**1.1)Relation A**

1.Superkeys:(EmpID),(SSN),(Email),(EmpID,Phone),(Email,Name),(SSN,Salary)

2.Candidate keys:EmpID,SSN,Email,Phone

3.Primary key: EmpID,I would choose it because it can not be null and stable

4.By the given table we can see that all phone numbers are unique. Numbers can not be the same because the sample confirms it as a candidate key.

**Relation B**

**1.**StudentID,Coursecode,Section,Semester and Year

2. StudentId – identifies which student

CourseCode- identifies which course

Section- different sectiond of same course in the same semester

Semester and Year- needed because same course could be repeated

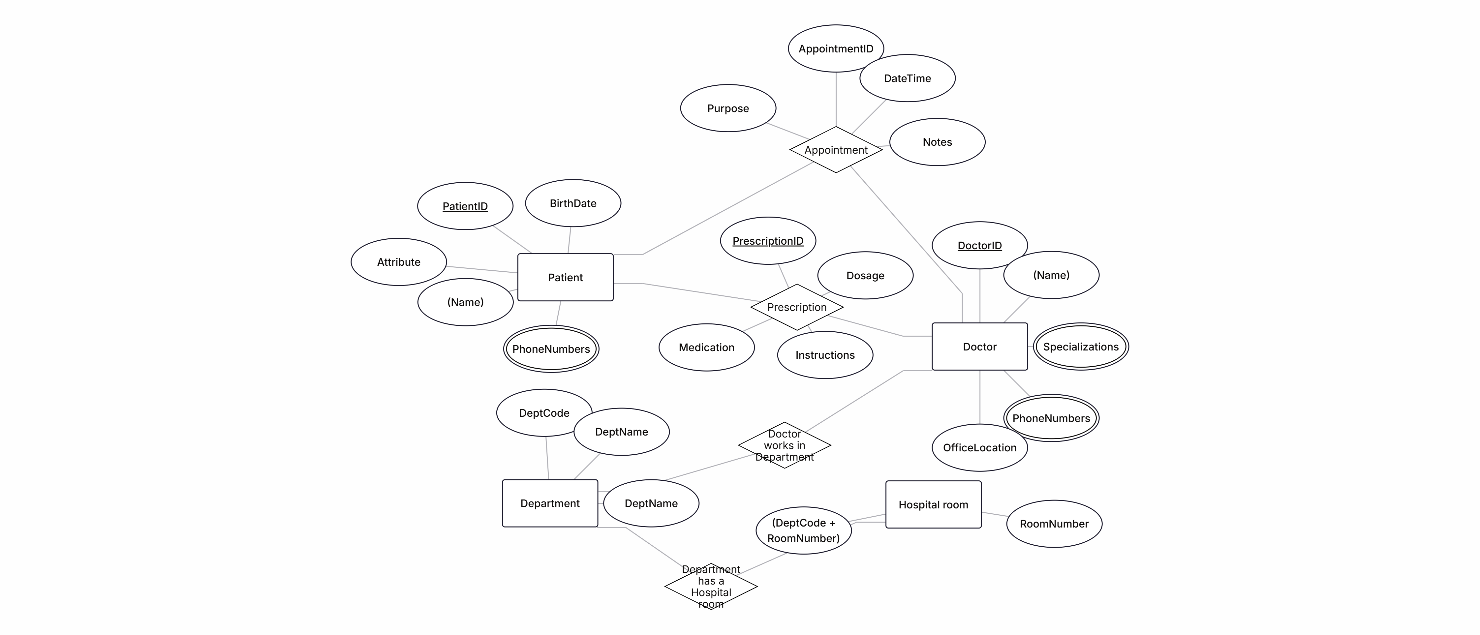
**3.**

Candidate key: (StudentID, CourseCode, Section, Semester, Year)

