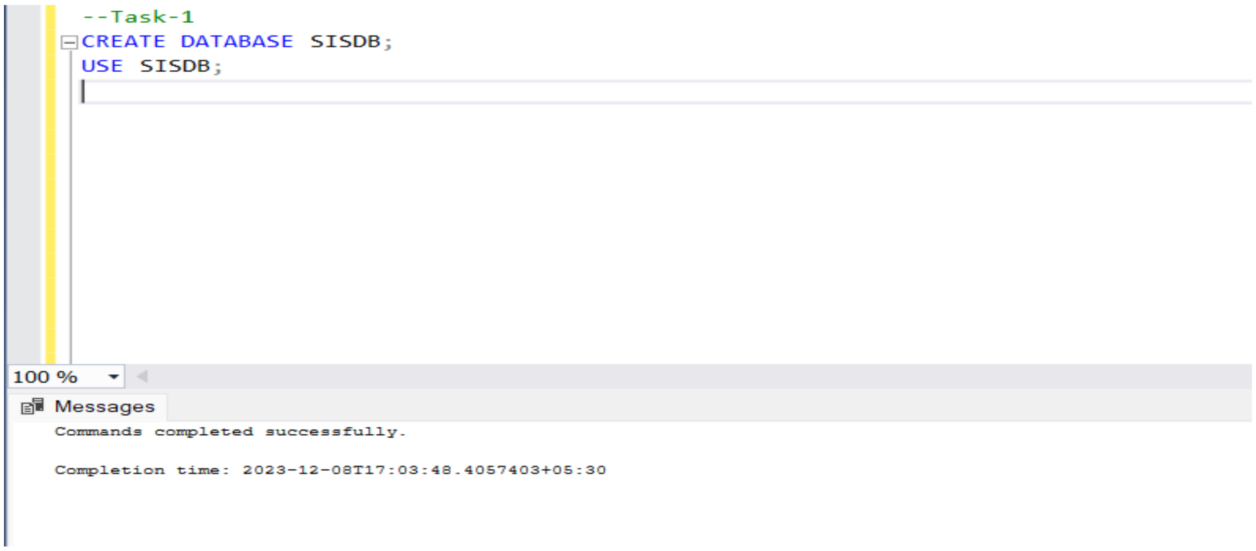
Assignment – 2
Student Information System

Task 1. Database Design:

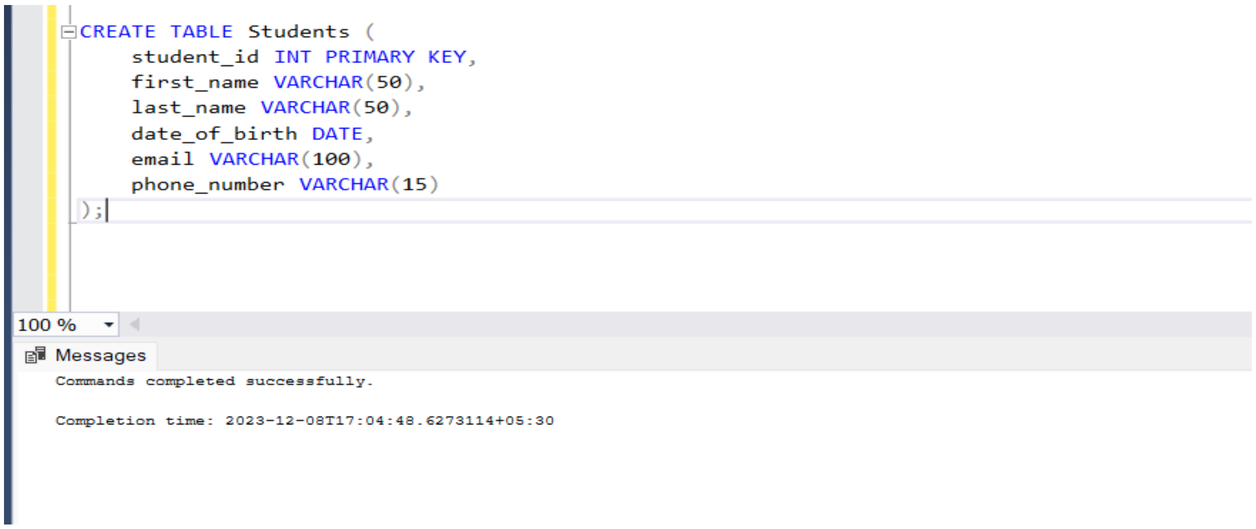
1.



```
--Task-1
CREATE DATABASE SISDB;
USE SISDB;
```

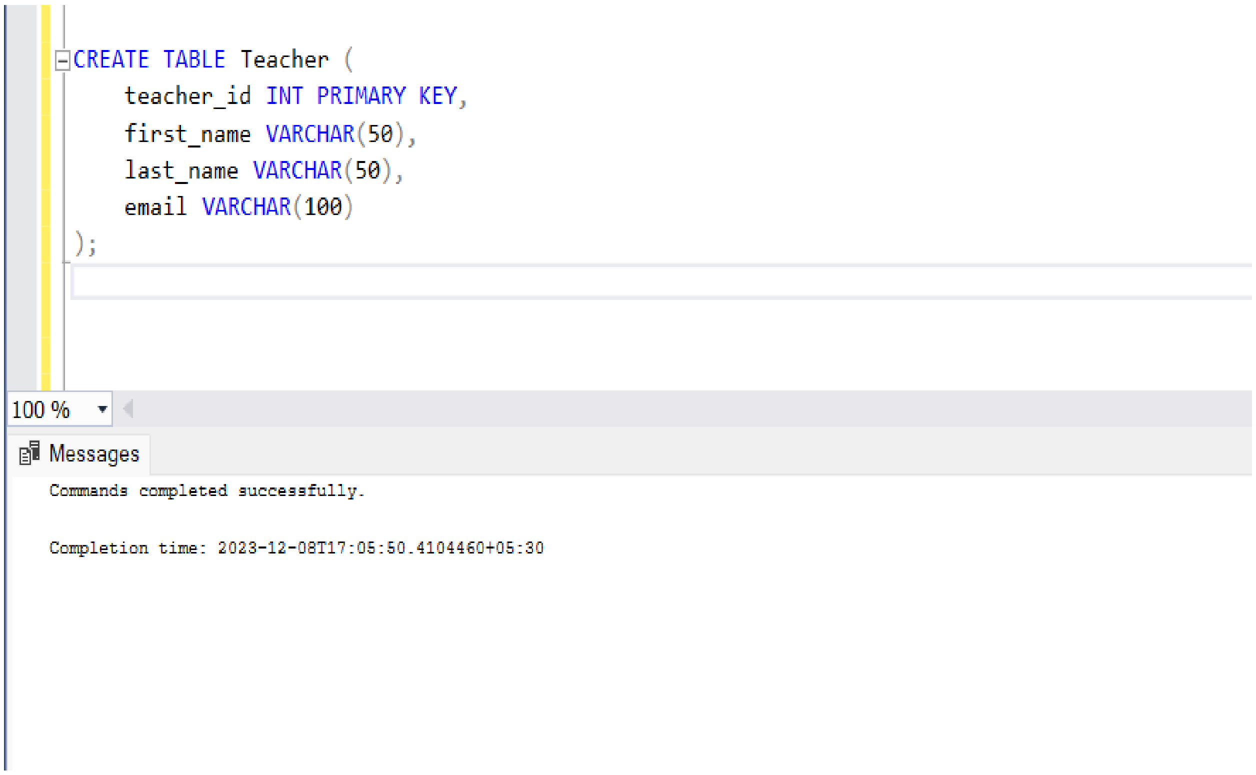
```
100 %
Messages
Commands completed successfully.

Completion time: 2023-12-08T17:03:48.4057403+05:30
```

2.



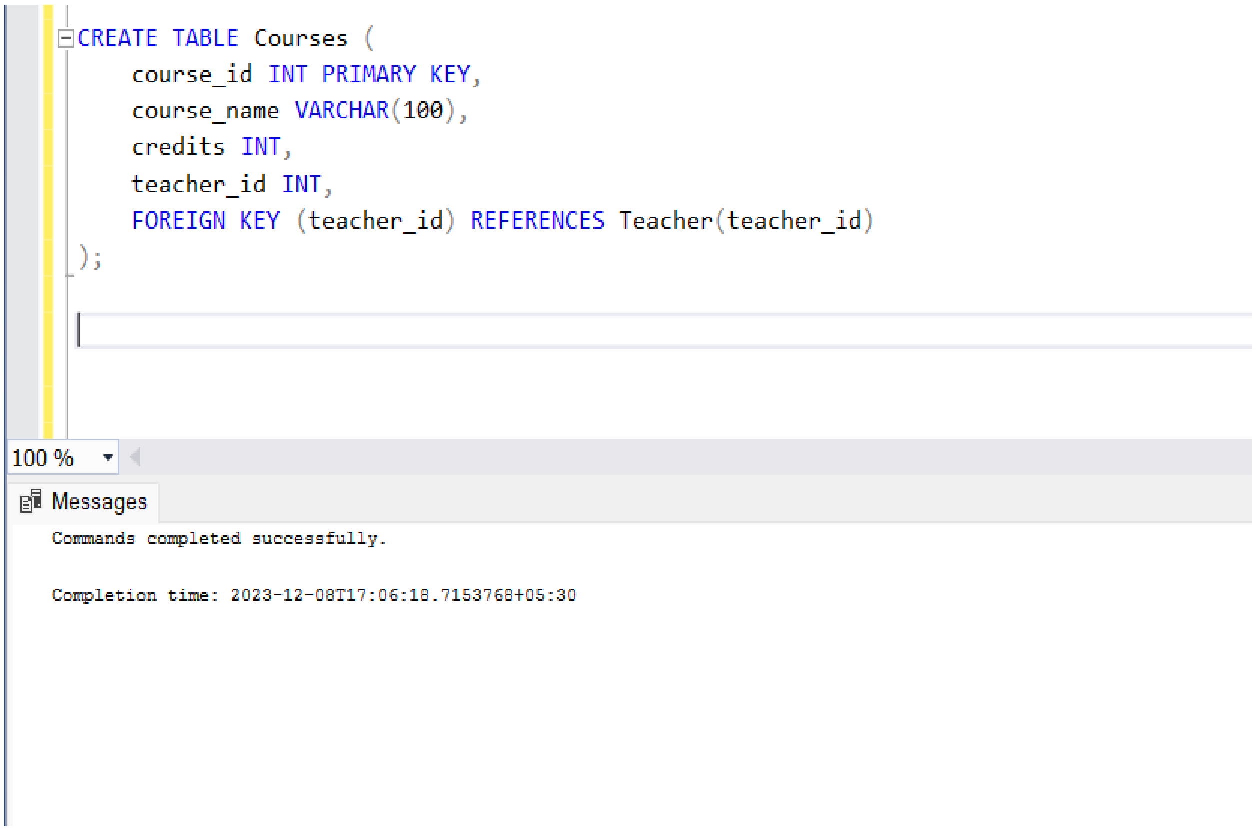
```
CREATE TABLE Students (
    student_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    date_of_birth DATE,
    email VARCHAR(100),
    phone_number VARCHAR(15)
);
```

```
100 %
Messages
Commands completed successfully.

Completion time: 2023-12-08T17:04:48.6273114+05:30
```



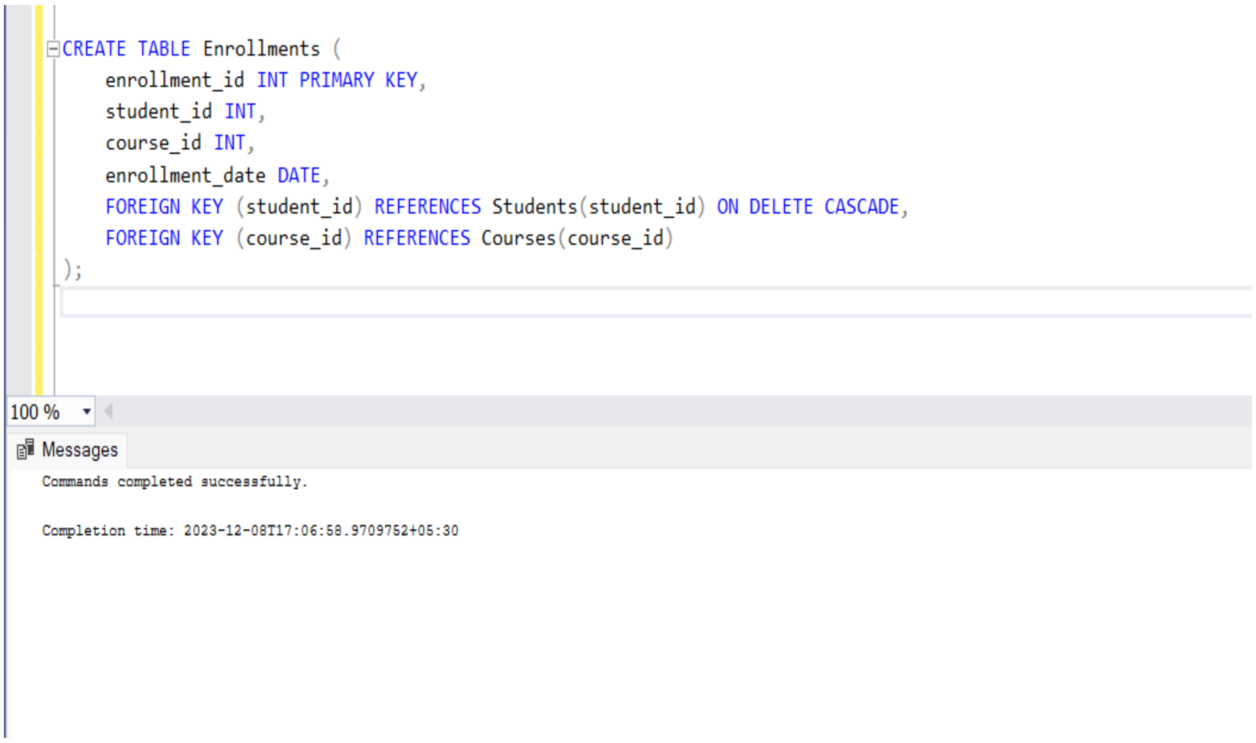
```
CREATE TABLE Teacher (
    teacher_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100)
);
```

```
Commands completed successfully.

Completion time: 2023-12-08T17:05:50.4104460+05:30
```



```
CREATE TABLE Courses (
    course_id INT PRIMARY KEY,
    course_name VARCHAR(100),
    credits INT,
    teacher_id INT,
    FOREIGN KEY (teacher_id) REFERENCES Teacher(teacher_id)
);
```

```
Commands completed successfully.

Completion time: 2023-12-08T17:06:18.7153768+05:30
```

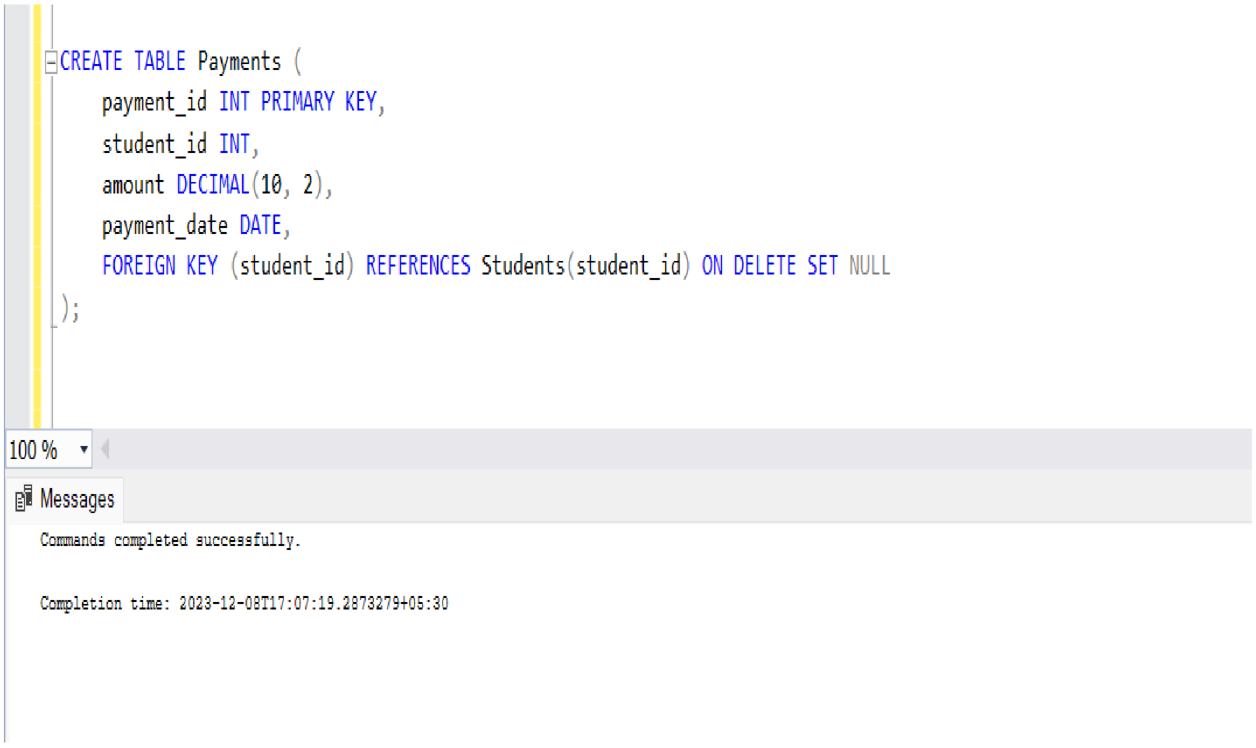


```
CREATE TABLE Enrollments (
    enrollment_id INT PRIMARY KEY,
    student_id INT,
    course_id INT,
    enrollment_date DATE,
    FOREIGN KEY (student_id) REFERENCES Students(student_id) ON DELETE CASCADE,
    FOREIGN KEY (course_id) REFERENCES Courses(course_id)
);
```

