Laboratory work #1

Task 1.1

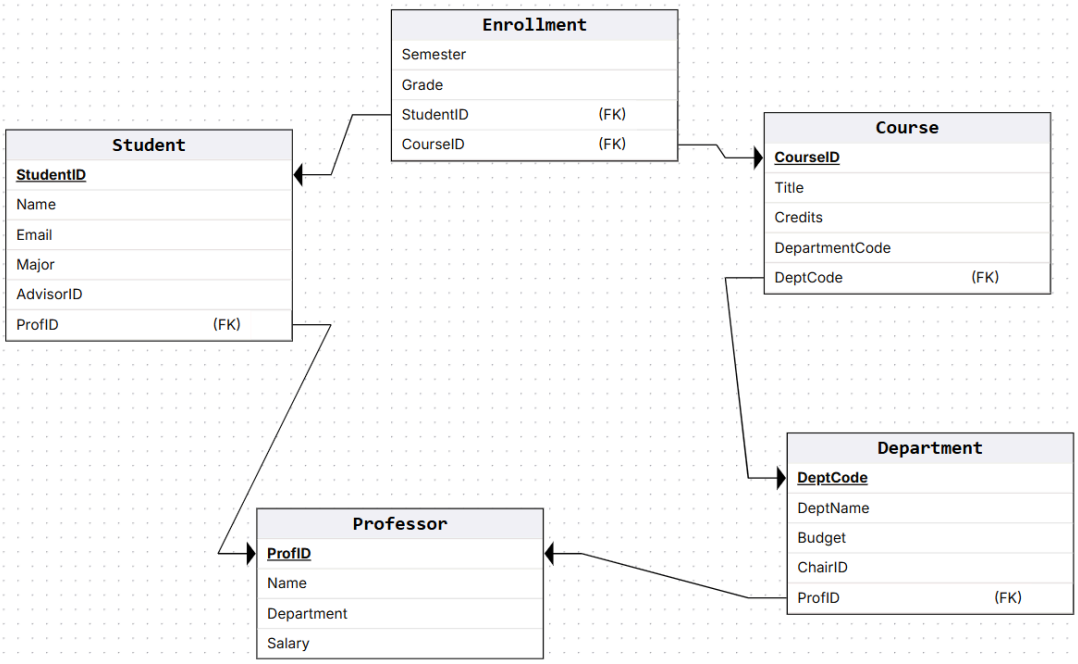
Relation A

1) {EmpID}, {SSN}, {Email}, {EmpID, Name}, {SSN, Department}, {EmpID, SSN, Email}
2) {EmpID}, {SSN}, {Email}
3) EmpID, because it is always unique, cannot be null and rarely changes. Unlike SSN, it is not sensitive data, so it's okay to be publicly known.
4) Based on the data shown, there are no same phone numbers. Talking in general, phone numbers could be shared among employees. For example, elderly married workers could use one phone number together.

