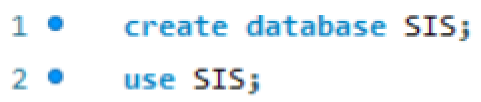
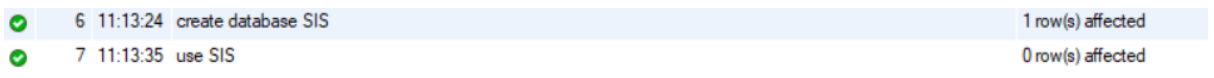
Assignment-2

Task-1

1, Create the database named "SISDB"



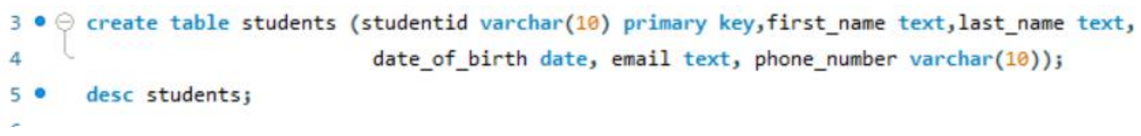
```
create database SIS;
use SIS;
```



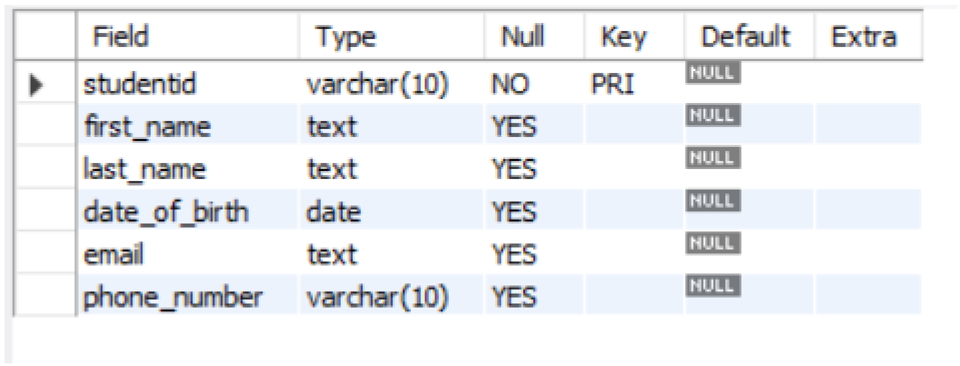
```
6  11:13:24  create database SIS        1 row(s) affected
7  11:13:35  use SIS                    0 row(s) affected
```

2, Define the schema for the Students, Courses, Enrollments, Teacher, and Payments tables based on the provided schema. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.

Students:

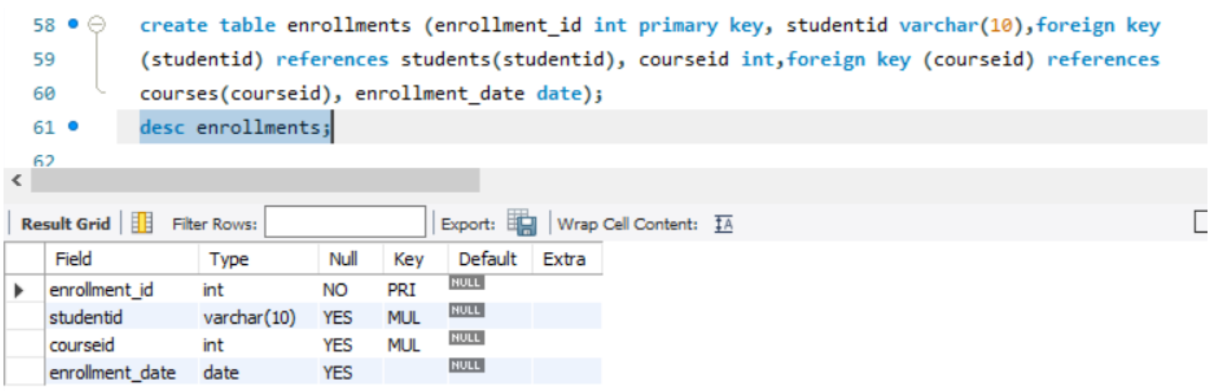


```
create table students (studentid varchar(10) primary key,first_name text,last_name text,
                       date_of_birth date, email text, phone_number varchar(10));
desc students;
```



Field	Type	Null	Key	Default	Extra
studentid	varchar(10)	NO	PRI	NULL	
first_name	text	YES		NULL	
last_name	text	YES		NULL	
date_of_birth	date	YES		NULL	
email	text	YES		NULL	
phone_number	varchar(10)	YES		NULL	