100 %

Messages

```
Commands completed successfully.

Completion time: 2023-12-08T17:06:58.9709752+05:30
```



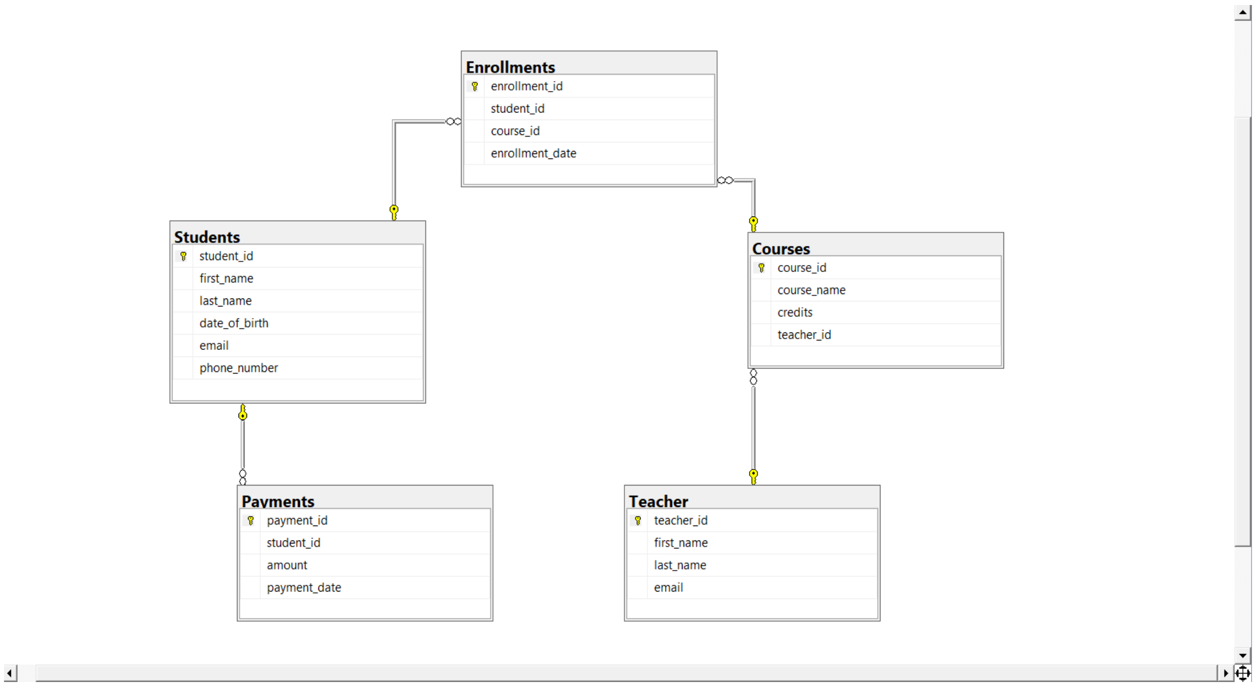
```
CREATE TABLE Payments (
    payment_id INT PRIMARY KEY,
    student_id INT,
    amount DECIMAL(10, 2),
    payment_date DATE,
    FOREIGN KEY (student_id) REFERENCES Students(student_id) ON DELETE SET NULL
);
```

100 %

Messages

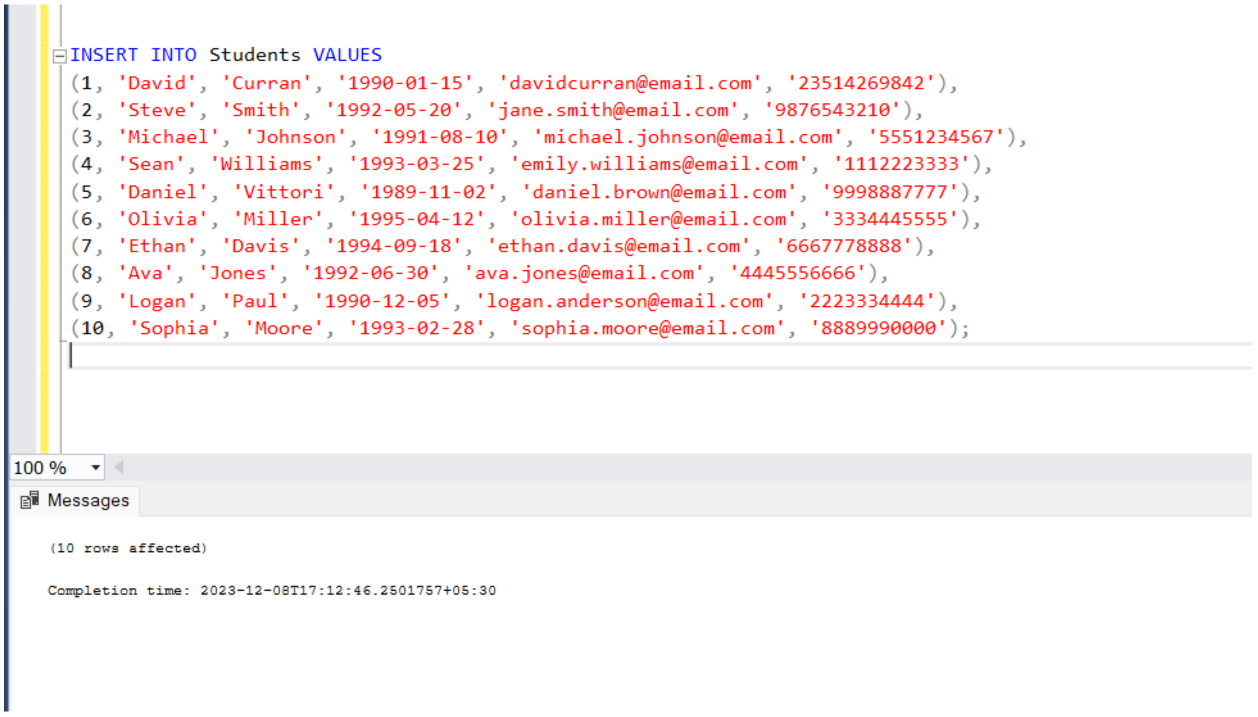
```
Commands completed successfully.

Completion time: 2023-12-08T17:07:19.2873279+05:30
```

3.



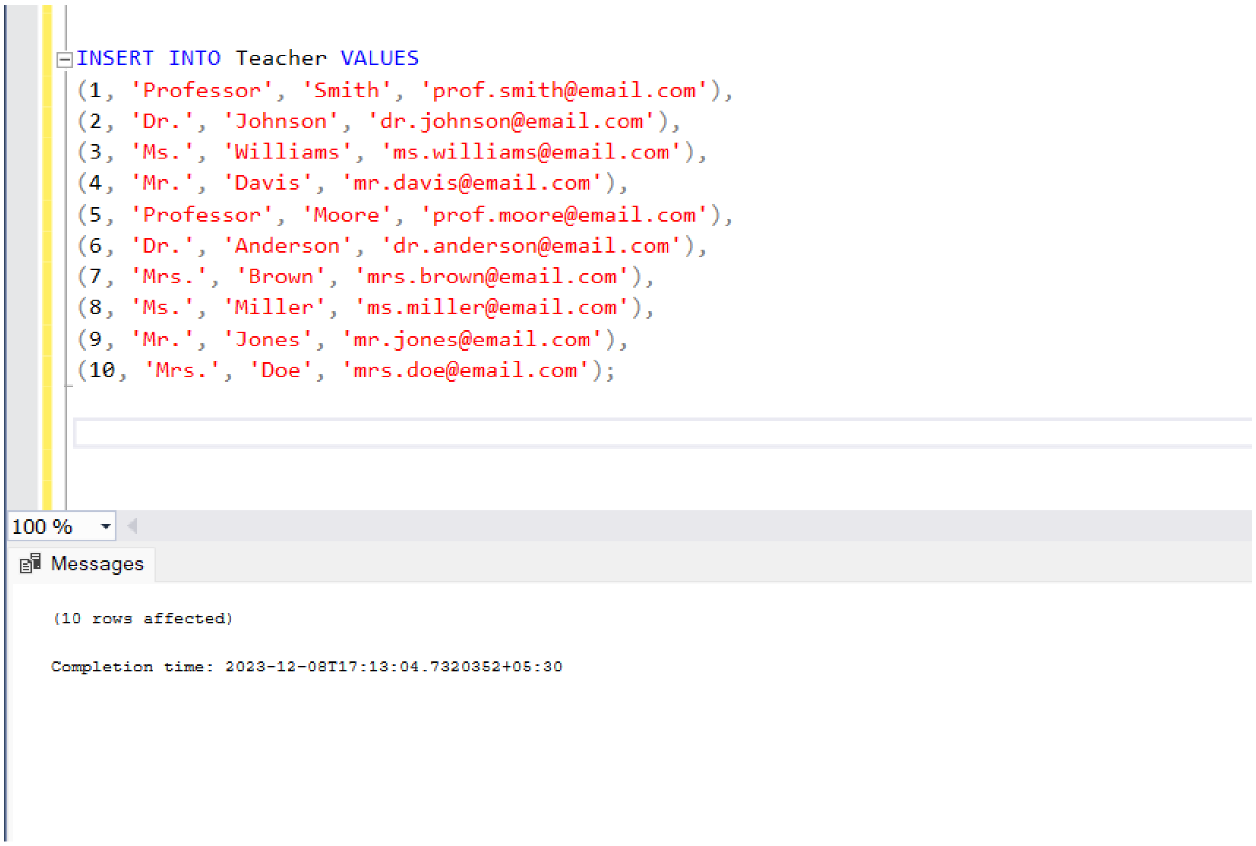
4. Created Appropriate Primary Key and Foreign Key while creating Table.

5.



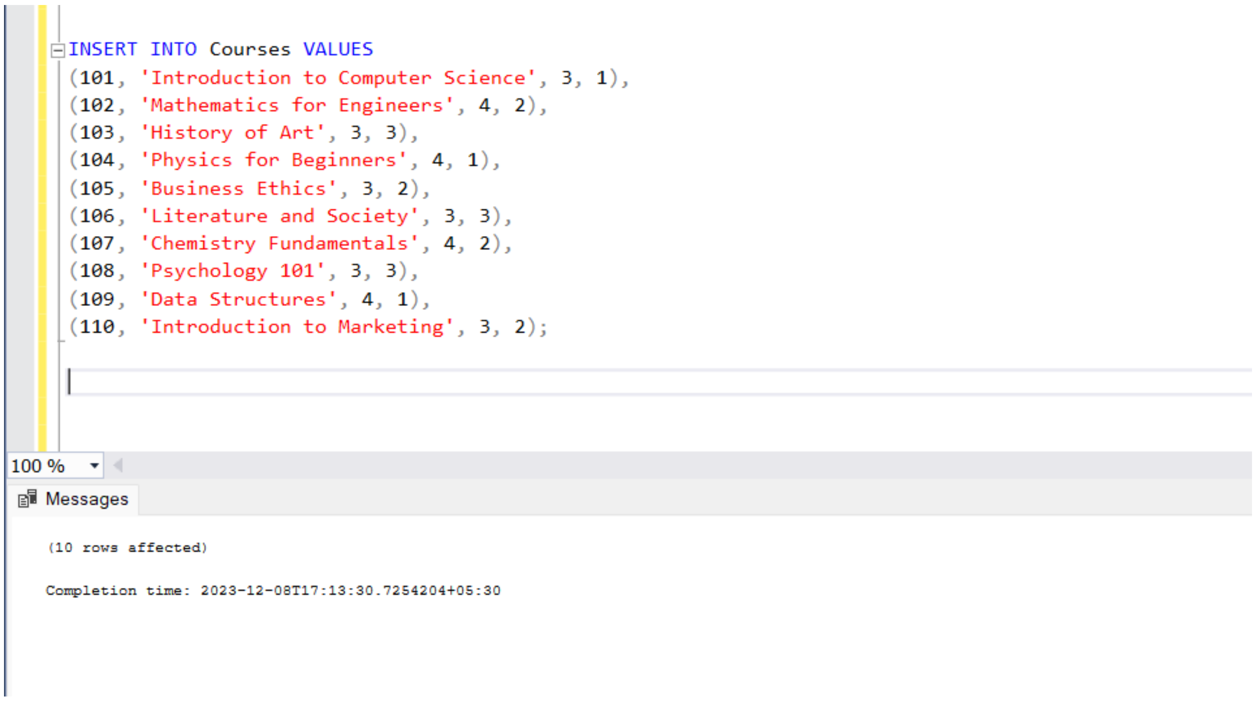
```
INSERT INTO Students VALUES
(1, 'David', 'Curran', '1990-01-15', 'davidcurran@email.com', '23514269842'),
(2, 'Steve', 'Smith', '1992-05-20', 'jane.smith@email.com', '9876543210'),
(3, 'Michael', 'Johnson', '1991-08-10', 'michael.johnson@email.com', '5551234567'),
(4, 'Sean', 'Williams', '1993-03-25', 'emily.williams@email.com', '1112223333'),
(5, 'Daniel', 'Vittori', '1989-11-02', 'daniel.brown@email.com', '9998887777'),
(6, 'Olivia', 'Miller', '1995-04-12', 'olivia.miller@email.com', '3334445555'),
(7, 'Ethan', 'Davis', '1994-09-18', 'ethan.davis@email.com', '6667778888'),
(8, 'Ava', 'Jones', '1992-06-30', 'ava.jones@email.com', '4445556666'),
(9, 'Logan', 'Paul', '1990-12-05', 'logan.anderson@email.com', '2223334444'),
(10, 'Sophia', 'Moore', '1993-02-28', 'sophia.moore@email.com', '8889990000');
```

```
(10 rows affected)

Completion time: 2023-12-08T17:12:46.2501757+05:30
```



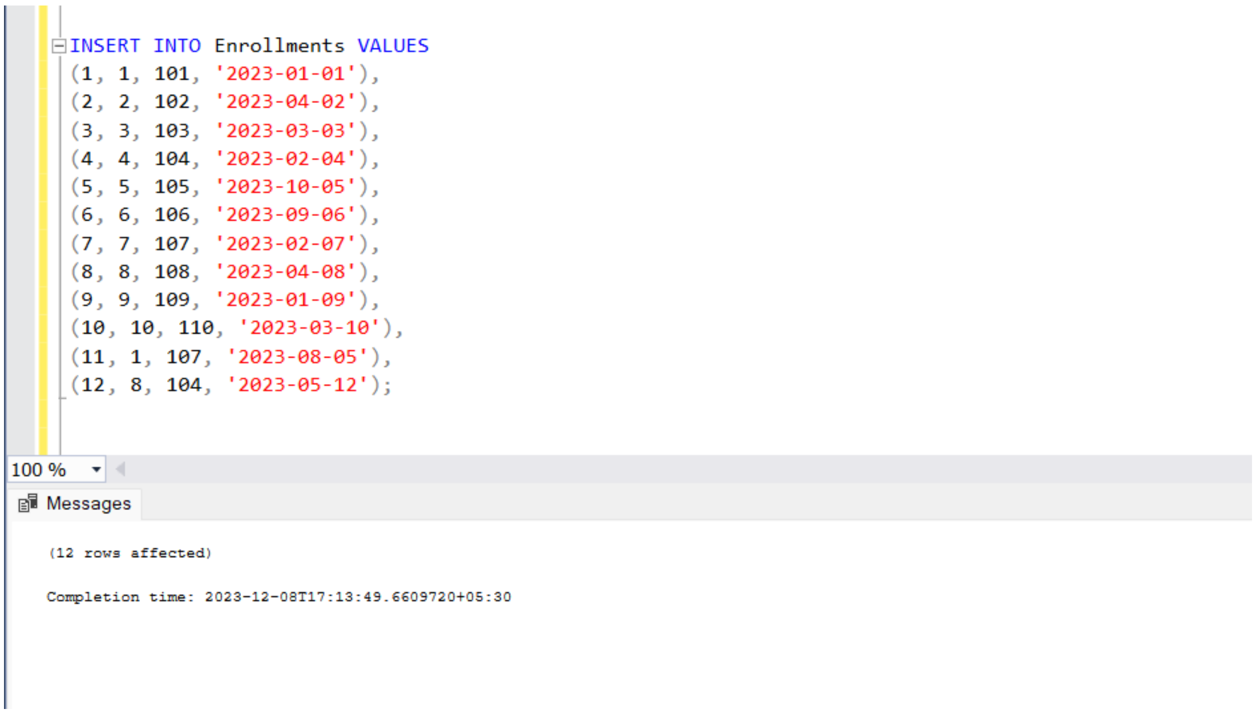
```sql
INSERT INTO Teacher VALUES
(1, 'Professor', 'Smith', 'prof.smith@email.com'),
(2, 'Dr.', 'Johnson', 'dr.johnson@email.com'),
(3, 'Ms.', 'Williams', 'ms.williams@email.com'),
(4, 'Mr.', 'Davis', 'mr.davis@email.com'),
(5, 'Professor', 'Moore', 'prof.moore@email.com'),
(6, 'Dr.', 'Anderson', 'dr.anderson@email.com'),
(7, 'Mrs.', 'Brown', 'mrs.brown@email.com'),
(8, 'Ms.', 'Miller', 'ms.miller@email.com'),
(9, 'Mr.', 'Jones', 'mr.jones@email.com'),
(10, 'Mrs.', 'Doe', 'mrs.doe@email.com');
```

```
(10 rows affected)

Completion time: 2023-12-08T17:13:04.7320352+05:30
```



