Database Development

Introduction:

The database design for 'Your Center for Skill Enhancement' is essential for maintaining structured, efficient, and scalable data management. It involves multiple interrelated entities such as users, courses, enrollments, progress, and certification. Each entity is represented through a dedicated table with specific attributes and relational keys to ensure data consistency and integrity.

Core Tables and Their Purpose:

1. users:

   - Stores all user-related data (admin, teacher, student).

   - Fields: user_id (PK), name, email, password_hash, role, created_at

2. courses:

   - Contains course data created by teachers.

   - Fields: course_id (PK), teacher_id (FK), title, category, description, price, created_at

3. sections:

   - Divides courses into sections for modular content.

   - Fields: section_id (PK), course_id (FK), title, content, order

4. enrollments:

   - Tracks student enrollments and payment status.

   - Fields: enrollment_id (PK), student_id (FK), course_id (FK), enrolled_on, payment_status

5. progress:

   - Records student's current progress in a course.

- Fields: progress_id (PK), student_id (FK), course_id (FK), section_id (FK), status, updated_at

6. payments:

- Manages financial transactions for paid courses.

- Fields: payment_id (PK), student_id (FK), course_id (FK), amount, payment_date, status

7. certificates:

- Stores completion certificates for students.

- Fields: certificate_id (PK), student_id (FK), course_id (FK), issue_date, download_link

Entity Relationships:

- One teacher can create many courses (1:N).

- Each course has multiple sections (1:N).

- Students can enroll in many courses, and a course can have many students (M:N via enrollments).

- A student can track progress section-wise (M:N).

- Payments are linked to both students and courses (M:N).

- Certificates are issued once a course is completed by a student (1:1 or 1:N depending on retakes).

Optimization Strategies:

- Normalize the schema up to 3NF to eliminate data redundancy.

- Use proper indexing on foreign keys for faster JOIN operations.

- Utilize ENUMs for fixed types like user roles and payment status.

- Archive old certificate and payment records for performance.

- Implement triggers or stored procedures for data validation or cascading deletions.

Security Considerations:

- Passwords must be hashed using secure algorithms (e.g., bcrypt).

- Use role-based access control (RBAC) to restrict data access.

- Validate all input to prevent SQL injection attacks.

- Use HTTPS for database-connected APIs to secure data in transit.

Conclusion:

An efficient database is the backbone of the Center for Skill Enhancement. This structure ensures smooth operations, enables scalability, supports analytics, and maintains data integrity throughout the platform's lifecycle.