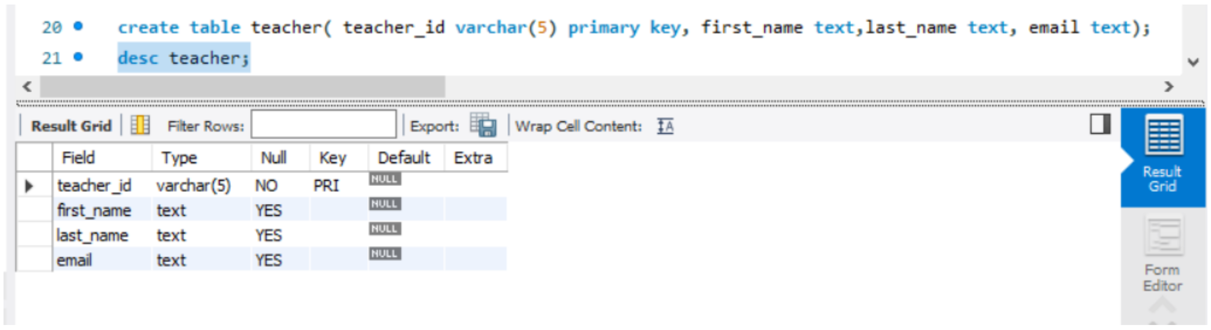
Enrollments:



```
create table enrollments (enrollment_id int primary key, studentid varchar(10),foreign key
(studentid) references students(studentid), courseid int,foreign key (courseid) references
courses(courseid), enrollment_date date);
desc enrollments;
```

Field	Type	Null	Key	Default	Extra
enrollment_id	int	NO	PRI	NULL	
studentid	varchar(10)	YES	MUL	NULL	
courseid	int	YES	MUL	NULL	
enrollment_date	date	YES		NULL	

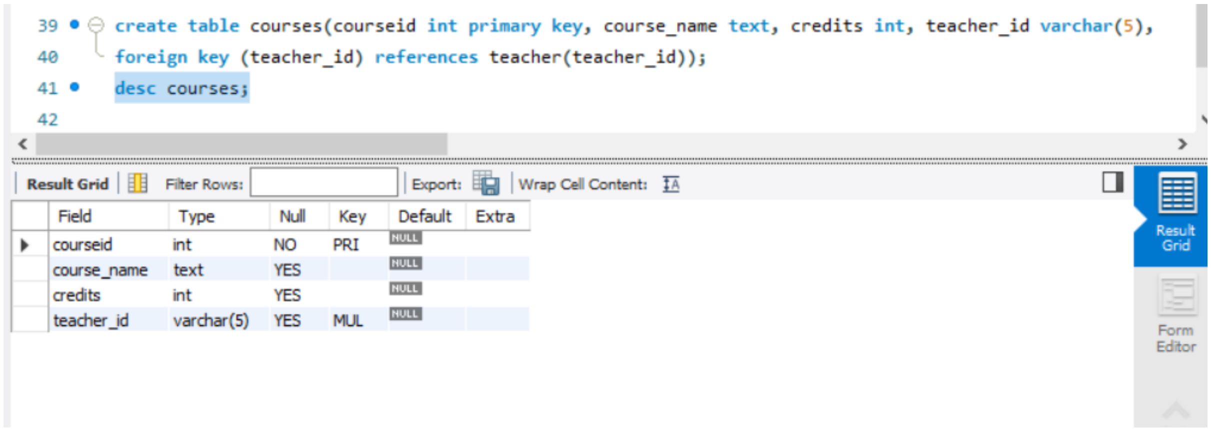
Teacher:



```
create table teacher( teacher_id varchar(5) primary key, first_name text,last_name text, email text);
desc teacher;
```

Field	Type	Null	Key	Default	Extra
teacher_id	varchar(5)	NO	PRI	NULL	
first_name	text	YES		NULL	
last_name	text	YES		NULL	
email	text	YES		NULL	

