Task 1.1

1. 6 different superkeys: EmplID , SSN, Email, Phone, EmplID+Name, SSN+Department
2. Candidate keys: EmplID, SSN, Email, Phone

(Cause they unique)

1. The best choice is EmplID for the primary key (It’s unique and stable, cannot change like email or phone number)
2. According to the data each employee has a different Phone number. If the Phone is included in the candidate keys, it means that it is unique.

Relation B

1)A minimal primary key is – { StudentID, CourseCode, Section, Semester, Year}

2) Student ID - Identifies which student is this and for what row belongs this student.

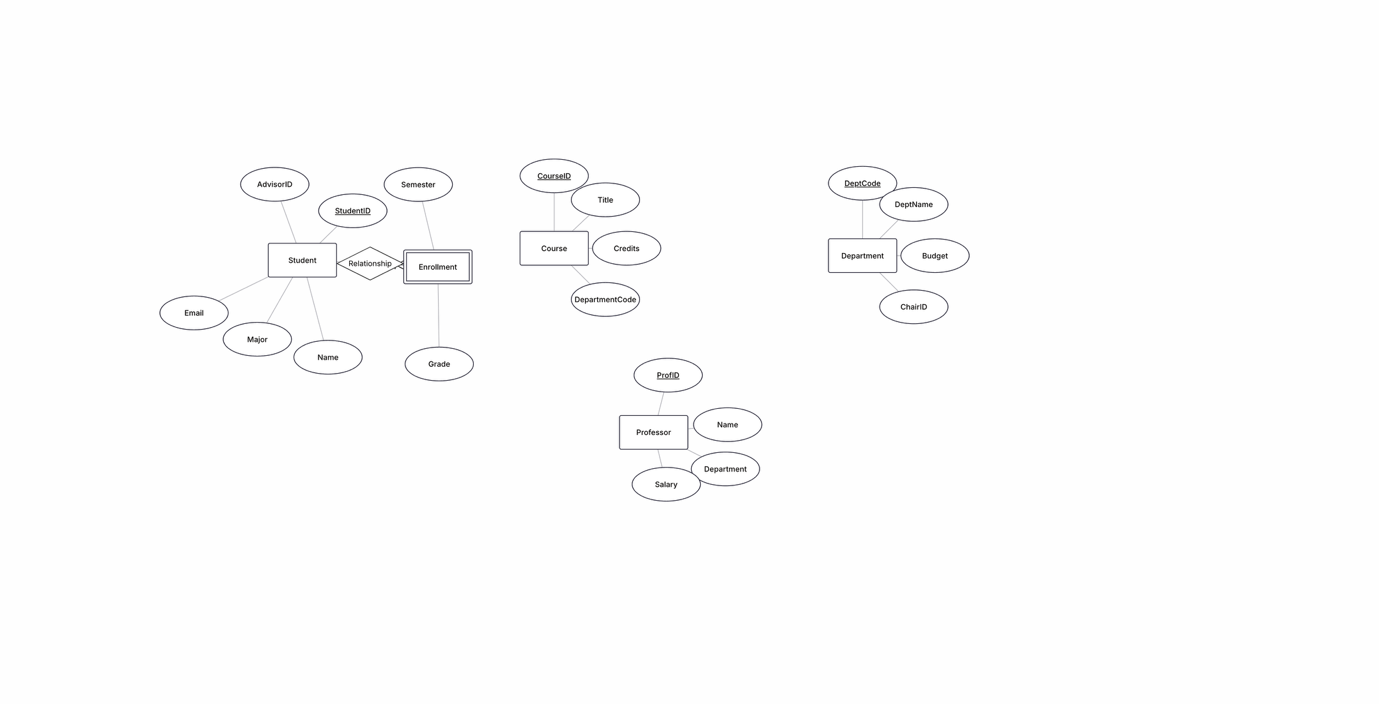
CoureCode – identifies the course

Section – in one course can be different section, so its necessary distinguish which section is this

