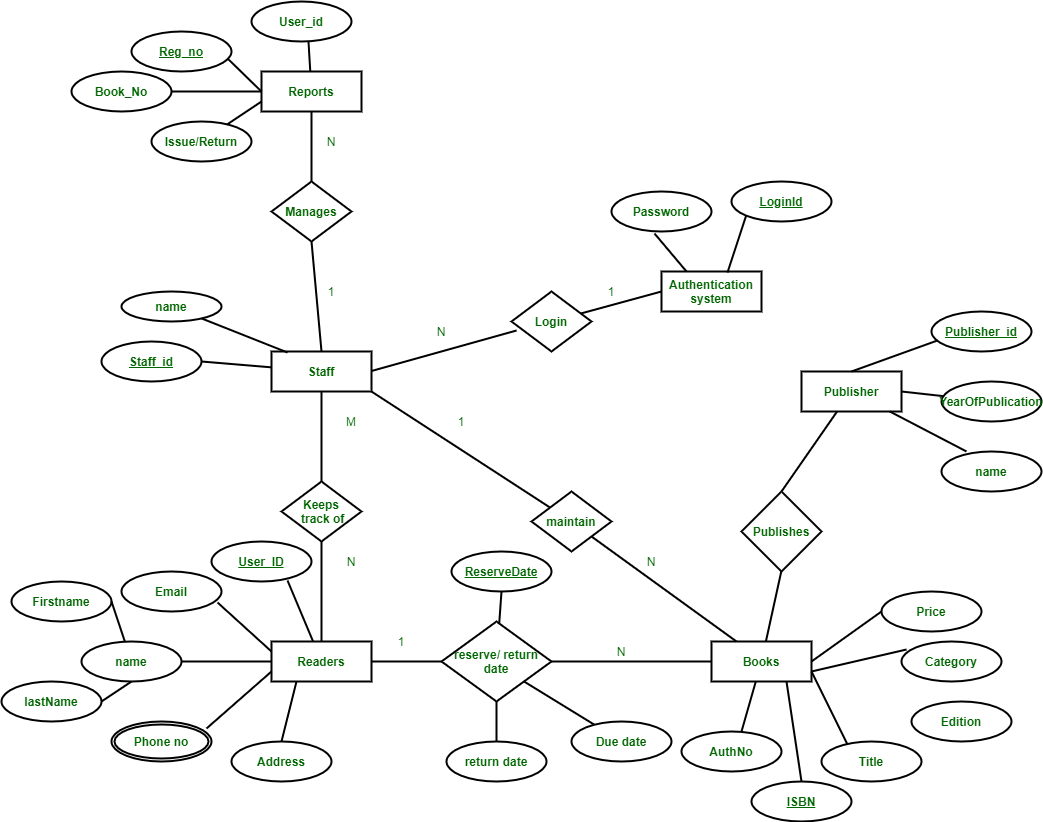
**🔹 Foreign Keys:**

| **Table** | **Foreign Key** | **References** |
| --- | --- | --- |
| MEMBER | LoginID | LOGIN(LoginID) |
| STAFF | LoginID | LOGIN(LoginID) |
| BOOK\_AUTHOR | BookID | BOOK(BookID) |
| BOOK\_EDITION | BookID | BOOK(BookID) |
| BORROWING | BookID | BOOK(BookID) |
| BORROWING | MemberID | MEMBER(MemberID) |
| BORROWING | ManagedBy | STAFF(StaffID) |
| RESERVATION | BookID | BOOK(BookID) |
| RESERVATION | MemberID | MEMBER(MemberID) |

**Functional Dependencies to Justify 3NF**

* **BOOK**: BookID → Title, ISBN, Publisher, YearPublished
* **MEMBER**: MemberID → Name, Email, Phone, Address, LoginID
* **LOGIN**: LoginID → Username, Password, Role
* **BORROWING**: BorrowID → BookID, MemberID, BorrowDate, etc.
* **RESERVATION**: ReservationID → BookID, MemberID, Date, Status

Each table has only one candidate key → in **3NF** since non-prime attributes are fully functionally dependent on the key.