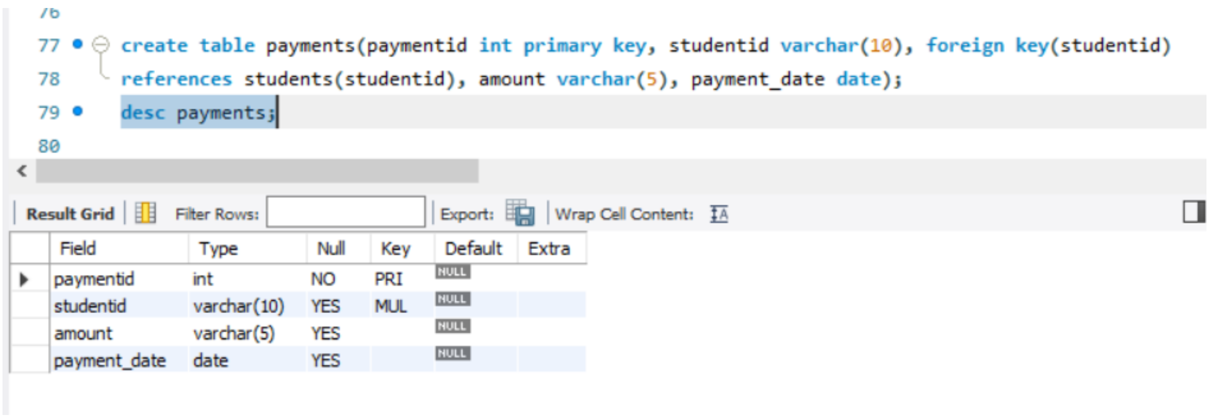
Courses:



```
create table courses(courseid int primary key, course_name text, credits int, teacher_id varchar(5),
foreign key (teacher_id) references teacher(teacher_id));
desc courses;
```

Field	Type	Null	Key	Default	Extra
courseid	int	NO	PRI	NULL	
course_name	text	YES		NULL	
credits	int	YES		NULL	
teacher_id	varchar(5)	YES	MUL	NULL	

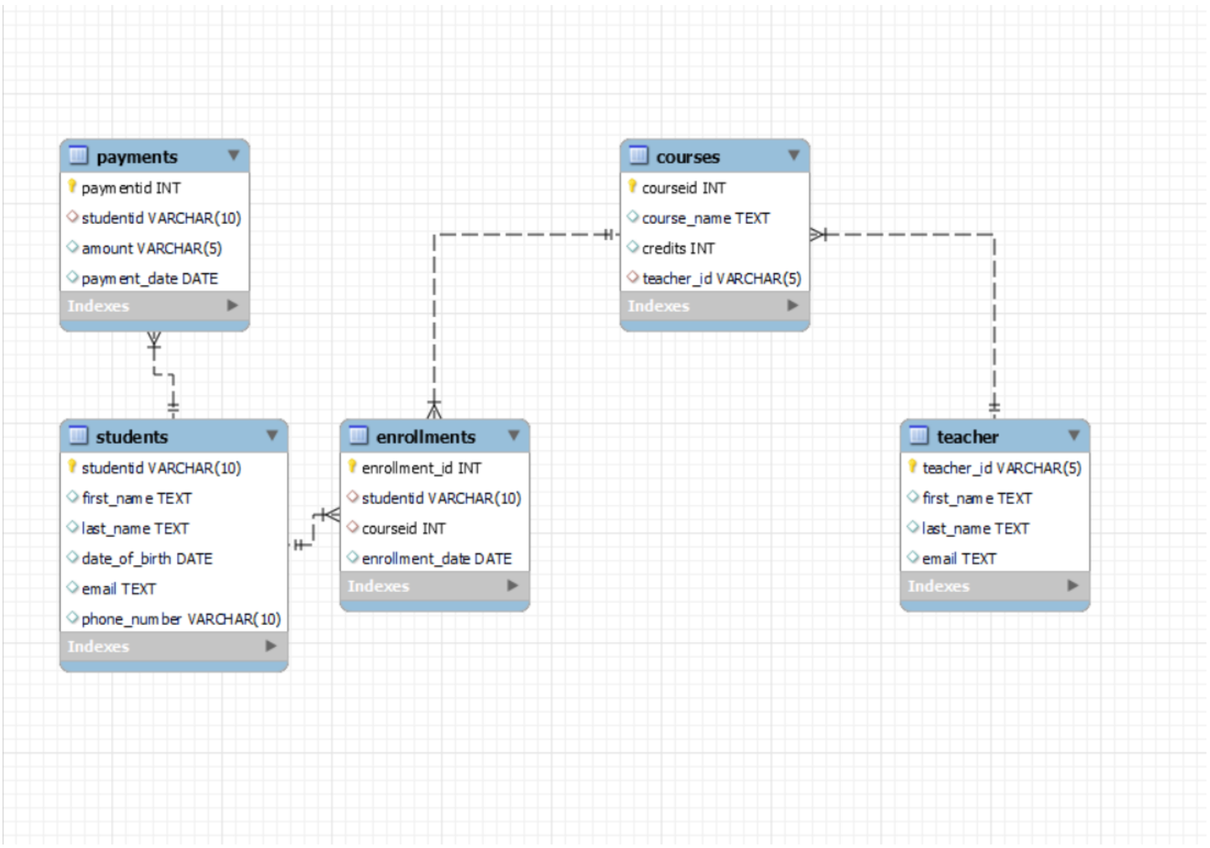
Payments:



```
create table payments(paymentid int primary key, studentid varchar(10), foreign key(studentid)
references students(studentid), amount varchar(5), payment_date date);
desc payments;
```