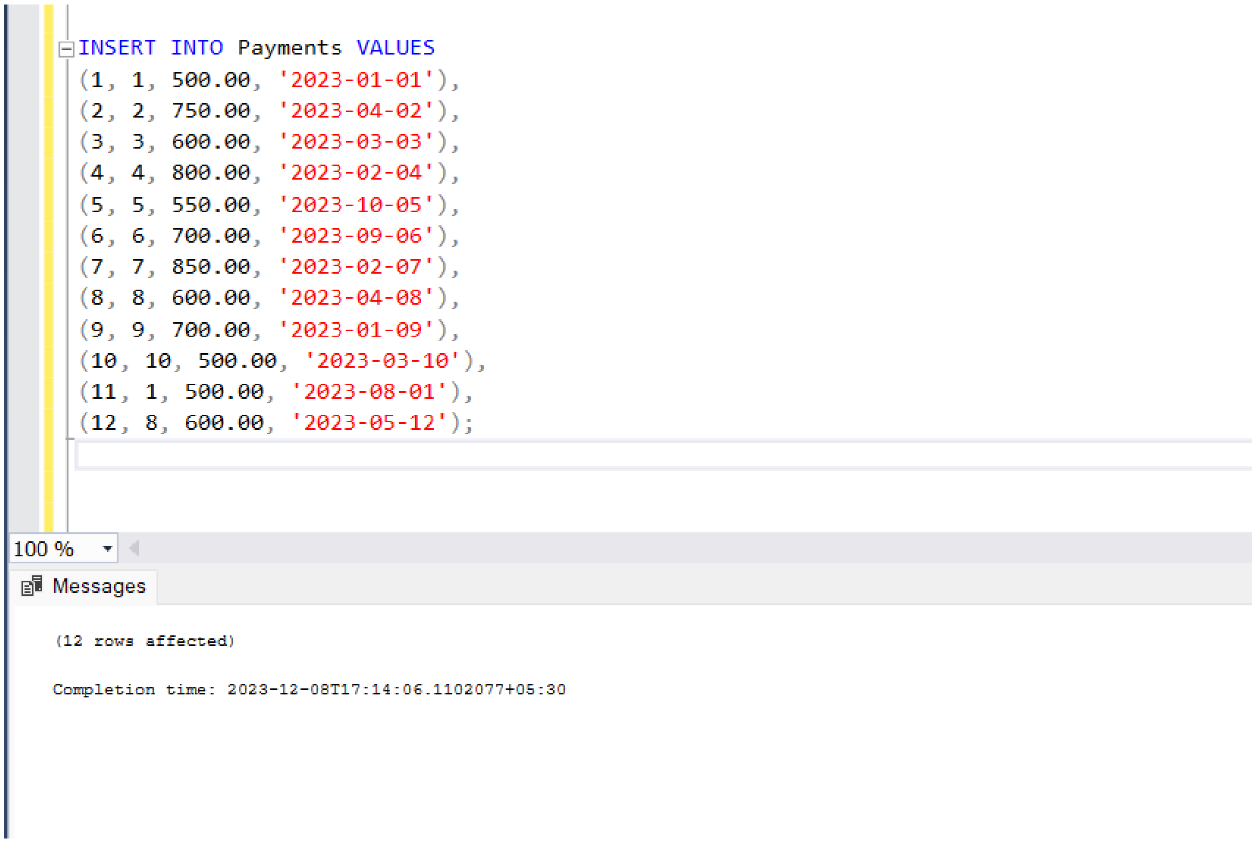
```sql
INSERT INTO Courses VALUES
(101, 'Introduction to Computer Science', 3, 1),
(102, 'Mathematics for Engineers', 4, 2),
(103, 'History of Art', 3, 3),
(104, 'Physics for Beginners', 4, 1),
(105, 'Business Ethics', 3, 2),
(106, 'Literature and Society', 3, 3),
(107, 'Chemistry Fundamentals', 4, 2),
(108, 'Psychology 101', 3, 3),
(109, 'Data Structures', 4, 1),
(110, 'Introduction to Marketing', 3, 2);
```

```
(10 rows affected)

Completion time: 2023-12-08T17:13:30.7254204+05:30
```



```
INSERT INTO Enrollments VALUES
(1, 1, 101, '2023-01-01'),
(2, 2, 102, '2023-04-02'),
(3, 3, 103, '2023-03-03'),
(4, 4, 104, '2023-02-04'),
(5, 5, 105, '2023-10-05'),
(6, 6, 106, '2023-09-06'),
(7, 7, 107, '2023-02-07'),
(8, 8, 108, '2023-04-08'),
(9, 9, 109, '2023-01-09'),
(10, 10, 110, '2023-03-10'),
(11, 1, 107, '2023-08-05'),
(12, 8, 104, '2023-05-12');
```

```
(12 rows affected)

Completion time: 2023-12-08T17:13:49.6609720+05:30
```

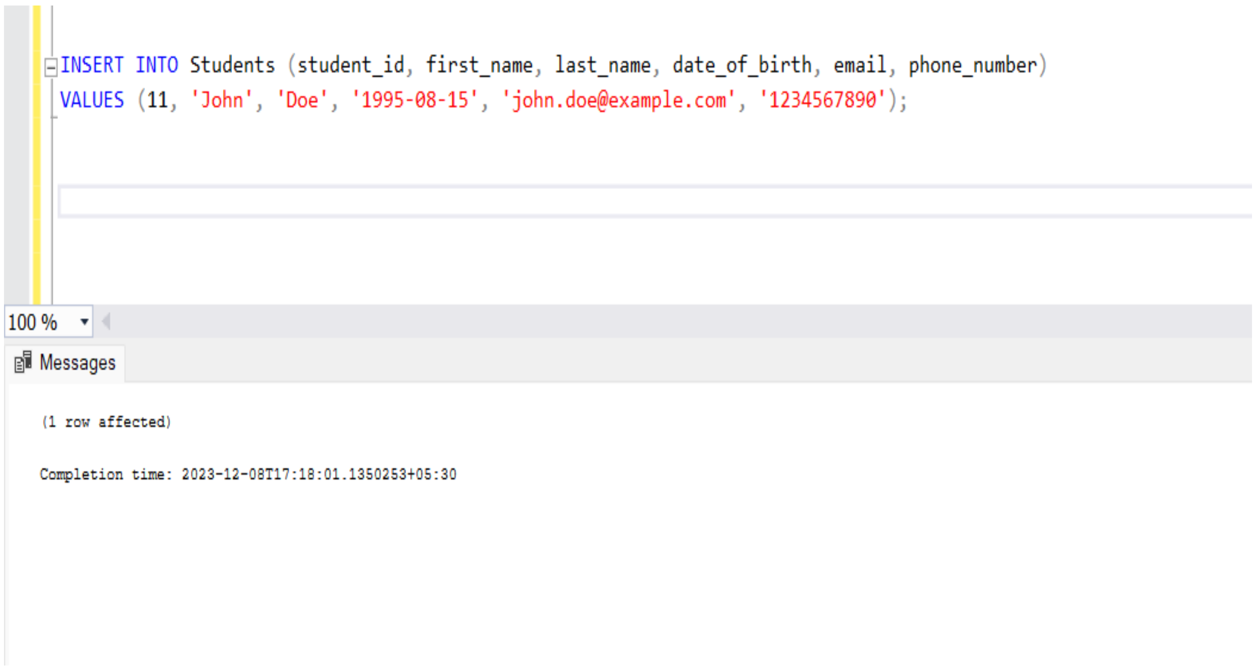


```
INSERT INTO Payments VALUES
(1, 1, 500.00, '2023-01-01'),
(2, 2, 750.00, '2023-04-02'),
(3, 3, 600.00, '2023-03-03'),
(4, 4, 800.00, '2023-02-04'),
(5, 5, 550.00, '2023-10-05'),
(6, 6, 700.00, '2023-09-06'),
(7, 7, 850.00, '2023-02-07'),
(8, 8, 600.00, '2023-04-08'),
(9, 9, 700.00, '2023-01-09'),
(10, 10, 500.00, '2023-03-10'),
(11, 1, 500.00, '2023-08-01'),
(12, 8, 600.00, '2023-05-12');
```

```
(12 rows affected)

Completion time: 2023-12-08T17:14:06.1102077+05:30
```

Tasks 2: Select, Where, Between, AND, LIKE:

1.



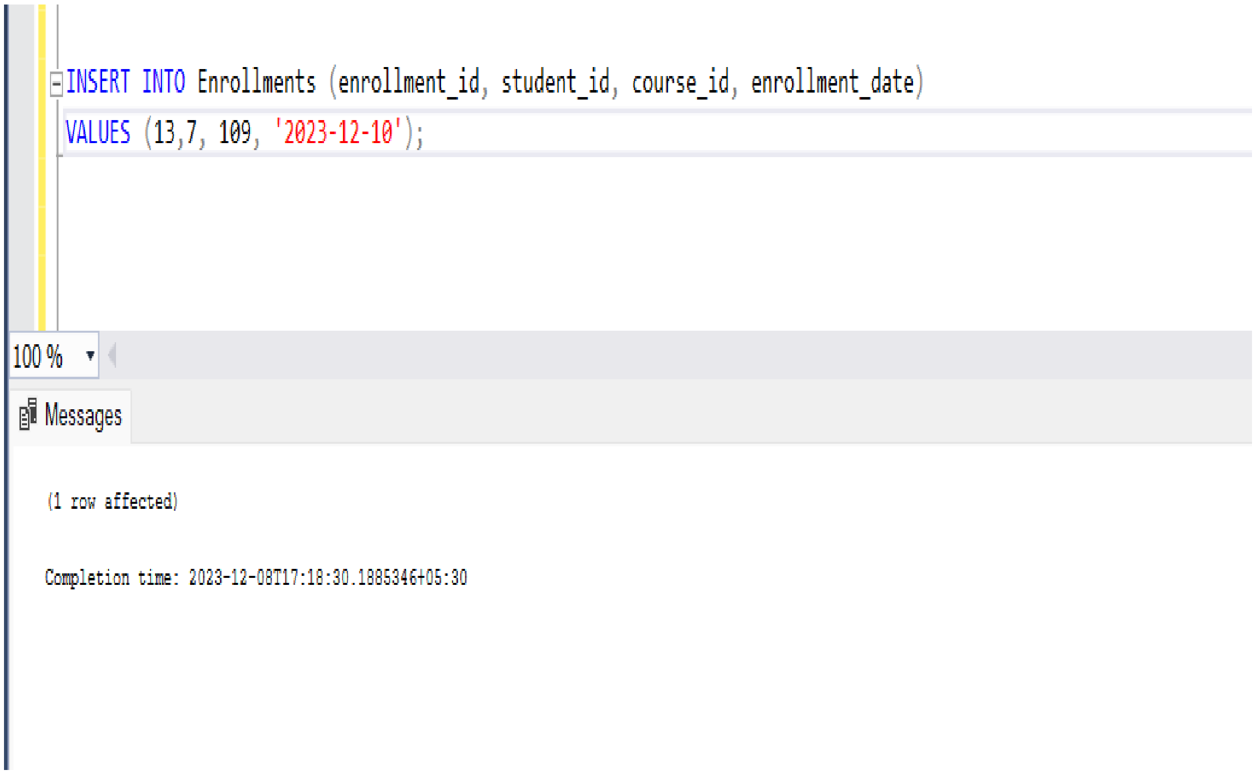
```sql
INSERT INTO Students (student_id, first_name, last_name, date_of_birth, email, phone_number)
VALUES (11, 'John', 'Doe', '1995-08-15', 'john.doe@example.com', '1234567890');
```

100 %

Messages

```
(1 row affected)

Completion time: 2023-12-08T17:18:01.1350253+05:30
```

2.



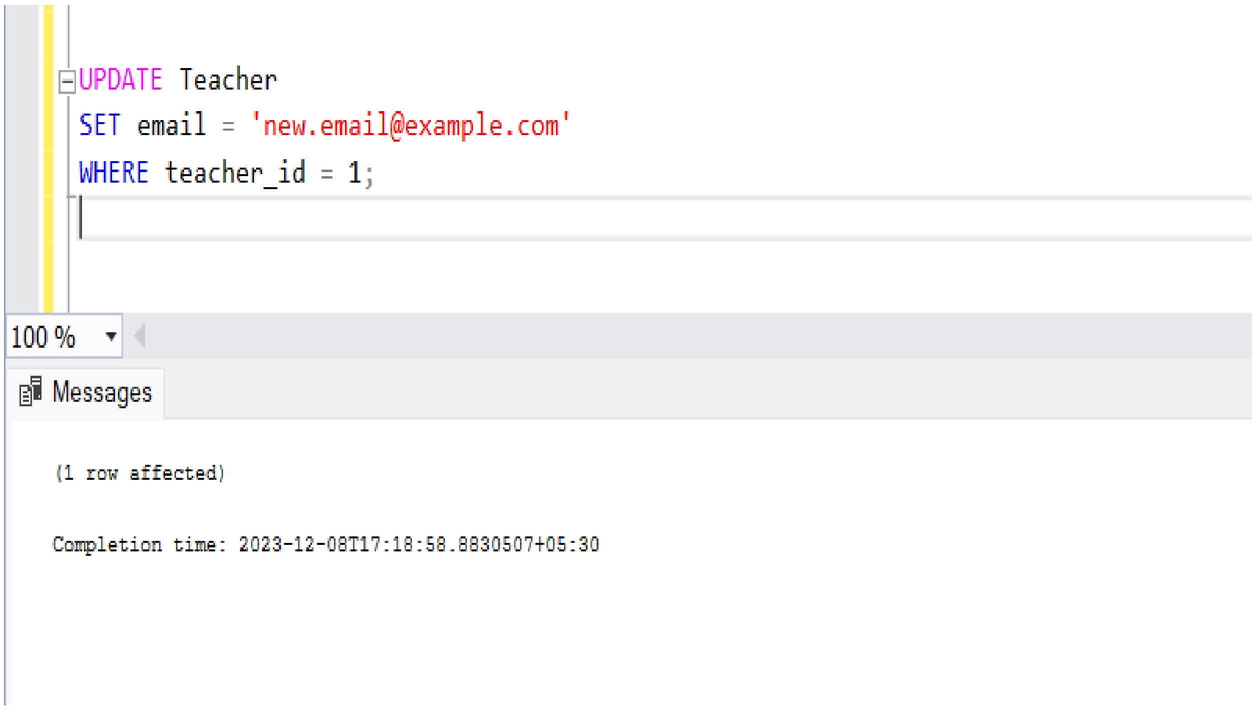
```sql
INSERT INTO Enrollments (enrollment_id, student_id, course_id, enrollment_date)
VALUES (13,7, 109, '2023-12-10');
```

100 %

Messages

```
(1 row affected)

Completion time: 2023-12-08T17:18:30.1885346+05:30
```

