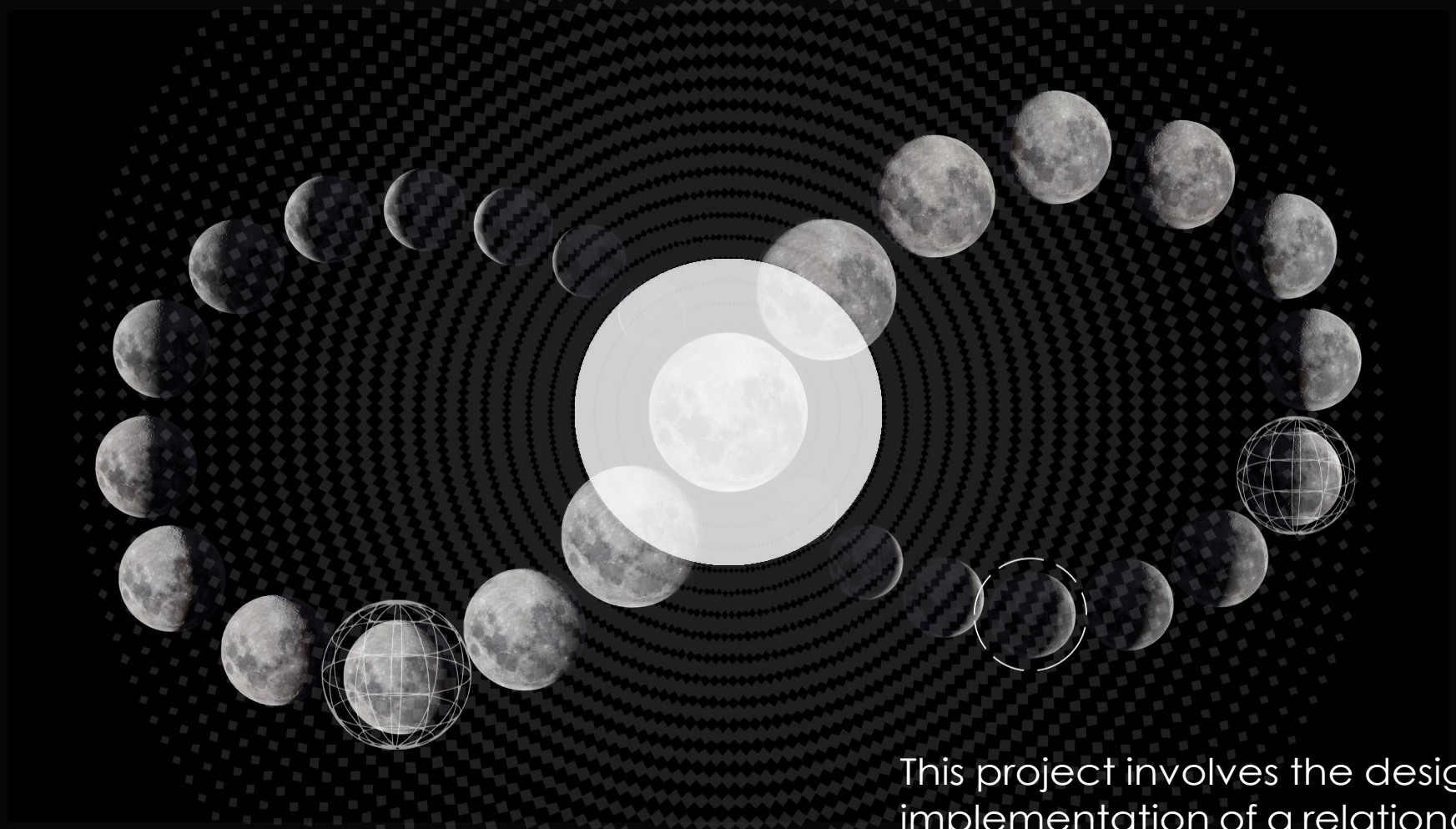
SQL
STUDENT_COURSE_MANAGEMENT



This project involves the design and implementation of a relational database to manage student course enrollments. The system includes tables for Students, Instructors, Courses, and Enrollments, with carefully structured relationships to efficiently manage and retrieve data. Advanced SQL queries were employed to perform operations such as joining tables, calculating aggregates, and utilizing subqueries.

