Problem 2: University Examination System

Design an Entity-Relationship schema for a university examination system that manages data about exams, students, faculty members, courses, and departments.
Each department has a unique name and is headed by a faculty member. A department can offer multiple courses, and each course has a unique course code, title, and is coordinated by a faculty member. Faculty members have an employee ID, name, and designation. They can teach multiple courses, coordinate specific courses, and also serve as heads of departments. A faculty member may handle multiple roles at once.
Students have a roll number and name, and each student belongs to one department. A student can enroll in multiple courses offered by that department. For each enrolled course, a student has an attendance percentage recorded.
Exams are created by faculty members. Each exam has a title, subject name (which is assumed to be the same as the course name), duration, date, type (internal or external), and is always linked to a specific course. Students may appear in multiple exams related to their courses, and for each exam, a student may have multiple attempts, with marks and attempt dates recorded for each.
All relationships between students, courses, faculty, and exams must reflect these associations clearly-such as student-course enrollment, faculty-course teaching, course-department mapping, and exam-course ownership.

1. Department Table

```
CREATE TABLE Department (
    department_id INT PRIMARY KEY AUTO_INCREMENT,
    department_name VARCHAR(255) NOT NULL UNIQUE,
    head_faculty_id INT UNIQUE, -- A faculty member heads at most one department
    CONSTRAINT fk_head_faculty FOREIGN KEY (head_faculty_id) REFERENCES
FacultyMember(employee_id)
);
```

2. FacultyMember Table

```
CREATE TABLE FacultyMember (
    employee_id INT PRIMARY KEY AUTO_INCREMENT,
```

```
    name VARCHAR(255) NOT NULL,
    designation VARCHAR(255)
);
```

3. Course Table

```
CREATE TABLE Course (
    course_code VARCHAR(50) PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    department_id INT NOT NULL,
    coordinator_faculty_id INT NOT NULL, -- Each course is coordinated by one faculty member
    CONSTRAINT fk_course_department FOREIGN KEY (department_id) REFERENCES
Department(department_id),
    CONSTRAINT fk_course_coordinator FOREIGN KEY (coordinator_faculty_id)
REFERENCES FacultyMember(employee_id)
);
```

4. Student Table

```
CREATE TABLE Student (
    roll_number VARCHAR(50) PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    department_id INT NOT NULL,
    CONSTRAINT fk_student_department FOREIGN KEY (department_id) REFERENCES
Department(department_id)
);
```

2. Exam Table

```
CREATE TABLE Exam (
    exam_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(255) NOT NULL,
    subject_name VARCHAR(255), -- Assumed to be same as course name
    duration_minutes INT,
    exam_date DATE NOT NULL,
    exam_type VARCHAR(50), -- e.g., 'Internal', 'External'
    course_code VARCHAR(50) NOT NULL,
    creator_faculty_id INT NOT NULL,
    CONSTRAINT fk_exam_course FOREIGN KEY (course_code) REFERENCES
Course(course_code),
```

```
    CONSTRAINT fk_exam_creator FOREIGN KEY (creator_faculty_id) REFERENCES
FacultyMember(employee_id),
    CONSTRAINT chk_exam_type CHECK (exam_type IN ('Internal', 'External'))
);
```

EXTRA DATA :

