Problem 2: University Examination System

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.

Faculty Table - - - -

```
CREATE TABLE Faculty (
    employee_id INT PRIMARY KEY,
    name VARCHAR(100),
    designation VARCHAR(50)
);
```

Department Table - - - -

```
CREATE TABLE Department (
    dept_name VARCHAR(100) PRIMARY KEY,
    head_id INT,
    FOREIGN KEY (head_id) REFERENCES Faculty(employee_id));
```

Course Table - - - -

```
CREATE TABLE Course (
    course_code VARCHAR(20) PRIMARY KEY,
    title VARCHAR(100),
    dept_name VARCHAR(100),
    coordinator_id INT,
    FOREIGN KEY (dept_name) REFERENCES Department(dept_name),
    FOREIGN KEY (coordinator_id) REFERENCES Faculty(employee_id)
);
```

Student Table - - - -

```
CREATE TABLE Student (
    roll_number INT PRIMARY KEY,
    name VARCHAR(100),
    dept_name VARCHAR(100),
    FOREIGN KEY (dept_name) REFERENCES Department(dept_name)
);
```

Exam Table - - - -

```
CREATE TABLE Exam (
    exam_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(100),
    subject_name VARCHAR(100),
    duration INT,
    date DATE,
    type ENUM('internal', 'external'),
    course_code VARCHAR(20),
    creator_id INT,
    FOREIGN KEY (course_code) REFERENCES Course(course_code),
    FOREIGN KEY (creator_id) REFERENCES Faculty(employee_id)
);
```

Teacher Table - - - -

```
CREATE TABLE Teaches (
```

```
    employee_id INT,
    course_code VARCHAR(20),
    PRIMARY KEY (employee_id, course_code),
    FOREIGN KEY (employee_id) REFERENCES Faculty(employee_id),
    FOREIGN KEY (course_code) REFERENCES Course(course_code)
);
```

Enrollment Table - - - -

```
CREATE TABLE Enrollment (
    roll_number INT,
    course_code VARCHAR(20),
    attendance_percentage DECIMAL(5,2),
    PRIMARY KEY (roll_number, course_code),
    FOREIGN KEY (roll_number) REFERENCES Student(roll_number),
    FOREIGN KEY (course_code) REFERENCES Course(course_code)
);
```

Attempts Table - - - -

```
CREATE TABLE Attempts (
    roll_number INT,
    exam_id INT,
```

```
    attempt_no INT,
    marks DECIMAL(5,2),
    attempt_date DATE,
    PRIMARY KEY (roll_number, exam_id, attempt_no),
    FOREIGN KEY (roll_number) REFERENCES Student(roll_number),
    FOREIGN KEY (exam_id) REFERENCES Exam(exam_id)
);
```

ER Diagram - - - -

Problem 2: University Examination System

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.

Faculty Table - - - -

```
CREATE TABLE Faculty (
    employee_id INT PRIMARY KEY,
    name VARCHAR(100),
    designation VARCHAR(50)
);
```

Department Table - - - -

```
CREATE TABLE Department (
    dept_name VARCHAR(100) PRIMARY KEY,
```

```
    head_id INT,
    FOREIGN KEY (head_id) REFERENCES Faculty(employee_id));
```

Course Table - - - -

```
CREATE TABLE Course (
    course_code VARCHAR(20) PRIMARY KEY,
    title VARCHAR(100),
    dept_name VARCHAR(100),
    coordinator_id INT,
    FOREIGN KEY (dept_name) REFERENCES Department(dept_name),
    FOREIGN KEY (coordinator_id) REFERENCES Faculty(employee_id)
);
```

Student Table - - - -

```
CREATE TABLE Student (
    roll_number INT PRIMARY KEY,
    name VARCHAR(100),
    dept_name VARCHAR(100),
    FOREIGN KEY (dept_name) REFERENCES Department(dept_name)
);
```

Exam Table - - - -

```
CREATE TABLE Exam (
```

```
    exam_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(100),
    subject_name VARCHAR(100),
    duration INT,
    date DATE,
    type ENUM('internal', 'external'),
    course_code VARCHAR(20),
    creator_id INT,
    FOREIGN KEY (course_code) REFERENCES Course(course_code),
    FOREIGN KEY (creator_id) REFERENCES Faculty(employee_id)
);
```

