College Database

Problem Statement

A college contains many departments. Each department can offer any number of courses. Many instructors can work in a department, but an instructor can work only in one department. For each department, there is a head, and an instructor can be head of only one department. Each instructor can take any number of courses, and a course can be taken by only one instructor. A student can enroll for any number of courses and each course can have any number of students. Requirements other than the above can be added.

Group Members **(Group III)**

- 510817017 **Pankaj Naik**
- 510817018 **Trishan Kumar Hazra**
- 510817019 **Varsha Goel**
- 510817020 **Anindya Kundu**
- 510817021 **Soumabrata Bhattacharya**
- 510817022 **Shounak Chatterjee**
- 510817024 **Md Fahad**

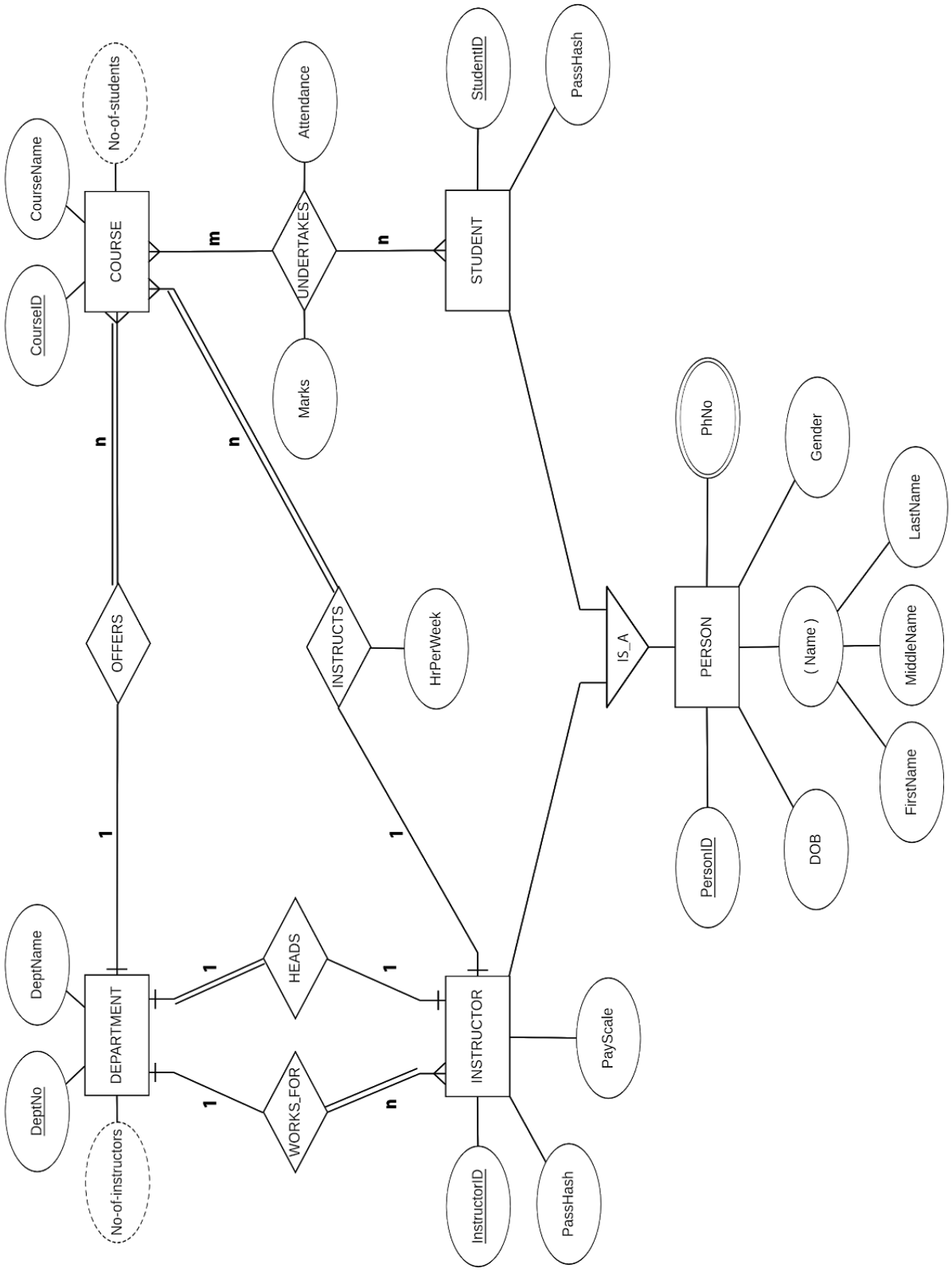
Requirement Analysis

- College has several departments.
- College has several instructors.
- Several instructors can work in one department.
- An instructor can work in only one department.
- Every department has a head who is an instructor.
- An instructor can head only one department.
- Each department can offer any number of courses.
- An instructor can only take a course offered by his department.
- Each instructor can take any number of courses.
- A course can be taken by only one instructor.
- College has several students.
- A student can enroll for any number of courses.
- Each course can have any number of students.