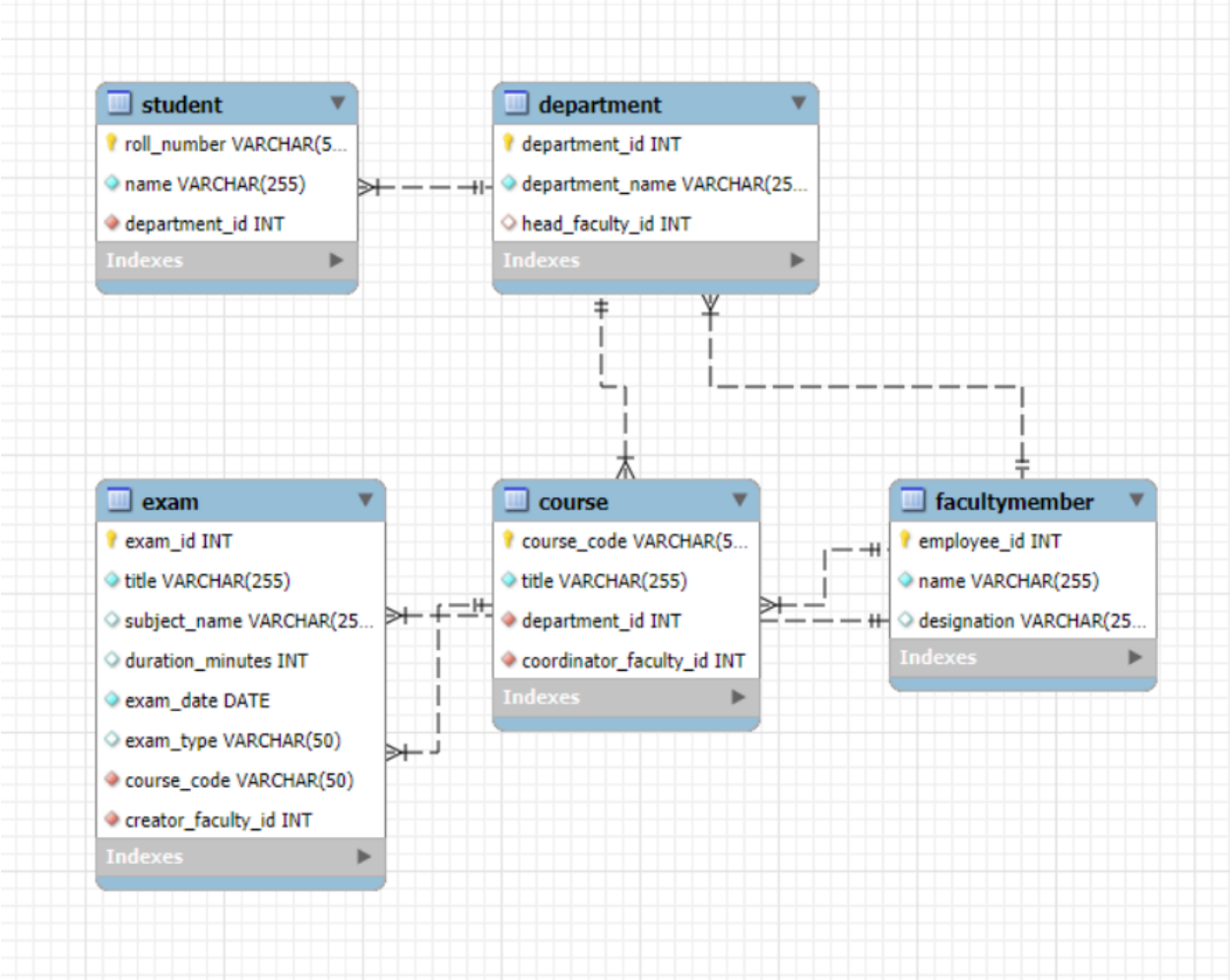
```
CREATE TABLE FacultyTeachesCourse (
    faculty_id INT,
    course_code VARCHAR(50),
    PRIMARY KEY (faculty_id, course_code),
    CONSTRAINT fk_teaches_faculty FOREIGN KEY (faculty_id) REFERENCES
FacultyMember(employee_id),
    CONSTRAINT fk_teaches_course FOREIGN KEY (course_code) REFERENCES
Course(course_code)
);


CREATE TABLE Enrollment (
    student_id VARCHAR(50),
    course_code VARCHAR(50),
    enrollment_date DATE NOT NULL,
    attendance_percentage DECIMAL(5,2) NOT NULL,
    PRIMARY KEY (student_id, course_code),
    CONSTRAINT fk_enrollment_student FOREIGN KEY (student_id) REFERENCES
Student(roll_number),
    CONSTRAINT fk_enrollment_course FOREIGN KEY (course_code) REFERENCES
Course(course_code),
    CONSTRAINT chk_attendance_percentage CHECK (attendance_percentage BETWEEN 0
AND 100)
);


CREATE TABLE ExamAttempt (
    student_id VARCHAR(50),
    exam_id INT,
    attempt_number INT, -- For multiple attempts by the same student on the same exam
    attempt_date DATE NOT NULL,
    marks DECIMAL(5,2),
    PRIMARY KEY (student_id, exam_id, attempt_number),
    CONSTRAINT fk_attempt_student FOREIGN KEY (student_id) REFERENCES
Student(roll_number),
    CONSTRAINT fk_attempt_exam FOREIGN KEY (exam_id) REFERENCES Exam(exam_id),
```

```
    CONSTRAINT chk_attempt_number CHECK (attempt_number > 0)
);
```

MAIN TABLE :



WITH EXTRA DATA :