@SALAH MOHAMED

Database Setup:

StudentCourseManagement.sql

```
CREATE DATABASE
StudentCourseManagement;
USE StudentCourseManagement;
```

Table Creation:
Students Table

StudentCourseManagement.sql

```
CREATE TABLE Students
 ( student_id INT PRIMARY
 KEY
IDENTITY(1,1),
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100),
    date of birth DATE
```

Table Creation:
Instructors Table

StudentCourseManagement.sql

```
CREATE TABLE Instructors
( instructor_id INT PRIMARY
KEY IDENTITY(1,1),
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100)
);
```

Table Creation:
Courses Table

StudentCourseManagement.sql

```
CREATE TABLE Courses
( course_id INT PRIMARY KEY
IDENTITY(1,1),
    course_name VARCHAR(100),
    course_description TEXT,
    instructor_id INT,
    FOREIGN KEY (instructor_id)
REFERENCES
Instructors(instructor_id)
);
```

Table Creation: Enrollments Table

StudentCourseManagement.sql

```
CREATE TABLE Enrollments
( enrollment_id INT PRIMARY KEY
IDENTITY(1,1),
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 Sample Data:

Students Table

StudentCourseManagement.sql

```
INSERT INTO Students (first_name,
last_name, email, date_of_birth)
VALUES
('Ahmed', 'El-Sayed',
'ahmed.elsayed@gmail.com', '2000-
01-15'),
('Mona', 'Hassan',
'mona.hassan@gmail.com', '1999-02-
20');
```

Insert Sample Data:

Instructors Table

StudentCourseManagement.sql

```
INSERT INTO Instructors
(first_name, last_name, email)
VALUES
('Hesham', 'Mansour',
'hesham.mansour@gmail.com'),
('Amina', 'Zaki',
'amina.zaki@gmail.com'),
('Karim', 'Mostafa',
'karim.mostafa@gmail.com');
```

Insert Sample Data:

Courses Table

StudentCourseManagement.sql

```
INSERT INTO Courses (course_name,
course_description, instructor_id)
VALUES
('Introduction to Programming',
'Basic concepts of programming
using Python.', 1),
('Data Structures', 'Understanding
and implementing various data
structures.', 2);
```

Insert Sample Data:

Enrollments Table

StudentCourseManagement.sql

```
INSERT INTO Enrollments
(student_id, course_id,
enrollment_date) VALUES
(1, 1,  '2024-01-10'),
(2, 2,  '2024-01-12'),
(3, 4,  '2024-01-14'),
(4, 4,  '2024-01-16'),
(5, 5,  '2024-01-18'),
(6, 1,  '2024-01-20'),
(7, 2,  '2024-01-22'),
(8, 3,  '2024-01-24');
```

Queries:

<Select all students>

StudentCourseManagement.sql

```
SELECT * FROM Students;
```

Queries:

<Select all courses>

StudentCourseManagement.sql

```
SELECT * FROM Courses;
```

Queries:

<Select all enrollments with student names and course names>

StudentCourseManagement.sql

```
SELECT
    E.enrollment_id,
    S.first_name + ' ' +
S.last_name AS student_name,
    C.course_name,
    E.enrollment_date
FROM
    Enrollments E
JOIN
    Students S ON E.student_id =
S.student_id
JOIN
    Courses C ON E.course_id =
C.course_id;
```

Queries:

<Select students who enrolled in a specific course>

StudentCourseManagement.sql

```
SELECT
    S.first_name, S.last_name,
C.course_name
FROM
    Enrollments E
JOIN
    Students S ON E.student_id =
S.student_id
JOIN
    Courses C ON E.course_id =
C.course_id
WHERE
    C.course_name = 'Database
Systems';
```

Queries:

<Select courses with more than 5 students>

StudentCourseManagement.sql

