MySQL Comprehensive Exercise

Objective: Build and Manage a University Database

Step 1: Database Design

1. Create the following tables with appropriate data types and relationships:

   - Students: student_id, first_name, last_name, email, date_of_birth, gender, major, enrollment_year

   - Courses: course_id, course_name, course_code, credits, department

   - Instructors: instructor_id, first_name, last_name, email, hire_date, department

   - Enrollments: enrollment_id, student_id, course_id, grade

   - Course Assignments: assignment_id, instructor_id, course_id, semester, year

Step 2: Insert Data

Insert at least 10 students, 5 instructors, and 5 courses. Make sure courses are assigned to instructors, and students are enrolled in at least 2 courses each.

Step 3: Queries

Basic Queries:

1. List all students along with their details.

2. Find the total number of courses offered by the university.

3. Show the names of all students enrolled in a specific course.

4. Retrieve the email addresses of all instructors in a department.

Intermediate Queries:

5. List all courses along with the number of students enrolled.

6. Find the students who were enrolled in a course with a grade of 'A'.

7. Retrieve the courses and the instructors assigned for a specific semester.

8. Find the average grade for a particular course.

Advanced Queries:

9. List students taking more than 3 courses in the current semester.

10. Generate a report of students with incomplete grades.

11. Show the student with the highest average grade across courses.

12. Find the department with the most courses taught this year.

13. List courses with no student enrollments.

Step 4: Functions and Aggregates

1. Create a function to calculate a student's age based on date_of_birth.

2. Create a stored procedure to enroll a student in a course.

3. Use aggregate functions to show average grades by department.

Step 5: Constraints and Transactions

1. Add a constraint to ensure unique student emails.

2. Write a transaction to enroll a student if the course capacity isn't exceeded.

Step 6: Optimization and Indexes

1. Create an index on the course_code to speed up searches.

2. Optimize a query using EXPLAIN to fetch students enrolled in a course.

Step 7: Joins

1. Write an inner join to fetch students and the courses they are enrolled in.

2. Write a left join to show instructors and courses they teach.

3. Write a query using union to list all students and instructors.

Step 8: Final Challenge

Generate a report showing each student's name, email, major, courses enrolled, instructor, grades, and total credits.