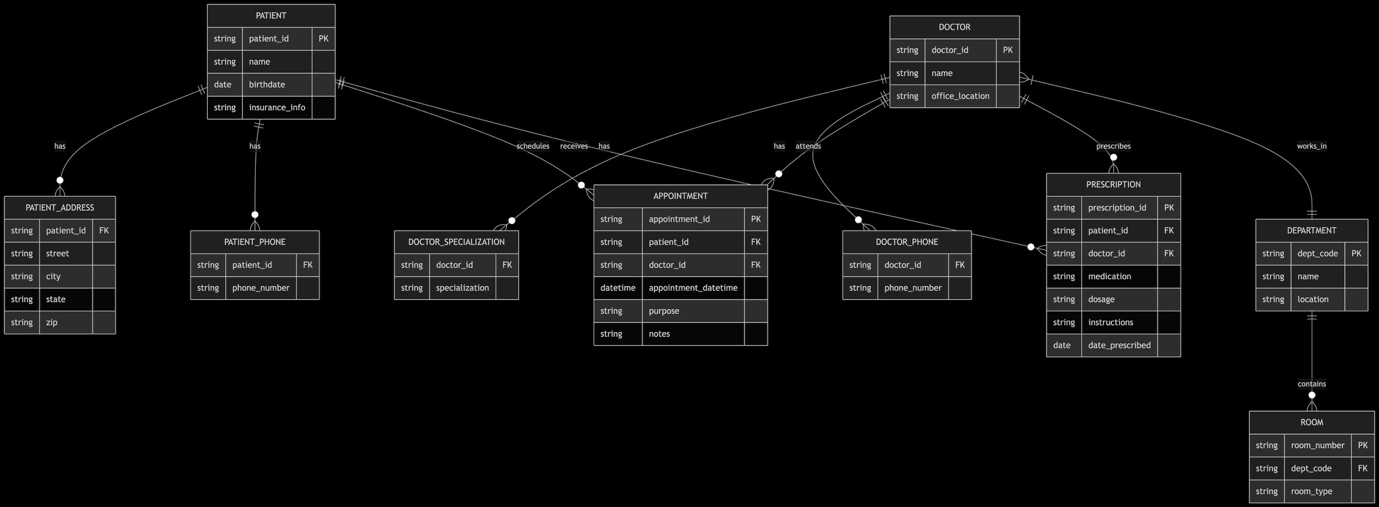
Semester and Year – a student can take same course in different year or semester, but in rule says student can take same curse in different semester, so we need to include semester+year.

3)For this case we cant choose candidate key only reordered previous key.