Relation B

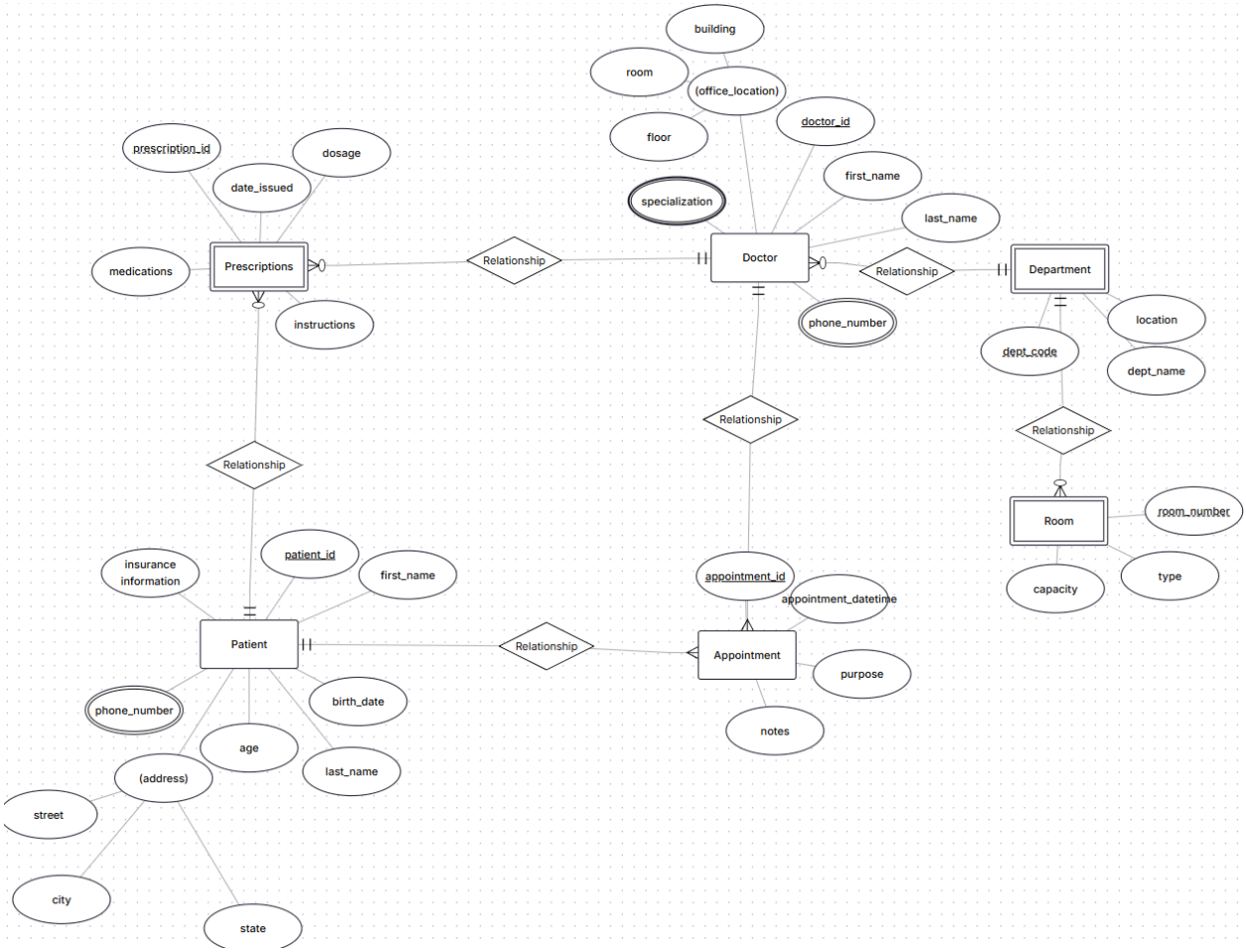
1) {StudentID, CourseCode, Section, Semester, Year}
2) StudentID, because a unique ID of a student is needed so we know who exactly registered.
   CourseCode, because each course can be identified by the unique code so the registration works correctly.
   Section, because a student cannot register for the same course section in the same semester, they can register for different sections of the same course and same semester.
   Semester, because a student can take the same course in different semesters, semester has to be in the primary key.
   Year, because without it data cannot be told apart from different years of the same course, student, section, and semester.
3) Additional candidate keys don't exist.

Task 1.2



1. AdvisorID references Professor(ProdID); Department references Department(DeptCode); DepartmentCode references Department(DepartmentCode); ChairID references Professor(ProfID); StudentID references Student(StudentID); CourseID references Course(CourseID)