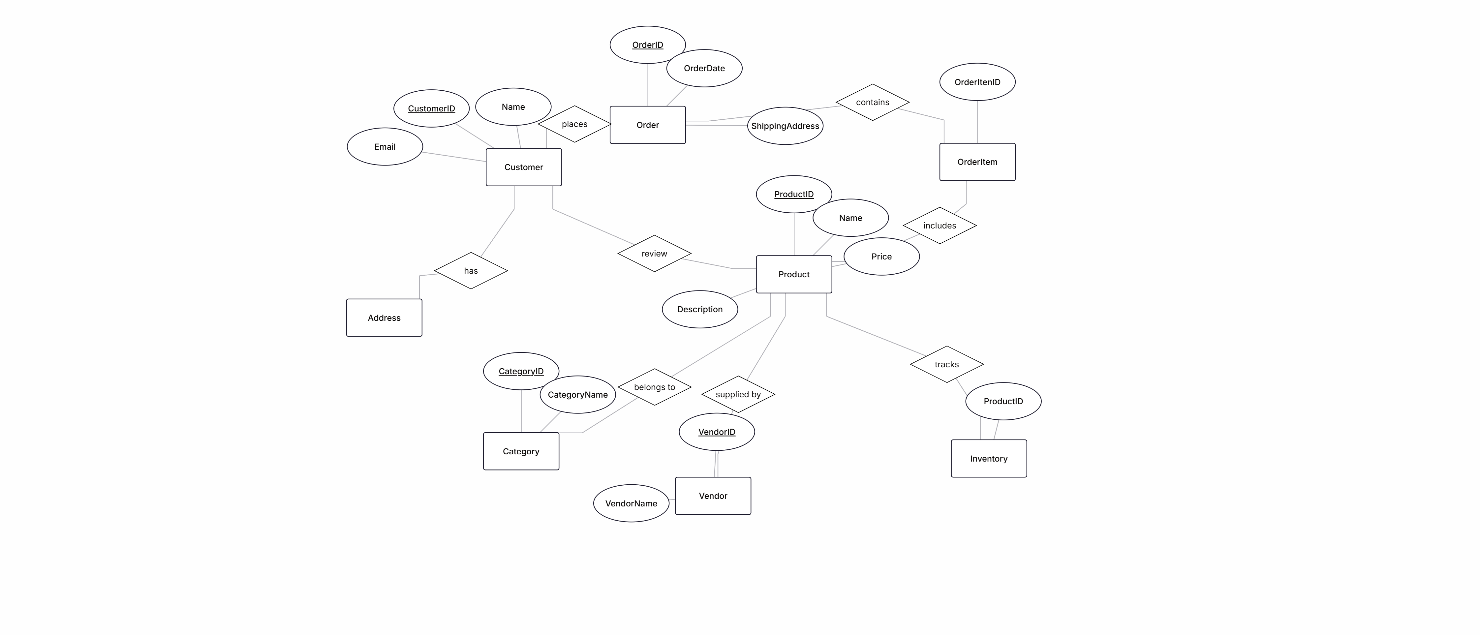
**1.2**

1. **Student – Professor**
   * 1. AdvisorID → ProfID
2. **Student – Department**
   * 1. Major - DeptCode
3. **Professor – Department**
   * 1. Department - DeptCode
4. **Department – Professor**
   * 1. ChairID - ProfID
5. **Course - Department**
   1. DepartmentCode - DeptCode
6. **Enrollment – Student**
   1. StudentID - Student.StudentID
7. **Enrollment – Course**
   1. CourseID - Course.CourseID

**2.1**



**2.2**



**4.1**

### 1. Functional Dependencies

* StudentID → StudentName, StudentMajor
* ProjectID → ProjectTitle, ProjectType, StartDate, EndDate
* SupervisorID → SupervisorName, SupervisorDept
* {StudentID, ProjectID} → Role, HoursWorked

**2.**

Redundancy:StufentName and StudentMajor repeats for every student project

SupervisorName and SupervusorDept repeats for supervisor

Update anomaly:changing students major requires updating multiple rows

Insertion Anomaly: Cannot add a new supervisor without assigning them to a project

Deletion Anomaly: Deleting the last student from a project could lose project information**3.**Already in 1NF

### 4. Apply 2NF

1. **Primary Key:** {StudentID, ProjectID}
2. **Partial dependencies:**

StudentID → StudentName, StudentMajor

ProjectID → ProjectTitle, ProjectType, StartDate, EndDate

SupervisorID → SupervisorName, SupervisorDept

1. **Decomposition (2NF):**

Student(StudentID, StudentName, StudentMajor)

Project(ProjectID, ProjectTitle, ProjectType, StartDate, EndDate, SupervisorID)

Supervisor(SupervisorID, SupervisorName, SupervisorDept)

StudentProject(StudentID, ProjectID, Role, HoursWorked)

### 5.Apply 3NF

1. Project → SupervisorID → SupervisorName, SupervisorDept

**3NF :**

Student(StudentID, StudentName, StudentMajor)

Project(ProjectID, ProjectTitle, ProjectType, StartDate, EndDate, SupervisorID)

Supervisor(SupervisorID, SupervisorName, SupervisorDept)

StudentProject(StudentID, ProjectID, Role, HoursWorked)

**4.2**

1.Primary key (StudentID, CourseID, TimeSlot)]

### 2. Functional Dependencies

StudentID → StudentMajor

CourseID → CourseName

InstructorID → InstructorName

{Room, TimeSlot} → Building

{CourseID, TimeSlot} → InstructorID, Room

{StudentID, CourseID, TimeSlot}

**3.** not superkeys

StudentID → StudentMajor

CourseID → CourseName

InstructorID → InstructorName

{Room, TimeSlot} → Building

**4.**

Student(StudentID, StudentMajor)