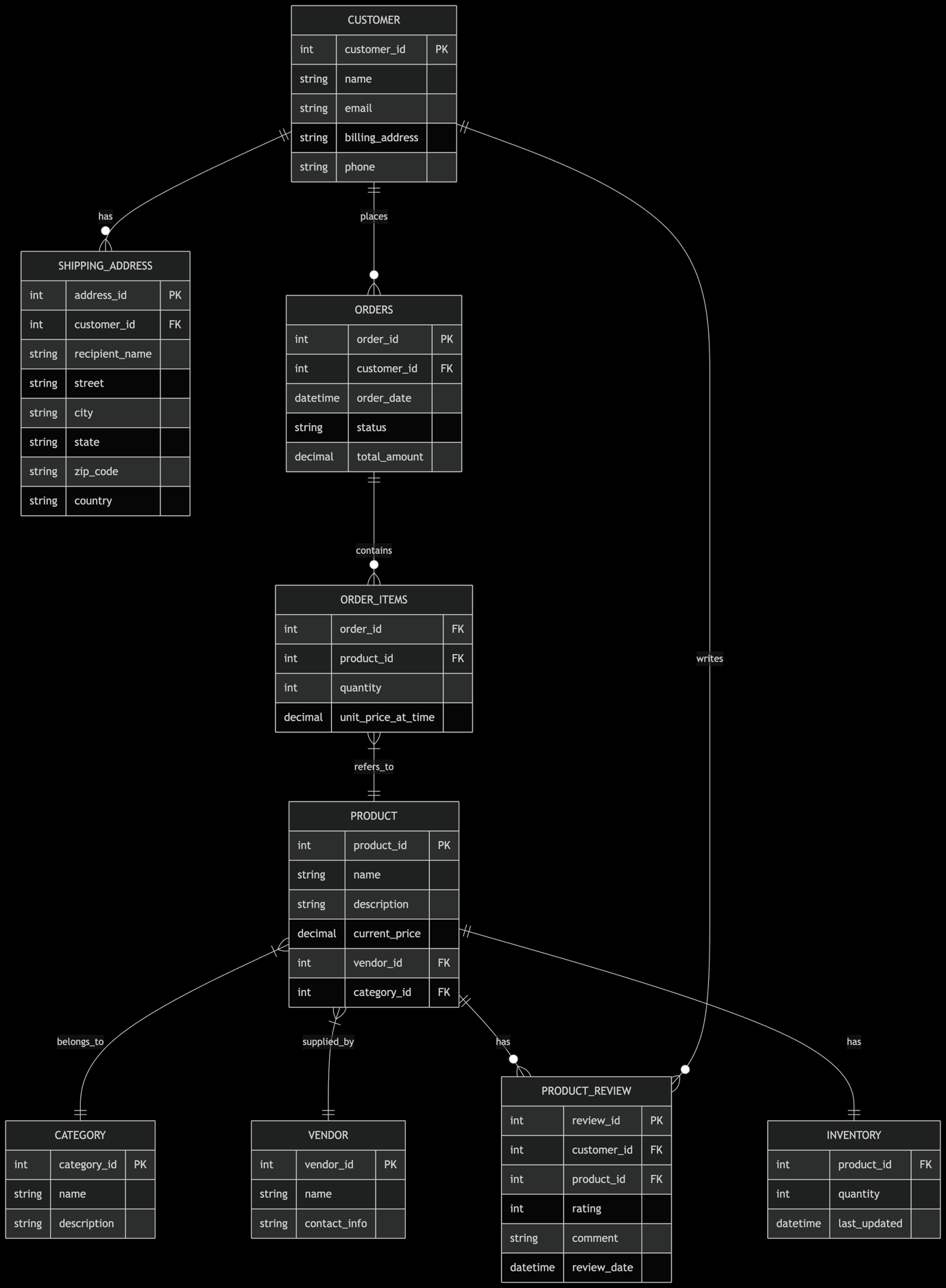
Task 1.2



Task 2.1



Task 2.2



Task 4.1

StudentID → StudentName, StudentMajor

ProjectID → ProjectTitle, ProjectType, SupervisorID

SupervisorID → SupervisorName, SupervisorDept

ProjectID, StudentID → Role, HoursWorked, StartDate, EndDate

1nf no

2nd decomposition

STUDENT(StudentID, StudentName, StudentMajor)

PROJECT(ProjectID, ProjectTitle, ProjectType, SupervisorID)

SUPERVISOR(SupervisorID, SupervisorName, SupervisorDept)

STUDENT\_PROJECT(StudentID, ProjectID, Role, HoursWorked, StartDate, EndDate)

3rd

STUDENT(StudentID, StudentName, StudentMajor)

PROJECT(ProjectID, ProjectTitle, ProjectType, SupervisorID)

SUPERVISOR(SupervisorID, SupervisorName, SupervisorDept)

STUDENT\_PROJECT(StudentID, ProjectID, Role, HoursWorked, StartDate, EndDate)

Task 4.2

1. Primary Key Determination

Primary Key: (StudentID, CourseID, TimeSlot)

2. Functional Dependencies

StudentID → StudentMajor

CourseID → CourseName

InstructorID → InstructorName

TimeSlot, Room → Building

CourseID, TimeSlot → InstructorID, Room, Building

StudentID, CourseID, TimeSlot → StudentMajor, CourseName, InstructorID, InstructorName, Room, Building

3. BCNF Check

The table is NOT in BCNF because there are functional dependencies where the determinant is not a superkey.

4. BCNF Decomposition

Final BCNF Decomposition:

STUDENT(StudentID, StudentMajor)

COURSE(CourseID, CourseName)

INSTRUCTOR(InstructorID, InstructorName)

ROOM\_SCHEDULE(TimeSlot, Room, Building)