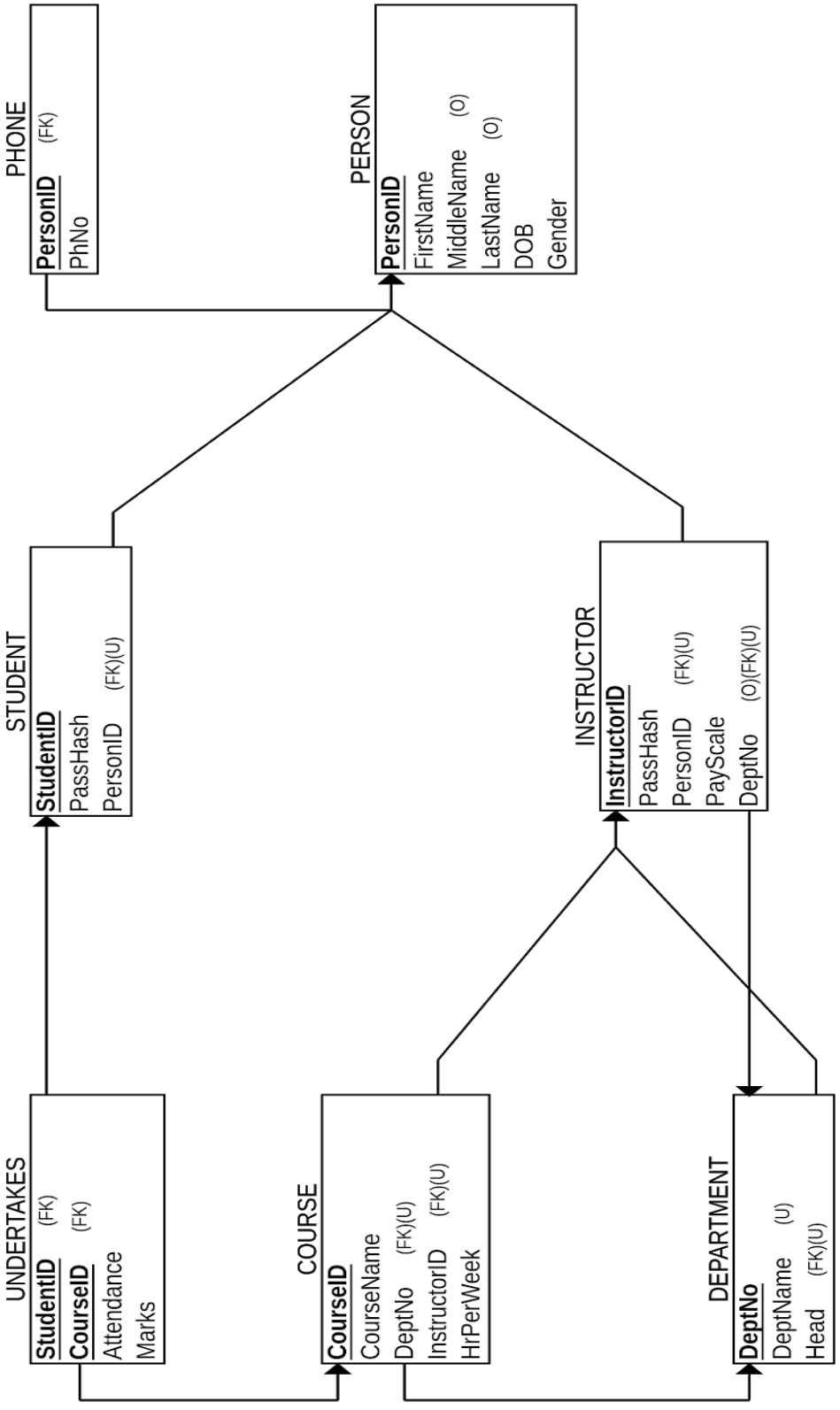
Assumptions

- Every department has a unique *Department Numbe*r and *Department Name*.
- Every course has a globally unique *Course ID* and *Course Name*.
- A person in general (instructor or student) has a *First Name*, *Middle Name* (optional), *Last Name* (optional), *Date of Birth*, and *Gender*.
- A person may have one or more *Phone Numbers*.
- Every instructor, in addition, has a unique *Instructor ID* and *Pay Scale*.
- Every student, in addition, has a unique *Student ID*.
- Both instructors and students have a saved *Password Hash* required for login.
- Every course has a count of the *Hours per Week* it is taken by its instructor.
- Every course associated with a student has a record of the *Attendance* and *Marks* obtained by the student in that course.

ER Diagram **(one generalisation included)**



Relational Schema



Tables

DEPARTMENT

DeptNo	DeptName	**Head**

PERSON

PersonID	FirstName	MiddleName	LastName	DOB	Gender

PHONE

PersonID	PhNo

INSTRUCTOR

InstructorID	PassHash	**PersonID**	PayScale	**DeptNo**

COURSE

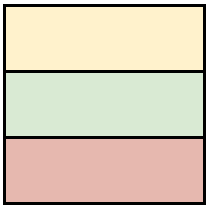
CourseID	CourseName	**DeptNo**	**InstructorID**	HrPerWeek

STUDENT

StudentID	PassHash	**PersonID**

UNDERTAKES

StudentID	**CourseID**	Attendance	Marks