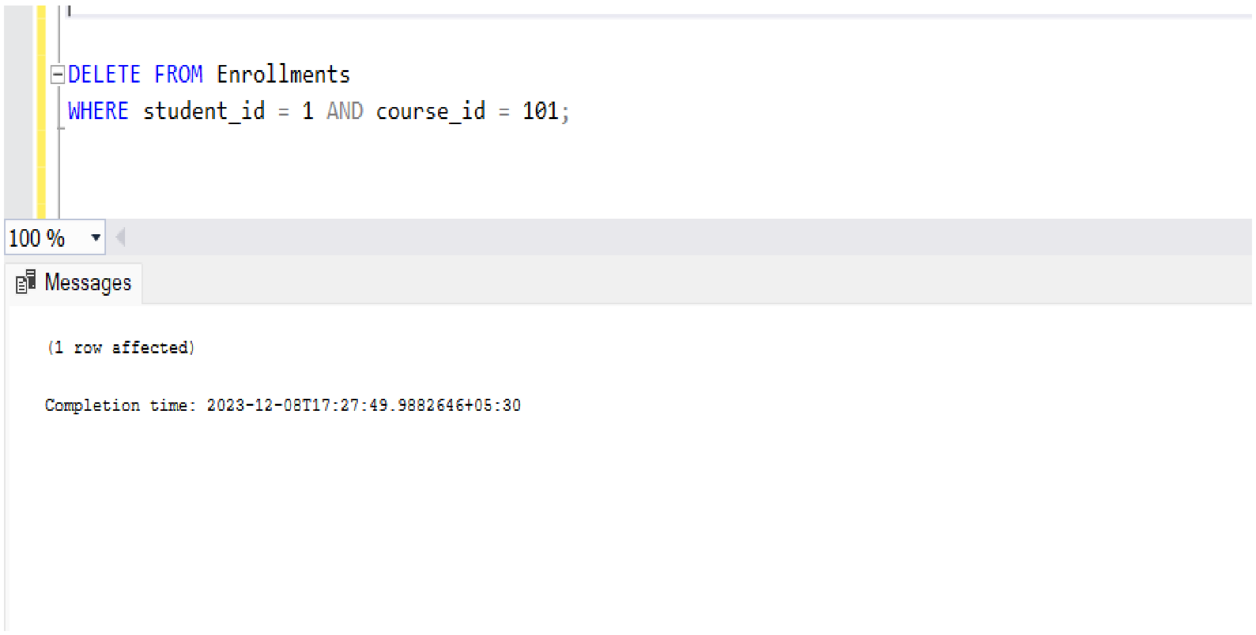
3.



```sql
UPDATE Teacher
SET email = 'new.email@example.com'
WHERE teacher_id = 1;
```

```
(1 row affected)

Completion time: 2023-12-08T17:18:58.8830507+05:30
```

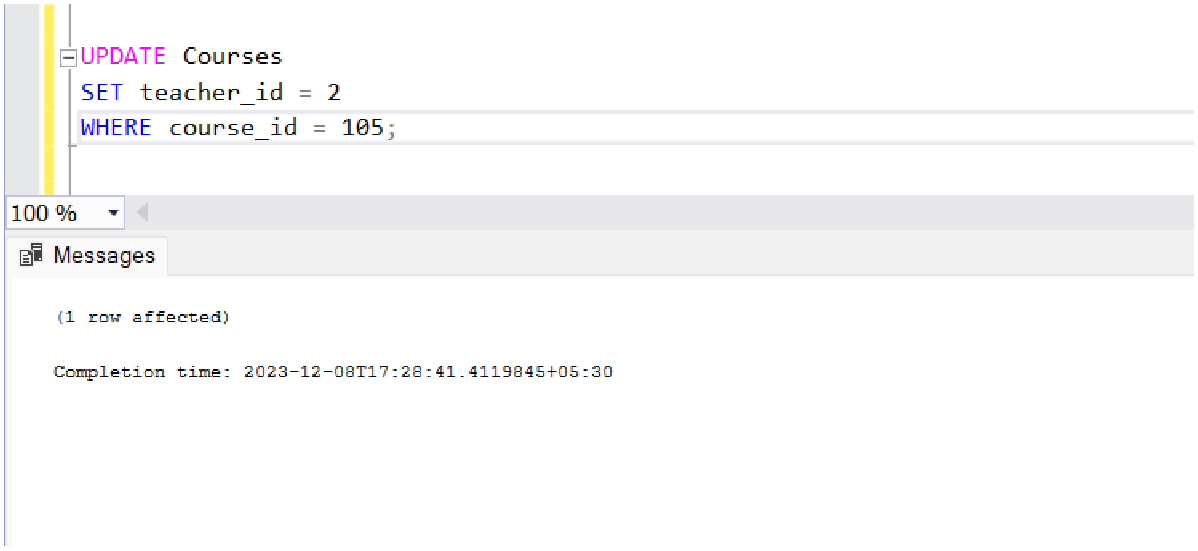
4.



```sql
DELETE FROM Enrollments
WHERE student_id = 1 AND course_id = 101;
```

```
(1 row affected)

Completion time: 2023-12-08T17:27:49.9882646+05:30
```

5.



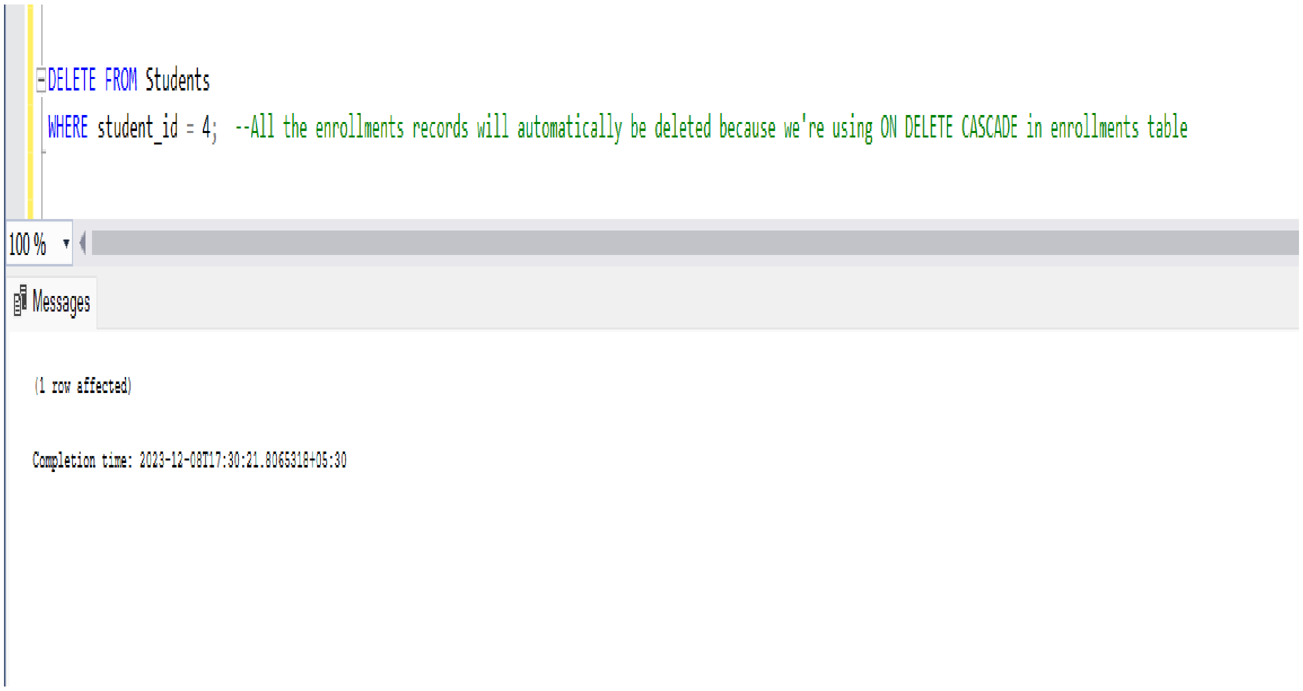
```
UPDATE Courses
SET teacher_id = 2
WHERE course_id = 105;
```

100 %

Messages

```
(1 row affected)

Completion time: 2023-12-08T17:28:41.4119845+05:30
```

6.



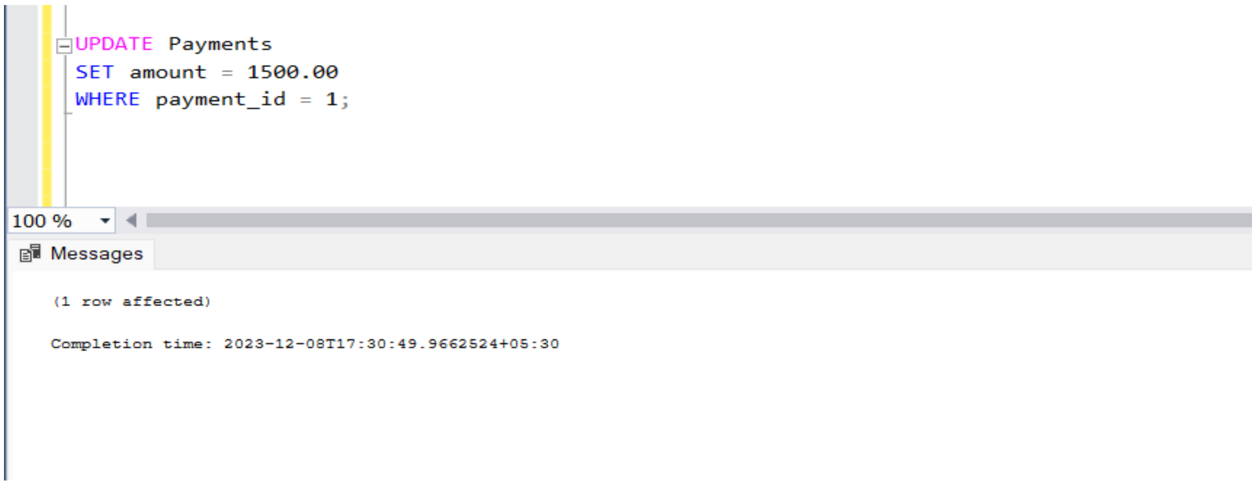
```
DELETE FROM Students
WHERE student_id = 4;  --All the enrollments records will automatically be deleted because we're using ON DELETE CASCADE in enrollments table
```

100 %

Messages

```
(1 row affected)

Completion time: 2023-12-08T17:30:21.8065318+05:30
```

7.

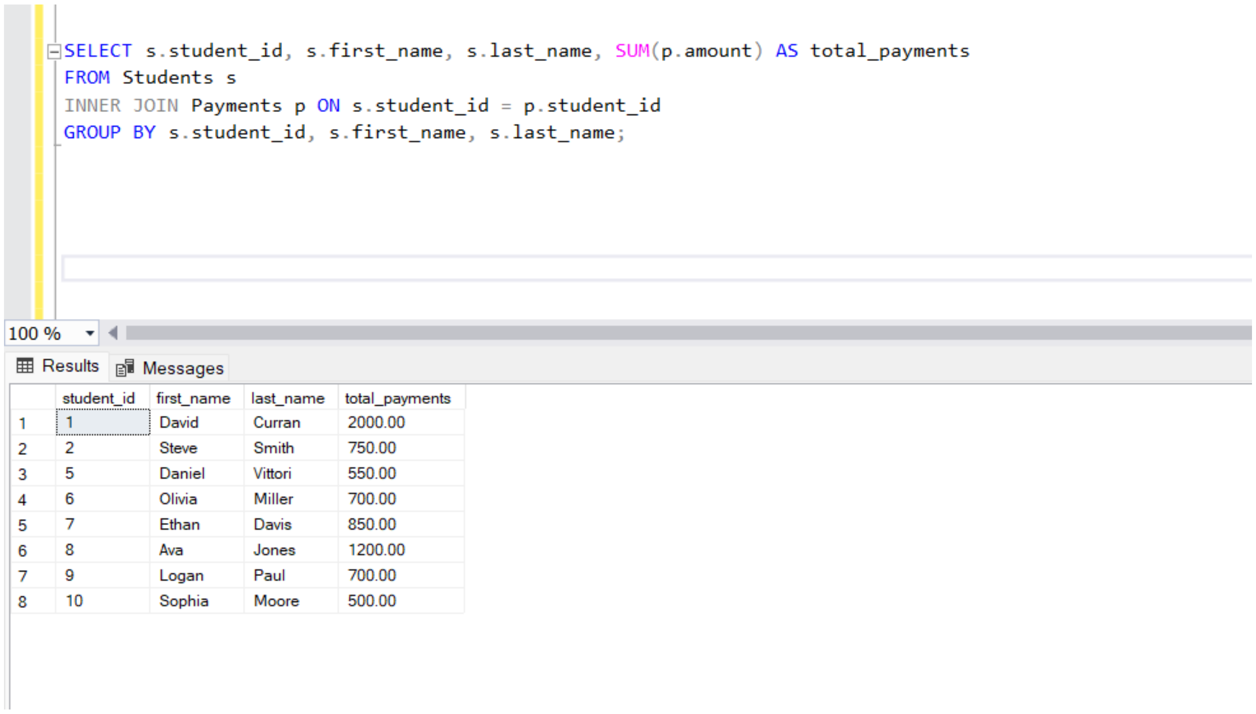


```
UPDATE Payments
SET amount = 1500.00
WHERE payment_id = 1;
```

```
(1 row affected)

Completion time: 2023-12-08T17:30:49.9662524+05:30
```

Task 3. Aggregate functions, Having, Order By, GroupBy and Joins:

1.



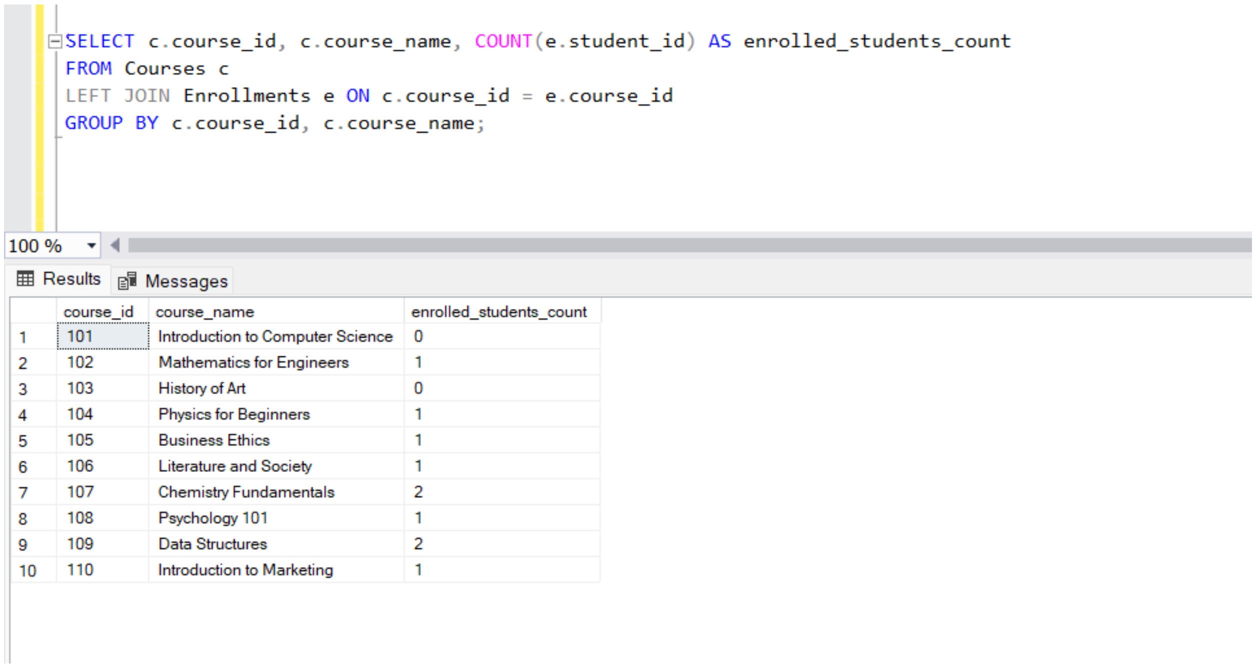
```
SELECT s.student_id, s.first_name, s.last_name, SUM(p.amount) AS total_payments
FROM Students s
INNER JOIN Payments p ON s.student_id = p.student_id
GROUP BY s.student_id, s.first_name, s.last_name;
```

100 %

Results | Messages

	student_id	first_name	last_name	total_payments
1	1	David	Curran	2000.00
2	2	Steve	Smith	750.00
3	5	Daniel	Vittori	550.00
4	6	Olivia	Miller	700.00
5	7	Ethan	Davis	850.00
6	8	Ava	Jones	1200.00
7	9	Logan	Paul	700.00
8	10	Sophia	Moore	500.00

2.



