SQL DATABASE PROJECT FOR LECTURE OF COMPUTER PROGRAMMING 2

(Grzegorz Pawłowski)

Student Name and Surname: Furkan Panayır

Student ID: 160477

Group: EM1

Semester: 3. Semester (2nd year)

Database Creation Procedure

```
CREATE DATABASE UniversityDatabase;
```

Students Table:

```
CREATE TABLE Students (
  StudentID INT AUTO_INCREMENT PRIMARY KEY,
  FirstName VARCHAR(50) NOT NULL,
  LastName VARCHAR(50) NOT NULL,
  IndexNumber VARCHAR(20) UNIQUE NOT NULL,
  YearOfStudy INT CHECK (YearOfStudy BETWEEN 1 AND 5),
  GroupName VARCHAR(20)
);
```

Courses Table:

```
CREATE TABLE Courses (
  CourseID SERIAL PRIMARY KEY,
  CourseName VARCHAR(100) NOT NULL,
  Credits INT CHECK (Credits BETWEEN 1 AND 10),
  Department VARCHAR(50)
);
```

Professors Table:

```
CREATE TABLE Professors (
```

ProfessorID SERIAL PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Department VARCHAR(50) NOT NULL

);

Enrollments Table:

CREATE TABLE Enrollments (

EnrollmentID SERIAL PRIMARY KEY,

StudentID INT NOT NULL,

CourseID INT NOT NULL,

ProfessorID INT NOT NULL,

EnrollmentDate DATE NOT NULL,

FOREIGN KEY (StudentID) REFERENCES Students(StudentID) ON DELETE CASCADE,

FOREIGN KEY (CourseID) REFERENCES Courses(CourseID) ON DELETE CASCADE,

FOREIGN KEY (ProfessorID) REFERENCES Professors(ProfessorID) ON DELETE CASCADE

);

RELATIONSHIPS:

Students → **Enrollments** (One-to-Many):

Courses → **Enrollments** (One-to-Many):

Professors → **Enrollments** (One-to-Many):

Query to Relationships:

SELECT

e.EnrollmentID,

CONCAT(s.FirstName, ' ', s.LastName) AS StudentName,

c.CourseName AS CourseName,

CONCAT(p.FirstName, ' ', p.LastName) AS ProfessorName,

```
    e.EnrollmentDate
FROM Enrollments e
JOIN Students s ON e.StudentID = s.StudentID
JOIN Courses c ON e.CourseID = c.CourseID
JOIN Professors p ON e.ProfessorID = p.ProfessorID;
```

DATA ENTRANCE:

Students Table:

```
INSERT INTO Students (FirstName, LastName, IndexNumber, YearOfStudy,
GroupName)
VALUES
('Alice', 'Smith', 'S101', 3, 'CS1'),
('Bob', 'Johnson', 'S102', 1, 'CS1'),
('Charlie', 'Brown', 'S103', 2, 'CS2'),
('Daisy', 'White', 'S104', 3, 'CS1'),
('Ella', 'Taylor', 'S105', 3, 'CS2'),
('Frank', 'Miller', 'S106', 2, 'CS1'),
('Grace', 'Wilson', 'S107', 2, 'CS3'),
('Harry', 'Davis', 'S108', 1, 'CS2'),
('Ivy', 'Clark', 'S109', 1, 'CS1'),
('Jack', 'Lewis', 'S110', 2, 'CS3');
```

Courses table:

```
INSERT INTO Courses (CourseName, Credits, Department)
VALUES
('Database Systems', 5, 'Computer Science'),
('Operating Systems', 4, 'Computer Science'),
('Data Structures', 4, 'Computer Science'),
('Artificial Intelligence', 5, 'AI Research'),
('Software Engineering', 3, 'Software Development'),
```

('Cybersecurity', 3, 'Cyber Defense');

Professors table:

```
INSERT INTO Professors (FirstName, LastName, Department)
VALUES
('Dr. Emily', 'Jones', 'Computer Science'),
('Dr. Michael', 'Taylor', 'AI Research'),
('Dr. Sarah', 'Brown', 'Software Development'),
('Dr. James', 'Wilson', 'Cyber Defense'),
('Dr. Anna', 'Davis', 'Computer Science');
```