Primary Key

Foreign Key

Both

Normalization

First Normal Form

In all tables, every attribute is atomic and takes only one value from its domain. Hence all are in their First Normal Form (1NF).

Second Normal Form

Super keys of the tables are as:

Table	Super Key
DEPARTMENT	DeptNo
PERSON	PersonID
PHONE	PersonID
INSTRUCTOR	InstructorID
COURSE	CourseID
STUDENT	StudentID
UNDERTAKES	StudentID, CourseID

In all but the last table, there is only one attribute forming the super key. Hence, in all such tables, the super key is the candidate key. Therefore, the relations are in their Second Normal Form (2NF).

In table *UNDERTAKES*, the functional dependencies are:

(*StudentID*, *CourseID*) $\longrightarrow$ *Attendance*

(*StudentID*, *CourseID*) $\longrightarrow$ *Marks*

Removing either StudentID or CourseID cannot identify the Attendance and Marks uniquely. Hence, the super key is the candidate key. Therefore, the relation is in its Second Normal Form (2NF).

Third Normal Form

The functional dependencies are:

DEPARTMENT:

DeptNo $\longrightarrow$ *DeptNo*

DeptNo $\longrightarrow$ *DeptName*

DeptNo $\longrightarrow$ *Head*

DeptName $\longrightarrow$ *Head*

We can remove the transitive dependency *DeptNo* $\longrightarrow$ *DeptName* $\longrightarrow$ *Head*.