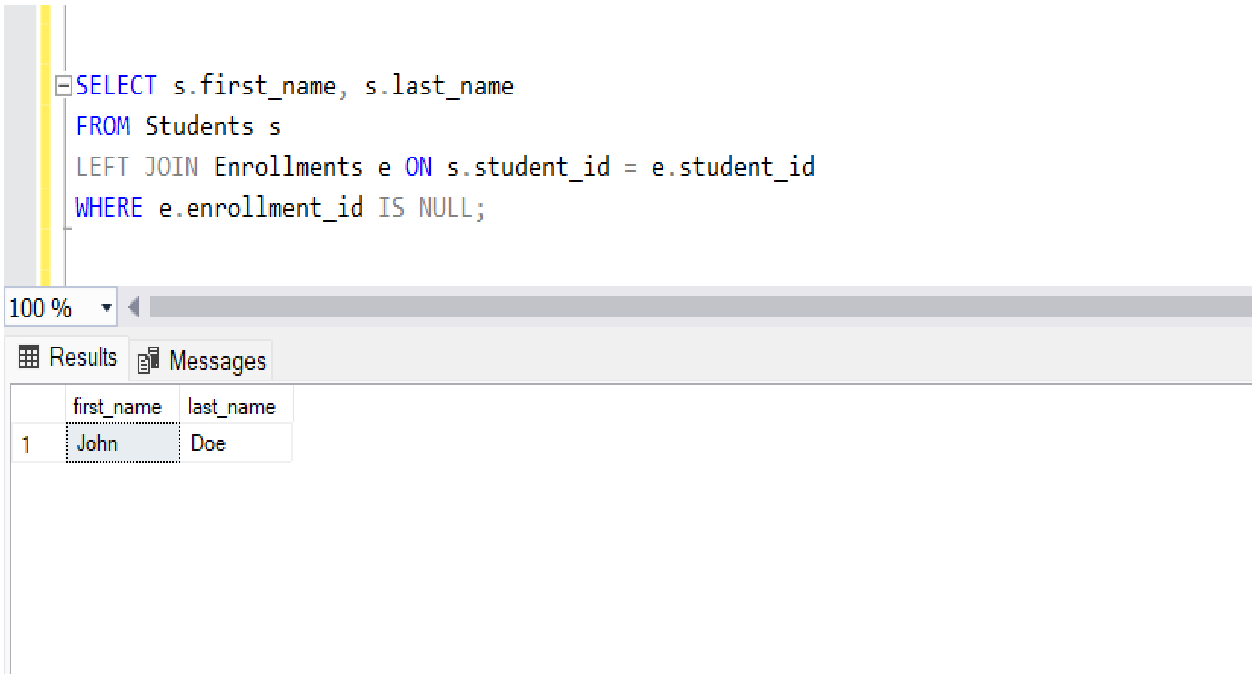
```
SELECT c.course_id, c.course_name, COUNT(e.student_id) AS enrolled_students_count
FROM Courses c
LEFT JOIN Enrollments e ON c.course_id = e.course_id
GROUP BY c.course_id, c.course_name;
```

100 %

Results | Messages

	course_id	course_name	enrolled_students_count
1	101	Introduction to Computer Science	0
2	102	Mathematics for Engineers	1
3	103	History of Art	0
4	104	Physics for Beginners	1
5	105	Business Ethics	1
6	106	Literature and Society	1
7	107	Chemistry Fundamentals	2
8	108	Psychology 101	1
9	109	Data Structures	2
10	110	Introduction to Marketing	1

3.



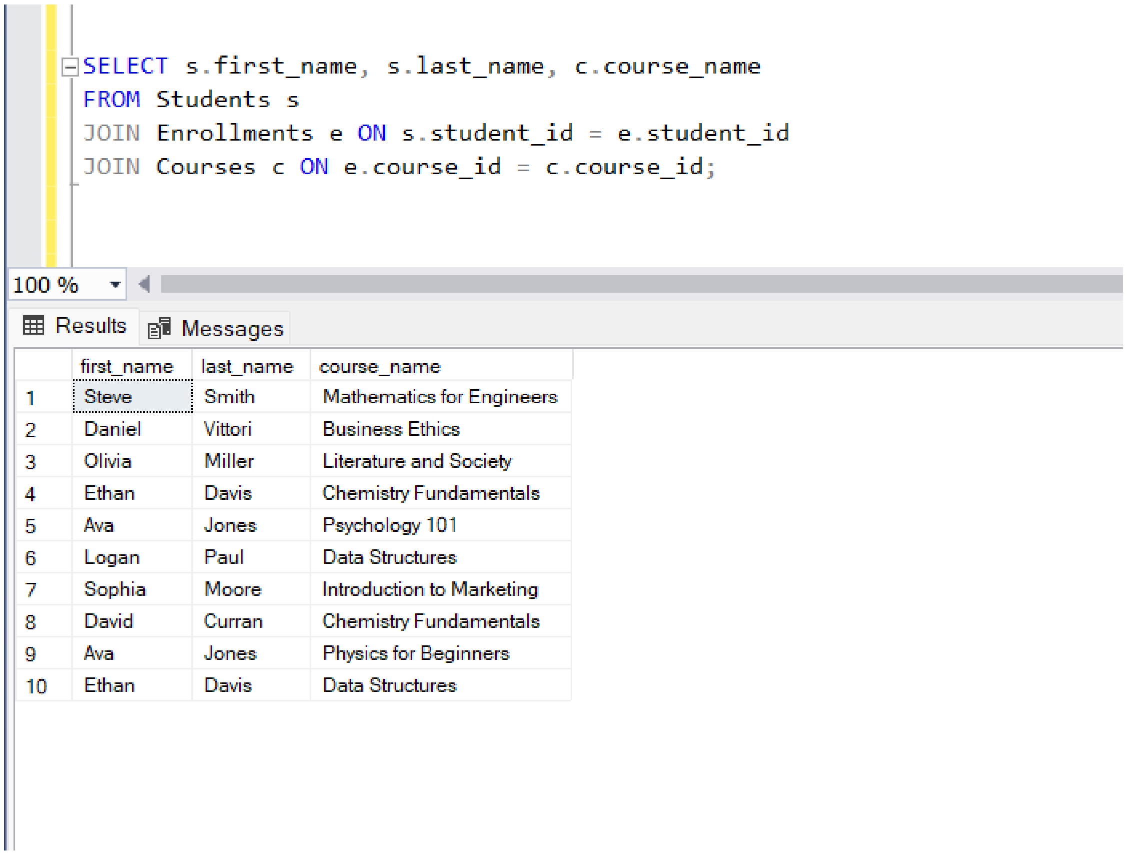
```
SELECT s.first_name, s.last_name
FROM Students s
LEFT JOIN Enrollments e ON s.student_id = e.student_id
WHERE e.enrollment_id IS NULL;
```

100 %

Results | Messages

	first_name	last_name
1	John	Doe

4.



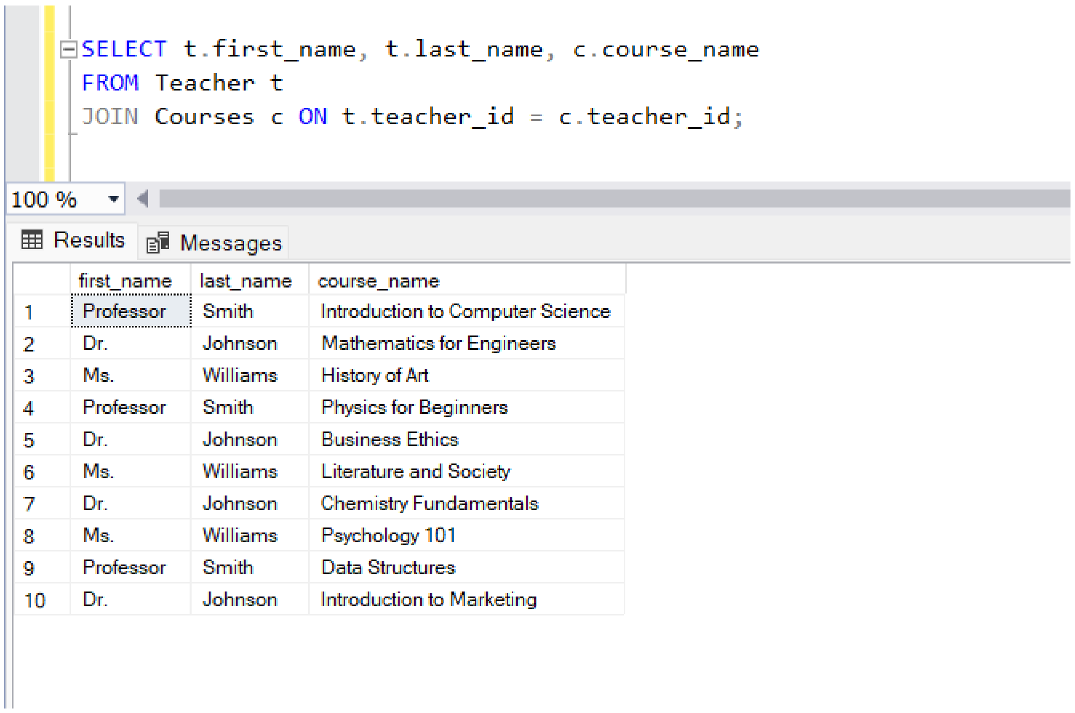
```
SELECT s.first_name, s.last_name, c.course_name
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id
JOIN Courses c ON e.course_id = c.course_id;
```

100 %

Results | Messages

	first_name	last_name	course_name
1	Steve	Smith	Mathematics for Engineers
2	Daniel	Vittori	Business Ethics
3	Olivia	Miller	Literature and Society
4	Ethan	Davis	Chemistry Fundamentals
5	Ava	Jones	Psychology 101
6	Logan	Paul	Data Structures
7	Sophia	Moore	Introduction to Marketing
8	David	Curran	Chemistry Fundamentals
9	Ava	Jones	Physics for Beginners
10	Ethan	Davis	Data Structures

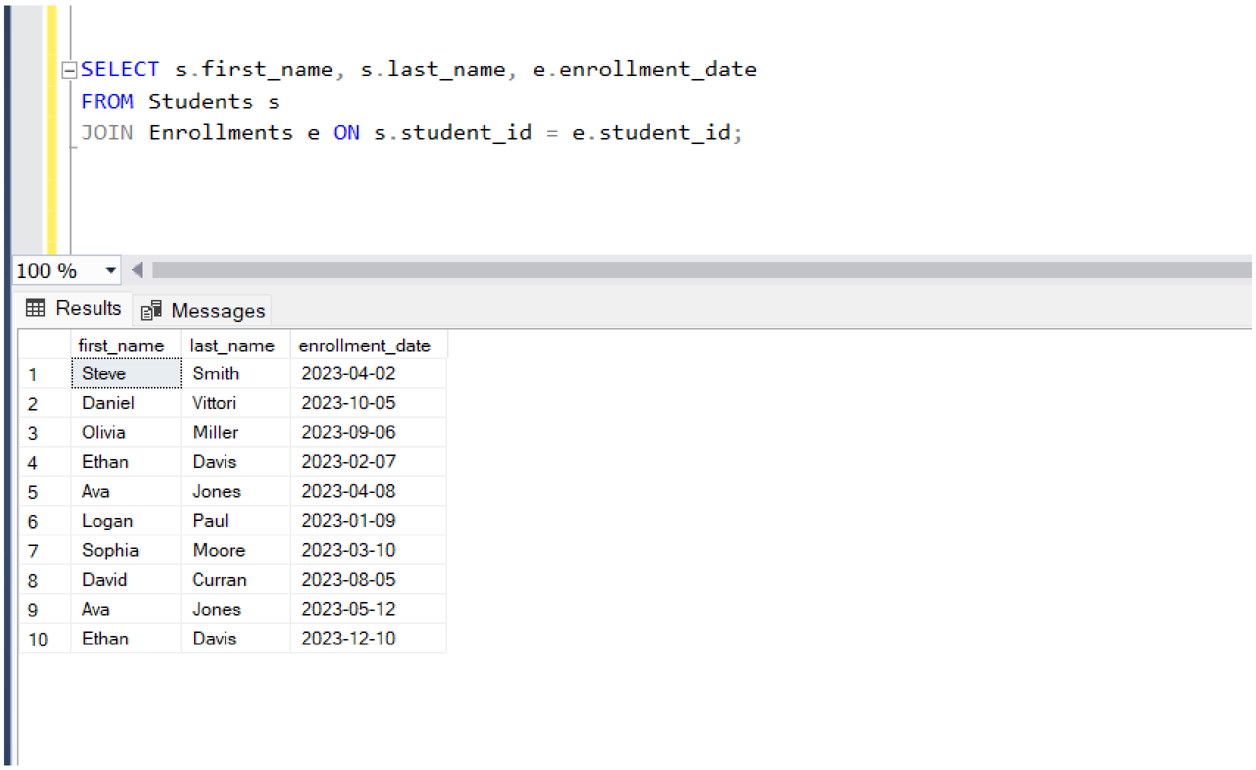
5.



```sql
SELECT t.first_name, t.last_name, c.course_name
FROM Teacher t
JOIN Courses c ON t.teacher_id = c.teacher_id;
```

	first_name	last_name	course_name
1	Professor	Smith	Introduction to Computer Science
2	Dr.	Johnson	Mathematics for Engineers
3	Ms.	Williams	History of Art
4	Professor	Smith	Physics for Beginners
5	Dr.	Johnson	Business Ethics
6	Ms.	Williams	Literature and Society
7	Dr.	Johnson	Chemistry Fundamentals
8	Ms.	Williams	Psychology 101
9	Professor	Smith	Data Structures
10	Dr.	Johnson	Introduction to Marketing

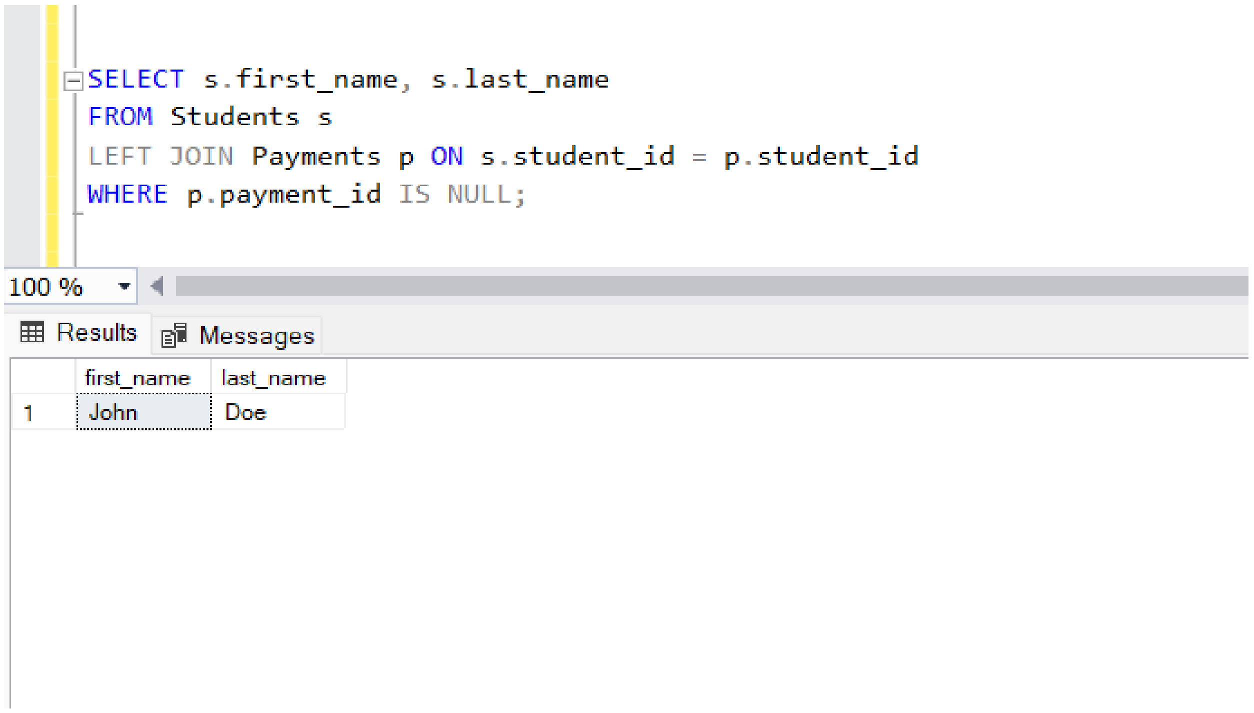
6.



```sql
SELECT s.first_name, s.last_name, e.enrollment_date
FROM Students s
JOIN Enrollments e ON s.student_id = e.student_id;
```

	first_name	last_name	enrollment_date
1	Steve	Smith	2023-04-02
2	Daniel	Vittori	2023-10-05
3	Olivia	Miller	2023-09-06
4	Ethan	Davis	2023-02-07
5	Ava	Jones	2023-04-08
6	Logan	Paul	2023-01-09
7	Sophia	Moore	2023-03-10
8	David	Curran	2023-08-05
9	Ava	Jones	2023-05-12
10	Ethan	Davis	2023-12-10

