SECOND SUBMISSION

CSE-202 DBMS PROJECT

Aditya Kantiwal 2023045
Devaj Rathore 2023190
Ansh Varma 2023101
Kanha Shrikant Kale 2023267

Step-by-Step ER to RS Conversion:

1. **Entity Conversion:**
   - Each entity in the ER diagram is converted into a relation (table) with its attributes.
   - The primary key of each entity is underlined.
2. **Entities and Corresponding Relations:**
   - **Courses** → `COURSES(CourseCode (PK), Name, Professor, Deadlines, TeachingAssistants, GradingScheme)`
   - **Instructors** → `INSTRUCTORS(InstructorID (PK), Name, Department)`
   - **Students** → `STUDENTS(RollNumber (PK), Branch, Batch, PredictedGrades, PredictedSGPA, CurrentCGPA, Program, TAInformation)`
3. **Many-to-Many Relationships Conversion:**
   - Many-to-many relationships are converted into separate relations (tables) with foreign keys referencing the primary keys of related entities.
   - The primary key of these relationship tables is usually a composite key.
4. **Relationships and Corresponding Relations:**
   - **Enrollment (Students enroll into Courses)**
     - `ENROLLS(RollNumber (FK → STUDENTS.RollNumber), CourseCode (FK → COURSES.CourseCode), Eligibility, Capacity, PRIMARY KEY (RollNumber, CourseCode))`
   - **Teaching (Instructors teach Courses)**

    - `TEACHES(InstructorID (FK → INSTRUCTORS.InstructorID), CourseCode (FK → COURSES.CourseCode), PRIMARY KEY (InstructorID, CourseCode))`
    - **Assigning Grades (Students receive grades for Courses)**
        - `ASSIGNS_GRADES(RollNumber (FK → STUDENTS.RollNumber), CourseCode (FK → COURSES.CourseCode), TA_Grades, StudentGrades, PRIMARY KEY (RollNumber, CourseCode))`

5. **One-to-Many and Many-to-One Relationships:**
    - The "Is TA of" relationship is captured within the `STUDENTS` table using the `TAInformation` attribute.
    - The instructor-student grading process is managed via `ASSIGNS_GRADES`, linking students, courses, and TAs.

Relational Schema

Departments

Departments(DeptID, Name)

- **DeptID**: `INT` (PK)
- **Name**: `VARCHAR(100)` NOT NULL, UNIQUE

Instructors

Instructors(InstructorID, Name, DeptID)

- **InstructorID**: `INT` (PK)
- **Name**: `VARCHAR(100)` NOT NULL
- **DeptID**: `INT`, FK → `Departments(DeptID)` ON DELETE SET NULL

Students

Students(RollNumber, Name, Branch, Batch, PredictedGrades, PredictedSGPA, CurrentCGPA)

- **RollNumber**: `VARCHAR(20)` (PK)

- **Name**: `VARCHAR(100)` NOT NULL
- **Branch**: `VARCHAR(50)` NOT NULL
- **Batch**: `YEAR`, CHECK (`Batch BETWEEN 2008 AND 2100`)
- **PredictedGrades**: `VARCHAR(255)` DEFAULT NULL
- **PredictedSGPA**: `DECIMAL(4,2)`, CHECK (`PredictedSGPA BETWEEN 0.00 AND 10.00`)
- **CurrentCGPA**: `DECIMAL(4,2)`, CHECK (`CurrentCGPA BETWEEN 0.00 AND 10.00`)

Courses

Courses(CourseCode, Name, ProfessorID, GradingScheme)

- **CourseCode**: `VARCHAR(10)` (PK)
- **Name**: `VARCHAR(100)` NOT NULL
- **ProfessorID**: `INT` NOT NULL, FK → `Instructors(InstructorID)` ON DELETE CASCADE
- **GradingScheme**: `ENUM('linear', 'gaussian', 'random')` NOT NULL

Enrolls

Enrolls(RollNumber, CourseCode, Eligibility, Capacity)

- **RollNumber**: `VARCHAR(20)` (PK, FK → `Students(RollNumber)`)
- **CourseCode**: `VARCHAR(10)` (PK, FK → `Courses(CourseCode)`)
- **Eligibility**: `BOOLEAN` DEFAULT TRUE
- **Capacity**: `INT`, CHECK (`Capacity > 0`)

Is_TA_Of

Is_TA_Of (RollNumber, CourseCode)

- **RollNumber**: `VARCHAR(20)` (PK, FK → `Students(RollNumber)`)
- **CourseCode**: `VARCHAR(10)` (PK, FK → `Courses(CourseCode)`)

Teaches

Teaches(InstructorID, CourseCode)

- **InstructorID**: `INT` (PK, FK → `Instructors(InstructorID)`)
- **CourseCode**: `VARCHAR(10)` (PK, FK → `Courses(CourseCode)`)

Assigns_Grades

Assigns_Grades(AssignerID, RollNumber, CourseCode, Grade)

- **AssignerID**: `VARCHAR(20)` (PK)
- **RollNumber**: `VARCHAR(20)` (PK, FK → `Students(RollNumber)`)
- **CourseCode**: `VARCHAR(10)` (PK, FK → `Courses(CourseCode)`)
- **Grade**: `DECIMAL(4,2)`, CHECK (`Grade BETWEEN 0.00 AND 10.00`)

GradedComponents

GradedComponents(ComponentID, CourseCode, ComponentName, StartDate, EndDate, Percentage)

- **ComponentID**: `INT` (PK, AUTO_INCREMENT)
- **CourseCode**: `VARCHAR(10)` NOT NULL, FK → `Courses(CourseCode)` ON DELETE CASCADE
- **ComponentName**: `VARCHAR(100)` NOT NULL
- **StartDate**: `DATE` DEFAULT NULL
- **EndDate**: `DATE` DEFAULT NULL
- **Percentage**: `DECIMAL(5,2)` NOT NULL, CHECK (`Percentage > 0 AND Percentage <= 100`)

StudentComponentScores

StudentComponentScores(RollNumber, ComponentID, Score)

- **RollNumber**: `VARCHAR(20)` (PK, FK → `Students(RollNumber)`)
- **ComponentID**: `INT` (PK, FK → `GradedComponents(ComponentID)`)
- **Score**: `DECIMAL(5,2)` NOT NULL, CHECK (`Score >= 0`)

THANK YOU