Teacher Table - - - -

```
CREATE TABLE Teaches (
    employee_id INT,
    course_code VARCHAR(20),
    PRIMARY KEY (employee_id, course_code),
    FOREIGN KEY (employee_id) REFERENCES Faculty(employee_id),
    FOREIGN KEY (course_code) REFERENCES Course(course_code)
);
```

Enrollment Table - - - -

```
CREATE TABLE Enrollment (
    roll_number INT,
    course_code VARCHAR(20),
    attendance_percentage DECIMAL(5,2),
    PRIMARY KEY (roll_number, course_code),
    FOREIGN KEY (roll_number) REFERENCES Student(roll_number),
    FOREIGN KEY (course_code) REFERENCES Course(course_code)
);
```

Attempts Table - - - -

```
CREATE TABLE Attempts (
    roll_number INT,
    exam_id INT,
    attempt_no INT,
    marks DECIMAL(5,2),
    attempt_date DATE,
    PRIMARY KEY (roll_number, exam_id, attempt_no),
    FOREIGN KEY (roll_number) REFERENCES Student(roll_number),
    FOREIGN KEY (exam_id) REFERENCES Exam(exam_id)
);
```

ER Diagram - - - -

Problem 2: University Examination System

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.

Faculty Table - - - -

```
CREATE TABLE Faculty (
    employee_id INT PRIMARY KEY,
    name VARCHAR(100),
    designation VARCHAR(50)
);
```

Department Table - - - -

```
CREATE TABLE Department (
    dept_name VARCHAR(100) PRIMARY KEY,
    head_id INT,
    FOREIGN KEY (head_id) REFERENCES Faculty(employee_id));
```

Course Table - - - -

```
CREATE TABLE Course (
    course_code VARCHAR(20) PRIMARY KEY,
    title VARCHAR(100),
    dept_name VARCHAR(100),
    coordinator_id INT,
    FOREIGN KEY (dept_name) REFERENCES Department(dept_name),
    FOREIGN KEY (coordinator_id) REFERENCES Faculty(employee_id)
```

```
);
```

Student Table - - - -

```
CREATE TABLE Student (
    roll_number INT PRIMARY KEY,
    name VARCHAR(100),
    dept_name VARCHAR(100),
    FOREIGN KEY (dept_name) REFERENCES Department(dept_name)
);
```

Exam Table - - - -

```
CREATE TABLE Exam (
    exam_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(100),
    subject_name VARCHAR(100),
    duration INT,
    date DATE,
    type ENUM('internal', 'external'),
    course_code VARCHAR(20),
    creator_id INT,
    FOREIGN KEY (course_code) REFERENCES Course(course_code),
    FOREIGN KEY (creator_id) REFERENCES Faculty(employee_id)
);
```

Teacher Table - - - -

```
CREATE TABLE Teaches (
    employee_id INT,
    course_code VARCHAR(20),
    PRIMARY KEY (employee_id, course_code),
    FOREIGN KEY (employee_id) REFERENCES Faculty(employee_id),
    FOREIGN KEY (course_code) REFERENCES Course(course_code)
);
```