Task 2.1

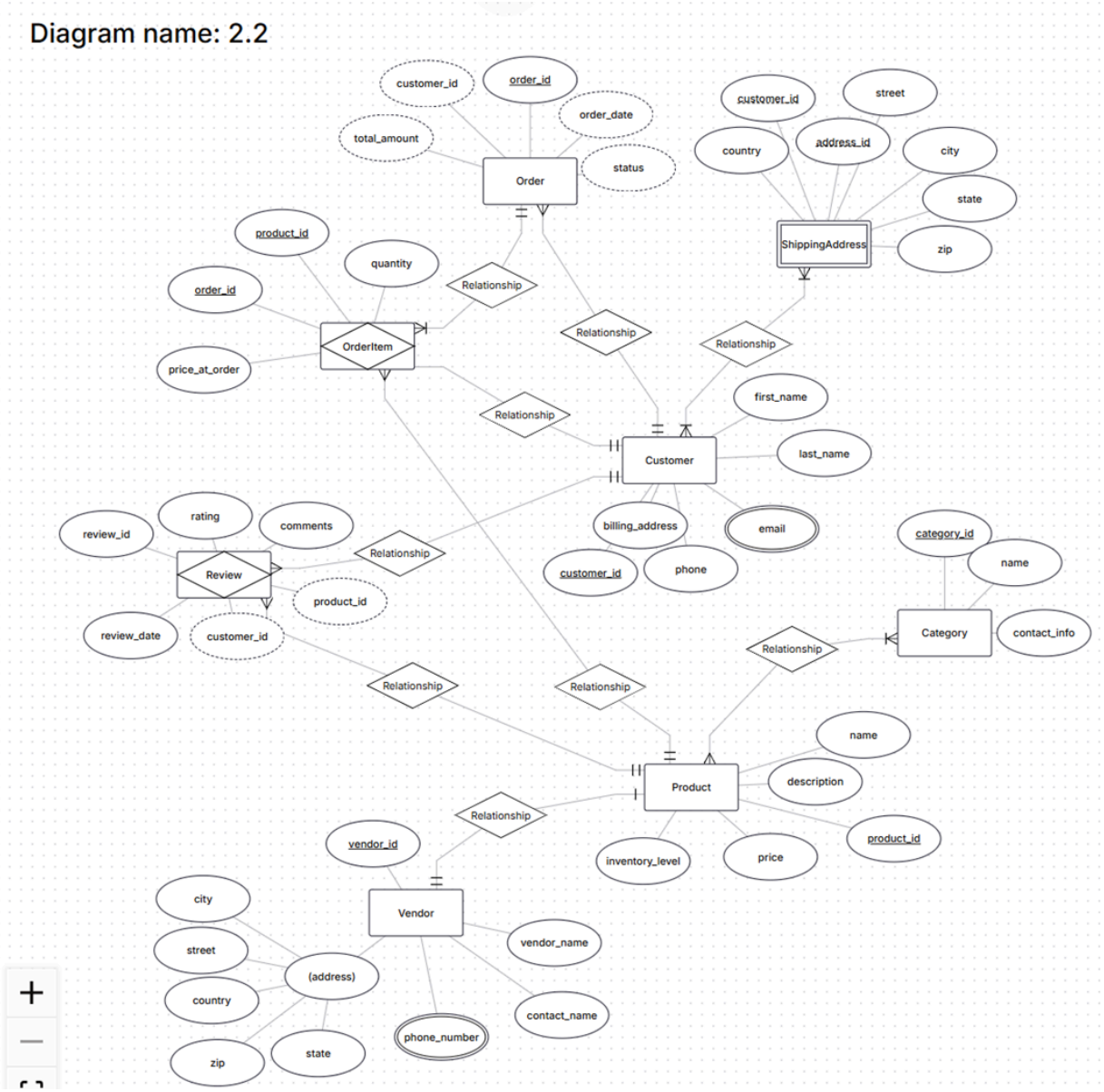


1) Strong: patient, doctor, department; Weak: appointments, prescription, room
2) Patient entity:
   Simple: patient_id, first_name, birth_date, last_name, age, street, city, state
   Composite: address
   Multi-valued: phone-number
   Appointment entity:
   Simple: appointment_id,
   Patient : patient_id (PK, simple); first_name (simple);last_name (simple);birth_date (simple);age (simple); address (composite)=street, city, state, zip (simple sub-attributes); phone_number (multi-valued)
   Doctor: doctor_id (PK, simple); first_name, last_name (simple);phone_number (multi-valued);office_location (multi-valued composite — e.g., building, floor, room); specialization(multi-valued)
   Department: dept_code (PK, simple);dept_name (simple);location(simple)