**School/University Management System with Exams**

**🔷 1. ER Diagram (Textual Description)**

**ENTITIES and ATTRIBUTES**

**STUDENT**

* **StudentID** *(PK)*
* Name
* Email
* Phone
* Gender
* DateOfBirth
* DepartmentID *(FK)*

**DEPARTMENT**

* **DepartmentID** *(PK)*
* DepartmentName

**COURSE**

* **CourseID** *(PK)*
* CourseName
* CreditHours
* DepartmentID *(FK)*

**INSTRUCTOR**

* **InstructorID** *(PK)*
* Name
* Email
* Phone
* LoginID *(FK)*

**LOGIN**

* **LoginID** *(PK)*
* Username
* Password
* Role *(ENUM: Admin, Instructor)*

**ENROLLMENT**

* **EnrollmentID** *(PK)*
* **StudentID** *(FK)*
* **CourseID** *(FK)*
* Semester
* Year

**COURSE\_INSTRUCTOR**

* **CourseID** *(FK)*
* **InstructorID** *(FK)*
* Role *(ENUM: Theory, Lab)*
* PK = (CourseID, InstructorID)

**ATTENDANCE**

* **AttendanceID** *(PK)*
* **StudentID** *(FK)*
* **CourseID** *(FK)*
* Date
* Status *(Present/Absent)*

**WARNING**

* **WarningID** *(PK)*
* **StudentID** *(FK)*
* **CourseID** *(FK)*
* Reason *(e.g., "Low Attendance")*
* WarningDate

**EXAM**

* **ExamID** *(PK)*
* **CourseID** *(FK)*
* ExamType *(ENUM: Midterm, Final, Quiz, Assignment)*
* Weight *(e.g., 0.3 for Midterm)*
* ExamDate

**GRADE**

* **GradeID** *(PK)*
* **StudentID** *(FK)*
* **ExamID** *(FK)*
* MarksObtained

**2. Normalized Relational Schema (3NF)**

**STUDENT (StudentID, Name, Email, Phone, Gender, DateOfBirth, DepartmentID)**

* PK: StudentID
* FK: DepartmentID → DEPARTMENT(DepartmentID)

**DEPARTMENT (DepartmentID, DepartmentName)**

* PK: DepartmentID

**COURSE (CourseID, CourseName, CreditHours, DepartmentID)**

* PK: CourseID
* FK: DepartmentID → DEPARTMENT(DepartmentID)

**INSTRUCTOR (InstructorID, Name, Email, Phone, LoginID)**

* PK: InstructorID
* FK: LoginID → LOGIN(LoginID)

**LOGIN (LoginID, Username, Password, Role)**

* PK: LoginID

**ENROLLMENT (EnrollmentID, StudentID, CourseID, Semester, Year)**

* PK: EnrollmentID
* FKs: StudentID → STUDENT(StudentID), CourseID → COURSE(CourseID)

**COURSE\_INSTRUCTOR (CourseID, InstructorID, Role)**

* PK: (CourseID, InstructorID)
* FKs: CourseID → COURSE(CourseID), InstructorID → INSTRUCTOR(InstructorID)

**ATTENDANCE (AttendanceID, StudentID, CourseID, Date, Status)**

* PK: AttendanceID
* FKs: StudentID → STUDENT(StudentID), CourseID → COURSE(CourseID)

**WARNING (WarningID, StudentID, CourseID, Reason, WarningDate)**

* PK: WarningID
* FKs: StudentID → STUDENT(StudentID), CourseID → COURSE(CourseID)

**EXAM (ExamID, CourseID, ExamType, Weight, ExamDate)**

* PK: ExamID
* FK: CourseID → COURSE(CourseID)
* Weight values: 0.3 (Midterm), 0.5 (Final), 0.2 (Assignment), etc.

**GRADE (GradeID, StudentID, ExamID, MarksObtained)**

* PK: GradeID
* FKs: StudentID → STUDENT(StudentID), ExamID → EXAM(ExamID)

**3. Candidate Keys & Foreign Keys**

| **Table** | **Candidate Keys** | **Foreign Keys** |
| --- | --- | --- |
| STUDENT | StudentID | DepartmentID |
| DEPARTMENT | DepartmentID | - |
| COURSE | CourseID | DepartmentID |
| INSTRUCTOR | InstructorID | LoginID |
| LOGIN | LoginID | - |
| ENROLLMENT | EnrollmentID | StudentID, CourseID |
| COURSE\_INSTRUCTOR | (CourseID, InstructorID) | CourseID, InstructorID |
| ATTENDANCE | AttendanceID | StudentID, CourseID |
| WARNING | WarningID | StudentID, CourseID |
| EXAM | ExamID | CourseID |
| GRADE | GradeID | StudentID, ExamID |

**4. Grading and Weighting System**

To compute final grades per course:

SELECT

g.StudentID,

e.CourseID,

SUM(g.MarksObtained \* ex.Weight) AS WeightedScore

FROM

GRADE g

JOIN

EXAM ex ON g.ExamID = ex.ExamID

JOIN

ENROLLMENT e ON e.StudentID = g.StudentID AND e.CourseID = ex.CourseID

GROUP BY

g.StudentID, e.CourseID;

