Experiment 1: Entity-Relationship (ER) Diagram

🎯 Objective:

To understand and apply the concepts of ER modeling by creating an ER diagram for a real-world application.

📚 Purpose:

The purpose of this workshop is to gain hands-on experience in designing ER diagrams that visually represent the structure of a database including entities, relationships, attributes, and constraints.

🧪 Choose One Scenario:

🔹 Scenario 1: University Database

Design a database to manage students, instructors, programs, courses, and student enrollments. Include prerequisites for courses.

User Requirements:

- Academic programs grouped under departments.
- Students have admission number, name, DOB, contact info.
- Instructors with staff number, contact info, etc.
- Courses have number, name, credits.
- Track course enrollments by students and enrollment date.
- Add support for prerequisites (some courses require others).

🔹 Scenario 2: Hospital Database

Design a database for patient management, appointments, medical records, and billing.

User Requirements:

- Patient details including contact and insurance.
- Doctors and their departments, contact info, specialization.
- Appointments with reason, time, patient-doctor link.
- Medical records with treatments, diagnosis, test results.
- Billing and payment details for each appointment.

📝 Tasks:

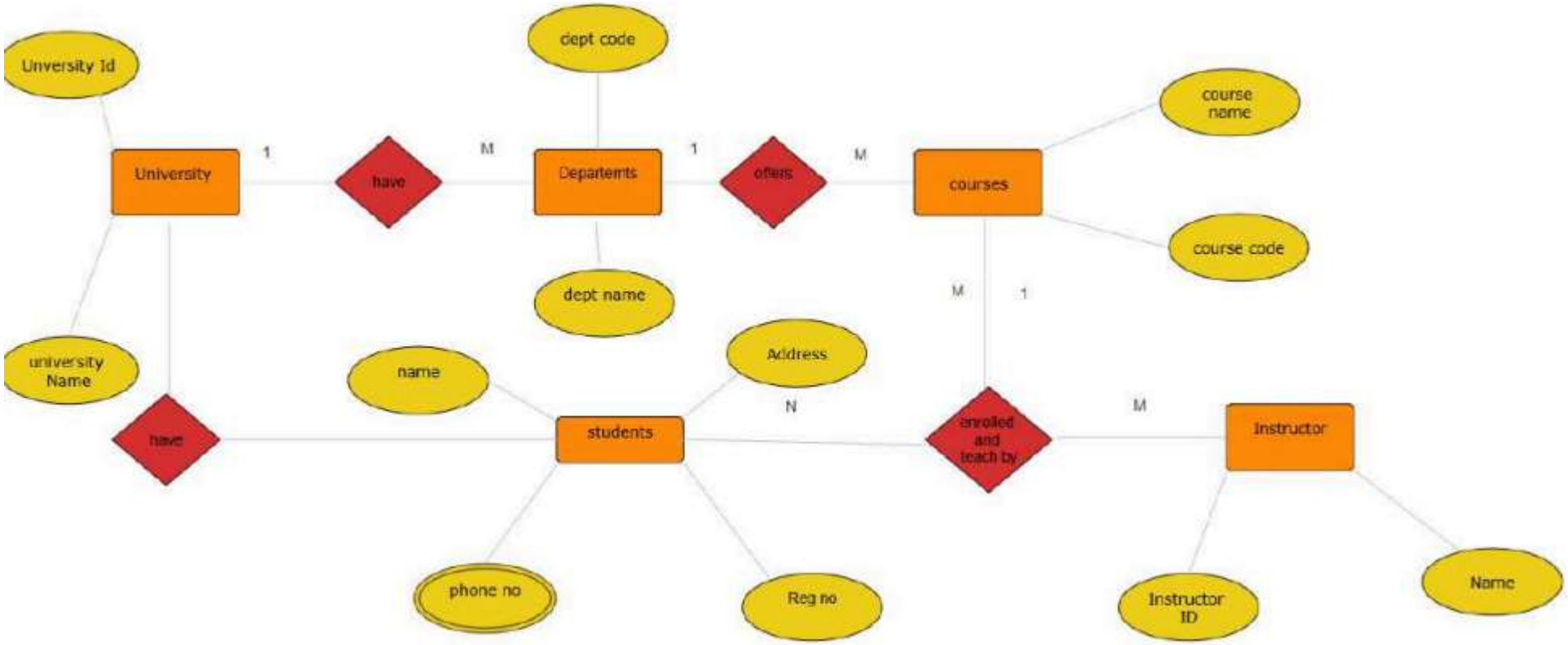
1. Identify entities, relationships, and attributes.
2. Draw the ER diagram using any tool (draw.io, dbdiagram.io, hand-drawn and scanned).
3. Include:
    - Cardinality & participation constraints
    - Prerequisites for University OR Billing for Hospital
4. Explain:
    - Why you chose the entities and relationships.
    - How you modeled prerequisites or billing.

ER Diagram Submission - Student Name : DHARANISH MS

Scenario Chosen:

University

ER Diagram:



Entities and Attributes:

- Student:Student Name,CGPA,Student Id,Semester
- Course: Course Code,Course Faculty,Course Name ...

Relationships and Constraints:

- Offers (University → Courses)
- Takes (Students → Courses) ...

Extension (Prerequisite / Billing):

Prerequisite: Not explicitly modeled in the diagram. Could be added as a recursive relationship on the Courses entity (e.g., "prerequisite_of").

Billing: Not modeled in the diagram. A possible extension could be a Billing entity with attributes like StudentID, Amount, Due Date, related to Students.

Design Choices:

Clear separation of entities: The core components (Students, Courses, University, Timetable) are logically represented.

Use of relationships: Proper use of diamond symbols to denote relationships like Takes, Has, and Offers.

Attributes placement: Most attributes are correctly associated with their respective entities, helping normalize the data structure.

Scalability: The design can be extended easily (e.g., adding Prerequisite or Billing) without disrupting the main schema.

RESULT

In this experiment, we successfully designed an Entity-Relationship (ER) diagram for a University Database System. The ER diagram identifies key entities such as University, Courses, Students, and Timetable, along with their respective attributes. The relationships among these entities—Offers, Takes, and Has—were established with appropriate cardinality and participation constraints.