```
SELECT
    C.course_name,
COUNT(E.student_id) AS
student_count
FROM
    Enrollments E
JOIN
    Courses C ON E.course_id =
C.course_id
GROUP BY
    C.course_name
HAVING
    COUNT(E.student_id) > 5;
```

Queries:

<Update a student's email>

StudentCourseManagement.sql

```
UPDATE
    Students
SET
    email = 'new.email@example.com'
WHERE
    student id = 1;
```

Queries:

<Delete a course that no students are enrolled in>

StudentCourseManagement.sql

```
DELETE FROM
    Courses
WHERE
course_id NOT IN (SELECT DISTINCT
course id FROM Enrollments);
```

Queries:

<Calculate the average age of students>

StudentCourseManagement.sql

```
SELECT
    AVG(DATEDIFF(YEAR,
date_of_birth, GETDATE())) AS
average_age
FROM
    Students;
```

Queries:

<Find the course with the maximum enrollments>

StudentCourseManagement.sql

```
SELECT
    TOP 1 C.course_name,
COUNT(E.enrollment_id) AS
enrollment_count
FROM
    Enrollments E
JOIN
    Courses C ON E.course_id =
C.course_id
GROUP BY
    C.course_name
ORDER BY
    COUNT(E.enrollment_id) DESC;
```

Queries:

<List courses along with the number of students enrolled (use GROUP BY)>

StudentCourseManagement.sql

```
SELECT
    C.course_name,
COUNT(E.student_id) AS
student_count
FROM
    Enrollments E
JOIN
    Courses C ON E.course_id =
C.course_id
GROUP BY
    C.course_name;
```

Queries:

<Select all students with their enrolled courses (use JOIN)>

StudentCourseManagement.sql

```
SELECT
    S.first_name, S.last_name,
C.course_name
FROM
    Students S
LEFT JOIN
    Enrollments E ON S.student_id =
E.student_id
LEFT JOIN
    Courses C ON E.course_id =
C.course_id;
```

Queries:

<List all instructors and their courses>

StudentCourseManagement.sql

```
SELECT
    I.first_name, I.last_name,
C.course_name
FROM
    Instructors I
LEFT JOIN
    Courses C ON I.instructor_id =
C.instructor id;
```

Queries:

<Find students who are not enrolled in any course>

StudentCourseManagement.sql

```
SELECT
    S.first_name, S.last_name
FROM
    Students S
LEFT JOIN
    Enrollments E ON S.student_id =
E.student_id
WHERE
    E.enrollment id IS NULL;
```

Queries:

<Select students enrolled in more than one course>

StudentCourseManagement.sql

```
SELECT
    S.first_name, S.last_name
FROM
    Students S
WHERE
    S.student_id IN
        ( SELECT
            E.student_id
        FROM
            Enrollments E
        GROUP BY
            E.student_id
        HAVING
            COUNT(E.course_id) > 1
    );
```

Queries:

<Find courses taught by a specific instructor>

StudentCourseManagement.sql

```
SELECT
    C.course_name, I.first_name,
I.last_name
FROM
    Courses C
JOIN
    Instructors I ON
C.instructor_id = I.instructor_id
WHERE
    I.first_name = 'Karim' AND
I.last_name = 'Mostafa';
```

Queries:

<Select the top 3 students with the most enrollments>

StudentCourseManagement.sql

```
SELECT
    TOP 3 S.first_name, S.last_name,
COUNT(E.enrollment_id) AS
enrollment_count
FROM
    Students S
JOIN
    Enrollments E ON S.student_id =
E.student_id
GROUP BY
    S.first_name, S.last_name
ORDER BY
    COUNT(E.enrollment_id) DESC;
```

Queries:

<Use UNION to combine results of two different SELECT queries>

StudentCourseManagement.sql