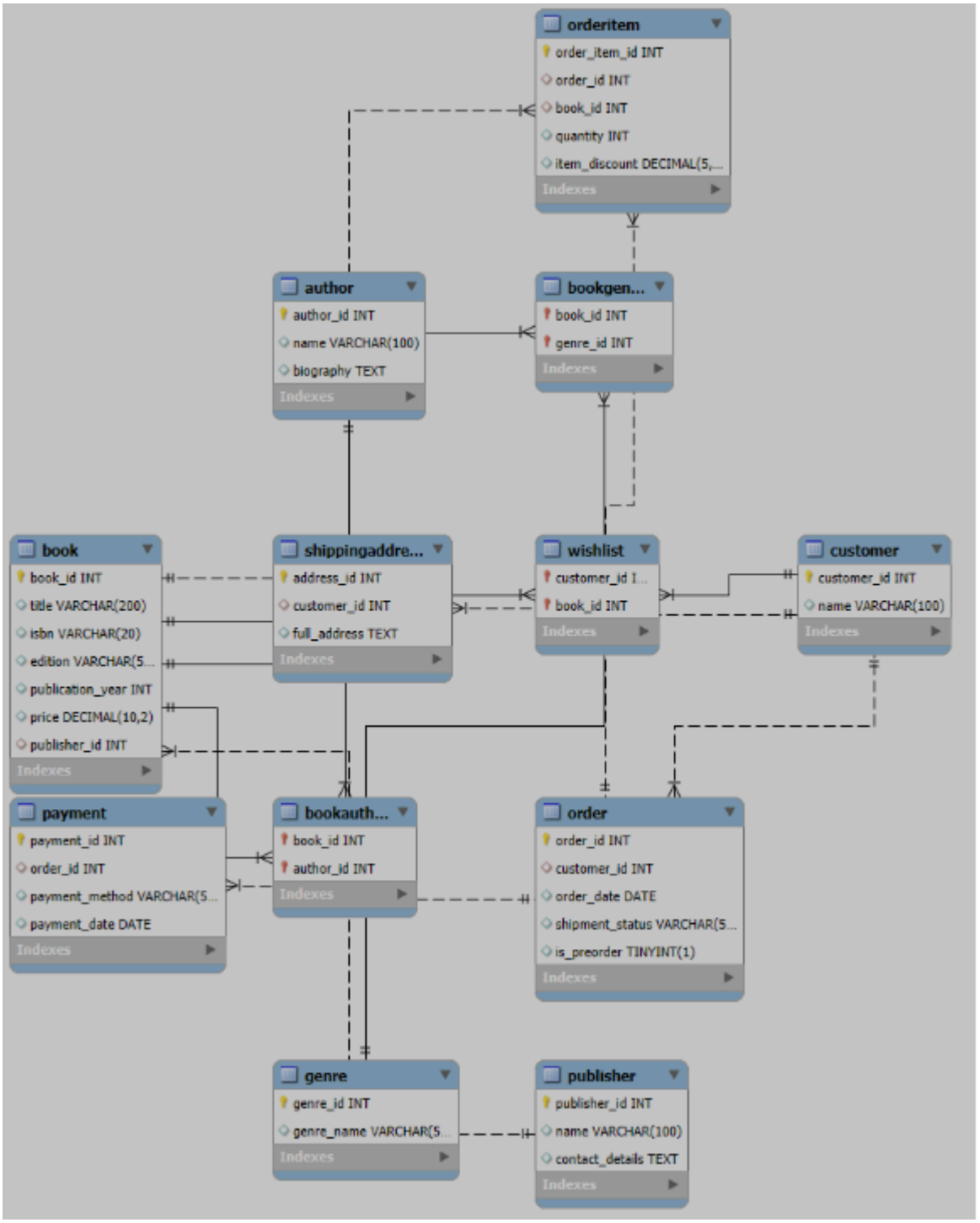
Enrollment Table - - - -

```
CREATE TABLE Enrollment (
    roll_number INT,
    course_code VARCHAR(20),
    attendance_percentage DECIMAL(5,2),
    PRIMARY KEY (roll_number, course_code),
    FOREIGN KEY (roll_number) REFERENCES Student(roll_number),
    FOREIGN KEY (course_code) REFERENCES Course(course_code)
);
```

Attempts Table - - - -

```
CREATE TABLE Attempts (
    roll_number INT,
    exam_id INT,
    attempt_no INT,
    marks DECIMAL(5,2),
    attempt_date DATE,
    PRIMARY KEY (roll_number, exam_id, attempt_no),
    FOREIGN KEY (roll_number) REFERENCES Student(roll_number),
    FOREIGN KEY (exam_id) REFERENCES Exam(exam_id)
);
```

ER Diagram - - - -


orderitem
order_item_id INT
order_id INT
book_id INT
quantity INT
item_discount DECIMAL(5,...
Indexes
author
author_id INT
name VARCHAR(100)
biography TEXT
Indexes
bookgen...
book_id INT
genre_id INT
Indexes
book
book_id INT
title VARCHAR(200)
isbn VARCHAR(20)
edition VARCHAR(5...
publication_year INT
price DECIMAL(10,2)
publisher_id INT
Indexes
shippingaddre...
address_id INT
customer_id INT
full_address TEXT
Indexes
wishlist
customer_id I...
book_id INT
Indexes
customer
customer_id INT
name VARCHAR(100)
Indexes
payment
payment_id INT
order_id INT
payment_method VARCHAR(5...
payment_date DATE
Indexes
bookauth...
book_id INT
author_id INT
Indexes
order
order_id INT
customer_id INT
order_date DATE
shipment_status VARCHAR(5...
is_preorder TINYINT(1)
Indexes
genre
genre_id INT
genre_name VARCHAR(5...
Indexes
publisher
publisher_id INT
name VARCHAR(100)
contact_details TEXT
Indexes