Part 1: Key idenification Exercises

Task 1.1:

Relation A:

1.superkesy = (EmpID) , (EmpID,SSN) ,(EmpID,SSN,Email) , (Email) , (Phone , SSN) , (Phone , Email)

2. candidate keys = (EmpID) , (SSN), (Email)

3. EmpID , because it's identification number

4 Yes, if they use the phone for work and communications with client and etc.

Relation B:

1

i. Minimal primary key is (StudentId , CourseCode, Section, Semester , Year)

ii. StudentId is necessary because it identifies student that registering on the course

iii CourseCode and section are related , because each course has many sections

iiii Semester and year are related , because student can register same course in different semesters , also we need year here , because there would be mistake if student take the same course with the same section at first semester of first and second year.\

2 there can't be any other candidate keys

Task 1.2

1

AdvisorId in student table

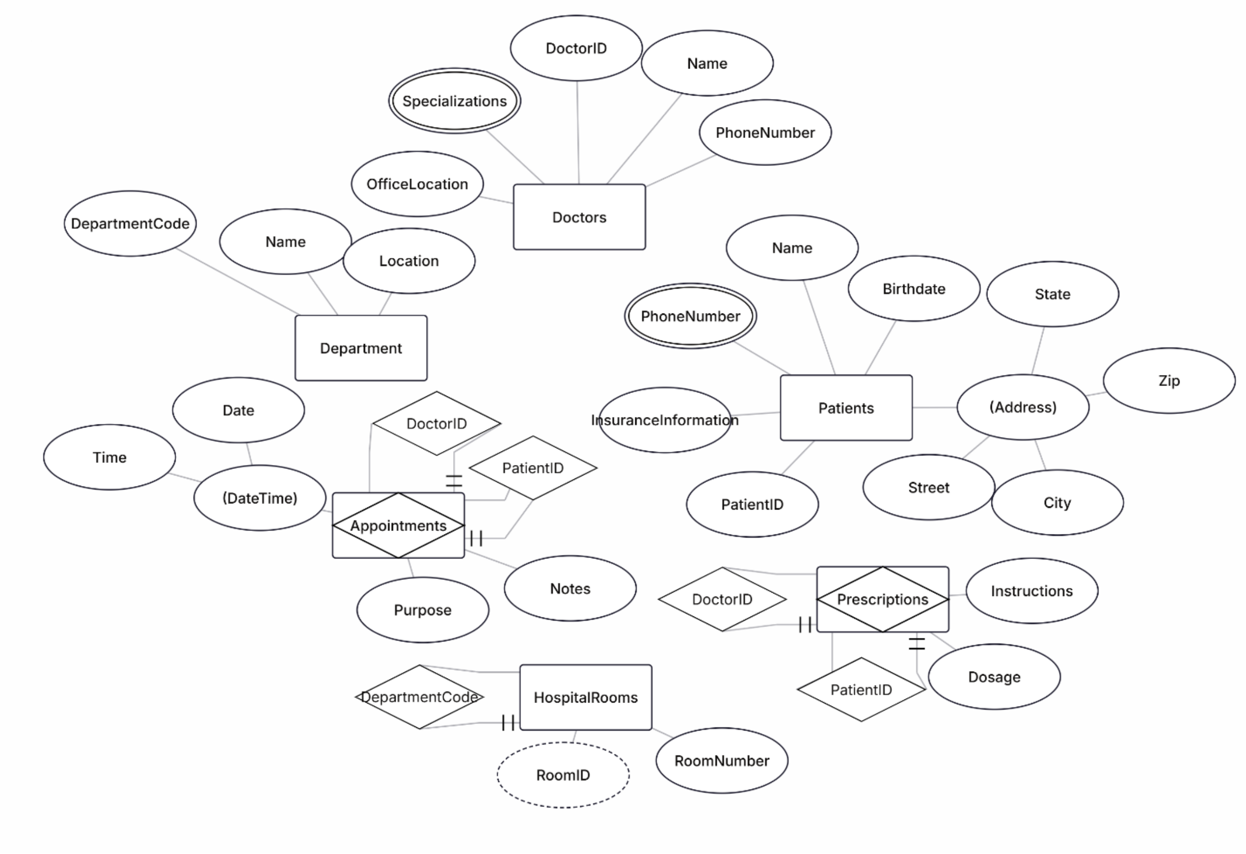
CourseId in Enrollment table

DepartmentCode in course table

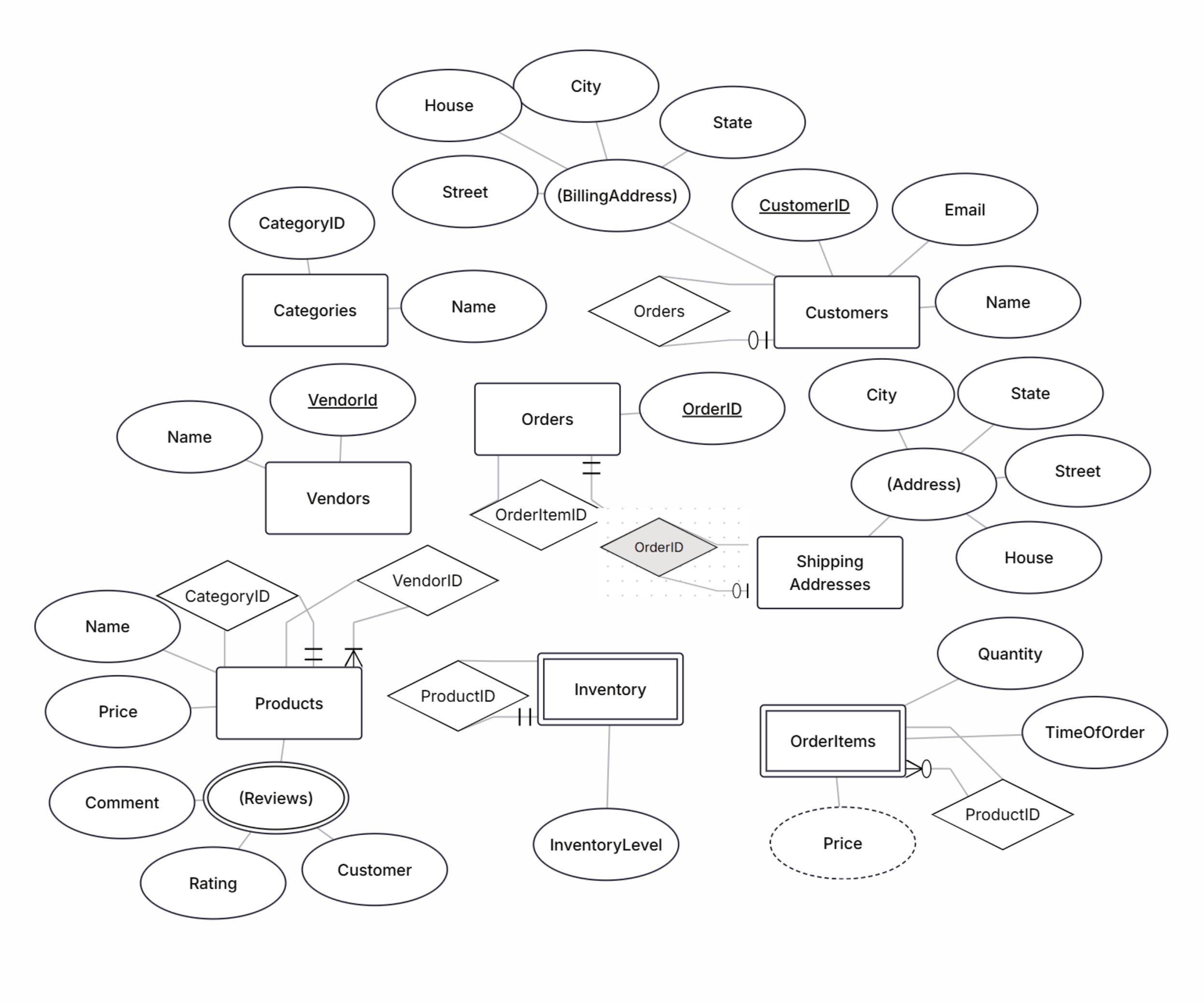
ChairId in department table

Part 2: ER Diagram Construction

Task 2.1



Task 2.2



Part 4: Normalization Workshop

Task 4.1

1 Functional Dependencies:

StudentId -> StudentName , StudentMajor;

ProjectId -> ProjectTitle, ProjectType , SupervisorId

SupervisorId -> SupervisorName, SupervisorDept

(StudentId , ProjectID) -> Roel, houseWorked , StartDate, EndDate

2

Update- If we want to update the role of some student if wiil change all roles of this student in all project that he participates.

Insert – if we want to add just empty project without any student , it won't be able

Delete – if we want to delete some student from some project , we will lose information about project and etc.

3 There is not any 1NF violations because each column contains atomic value , and there is no composite columns

4 Primary key is {StudentID , ProjectID}

Partial Dependencies - Student Name, StudentMajor depends on StudentID , Project Title , ProjectType , StartDate ,EndDtate,SupevisorID depend on ProjectID)

2NF decomposition:

Student{StudentID , StudentMajor , StudentName}

Project{ProjectID , ProjectTitle,ProjectType , SupervisorID }

StudentProject{StudentId , ProjectID , Role , HoursWorked,StartDate , EndDate}

Supervisor{SupervisorID , SupervisorName, SupervisorDept}

5 there is no any transitive dependencies in my 2NF decomposition!

