### **1. ER Diagram**

**Entities & Attributes:**

* **Student**: StudentID (PK), Name, Program, Year, DepartmentID (FK)
* **Faculty**: FacultyID (PK), Name, DepartmentID (FK), Type (FullTime / Visiting) → *Use generalization*
* **Department**: DepartmentID (PK), Name
* **Course**: CourseID (PK), Name, DepartmentID (FK), FacultyID (FK)
* **Enrollment**: StudentID (FK), CourseID (FK), GPA → *Associative Entity*
* **Classroom**: RoomID (PK), Capacity
* **Schedule**: ScheduleID (PK), RoomID (FK), CourseID (FK), TimeSlot
* **Event**: EventID (PK), Name, Date, ClubName, DepartmentID (FK) → *Use aggregation if multiple departments involved*
* **Inventory**: ItemID (PK), Name, Type, Quantity
* **IssueRecord**: RecordID (PK), ItemID (FK), IssuedTo (StudentID/FacultyID), IssueDate, ReturnDate

### **2. Relational Schema Design**

**Normalize up to 3NF or BCNF.**

STUDENT(StudentID INT PRIMARY KEY, Name VARCHAR(100), Program VARCHAR(50), Year INT, DepartmentID INT FOREIGN KEY)

FACULTY(FacultyID INT PRIMARY KEY, Name VARCHAR(100), DepartmentID INT FOREIGN KEY, Type VARCHAR(20))

DEPARTMENT(DepartmentID INT PRIMARY KEY, Name VARCHAR(100))

COURSE(CourseID INT PRIMARY KEY, Name VARCHAR(100), DepartmentID INT, FacultyID INT)

ENROLLMENT(StudentID INT, CourseID INT, GPA FLOAT, PRIMARY KEY(StudentID, CourseID))

CLASSROOM(RoomID INT PRIMARY KEY, Capacity INT)

SCHEDULE(ScheduleID INT PRIMARY KEY, RoomID INT, CourseID INT, TimeSlot VARCHAR(50))

EVENT(EventID INT PRIMARY KEY, Name VARCHAR(100), Date DATE, ClubName VARCHAR(100), DepartmentID INT)

INVENTORY(ItemID INT PRIMARY KEY, Name VARCHAR(100), Type VARCHAR(50), Quantity INT)

InventoryRecord(RecordID INT PRIMARY KEY, ItemID INT, IssuedTo INT, IssuerType VARCHAR(10), IssueDate DATE, ReturnDate DATE)

**6. Indexing Proposal**

Use:

* **B+ Tree Indexing** on StudentID, FacultyID, CourseID → for range and sorted queries.
* **Hash Indexing** on RoomID, ItemID → for fast lookups.

**Justification:**

* B+ Trees help in range queries like “Students with GPA > 8”.
* Hash is better for exact match (e.g., Room availability).

### **7. User Roles (Optional)**

Create views or simple CLI forms:

* **Admin**: Full access
* **Faculty**: Can view assigned courses, issue records
* **Student**: View grades, events, issue history

Example CLI using Python + SQLite or PHP + MySQL for form-based access.