```
SELECT
    first_name, last_name
FROM
    Students
WHERE
    date_of_birth < '2000-01-01'
UNION
SELECT
    first_name, last_name
FROM
    Instructors
WHERE
    last name LIKE 'A%';
```

Queries:

<Create a stored procedure to add a new student>

StudentCourseManagement.sql

```
CREATE PROCEDURE Add_Student
    @first_name VARCHAR(50),
    @last_name VARCHAR(50),
    @email VARCHAR(100),
    @date_of_birth DATE
AS
BEGIN
    INSERT INTO Students
(first_name, last_name, email,
date_of_birth)
    VALUES (@first_name,
@last_name, @email,
@date_of_birth);
END;
```

Queries:

<Calculate the total number of students>

StudentCourseManagement.sql

```
SELECT COUNT(*) AS total_students
FROM Students;
```

Queries:

<Calculate the average, minimum, and maximum number of enrollments per course>

StudentCourseManagement.sql

```
SELECT
    AVG(student_count) AS
average_enrollments,
    MIN(student_count) AS
min_enrollments,
    MAX(student_count) AS
max_enrollments
FROM (
    SELECT
        course_id,
COUNT(student_id) AS student_count
    FROM
        Enrollments
    GROUP BY
        course_id
) AS enrollments_per_course;
```

Queries:

<Create aliases for complex column names>

StudentCourseManagement.sql

```
SELECT
    S.first_name AS 'Student First
Name',
    S.last_name AS 'Student Last
Name',
    C.course_name AS 'Course Name'
FROM
    Students S
JOIN
    Enrollments E ON S.student_id =
E.student_id
JOIN
    Courses C ON E.course_id =
C.course_id;
```

Queries:

<Use CASE to categorize students based on their age>

StudentCourseManagement.sql

```
SELECT
    first_name,
    last_name,
    CASE
        WHEN DATEDIFF(YEAR,
date_of_birth, GETDATE()) < 20 THEN
'Teenager'
        WHEN DATEDIFF(YEAR,
date_of_birth, GETDATE()) BETWEEN
20 AND 25 THEN 'Young Adult'
        ELSE 'Adult'
    END AS age_group
FROM
    Students;
```

Queries:

<Use EXISTS to find courses with at least one enrolled student>

StudentCourseManagement.sql

```
SELECT
    course_name
FROM
    Courses C
WHERE
    EXISTS (
        SELECT
            1
        FROM
            Enrollments E
        WHERE
            E.course_id =
C.course_id
    );
```

Queries:

<Create comments in SQL for clarity>

StudentCourseManagement.sql