Room: partial key: room_number, room_number (partial key; simple), type (simple — e.g., ICU / Regular), capacity (simple)
Identification**:** Room is identified by (dept_code, room_number)
Appointment: appointment_id, appointment_id (PK), appointment_datetime (simple), purpose (simple), notes (simple)
Prescription: prescription_id (PK); date_issued (simple); dosage (simple); instructions (simple)

3) Doctors and Departments (N:1);
   Departments and HospitalRooms (1:N);
   Doctors and Prescriptions (1:N)
   Prescriptions and Patients (N:1)
   Patients and Appointments (1:N)
   Appointments and Doctors (N:1)

Task 2.2



2) ShippingAddress is weak because it is dependent on the Customer entity. Because it has address_id and customer_id primary keys, which is the data from different entities, making it a weak entity.
3) Customers and Products through Reviews. A customer can leave reviews for multiple products, and each product can receive reviews from many customers.

Task 4.1

1) StudentID → StudentName, StudentMajor;
   ProjectID → ProjectTitle, ProjectType
   SupervisorID → SupervisorName, SupervisorDept
   (StudentID, ProjectID, SupervisorID) → Role, HoursWorked, StartDate, EndDate

2) Redundancy: student details repeat for every project they're in. Project details repeat for every student working on it. Supervisor details repeat for every project/student.

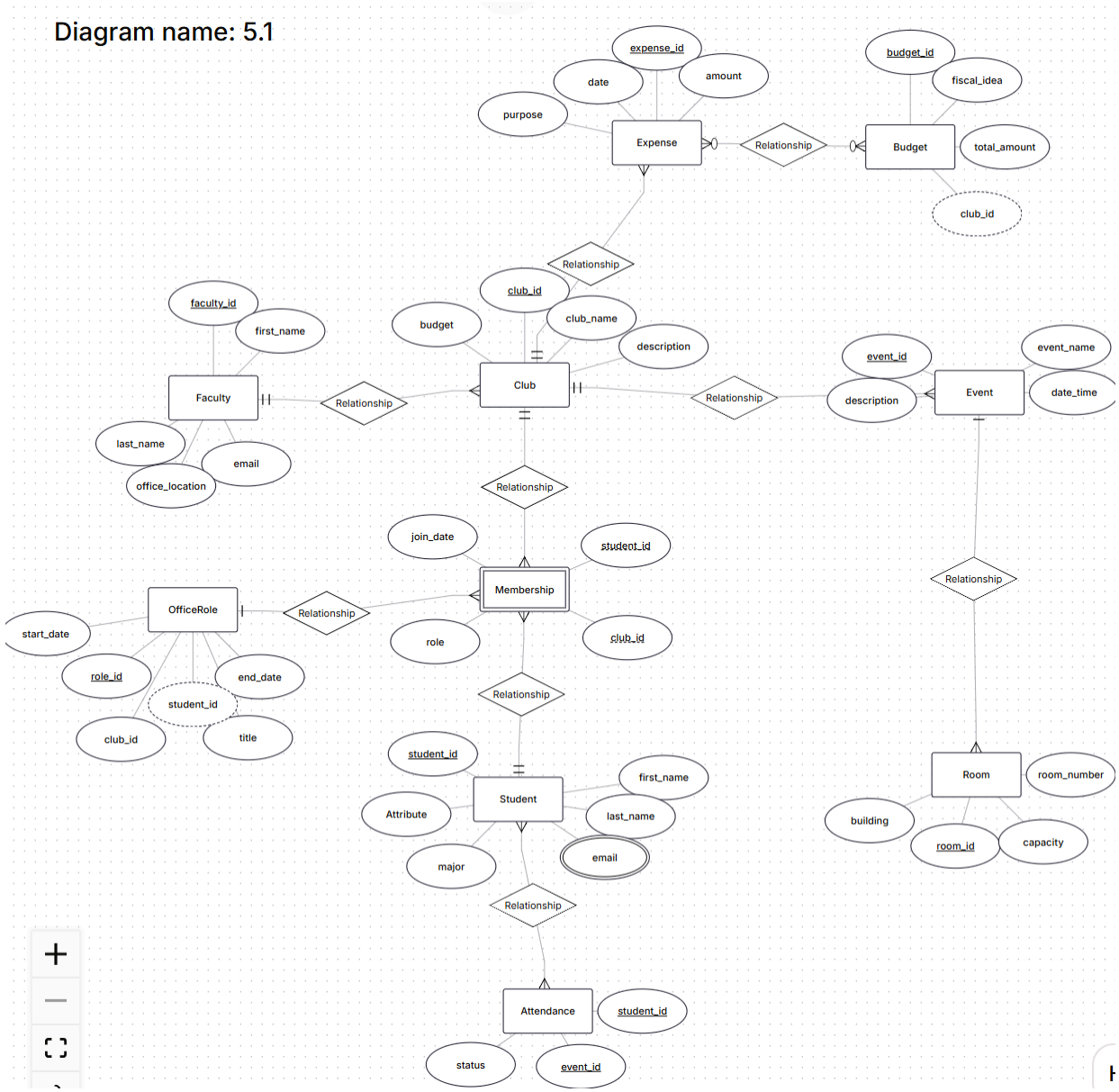
   Update**:**If a student changes major, you must update it in all rows where that student appears. Insert**:**Can't add a new student until they are assigned to a project.Can't add a new supervisor until they supervise a project. Delete**:** If you remove the last project a student is in, you also lose the student's record entirely.
3) No violations
4) Primary key - (StudentID, ProjectID, SupervisorID)