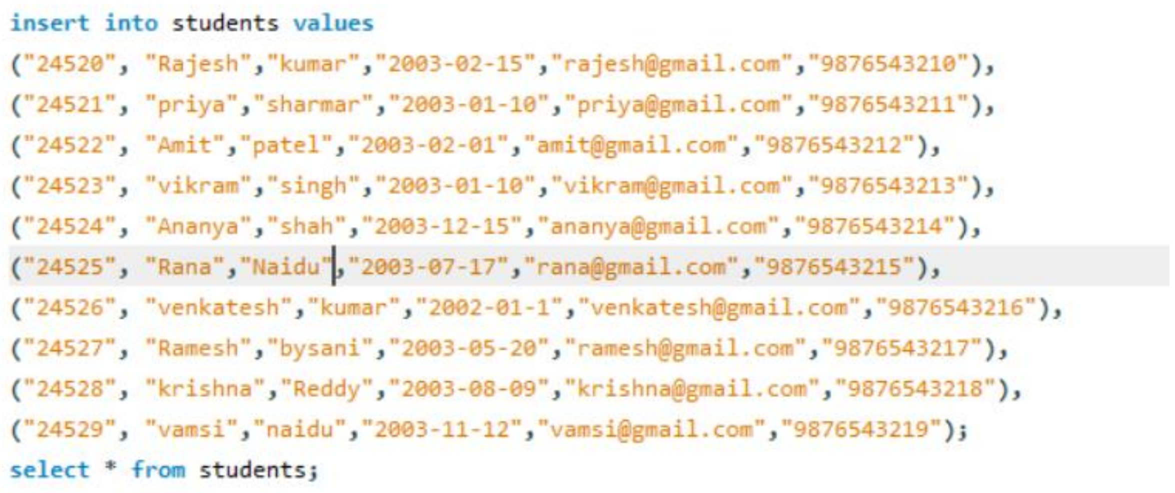
Field	Type	Null	Key	Default	Extra
paymentid	int	NO	PRI	NULL	
studentid	varchar(10)	YES	MUL	NULL	
amount	varchar(5)	YES		NULL	
payment_date	date	YES		NULL	

3, Create an ERD (Entity Relationship Diagram) for the database.

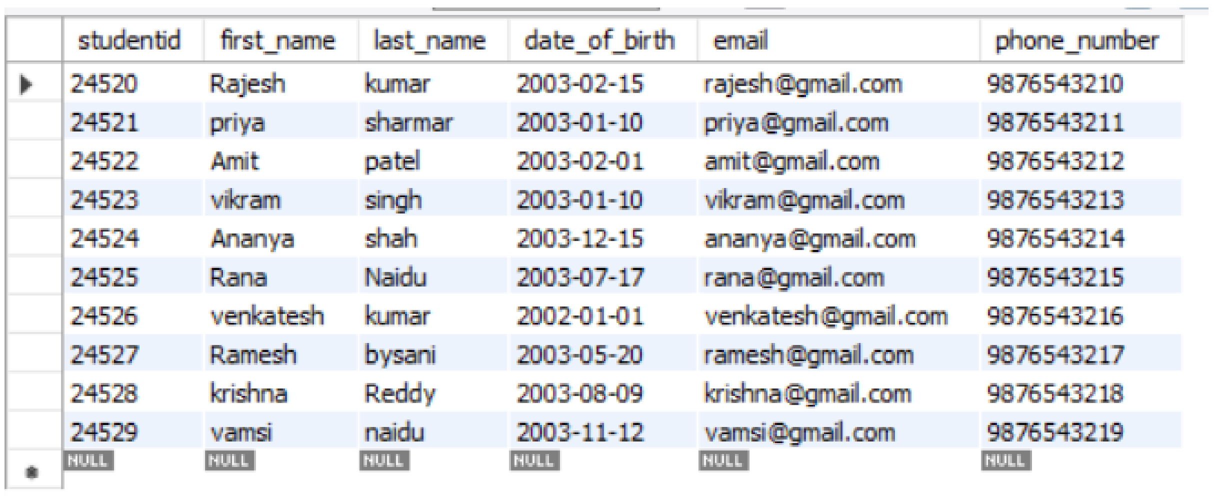


5, Insert at least 10 sample records into each of the following tables. i. Students ii. Courses iii. Enrollments iv. Teacher v. Payments