```
-- Single-line comment: This query
retrieves all students
SELECT COUNT(*) AS total_students
FROM Students;
-- Multi-line comment: This query
retrieves all students
/*
This query retrieves all students
*/
```

Outputs:

<Select all students>

	student_id	first_name	last_name	email	date_of_birth
1	1	Ahmed	El-Sayed	ahmed.elsayed@gmail.com	2000-01-15
2	2	Mona	Hassan	mona.hassan@gmail.com	1999-02-20
3	3	Mohamed	Ali	mohamed.ali@gmail.com	2001-03-10
4	4	Sara	Fathy	sara.fathy@gmail.com	1998-04-25
5	5	Omar	Khaled	omar.khaled@gmail.com	2000-05-30
6	6	Laila	Nabil	laila.nabil@gmail.com	1999-06-15
7	7	Hassan	Gamal	hassan.gamal@gmail.com	2001-07-20
8	8	Dina	Youssef	dina.youssef@gmail.com	1998-08-05
9	9	Tamer	Khalil	tamer.khalil@gmail.com	1999-09-25
10	10	Nadia	Farouk	nadia.farouk@gmail.com	2000-10-10
11	11	Yasser	Adel	yasser.adel@gmail.com	2001-11-15
12	12	Fatma	Saleh	fatma.saleh@gmail.com	1998-12-05
13	13	Khaled	Mohsen	khaled.mohsen@gmail.com	2002-03-15

<Select all courses>

	course_id	course_name	course_description	instructor_id
1	1	Introduction to Programming	Basic concepts of programming using Python.	1
2	2	Data Structures	Understanding and implementing various data structu...	2
3	3	Database Systems	Introduction to database design and SQL.	3
4	4	Computer Networks	Fundamentals of computer networking and protocols.	4
5	5	Operating Systems	Concepts of operating systems and process manage...	5
6	6	Advanced Algorithms	In-depth study of advanced algorithmic techniques.	3

<Select all enrollments with student names and course names>

	enrollment_id	student_name	course_name	enrollment_date
1	1	Ahmed El-Sayed	Introduction to Programming	2024-01-10
2	2	Mona Hassan	Data Structures	2024-01-12
3	3	Mohamed Ali	Database Systems	2024-01-14
4	4	Sara Fathy	Computer Networks	2024-01-16
5	5	Omar Khaled	Operating Systems	2024-01-18
6	6	Laila Nabil	Introduction to Programming	2024-01-20
7	7	Hassan Gamal	Data Structures	2024-01-22
8	8	Dina Youssef	Database Systems	2024-01-24
9	9	Tamer Khalil	Computer Networks	2024-01-26
10	10	Nadia Farouk	Operating Systems	2024-01-28
11	11	Yasser Adel	Introduction to Programming	2024-01-30
12	12	Fatma Saleh	Data Structures	2024-02-01
13	13	Ahmed El-Sayed	Database Systems	2024-02-03
14	14	Mona Hassan	Computer Networks	2024-02-05
15	15	Mohamed Ali	Operating Systems	2024-02-07
16	16	Sara Fathy	Introduction to Programming	2024-02-10

<Select students who enrolled in a specific course>

	first_name	last_name	course_name
1	Mohamed	Ali	Database Systems
2	Dina	Youssef	Database Systems
3	Ahmed	El-Sayed	Database Systems
4	Laila	Nabil	Database Systems
5	Yasser	Adel	Database Systems

<Select courses with more than 5 students>

	course_name	student_count
1	Computer Networks	6

<Calculate the average age of students>

	average_age
1	24

<Find the course with the maximum enrollments>

	course_name	enrollment_count
1	Computer Networks	6

<List courses along with the number of students enrolled (use GROUP BY)>

	course_name	student_count
1	Computer Networks	6
2	Data Structures	5
3	Database Systems	4
4	Introduction to Programming	5
5	Operating Systems	4

<Select all students with their enrolled courses (use JOIN)>

	first_name	last_name	course_name
1	Ahmed	El-Sayed	Introduction to Programming
2	Ahmed	El-Sayed	Database Systems
3	Mona	Hassan	Data Structures
4	Mona	Hassan	Computer Networks
5	Mohamed	Ali	Computer Networks
6	Mohamed	Ali	Operating Systems
7	Sara	Fathy	Computer Networks
8	Sara	Fathy	Introduction to Programming
9	Omar	Khaled	Operating Systems
10	Omar	Khaled	Data Structures
11	Laila	Nabil	Introduction to Programming
12	Laila	Nabil	Database Systems
13	Hassan	Gamal	Data Structures
14	Hassan	Gamal	Computer Networks
15	Dina	Youssef	Database Systems
16	Dina	Youssef	Operating Systems

<List all instructors and their courses>

	first_name	last_name	course_name
1	Hesham	Mansour	Introduction to Programming
2	Amina	Zaki	Data Structures
3	Karim	Mostafa	Database Systems
4	Rania	Adel	Computer Networks
5	Tarek	Nassar	Operating Systems

<Find students who are not enrolled in any course.>

	first_name	last_name
1	Khaled	Mohsen

<Select students enrolled in more than one course>

	first_name	last_name
1	Ahmed	El-Sayed
2	Mona	Hassan
3	Mohamed	Ali
4	Sara	Fathy
5	Omar	Khaled
6	Laila	Nabil
7	Hassan	Gamal
8	Dina	Youssef
9	Tamer	Khalil
10	Nadia	Farouk
11	Yasser	Adel
12	Fatma	Saleh

<Find courses taught by a specific instructor>

	course_name	first_name	last_name
1	Database Systems	Karim	Mostafa

<Select the top 3 students with the most enrollments>

	first_name	last_name	enrollment_count
1	Nadia	Farouk	2
2	Ahmed	El-Sayed	2
3	Mohamed	Ali	2

<Use UNION to combine results of two different SELECT queries>

	first_name	last_name
1	Dina	Youssef
2	Fatma	Saleh
3	Laila	Nabil
4	Mona	Hassan
5	Rania	Adel
6	Sara	Fathy
7	Tamer	Khalil

<Create a stored procedure to add a new student>

