DAY 2:

Create queries:

-- Students table

CREATE TABLE Students (

student\_id INT PRIMARY KEY,

student\_name VARCHAR(50),

student\_age INT,

student\_grade\_id INT,

FOREIGN KEY (student\_grade\_id) REFERENCES Grades(grade\_id)

);

-- Grades table

CREATE TABLE Grades (

grade\_id INT PRIMARY KEY,

grade\_name VARCHAR(10)

);

-- Courses table

CREATE TABLE Courses (

course\_id INT PRIMARY KEY,

course\_name VARCHAR(50)

);

-- Enrollments table

CREATE TABLE Enrollments (

enrollment\_id INT PRIMARY KEY,

student\_id INT,

course\_id INT,

enrollment\_date DATE,

FOREIGN KEY (student\_id) REFERENCES Students(student\_id),

FOREIGN KEY (course\_id) REFERENCES Courses(course\_id)

);

Insert queries:

-- Insert into Grades table

INSERT INTO Grades (grade\_id, grade\_name) VALUES

(1, 'A'),

(2, 'B'),

(3, 'C');

-- Insert into Courses table

INSERT INTO Courses (course\_id, course\_name) VALUES

(101, 'Math'),

(102, 'Science'),

(103, 'History');

-- Insert into Students table

INSERT INTO Students (student\_id, student\_name, student\_age, student\_grade\_id) VALUES

(1, 'Alice', 17, 1),

(2, 'Bob', 16, 2),

(3, 'Charlie', 18, 1),

(4, 'David', 16, 2),

(5, 'Eve', 17, 1),

(6, 'Frank', 18, 3),

(7, 'Grace', 17, 2),

(8, 'Henry', 16, 1),

(9, 'Ivy', 18, 2),

(10, 'Jack', 17, 3);

-- Insert into Enrollments table

INSERT INTO Enrollments (enrollment\_id, student\_id, course\_id, enrollment\_date) VALUES

(1, 1, 101, '2023-09-01'),

(2, 1, 102, '2023-09-01'),

(3, 2, 102, '2023-09-01'),

(4, 3, 101, '2023-09-01'),

(5, 3, 103, '2023-09-01'),

(6, 4, 101, '2023-09-01'),

(7, 4, 102, '2023-09-01'),

(8, 5, 102, '2023-09-01'),

(9, 6, 101, '2023-09-01'),

(10, 7, 103, '2023-09-01');

Questions:

1. Find all students enrolled in the Math course.
2. List all courses taken by students named Bob.
3. Find the names of students who are enrolled in more than one course.
4. List all students who are in Grade A (grade\_id = 1).
5. Find the number of students enrolled in each course.
6. Retrieve the course with the highest number of enrollments.
7. List students who are enrolled in all available courses.
8. Find students who are not enrolled in any courses.
9. Retrieve the average age of students enrolled in the Science course.
10. Find the grade of students enrolled in the History course.

**Assignment:**

Please design and create the necessary tables (Books, Authors, Publishers, Customers, Orders, Book\_Authors, Order\_Items) for an online bookstore database. Ensure each table includes appropriate columns, primary keys, and foreign keys where necessary. Consider the relationships between these tables and how they should be defined.

**Conceptual Modeling:**

1. **Identify Entities and Relationships:**
   * **Entities**:
     + Book (with attributes like book\_id, title, author, genre, publisher, publication\_year)
     + Author (with attributes like author\_id, author\_name, birth\_date, nationality)
     + Publisher (with attributes like publisher\_id, publisher\_name, country)
     + Customer (with attributes like customer\_id, customer\_name, email, address)
     + Order (with attributes like order\_id, order\_date, customer\_id, total\_amount)
   * **Relationships**:
     + Books are written by Authors (many-to-many relationship)
     + Books are published by Publishers (many-to-one relationship)
     + Customers place Orders (one-to-many relationship)
     + Orders contain Books (many-to-many relationship)

**2. Conceptual Model Representation**:

* Use an Entity-Relationship Diagram (ERD) to visually represent entities, attributes, and relationships.

**Logical Schema Design:**

1. **Translate Entities to Tables:**
   * **Tables**:
     + Books table (with columns: book\_id, title, genre, publisher\_id, publication\_year)
     + Authors table (with columns: author\_id, author\_name, birth\_date, nationality)
     + Publishers table (with columns: publisher\_id, publisher\_name, country)
     + Customers table (with columns: customer\_id, customer\_name, email, address)
     + Orders table (with columns: order\_id, order\_date, customer\_id, total\_amount)
     + Book\_Authors table (to manage the many-to-many relationship between Books and Authors)
     + Order\_Items table (to manage the many-to-many relationship between Orders and Books)
2. **Define Relationships and Constraints:**
   * **Primary Keys**:
     + book\_id in Books
     + author\_id in Authors
     + publisher\_id in Publishers
     + customer\_id in Customers
     + order\_id in Orders
   * **Foreign Keys**:
     + publisher\_id in Books references publisher\_id in Publishers
     + customer\_id in Orders references customer\_id in Customers
     + book\_id and author\_id in Book\_Authors reference book\_id and author\_id in Books and Authors, respectively
     + order\_id and book\_id in Order\_Items reference order\_id in Orders and book\_id in Books, respectively