# Database lab1

Part1

Task1.1

1){Emp ID},{SSN},{Email},{Phone},{Emp ID,Name},{Emp ID,Email}

2){Emp ID}.{SSN},{Email}

3)Emp ID, because in any situation it cannot be duplicated and cannot be changed

4)No, because based on the data shown everyone has different phone numbers

Relation B

1){Student ID, Course code, Section, Semester, Year}

2)Student ID-identifies the student

Course code-identifies the course

Section-

Semester, Year-differs the passing of same course in different semester and year

3)They do not exist

Task 1.2

Student(AdvisorID) - Professor(ProfID)

Professor(Department) – Department(DeptCode)

Course(DepartmentCode) - Department(DeptCode)

Department(ChairID) – Professor(ProfID)

Enrollment(StudentID) – Student(StudentID)

Enrollment(CourseID) – Course(CourseID)

Task 2

1)patient-strong

Doctors-strong

Department-strong

Appointment-weak, because it is dependent on patient and doctor

Prescriptons-weak, because it is dependent on patient and doctor

Room-weak because it is dependent on department

2)Patient:

PatientID (PK, simple)

Name (simple)

Birthdate (simple)

Address (composite: Street, City, State, Zip)

Phone (multi-valued)

Insurance Info (simple)

Doctors:

Doctor ID(PK,simple)

Name(simple)

Specialization(multi-valued)

Phone numbers(multi-valued)

Office locations(

Department:

Department code(PK,simple)

Name(simple)

Location(composite)

Appointment:

Patient ID(simple)

Doctor ID(simple)

Date/time(simple)

Purpose(

Notes(

Prescriptions:

Medication(simple)

Patient ID(simple)

Doctor ID(simple)

Dosage(

Instruction(

Hospital room:

Number(

Department code(

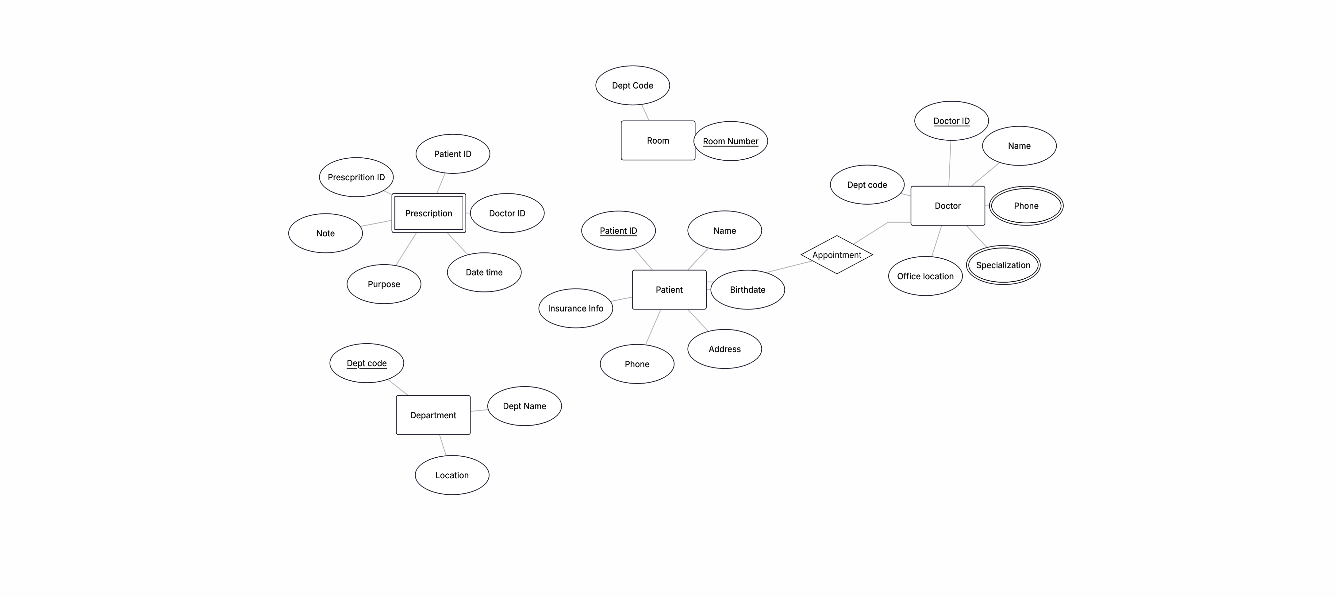
3)Patient-Appointment-Doctor(M:N)