```
EXEC Add_Student
    @first_name = 'Donia',
    @last_name = 'Mohamed',
    @email = 'donia.mohamed@example.com',
    @date_of_birth = '2000-01-01';
```

```
SELECT * FROM Students;
```

Results | Messages

student_id	first_name	last_name	email	date_of_birth
1	Ahmed	El-Sayed	new.email@example.com	2000-01-15
2	Mona	Hassan	mona.hassan@gmail.com	1999-02-20
3	Mohamed	Ali	mohamed.ali@gmail.com	2001-03-10
4	Sara	Fathy	sara.fathy@gmail.com	1998-04-25
5	Omar	Khaled	omar.khaled@gmail.com	2000-05-30
6	Laila	Nabil	laila.nabil@gmail.com	1999-06-15
7	Hassan	Gamal	hassan.gamal@gmail.com	2001-07-20
8	Dina	Youssef	dina.youssef@gmail.com	1998-08-05
9	Tamer	Khalil	tamer.khalil@gmail.com	1999-09-25
10	Nadia	Farouk	nadia.farouk@gmail.com	2000-10-10
11	Yasser	Adel	yasser.adel@gmail.com	2001-11-15
12	Fatma	Saleh	fatma.saleh@gmail.com	1998-12-05
13	Khaled	Mohsen	khaled.mohsen@gmail.com	2002-03-15
14	Donia	Mohamed	donia.mohamed@example.com	2000-01-01

<Calculate the total number of students>

	total_students
1	13

<Calculate the average, minimum, and maximum number of enrollments per course>

	average_enrollments	min_enrollments	max_enrollments
1	4	4	6

<Create aliases for complex column names>

	Student First Name	Student Last Name	Course Name
1	Ahmed	El-Sayed	Introduction to Programming
2	Mona	Hassan	Data Structures
3	Mohamed	Ali	Computer Networks
4	Sara	Fathy	Computer Networks
5	Omar	Khaled	Operating Systems
6	Laila	Nabil	Introduction to Programming
7	Hassan	Gamal	Data Structures
8	Dina	Youssef	Database Systems
9	Tamer	Khalil	Computer Networks
10	Nadia	Farouk	Operating Systems
11	Yasser	Adel	Introduction to Programming
12	Fatma	Saleh	Data Structures
13	Ahmed	El-Sayed	Database Systems
14	Mona	Hassan	Computer Networks
15	Mohamed	Ali	Operating Systems
16	Sara	Fathy	Introduction to Programming

Use CASE to categorize students based on their age

	first_name	last_name	age_group
1	Ahmed	El-Sayed	Young Adult
2	Mona	Hassan	Young Adult
3	Mohamed	Ali	Young Adult
4	Sara	Fathy	Adult
5	Omar	Khaled	Young Adult
6	Laila	Nabil	Young Adult
7	Hassan	Gamal	Young Adult
8	Dina	Youssef	Adult
9	Tamer	Khalil	Young Adult
10	Nadia	Farouk	Young Adult
11	Yasser	Adel	Young Adult
12	Fatma	Saleh	Adult
13	Khaled	Mohsen	Young Adult

Use EXISTS to find courses with at least one enrolled student

	course_name
1	Introduction to Programming
2	Data Structures
3	Database Systems
4	Computer Networks
5	Operating Systems

Thank you

check the Github link!

click here