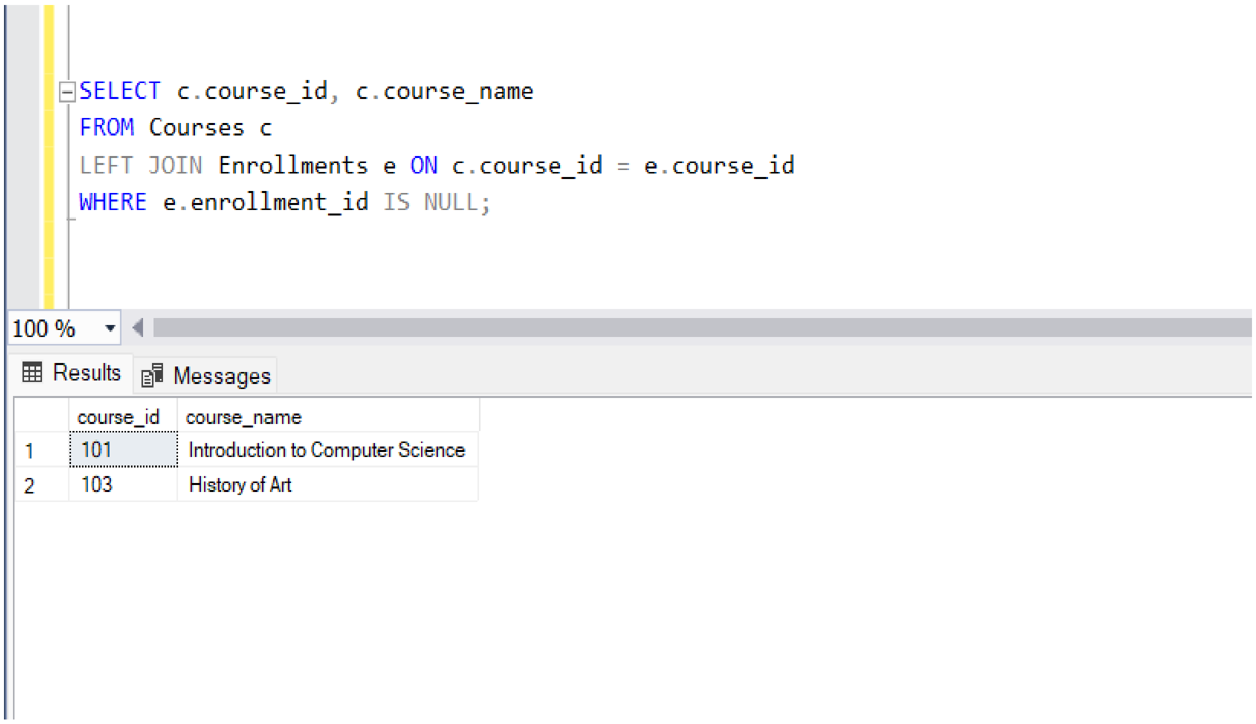
7.



```sql
SELECT s.first_name, s.last_name
FROM Students s
LEFT JOIN Payments p ON s.student_id = p.student_id
WHERE p.payment_id IS NULL;
```

	first_name	last_name
1	John	Doe

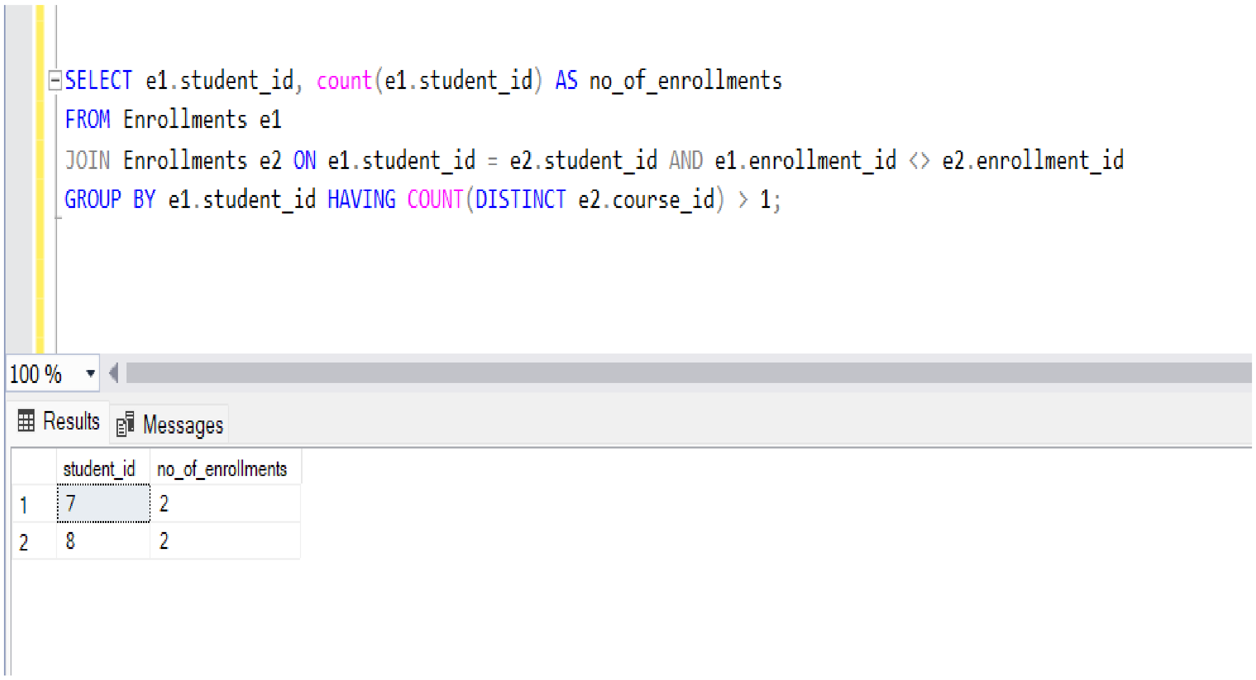
8.



```sql
SELECT c.course_id, c.course_name
FROM Courses c
LEFT JOIN Enrollments e ON c.course_id = e.course_id
WHERE e.enrollment_id IS NULL;
```

	course_id	course_name
1	101	Introduction to Computer Science
2	103	History of Art

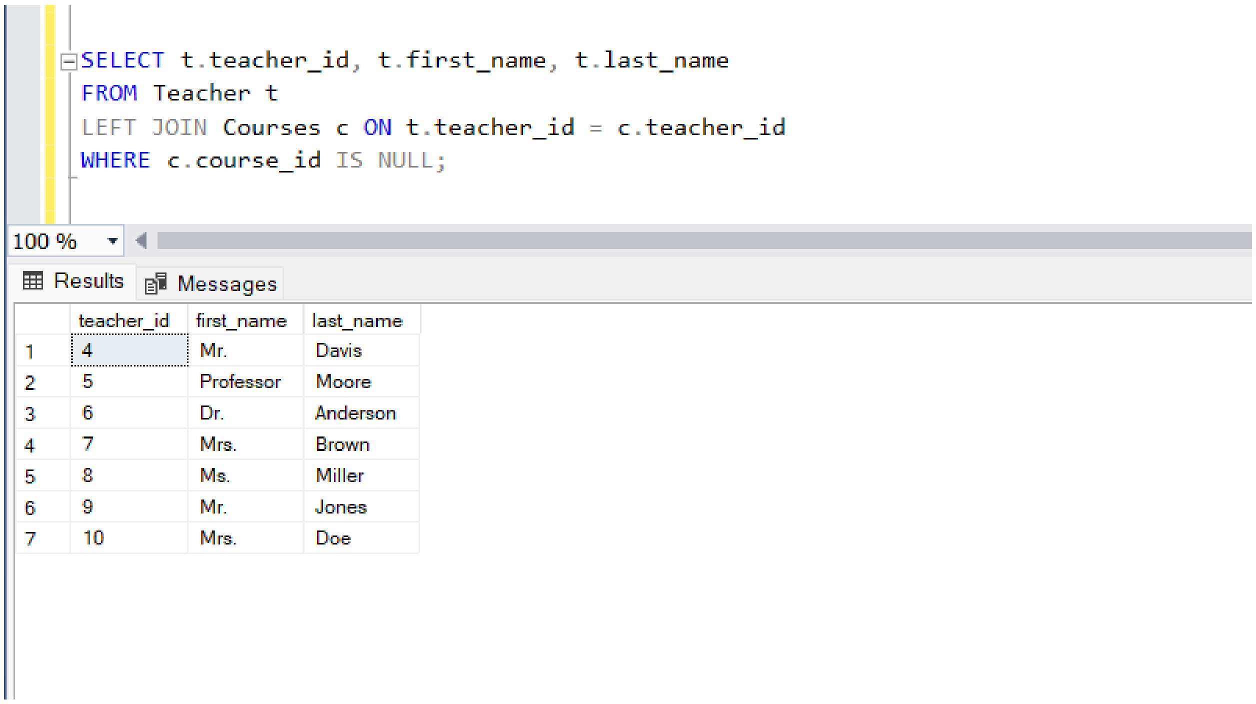
9.



```
SELECT e1.student_id, count(e1.student_id) AS no_of_enrollments
FROM Enrollments e1
JOIN Enrollments e2 ON e1.student_id = e2.student_id AND e1.enrollment_id <> e2.enrollment_id
GROUP BY e1.student_id HAVING COUNT(DISTINCT e2.course_id) > 1;
```

	student_id	no_of_enrollments
1	7	2
2	8	2

10.

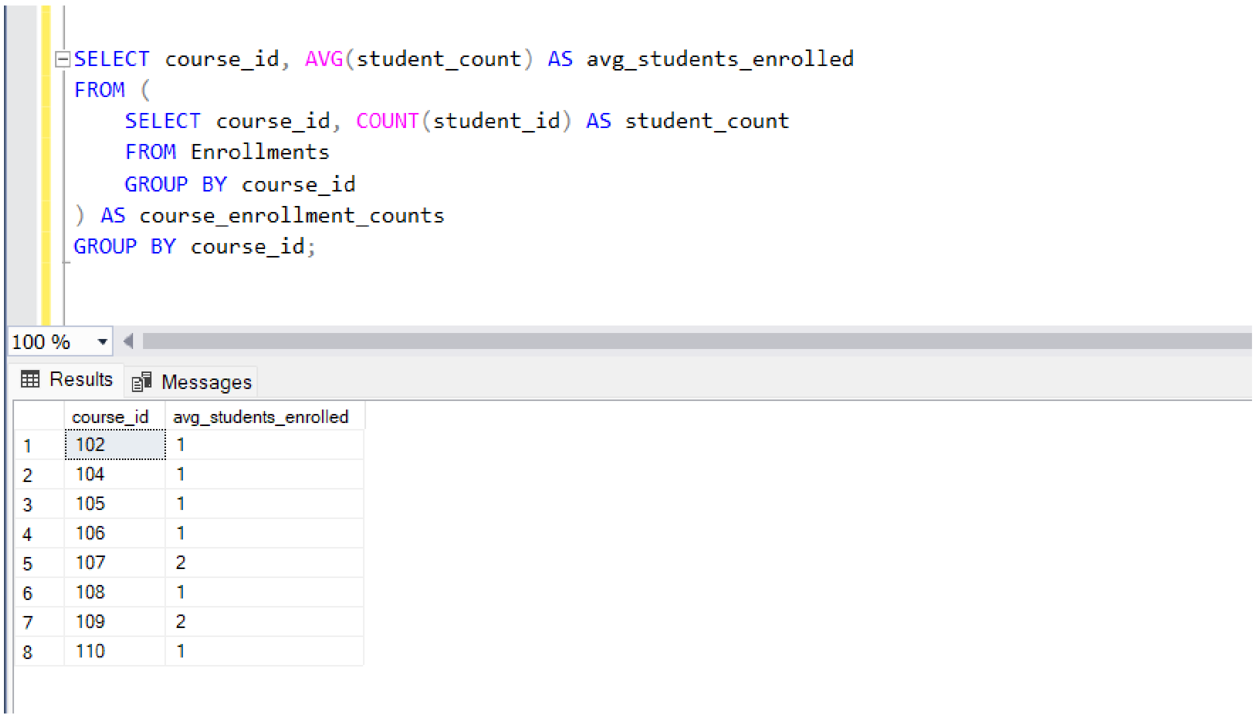


```
SELECT t.teacher_id, t.first_name, t.last_name
FROM Teacher t
LEFT JOIN Courses c ON t.teacher_id = c.teacher_id
WHERE c.course_id IS NULL;
```

	teacher_id	first_name	last_name
1	4	Mr.	Davis
2	5	Professor	Moore
3	6	Dr.	Anderson
4	7	Mrs.	Brown
5	8	Ms.	Miller
6	9	Mr.	Jones
7	10	Mrs.	Doe

Task 4. Subquery and its type:

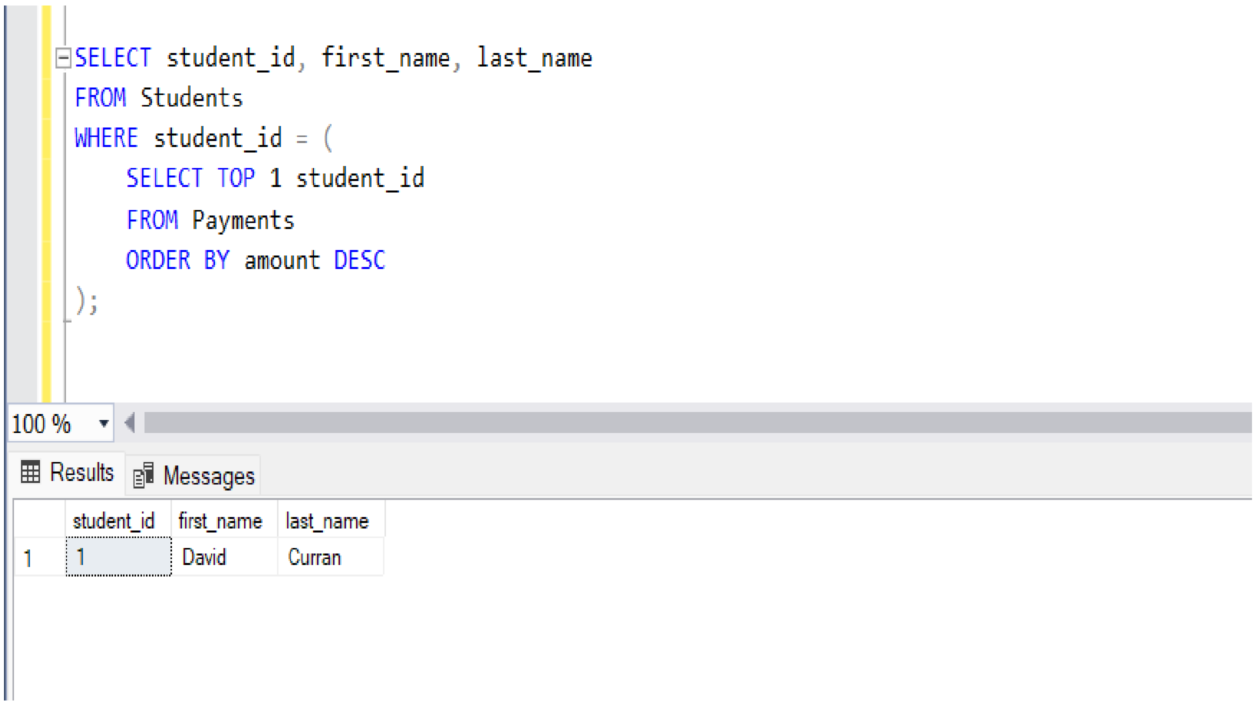
1.



```sql
SELECT course_id, AVG(student_count) AS avg_students_enrolled
FROM (
    SELECT course_id, COUNT(student_id) AS student_count
    FROM Enrollments
    GROUP BY course_id
) AS course_enrollment_counts
GROUP BY course_id;
```

	course_id	avg_students_enrolled
1	102	1
2	104	1
3	105	1
4	106	1
5	107	2
6	108	1
7	109	2
8	110	1

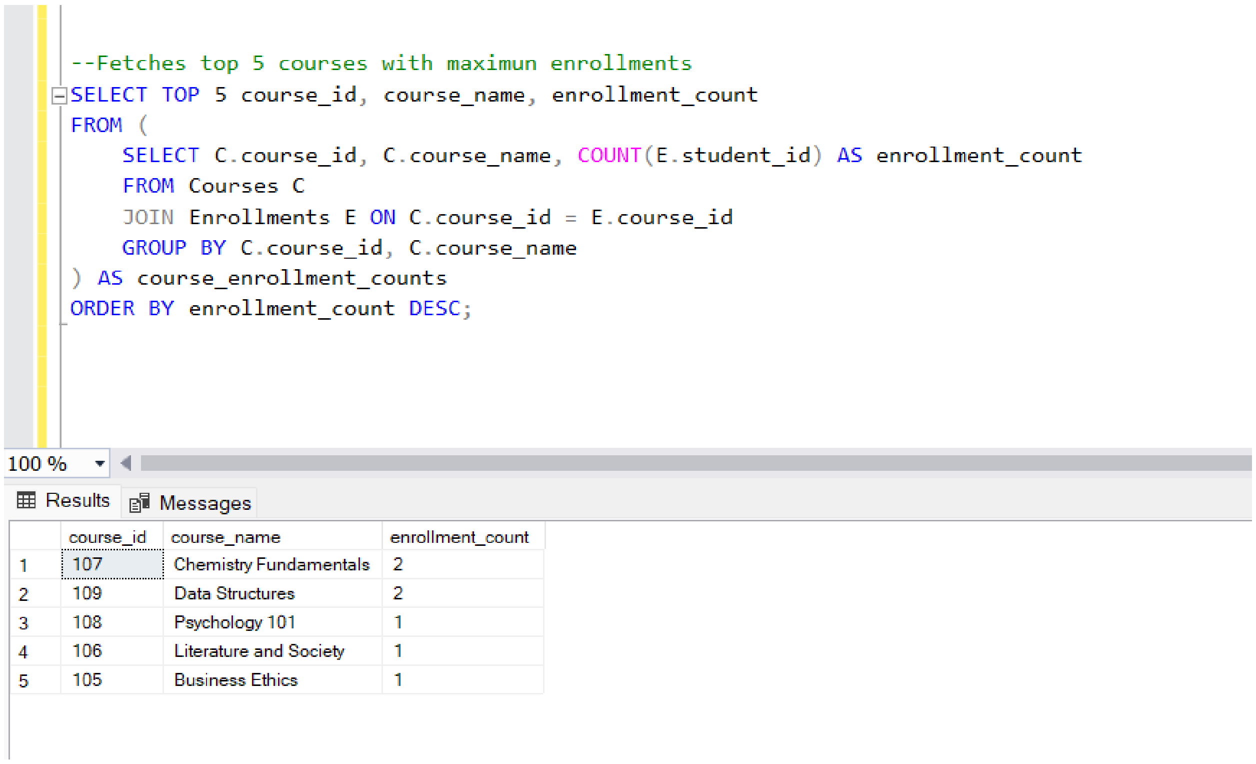
2.



```sql
SELECT student_id, first_name, last_name
FROM Students
WHERE student_id = (
    SELECT TOP 1 student_id
    FROM Payments
    ORDER BY amount DESC
);
```

	student_id	first_name	last_name
1	1	David	Curran

3.