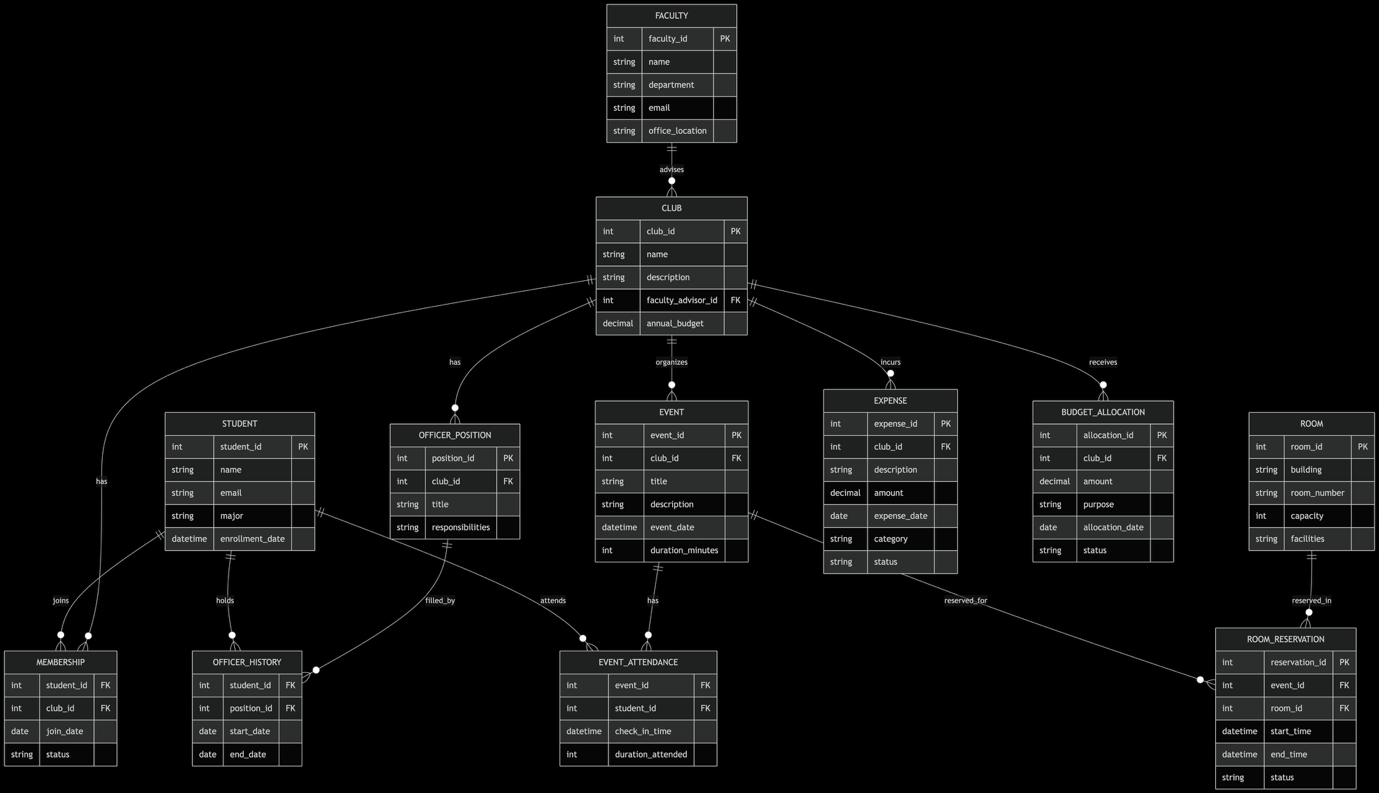
COURSE\_SCHEDULE(CourseID, TimeSlot, InstructorID, Room)

STUDENT\_ENROLLMENT(StudentID, CourseID, TimeSlot)

5. Potential Loss of Information

The decomposition is lossless - it preserves all functional dependencies and the natural join of all decomposed tables will reconstruct the original table without loss of information.

Task 5.1



Normalized Relational Schema

STUDENT (student\_id, name, email, major, enrollment\_date)

FACULTY (faculty\_id, name, department, email, office\_location)

CLUB (club\_id, name, description, faculty\_advisor\_id, annual\_budget)

MEMBERSHIP (student\_id, club\_id, join\_date, status)

OFFICER\_POSITION (position\_id, club\_id, title, responsibilities)

OFFICER\_HISTORY (student\_id, position\_id, start\_date, end\_date)

EVENT (event\_id, club\_id, title, description, event\_date, duration\_minutes)

EVENT\_ATTENDANCE (event\_id, student\_id, check\_in\_time, duration\_attended)

ROOM (room\_id, building, room\_number, capacity, facilities)

ROOM\_RESERVATION (reservation\_id, event\_id, room\_id, start\_time, end\_time, status)

EXPENSE (expense\_id, club\_id, description, amount, expense\_date, category, status)

BUDGET\_ALLOCATION (allocation\_id, club\_id, amount, purpose, allocation\_date, status)

Design Decision with Multiple Options

Option for tracking officer positions:

1. Create separate boolean columns for each position (is\_president, is\_treasurer, etc.)

2. Create a separate OFFICER\_POSITION table with position types

Chosen approach: Option 2 with a separate OFFICER\_POSITION table

Justification: This approach is more flexible as it allows clubs to define custom positions beyond the standard ones. It also maintains better normalization by avoiding multiple boolean columns and makes it easier to track the history of officer positions over time.

Example Queries

1. "Find all students who are officers in the Computer Science Club and their respective positions"

2. "List all events scheduled for next week with their room reservations, including building and room number"

3. "Calculate the total expenses for each club in the current semester, ordered by highest spending"