   Partial dependencies - StudentID → StudentName, StudentMajor
   ProjectID → ProjectTitle, ProjectType
   SupervisorID → SupervisorName, SupervisorDept

   2NF:
   Student(StudentID, StudentName, StudentMajor)
   Project(ProjectID, ProjectTitle, ProjectType)
   Supervisor(SupervisorID, SupervisorName, SupervisorDept)
   StudentProject(StudentID, ProjectID, SupervisorID, Role, HoursWorked, StartDate, EndDate)
5) No transitive dependencies. In Student, non-key attributes depend only on StudentID → no issue.In Project, non-key attributes depend only on ProjectID. In Supervisor, non-key attributes depend only on SupervisorID.In StudentProject, all attributes depend on the whole composite key (StudentID, ProjectID, SupervisorID).

   3NF:
   Student(StudentID, StudentName, StudentMajor)
   Project(ProjectID, ProjectTitle, ProjectType)
   Supervisor(SupervisorID, SupervisorName, SupervisorDept)
   StudentProject(StudentID, ProjectID, SupervisorID, Role, HoursWorked, StartDate, EndDate)

Task 5.1



2. Student(StudentID PK, Name, Major, Year)
Faculty(FacultyID PK, Name, Department)
Club(ClubID PK, ClubName, Budget, AdvisorID FK → Faculty(FacultyID))
Membership(StudentID FK → Student, ClubID FK → Club, JoinDate, PRIMARY KEY(StudentID, ClubID))
OfficerPosition(StudentID FK → Student, ClubID FK → Club, PositionName, StartDate, EndDate, PRIMARY KEY(StudentID, ClubID, PositionName))
Event(EventID PK, ClubID FK → Club, EventName, Date, Time, RoomID FK → Room)
EventAttendance(StudentID FK → Student, EventID FK → Event, PRIMARY KEY(StudentID, EventID))
Room(RoomID PK, Building, Capacity)

3. Two choices: first one to include officer info in Membership table and second one, the chosen one, to create a separate OfficerPosition table. Because a student may hold many positions over time, even in the same club. Separating positions allows us to keep up with positions without repeating membership info, and reducing redundancy problems.
4. Find all students who are officers (e.g., Presidents) of any club, along with the name of the club they lead.
Generate a list of all students who attended events for the 'Data Science Club' during the Fall 2023 semester, ordered by the number of events they attended.
For the 'Robotics Club', show the total allocated budget, the sum of all expenses, and the remaining balance for the 2024 academic year