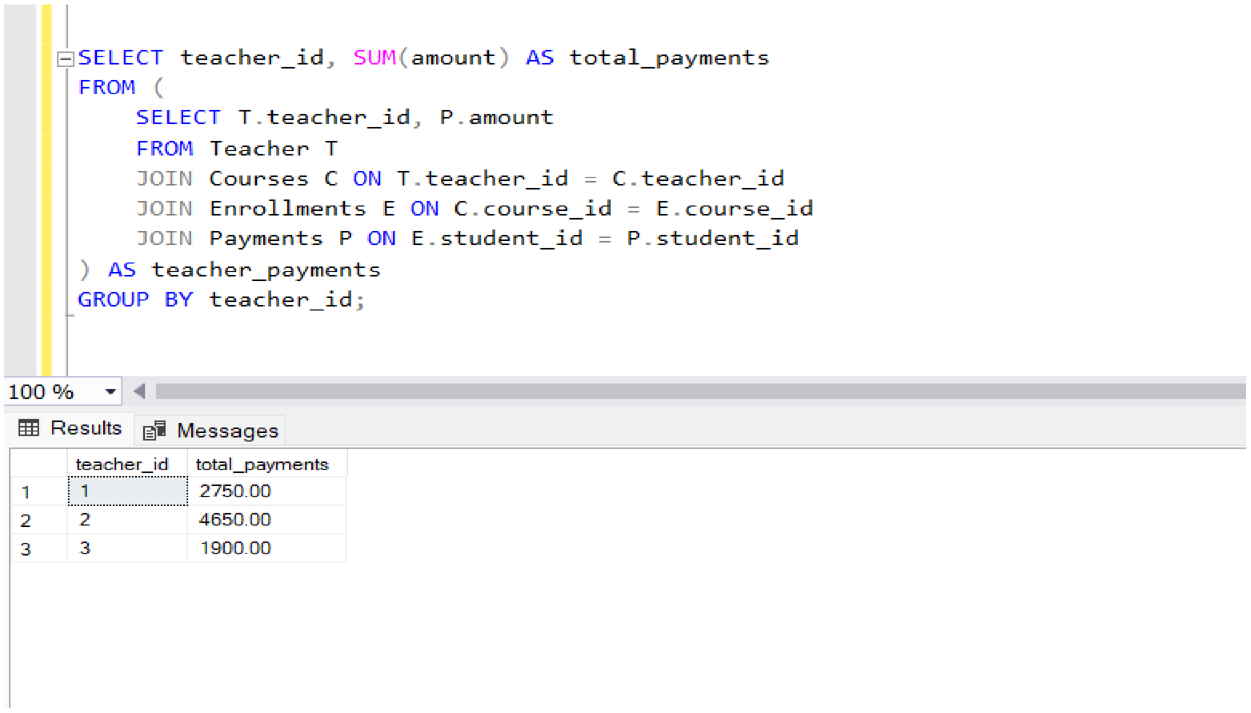
```
--Fetches top 5 courses with maximun enrollments
SELECT TOP 5 course_id, course_name, enrollment_count
FROM (
    SELECT C.course_id, C.course_name, COUNT(E.student_id) AS enrollment_count
    FROM Courses C
    JOIN Enrollments E ON C.course_id = E.course_id
    GROUP BY C.course_id, C.course_name
) AS course_enrollment_counts
ORDER BY enrollment_count DESC;
```

100 %

Results | Messages

	course_id	course_name	enrollment_count
1	107	Chemistry Fundamentals	2
2	109	Data Structures	2
3	108	Psychology 101	1
4	106	Literature and Society	1
5	105	Business Ethics	1

4.



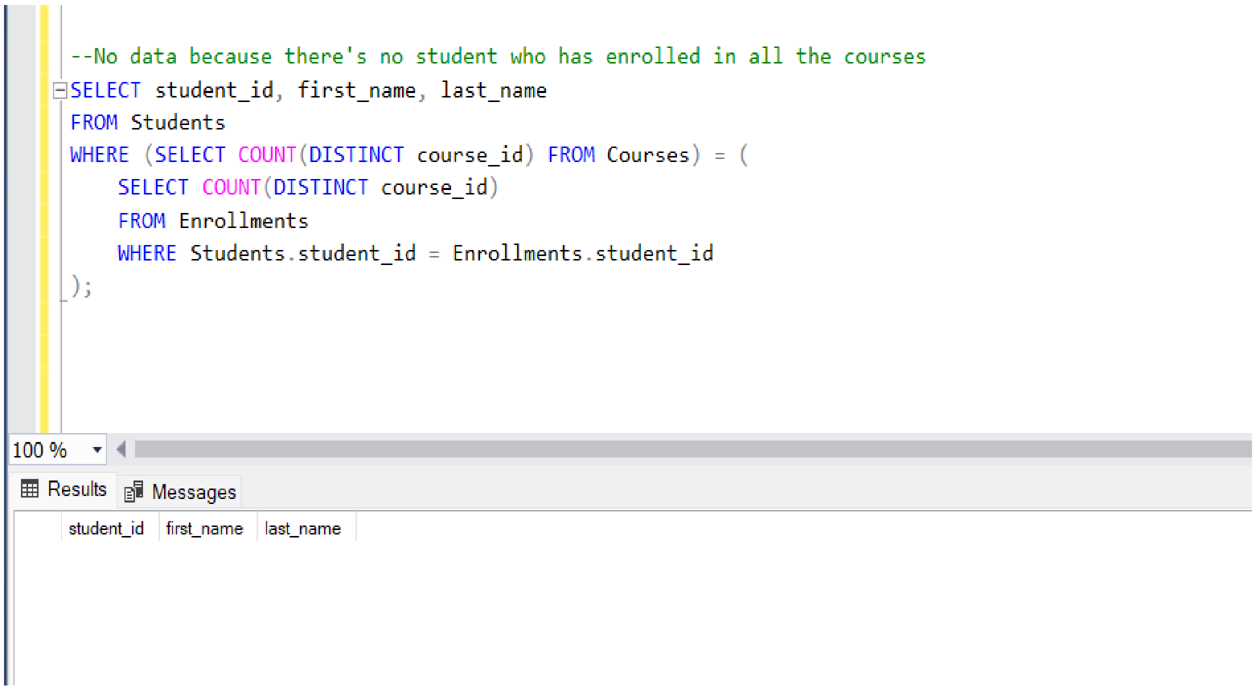
```
SELECT teacher_id, SUM(amount) AS total_payments
FROM (
    SELECT T.teacher_id, P.amount
    FROM Teacher T
    JOIN Courses C ON T.teacher_id = C.teacher_id
    JOIN Enrollments E ON C.course_id = E.course_id
    JOIN Payments P ON E.student_id = P.student_id
) AS teacher_payments
GROUP BY teacher_id;
```

100 %

Results | Messages

	teacher_id	total_payments
1	1	2750.00
2	2	4650.00
3	3	1900.00

5.



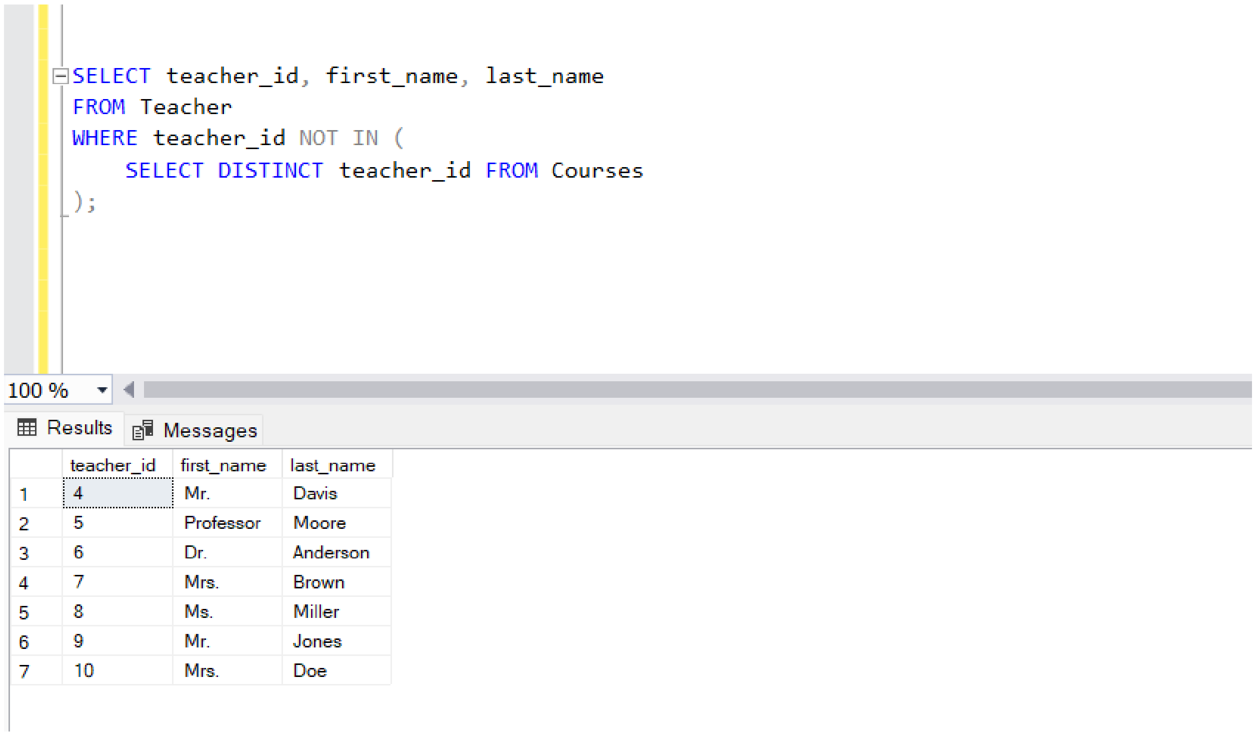
```
--No data because there's no student who has enrolled in all the courses
SELECT student_id, first_name, last_name
FROM Students
WHERE (SELECT COUNT(DISTINCT course_id) FROM Courses) = (
    SELECT COUNT(DISTINCT course_id)
    FROM Enrollments
    WHERE Students.student_id = Enrollments.student_id
);
```

100 %

Results | Messages

student_id	first_name	last_name

6.



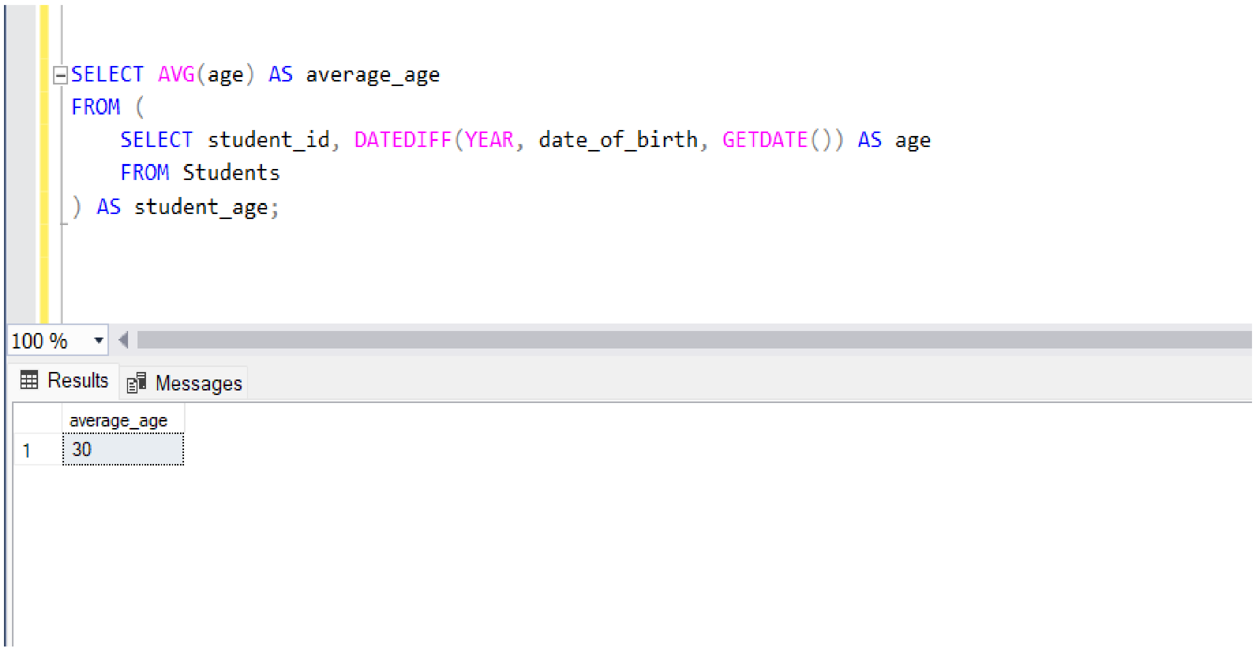
```
SELECT teacher_id, first_name, last_name
FROM Teacher
WHERE teacher_id NOT IN (
    SELECT DISTINCT teacher_id FROM Courses
);
```

100 %

Results | Messages

	teacher_id	first_name	last_name
1	4	Mr.	Davis
2	5	Professor	Moore
3	6	Dr.	Anderson
4	7	Mrs.	Brown
5	8	Ms.	Miller
6	9	Mr.	Jones
7	10	Mrs.	Doe

7.



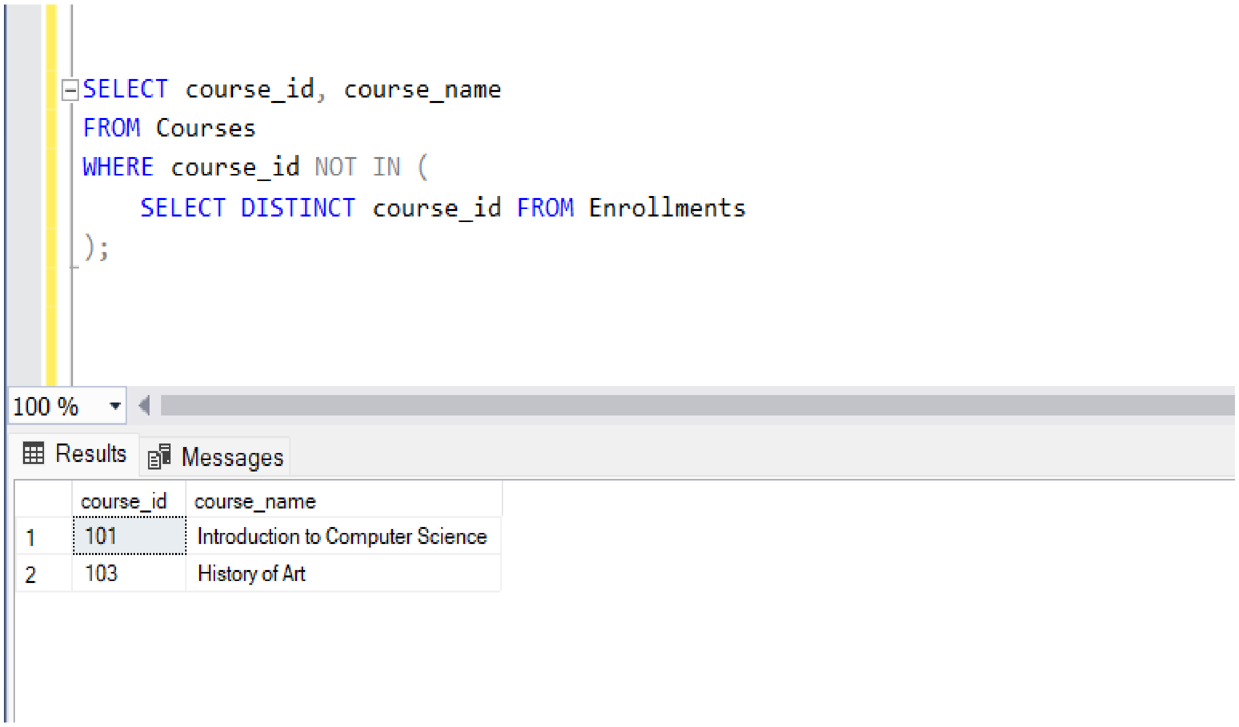
```
SELECT AVG(age) AS average_age
FROM (
    SELECT student_id, DATEDIFF(YEAR, date_of_birth, GETDATE()) AS age
    FROM Students
) AS student_age;
```

100 %

Results | Messages

	average_age
1	30

8.