Hence, new FDs are:

DeptNo $\longrightarrow$ *DeptNo*

DeptNo $\longrightarrow$ *DeptName*

DeptNo $\longrightarrow$ *Head*

PERSON:

PersonID $\longrightarrow$ *PersonID*

PersonID $\longrightarrow$ *FirstName*

PersonID $\longrightarrow$ *MiddleName*

PersonID $\longrightarrow$ *LastName*

PersonID $\longrightarrow$ *DOB*

PersonID $\longrightarrow$ *Gender*

There are no transitive dependencies; relation is in its Third Normal Form (3NF).

PHONE:

PersonID $\longrightarrow$ *PersonID*

PersonID $\longrightarrow$ *PhNo*

There are no transitive dependencies; relation is in its Third Normal Form (3NF).

INSTRUCTOR:

InstructorID ⟶ *InstructorID*

InstructorID ⟶ *PassHash*

InstructorID ⟶ *PersonID*

InstructorID ⟶ *PayScale*

InstructorID ⟶ *DeptNo*

There are no transitive dependencies; relation is in its Third Normal Form (3NF).

COURSE:

CourseID ⟶ *CourseID*

CourseID ⟶ *CourseName*

CourseID ⟶ *DeptNo*

CourseID ⟶ *InstructorID*

CourseID ⟶ *HrPerWeek*

There are no transitive dependencies; relation is in its Third Normal Form (3NF).

STUDENT:

StudentID ⟶ *StudentID*

StudentID ⟶ *PassHash*

StudentID ⟶ *PersonID*

There are no transitive dependencies; relation is in its Third Normal Form (3NF).

UNDERTAKES:

(*StudentID*, *CourseID*) ⟶ *StudentID, CourseID*

(*StudentID*, *CourseID*) ⟶ *Attendance*

(*StudentID*, *CourseID*) ⟶ *Marks*

There are no transitive dependencies; relation is in its Third Normal Form (3NF).

Boyce-Codd Normal Form

In all tables, for all functional dependencies, the L.H.S. is a super key. Hence, all relations are in their Boyce-Codd Normal Form (BCNF).

Normalised Tables

DEPARTMENT

DeptNo	DeptName

HEAD

DeptNo	**Head**

PERSON

PersonID	FirstName	MiddleName	LastName	DOB	Gender

PHONE

PersonID	PhNo

INSTRUCTOR

InstructorID	PassHash	**PersonID**	PayScale	**DeptNo**

COURSE

CourseID	CourseName	**DeptNo**	**InstructorID**	HrPerWeek

STUDENT

StudentID	PassHash	**PersonID**

UNDERTAKES

StudentID	**CourseID**	Attendance	Marks

	Primary Key		Foreign Key		Both

Proposed Application

Objectives

- Each user can login using respective ID and password.
- Privileges:
  - Administrator can **edit** *all information* about departments, courses, instructors, and students.
  - Instructors can **edit** *attendance* and *marks* of students in a course.
  - Students can **view** their *attendance* and *marks* in each course.

Application

The application opens up to a login screen. Users enter their respective ID, password, and the category (administrator, instructor, student) they belong to.

Each kind of user has their respective page. The student's page lists information about the student as listed in *Person* table, and followingly lists information about his/her undertaken courses, attendance, and the marks obtained in it.

The instructor's home page is similar to the students'. Apart from his/her information, it lists information about the courses he instructs. For every course there is a button to add attendance and marks of students in it. Clicking on such one button directs to list of students in the respective course, followed by fields to enter the attendance, or marks obtained per student.

The administrator's page opens up to a few buttons controlling the state of the database (edit "*XYZ*", or reset database). Editing is similar as instructors'. There is a respective list and corresponding fields. The administrator gets a query ability as well. He/she can look up discrete and derived information about departments, courses, instructors, and students.