Course(CourseID, CourseName)

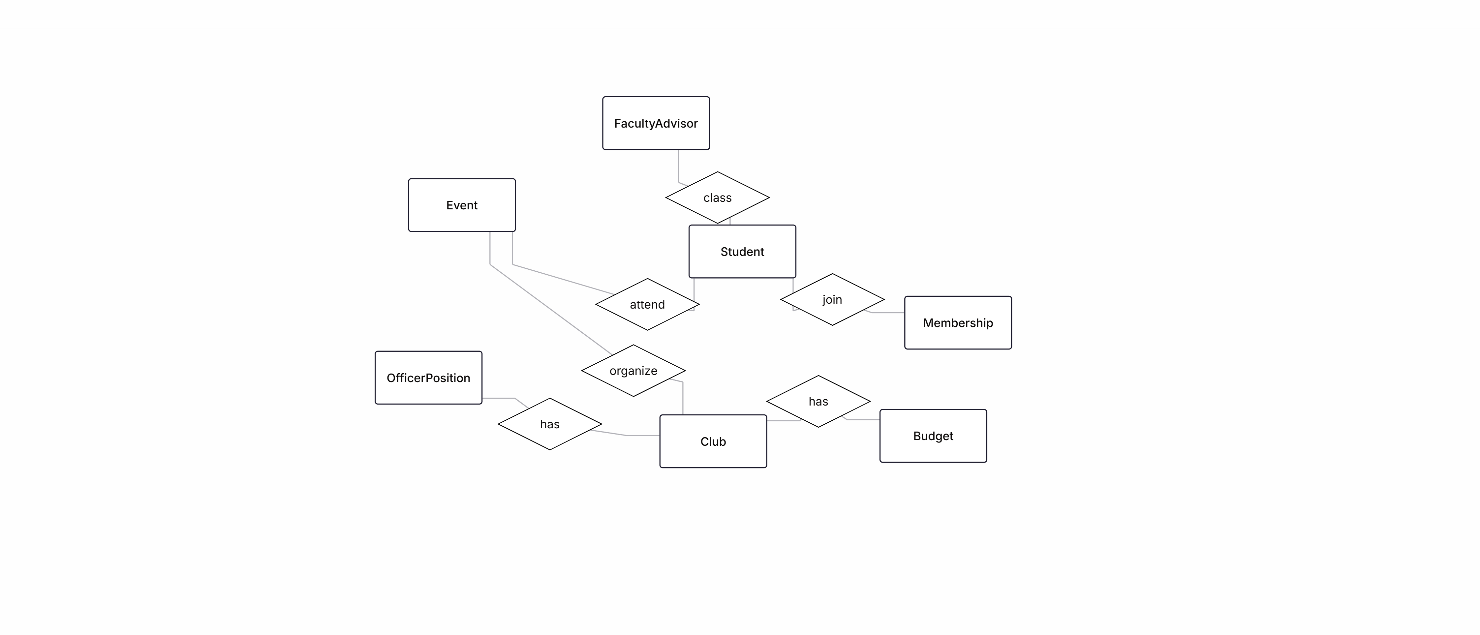
Instructor(InstructorID, InstructorName)

Room(Room, TimeSlot, Building)

CourseSchedule(StudentID, CourseID, TimeSlot, InstructorID, Room)

**5**.no loss

**5.**



**5.2**

### Normalized Relational Schema (3NF)

**Student(StudentID PK, StudentName, Major, Email, Phone)**

**Club(ClubID PK, ClubName, Description, AdvisorID FK)**

**FacultyAdvisor(AdvisorID PK, Name, Department, Email)**

**Membership(StudentID FK, ClubID FK, JoinDate, Role, PRIMARY KEY(StudentID, ClubID))**

**OfficerPosition(PositionID PK, ClubID FK, StudentID FK, Title, StartDate, EndDate)**

**Event(EventID PK, ClubID FK, Title, EventDate, RoomID FK)**

**Attendance(StudentID FK, EventID FK, Status, PRIMARY KEY(StudentID, EventID))**

**Room(RoomID PK, Building, Capacity)**

**Budget(BudgetID PK, ClubID FK, Year, AmountAllocated)**

**Expense(ExpenseID PK, BudgetID FK, Description, Amount, Date)**

**3.**

Option 1: Store officer role as an attribute in Membership.

Option 2: Create a separate OfficerPosition entity.

**4.**

1. Find all students who are officers in the Computer Science Club.
2. List all events scheduled for next week along with their room reservations.
3. Show the total allocated budget and total expenses for each club this semester.