3NF decomposition:

Student{StudentID , StudentMajor , StudentName}

Project{ProjectID , ProjectTitle,ProjectType , SupervisorID }

StudentProject{StudentId , ProjectID , Role , HoursWorked,StartDate , EndDate}

Supervisor{SupervisorID , SupervisorName, SupervisorDept}

Task 4.2

1 Primary key - {StudentId , CourseId , TimeSlot , Room}

2 Fucntional Dependencies :

StudentId - > StudentMajor

CourseId -> CourseName

InstructorId -> InstructorName

(Room , TimeSlot) -> building

(CourseId, TimeSlot , Room) -> InstructorID

3

It s not in bcnf , the table doesn't follow 2nf rules

4

1NF are followed

2NF Decompositon:

Student(StudentId , StudentName)

Course(CourseId , CourseName)

Instructor(InstructorId ,InstructorName)

Room(RoomId , Building)

CourseSchedule(CourseID ,TimeSlot ,Room , InstructorId)

Enrollment(StudentId , CoureId , TimeSlot , Room)

3NF Decomposition:

Same as 2NF

BCNF Decomposition:

It already follows all rules of BCNF

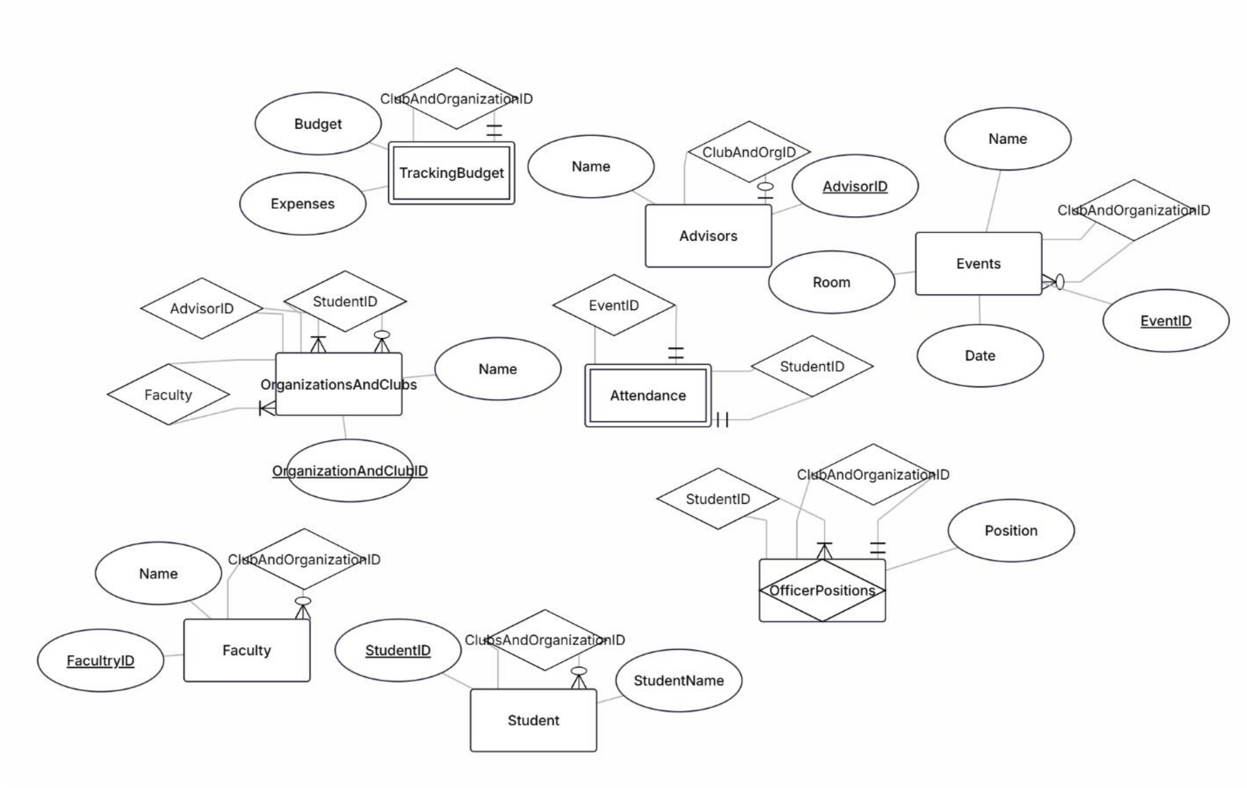
5

There is no any loss of information , we properly divided table into new 6 tables. We can take all relations by using JOIN

Part 5: Design Challenge

Task 5.1

1



2 It's already normalized

3 I could put attendance in events table but I decided to split it into two tables.

4 i. Find all event that are holding by ICPC organization

Ii. List all students that have attended on Chess Club event

iii. Select all OrganizationorClubs which budget is more than $1000