SQL queries with explanations

1. Database Structure and Table Descriptions

a. Students Table

- **Purpose**: The **Students Table** stores all personal information about the students enrolled in the university.
- **Columns**:
  - `student_id`: Unique identifier for each student (auto-incremented integer).
  - `first_name`: The first name of the student (string).
  - `last_name`: The last name of the student (string).
  - `date_of_birth`: The birth date of the student (date).
  - `email`: A unique email address for the student (string). It must be unique across all students to prevent duplication.

  Key Constraints:

  - `student_id` is the primary key, meaning it uniquely identifies each student.
  - `email` is unique, ensuring no two students share the same email address.

b. Departments Table

- **Purpose**: The **Departments Table** stores information about the different academic departments within the university.
- **Columns**:
  - `department_id`: Unique identifier for each department (auto-incremented integer).
  - `name`: The name of the department (e.g., Computer Science, Mechanical Engineering, etc.).

  Key Constraints:

  - `department_id` is the primary key, uniquely identifying each department.

c. Faculty Table

- **Purpose**: The **Faculty Table** stores information about faculty members, including their personal details and the department they belong to.
- **Columns**:

- `faculty_id`: Unique identifier for each faculty member (auto-incremented integer).
- `first_name`: The first name of the faculty member (string).
- `last_name`: The last name of the faculty member (string).
- `email`: A unique email address for the faculty member (string).
- `department_id`: Foreign key referencing the **Departments Table** that links each faculty member to a specific department.

Key Constraints:

- `faculty_id` is the primary key.
- `email` is unique across all faculty members.
- `department_id` is a foreign key that references the `department_id` in the **Departments Table**.
- A **faculty member** can belong to only one department at a time.

d. Courses Table

- **Purpose**: The **Courses Table** stores information about the courses offered at the university. Each course is linked to a faculty member who teaches it.
- **Columns**:
  - `course_id`: Unique identifier for each course (auto-incremented integer).
  - `code`: The course code (e.g., CS101, MATH202).
  - `title`: The title or name of the course (string).
  - `credits`: The number of credits associated with the course (integer).
  - `faculty_id`: Foreign key that references the **Faculty Table** to identify the faculty member teaching the course.

Key Constraints:

- `course_id` is the primary key.
- `faculty_id` is a foreign key referencing the `faculty_id` in the **Faculty Table**.

e. Enrollments Table

- **Purpose**: The **Enrollments Table** tracks the enrollment of students in courses. It links students to the courses they are taking and stores their grades.
- **Columns**:
  - `enrollment_id`: Unique identifier for each enrollment record (auto-incremented integer).
  - `student_id`: Foreign key referencing the **Students Table** to identify which student is enrolled.
  - `course_id`: Foreign key referencing the **Courses Table** to identify which course the student is enrolled in.
  - `enrollment_date`: The date when the student enrolled in the course (date).
  - `grade`: The grade the student received for the course (string or NULL).

Key Constraints:

- `enrollment_id` is the primary key.
- `student_id` and `course_id` are foreign keys referencing the `student_id` in the **Students Table** and the `course_id` in the **Courses Table**.
- The `grade` field can hold one of the predefined grades ('A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'D', 'F') or NULL (if the grade is not yet assigned).
- The `student_id` and `course_id` together create a composite relationship that ensures each student can enroll in multiple courses, but the combination of student and course is unique.

Foreign Key Constraints:

- **ON DELETE CASCADE**: When a student is deleted from the system, their corresponding enrollments will also be deleted.
- **ON DELETE RESTRICT**: If a course is deleted, its enrollments cannot be deleted, ensuring that enrollments tied to a course are preserved for historical purposes.

2. Data Insertion Overview

The following is a summary of the data insertion steps:

1. **Insert Students**: Six students are added to the **Students Table**, each with a unique first name, last name, date of birth, and email address.
2. **Insert Departments**: Six departments are inserted into the **Departments Table**, including fields such as 'Computer Science', 'Electrical Engineering', etc.
3. **Insert Faculty Members**: Six faculty members are inserted into the **Faculty Table**, each associated with a department via `department_id`.
4. **Insert Courses**: Six courses are added to the **Courses Table**, each linked to a faculty member.
5. **Insert Enrollments**: Enrollment records are added to the **Enrollments Table**, indicating which students are enrolled in which courses, the dates of enrollment, and their grades.

3. Indexes

Indexes are created on the **enrollments** table to speed up query performance:

- **`idx_student_id`**: This index is created on the `student_id` field, improving the performance of queries that filter by student.
- **`idx_course_id`**: This index is created on the `course_id` field, improving the performance of queries that filter by course.

4. Queries Overview

Here are some important queries that can be run on the database:

1. **Retrieve Students Enrolled in a Specific Course**: This query finds all students who are enrolled in a specific course, including their first and last names, enrollment date, and grade.
2. **Find Faculty in a Specific Department**: This query retrieves all faculty members who belong to a specific department, showing their name and email.
3. **List All Courses for a Student**: This query retrieves a list of all courses a student is enrolled in, along with the course details and grades.
4. **Find Students Not Enrolled in Any Courses**: This query retrieves all students who have not enrolled in any courses.
5. **Find Average Grade for a Specific Course**: This query calculates the average grade for a course, using numerical values for each grade ('A' = 4.0, 'B' = 3.0, etc.) to compute the average.

5. Summary of Relationships

- **Students and Enrollments**: Each student can be enrolled in multiple courses. The **Enrollments Table** is used to record these relationships.
- **Faculty and Courses**: Each course is taught by a faculty member, represented by a foreign key relationship in the **Courses Table**.
- **Departments and Faculty**: Each faculty member belongs to a specific department, represented by a foreign key relationship in the **Faculty Table**.
- **Students and Grades**: The **Enrollments Table** records the grade a student receives for each course they take.