Doctor-Prescription-Patient(M:N)

Department-Doctor(1:N)

Department-room(1:N)

4)



Task2.2

1)A diagram of a product

AI-generated content may be incorrect.

2)Order Item-weak, because it is dependent on Order and Product

3)Order-Order Item-Product(M:N)

Task 4

StudentID - StudentName, StudentMajor

ProjectID - ProjectTitle, ProjectType, SupervisorID

SupervisorID - SupervisorName, SupervisorDept

(StudentID, ProjectID) - Role, HoursWorked, StartDate, EndDate

2)

3) The table is already in 1NF

4) Student(StudentID, StudentName, StudentMajor)

Project(ProjectID, ProjectTitle, ProjectType, SupervisorID)

Supervisor(SupervisorID, SupervisorName, SupervisorDept)

StudentProject(StudentID, ProjectID, Role, HoursWorked, StartDate, EndDate)

Partialdependencies**:**

StudentID - StudentName, StudentMajor

ProjectID - ProjectTitle, ProjectType, SupervisorID

5) Student(StudentID PK, StudentName, StudentMajor)

Supervisor(SupervisorID PK, SupervisorName, SupervisorDept)

Project(ProjectID PK, ProjectTitle, ProjectType, SupervisorID FK)

StudentProject(StudentID FK, ProjectID FK, Role, HoursWorked, StartDate, EndDate, PK = (StudentID, ProjectID))

Transitive dependency: ProjectID - SupervisorID - SupervisorName, SupervisorDept

Task 4.2

1)PK = (StudentID, CourseID, TimeSlot, Room)

2) StudentID - StudentMajor

CourseID - CourseName

InstructorID - InstructorName

(TimeSlot, Room) - Building

(CourseID, TimeSlot, Room) - InstructorID

(StudentID, CourseID, TimeSlot, Room) - all other enrollment details

3) The table is not in BCNF because:

CourseID - CourseName (CourseID is not a superkey).

InstructorID - InstructorName (InstructorID is not a superkey).

(TimeSlot, Room) - Building (not a superkey either).

4) Student(StudentID PK, StudentMajor)

Course(CourseID PK, CourseName)

Instructor(InstructorID PK, InstructorName)

Room(Room PK, Building)