Enrollments Table:

```
INSERT INTO Enrollments (StudentID, CourseID, ProfessorID, EnrollmentDate)
VALUES
(1, 1, 1, '2025-01-12'),
(2, 3, 2, '2025-01-15'),
(3, 2, 1, '2025-01-20'),
(4, 4, 3, '2025-01-25'),
(5, 5, 4, '2025-01-30'),
(6, 6, 5, '2025-02-01'),
(7, 1, 1, '2025-02-05'),
(8, 3, 2, '2025-02-10'),
(9, 4, 3, '2025-02-15'),
(10, 5, 4, '2025-02-20');
```

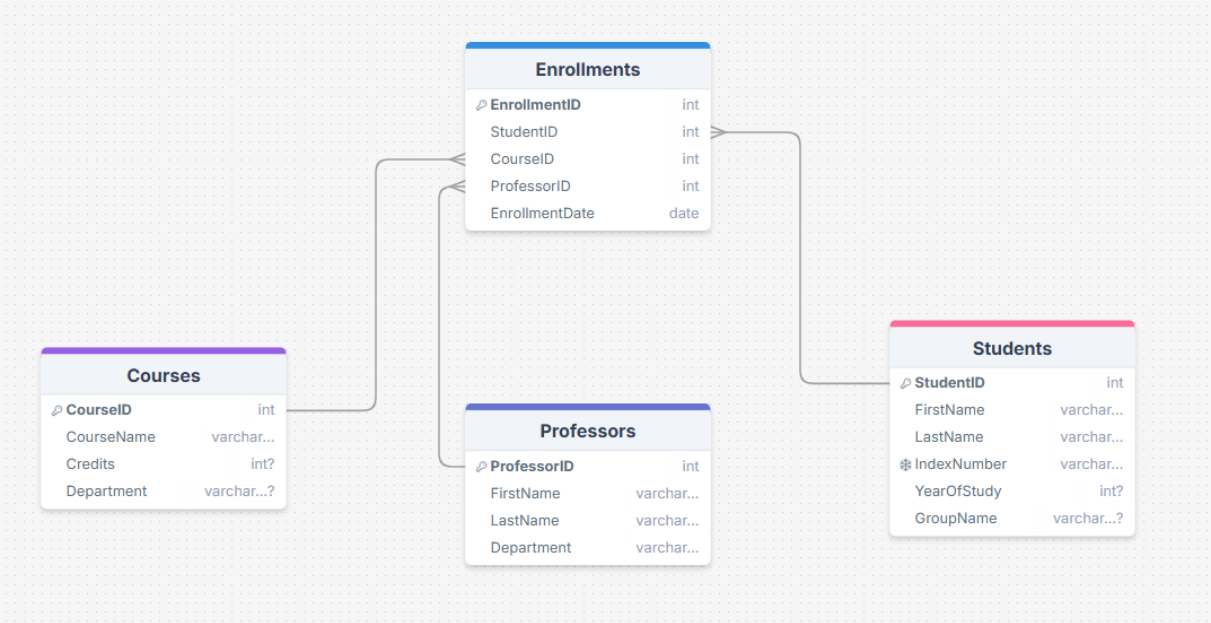
Enrollments table creating a view for names:

```
 CREATE VIEW EnrollmentDetails AS
SELECT
   e.EnrollmentID,
   s.FirstName || ' ' || s.LastName AS StudentName,
   c.CourseName,
```

```
    p.FirstName || ' ' || p.LastName AS ProfessorName,
    e.EnrollmentDate
FROM Enrollments e
JOIN Students s ON e.StudentID = s.StudentID
JOIN Courses c ON e.CourseID = c.CourseID
JOIN Professors p ON e.ProfessorID = p.ProfessorID;

SELECT * FROM EnrollmentDetails;
```

DIAGRAM/SCHEME:



QUERIES:

1- List all courses with departments:

SELECT * FROM Courses;

	courseid [PK] integer	coursename character varying (100)	credits integer	department character varying (50)
1	1	Database Systems	5	Computer Science
2	2	Operating Systems	4	Computer Science
3	3	Data Structures	4	Computer Science
4	4	Artificial Intelligence	5	AI Research
5	5	Software Engineering	3	Software Development
6	6	Cybersecurity	3	Cyber Defense

2-List all students:

