Task 1.1: Superkey and Candidate Key Analysis

- **Superkeys (at least 6):**
    1. {EmpID}
    2. {SSN}
    3. {Email}
    4. {EmpID, Name}
    5. {SSN, Department}
    6. {Email, Phone}
- **Candidate Keys:**
    - {EmpID}, {SSN}, {Email}
- **Primary Key (chosen):**
    - EmpID
- **Phone number uniqueness:**
    - Two employees can have the same phone number, so Phone is not a key.

Task 1.2: Foreign Key Design

- `Student.AdvisorID` → `Professor.ProfID`
- `Course.DepartmentCode` → `Department.DeptCode`
- `Department.ChairID` → `Professor.ProfID`
- `Enrollment.StudentID` → `Student.StudentID`
- `Enrollment.CourseID` → `Course.CourseID`

Task 4.1: StudentProject Normalization

Given Table:

```
StudentProject(
  StudentID, StudentName, StudentMajor,
  ProjectID, ProjectTitle, ProjectType,
  SupervisorID, SupervisorName, SupervisorDept,
  Role, HoursWorked, StartDate, EndDate
)
```

1. Functional Dependencies (FDs):

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

2. Problems:

- **Redundancy:** student, project, and supervisor information is repeated.
- **Update anomaly:** changing supervisor's department requires multiple updates.
- **Insert anomaly:** cannot add a new student without a project.
- **Delete anomaly:** deleting a project may remove information about a student or supervisor.

3. 1NF:

- The table already satisfies 1NF since all attributes are atomic.

4. 2NF:

- Primary Key: (StudentID, ProjectID)
- Partial dependencies exist:
  - StudentID → StudentName, StudentMajor
  - ProjectID → ProjectTitle, ProjectType
- Decomposition into 2NF:
  1. Student(StudentID, StudentName, StudentMajor)
  2. Project(ProjectID, ProjectTitle, ProjectType)
  3. Supervisor(SupervisorID, SupervisorName, SupervisorDept)
  4. StudentProject(StudentID, ProjectID, SupervisorID, Role, HoursWorked, StartDate, EndDate)

5. 3NF:

- Check for transitive dependencies → none exist in decomposed tables.
- Final 3NF schema:
  1. **Student**(StudentID, StudentName, StudentMajor)
  2. **Project**(ProjectID, ProjectTitle, ProjectType)
  3. **Supervisor**(SupervisorID, SupervisorName, SupervisorDept)
  4. **StudentProject**(StudentID, ProjectID, SupervisorID, Role, HoursWorked, StartDate, EndDate)

Task 4.2: CourseSchedule Normalization (to BCNF)

Given Table:

```
CourseSchedule(
  StudentID, StudentMajor, CourseID, CourseName,
  InstructorID, InstructorName, TimeSlot, Room, Building
)
```

1. Primary Key:

- (StudentID, CourseID, TimeSlot)
- **2. Functional Dependencies (FDs):**

- StudentID → StudentMajor
- CourseID → CourseName
- InstructorID → InstructorName
- (Room, TimeSlot) → Building
- (CourseID, TimeSlot, Room) → InstructorID
- (StudentID, CourseID, TimeSlot) → all other attributes

3. Is the table in BCNF?

- No, because there are dependencies where the determinant is not a key (e.g., StudentID → StudentMajor).

4. Decomposition to BCNF:

1. **Student**(StudentID, StudentMajor)
2. **Course**(CourseID, CourseName)
3. **Instructor**(InstructorID, InstructorName)
4. **RoomSchedule**(Room, TimeSlot, Building)
5. **CourseSchedule**(StudentID, CourseID, TimeSlot, Room, InstructorID)

5. Loss of Information:

- No loss of information occurs because decompositions preserve all original attributes and relationships.