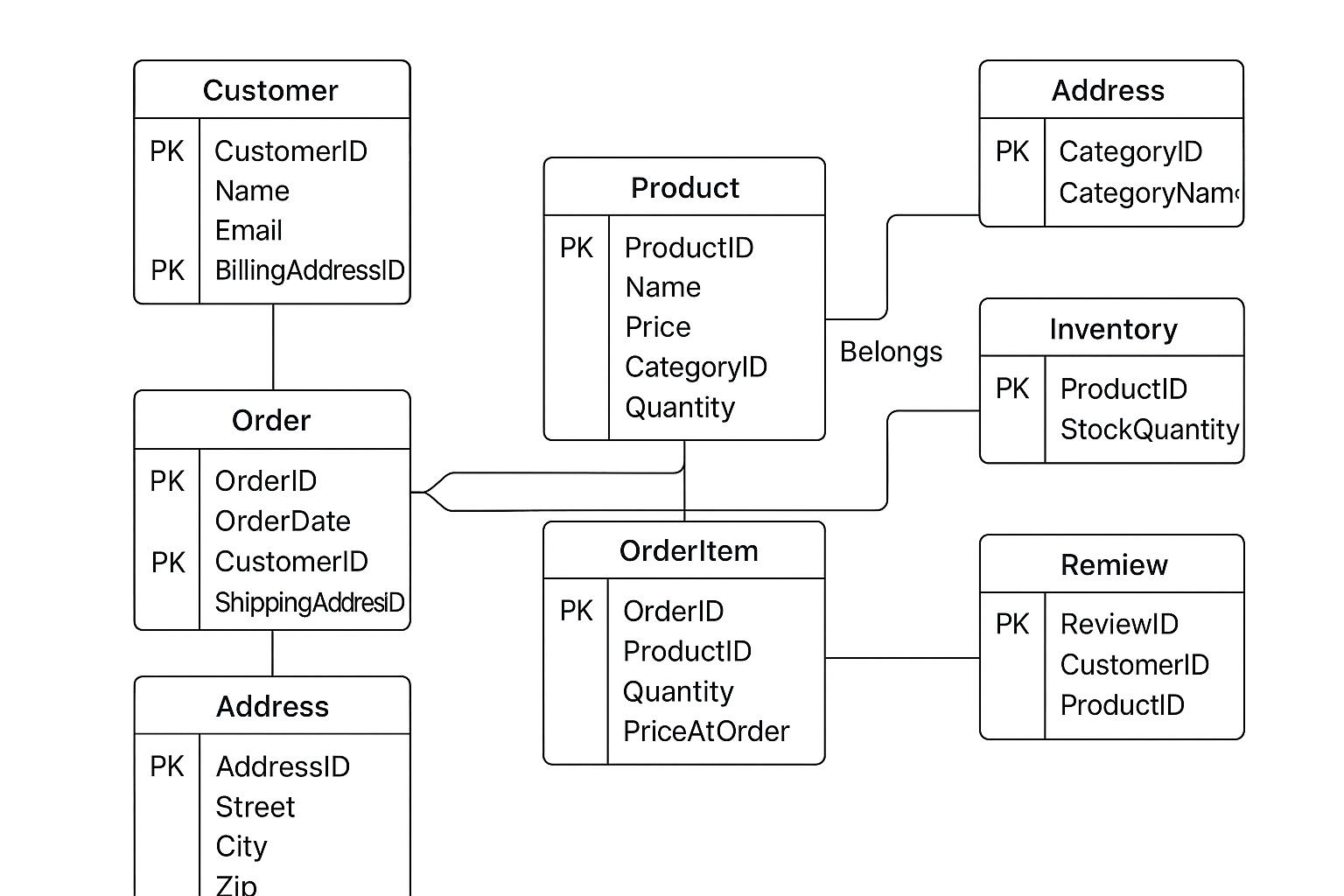
CourseSection(CourseID FK, InstructorID FK, TimeSlot, Room,  
  PK = (CourseID, TimeSlot, Room))

Enrollment(StudentID FK, CourseID, TimeSlot, Room,  
  PK = (StudentID, CourseID, TimeSlot, Room))

5) There is no information loss because all functional dependencies are preserved in separate relations.

However, reconstructing the original table requires multiple joins.

Task 5

1)

2)Student(StudentID PK, Name, Email, Major)

Club(ClubID PK, ClubName, Budget, FacultyAdvisorID FK)

Faculty(FacultyID PK, Name, Department)

Membership(StudentID FK, ClubID FK, JoinDate, Role, PK = (StudentID, ClubID))

Officer(StudentID FK, ClubID FK, Position, PK = (StudentID, ClubID, Position))

Event(EventID PK, ClubID FK, EventName, EventDate, Purpose)

Attendance(StudentID FK, EventID FK, Status, PK = (StudentID, EventID))

Room(RoomID PK, Location, Capacity)

Reservation(ReservationID PK, EventID FK, RoomID FK, DateTime)

Expense(ExpenseID PK, ClubID FK, Amount, Description, Date)

3) For officers, I used a separate weak entity Officer instead of just adding a “Position” field in Membership.

This allows one student to be both a member and an officer at the same time, possibly in multiple clubs. It keeps officer roles flexible and extensible.

4) Find all students who are officers in the Computer Science Club.

List all events scheduled for next week with their reserved rooms.

Show the total expenses for each club in the last semester.