SELECT * FROM Students;

	studentid [PK] integer	firstname character varying (50)	lastname character varying (50)	indexnumber character varying (20)	yearofstudy integer	groupname character varying (20)
1	1	Alice	Smith	S101	3	CS1
2	2	Bob	Johnson	S102	1	CS1
3	3	Charlie	Brown	S103	2	CS2
4	4	Daisy	White	S104	4	CS1
5	5	Ella	Taylor	S105	3	CS2
6	6	Frank	Miller	S106	5	CS1
7	7	Grace	Wilson	S107	2	CS3
8	8	Harry	Davis	S108	1	CS2
9	9	Ivy	Clark	S109	4	CS1
10	10	Jack	Lewis	S110	5	CS3

3-Find students enrolled in a specific course

```
SELECT s.FirstName, s.LastName, c.CourseName
FROM Enrollments e
JOIN Students s ON e.StudentID = s.StudentID
JOIN Courses c ON e.CourseID = c.CourseID
WHERE c.CourseName = 'Database Systems';
```

	firstname character varying (50)	lastname character varying (50)	coursename character varying (100)
1	Alice	Smith	Database Systems
2	Grace	Wilson	Database Systems

4-Counts students per course:

```
SELECT c.CourseName, COUNT(e.StudentID) AS StudentCount
FROM Enrollments e
JOIN Courses c ON e.CourseID = c.CourseID
GROUP BY c.CourseName;
```

	coursename character varying (100)	studentcount bigint
1	Cybersecurity	1
2	Database Systems	2
3	Artificial Intelligence	2
4	Data Structures	2
5	Software Engineering	2
6	Operating Systems	1

5-List professors teaching more than 2 courses:

```
SELECT p.FirstName, p.LastName, COUNT(e.CourseID) AS CoursesTaught
FROM Enrollments e
JOIN Professors p ON e.ProfessorID = p.ProfessorID
GROUP BY p.ProfessorID
HAVING COUNT(e.CourseID) > 1;
```

	firstname character varying (50)	lastname character varying (50)	coursestaught bigint
1	Dr. Sarah	Brown	2
2	Dr. Anna	Davis	1
3	Dr. James	Wilson	2
4	Dr. Michael	Taylor	2
5	Dr. Emily	Jones	3

6-Count the number of courses each professor is teaching:

```
SELECT
   p.FirstName || ' ' || p.LastName AS ProfessorName,
   COUNT(DISTINCT e.CourseID) AS TotalCourses
FROM Enrollments e
JOIN Professors p ON e.ProfessorID = p.ProfessorID
GROUP BY p.ProfessorID, p.FirstName, p.LastName;
```

	professorname text	totalcourses bigint
1	Dr. Emily Jones	2
2	Dr. Michael Taylor	1
3	Dr. Sarah Brown	1
4	Dr. James Wilson	1
5	Dr. Anna Davis	1

7- List students grouped by their year of study:

```
SELECT
   s.YearOfStudy,
   STRING_AGG(s.FirstName || ' ' || s.LastName, ', ') AS Students
FROM Students s
GROUP BY s.YearOfStudy
ORDER BY s.YearOfStudy;
```

	yearofstudy integer	students text
1	1	Bob Johnson, Harry Davis
2	2	Charlie Brown, Grace Wilson
3	3	Alice Smith, Ella Taylor
4	4	Daisy White, Ivy Clark
5	5	Frank Miller, Jack Lewis

8-List professors and the total number of students they are teaching:

```
SELECT
   p.FirstName || ' ' || p.LastName AS ProfessorName,
   COUNT(e.StudentID) AS TotalStudents
FROM Enrollments e
JOIN Professors p ON e.ProfessorID = p.ProfessorID
GROUP BY p.ProfessorID, p.FirstName, p.LastName
ORDER BY TotalStudents DESC;
```

	professorname text	totalstudents bigint
1	Dr. Emily Jones	3
2	Dr. Sarah Brown	2
3	Dr. James Wilson	2
4	Dr. Michael Taylor	2
5	Dr. Anna Davis	1

9-Find courses with the most students enrolled:

```
SELECT
   c.CourseName,
   COUNT(e.StudentID) AS TotalStudents
FROM Enrollments e
JOIN Courses c ON e.CourseID = c.CourseID
GROUP BY c.CourseName
ORDER BY TotalStudents DESC
```

	coursename character varying (100)	totalstudents bigint
1	Database Systems	2
2	Artificial Intelligence	2
3	Data Structures	2
4	Software Engineering	2
5	Cybersecurity	1