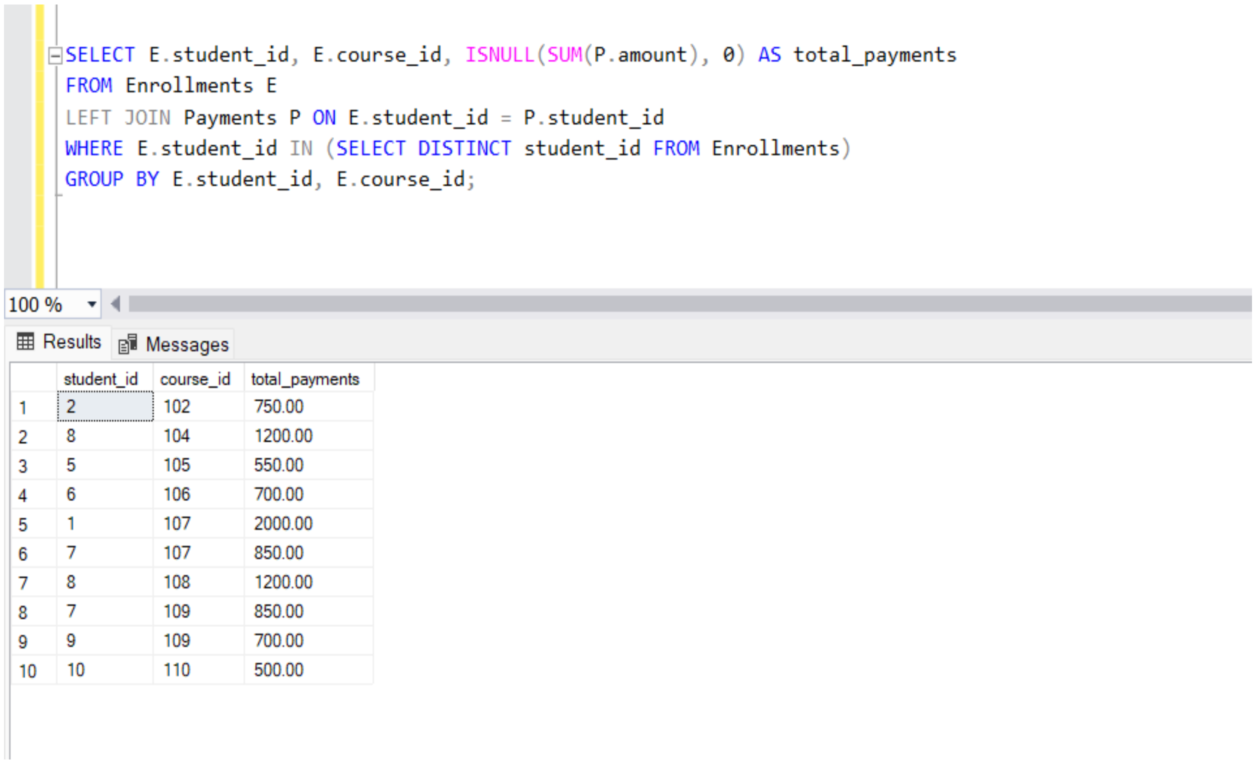
```
SELECT course_id, course_name
FROM Courses
WHERE course_id NOT IN (
    SELECT DISTINCT course_id FROM Enrollments
);
```

100 %

Results | Messages

	course_id	course_name
1	101	Introduction to Computer Science
2	103	History of Art

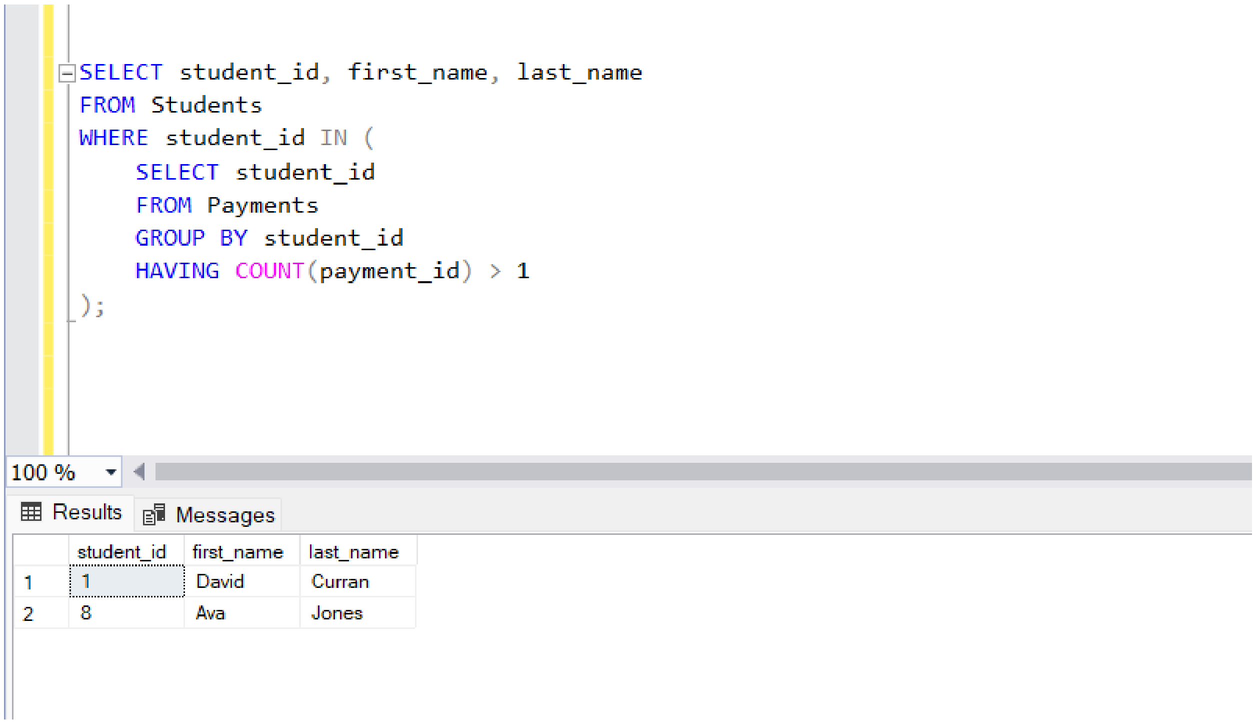
9.



```
SELECT E.student_id, E.course_id, ISNULL(SUM(P.amount), 0) AS total_payments
FROM Enrollments E
LEFT JOIN Payments P ON E.student_id = P.student_id
WHERE E.student_id IN (SELECT DISTINCT student_id FROM Enrollments)
GROUP BY E.student_id, E.course_id;
```

	student_id	course_id	total_payments
1	2	102	750.00
2	8	104	1200.00
3	5	105	550.00
4	6	106	700.00
5	1	107	2000.00
6	7	107	850.00
7	8	108	1200.00
8	7	109	850.00
9	9	109	700.00
10	10	110	500.00

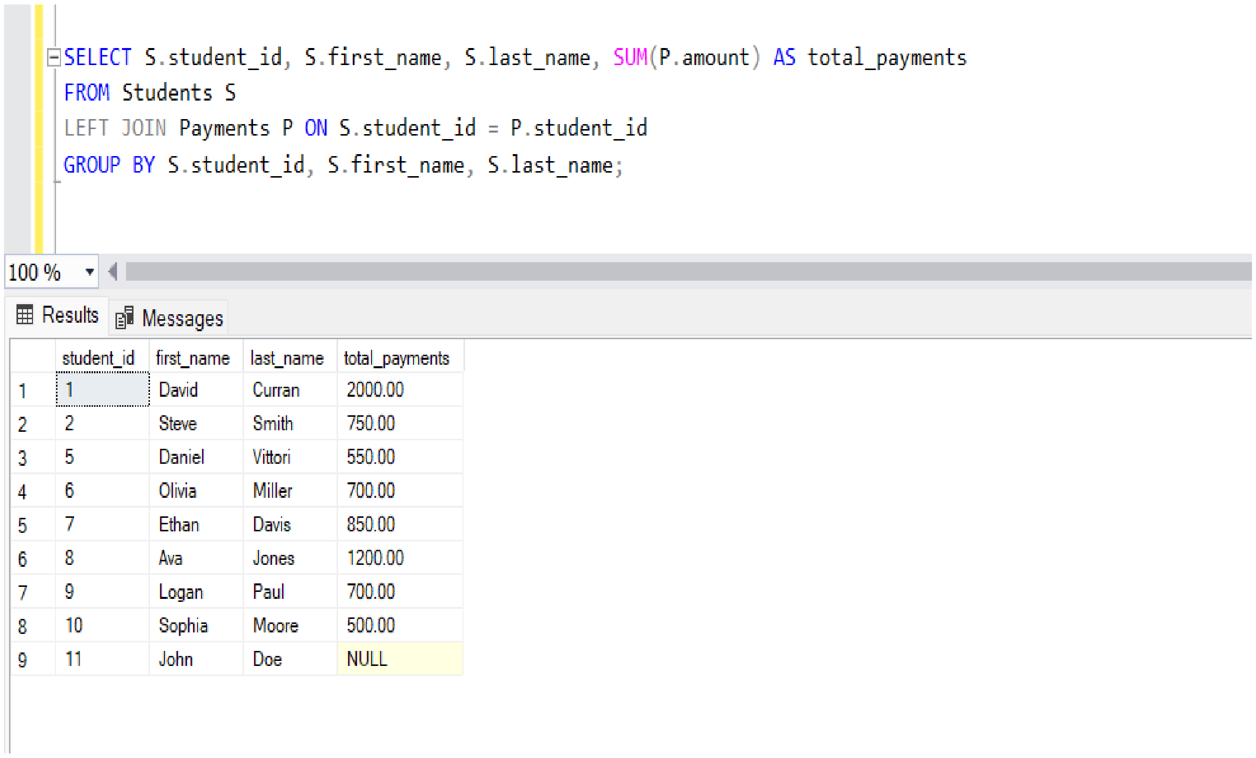
10.



```
SELECT student_id, first_name, last_name
FROM Students
WHERE student_id IN (
    SELECT student_id
    FROM Payments
    GROUP BY student_id
    HAVING COUNT(payment_id) > 1
);
```

	student_id	first_name	last_name
1	1	David	Curran
2	8	Ava	Jones

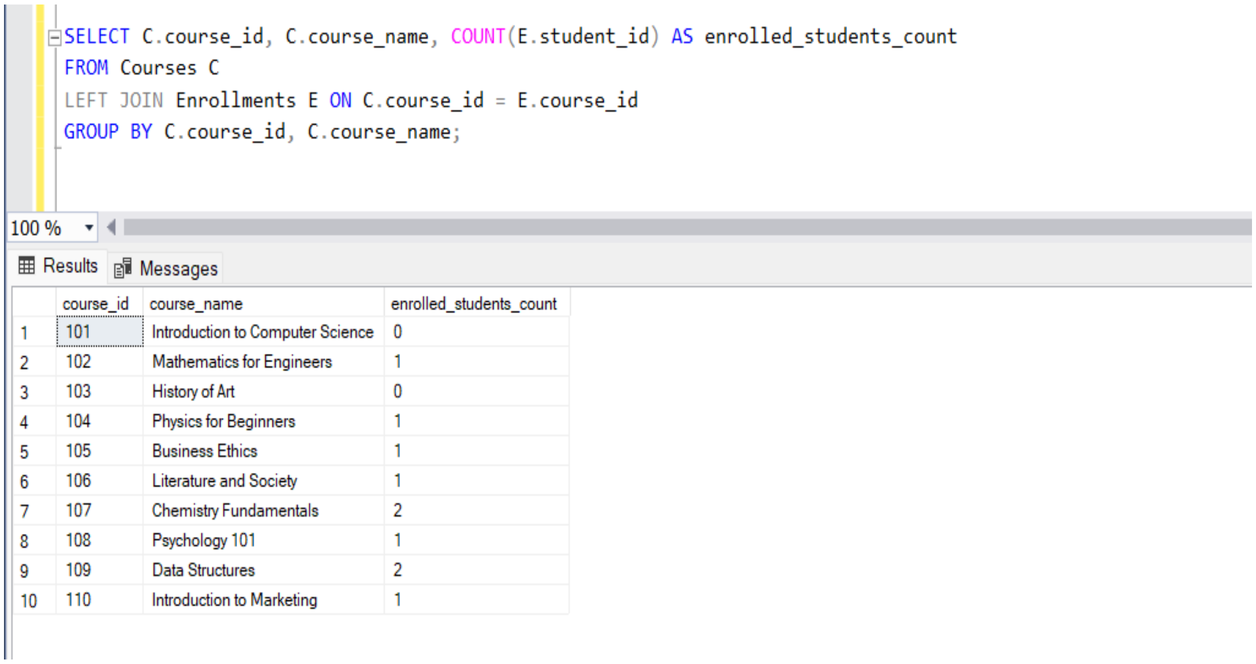
11.



```sql
SELECT S.student_id, S.first_name, S.last_name, SUM(P.amount) AS total_payments
FROM Students S
LEFT JOIN Payments P ON S.student_id = P.student_id
GROUP BY S.student_id, S.first_name, S.last_name;
```

	student_id	first_name	last_name	total_payments
1	1	David	Curran	2000.00
2	2	Steve	Smith	750.00
3	5	Daniel	Vittori	550.00
4	6	Olivia	Miller	700.00
5	7	Ethan	Davis	850.00
6	8	Ava	Jones	1200.00
7	9	Logan	Paul	700.00
8	10	Sophia	Moore	500.00
9	11	John	Doe	NULL

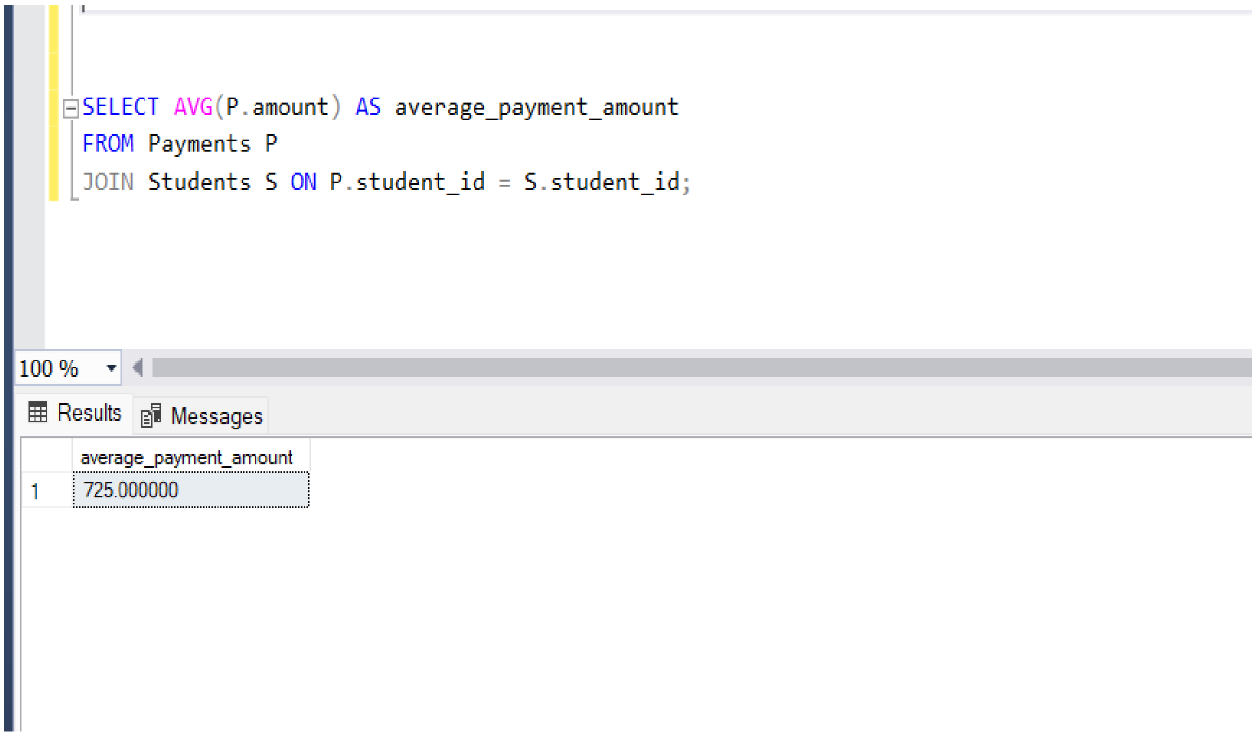
12.



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

	course_id	course_name	enrolled_students_count
1	101	Introduction to Computer Science	0
2	102	Mathematics for Engineers	1
3	103	History of Art	0
4	104	Physics for Beginners	1
5	105	Business Ethics	1
6	106	Literature and Society	1
7	107	Chemistry Fundamentals	2
8	108	Psychology 101	1
9	109	Data Structures	2
10	110	Introduction to Marketing	1

13.



```
SELECT AVG(P.amount) AS average_payment_amount
FROM Payments P
JOIN Students S ON P.student_id = S.student_id;
```

```
100 %
Results  Messages
   average_payment_amount
1  725.000000
```