Students:

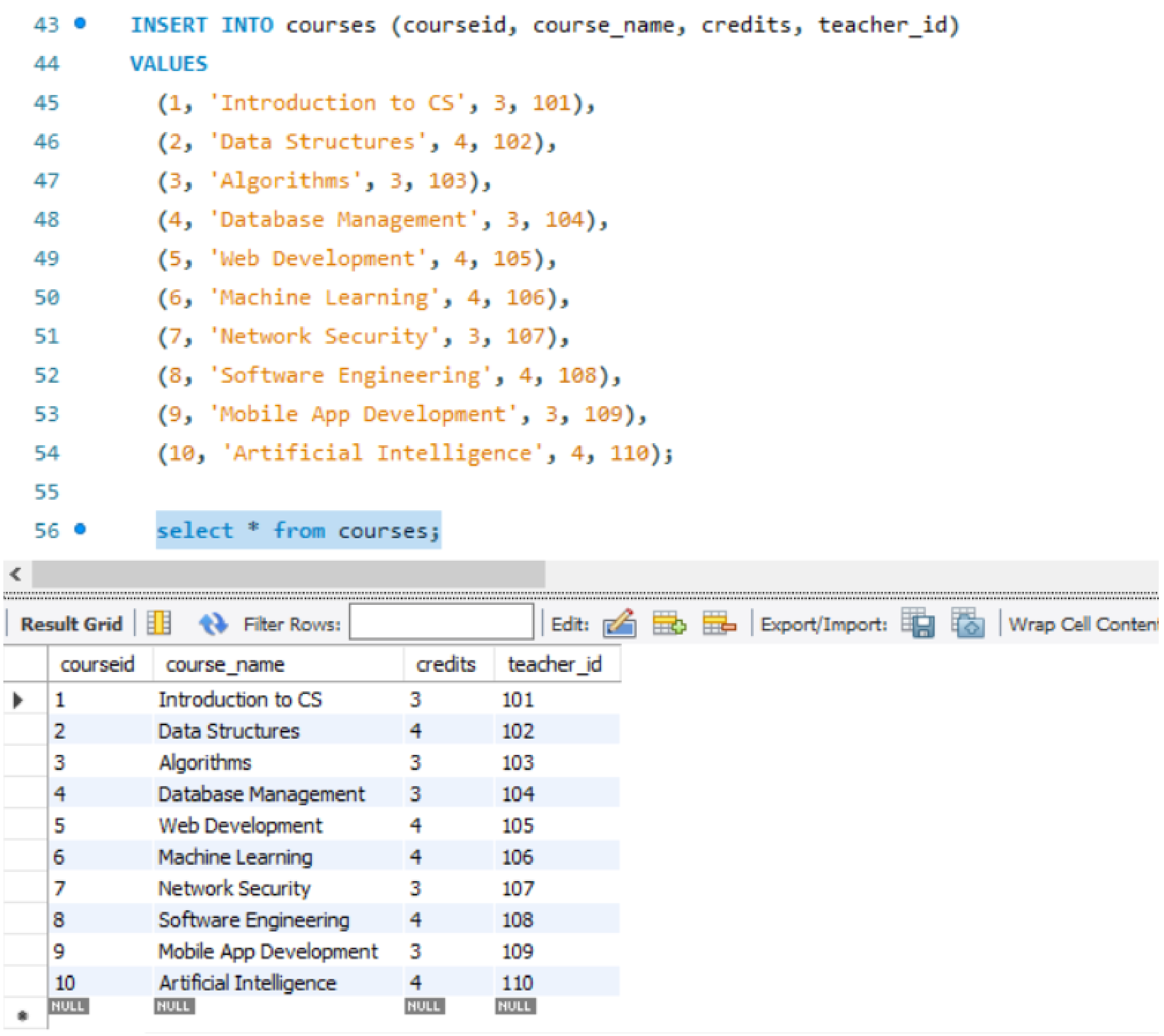


```
insert into students values
("24520", "Rajesh","kumar","2003-02-15","rajesh@gmail.com","9876543210"),
("24521", "priya","sharmar","2003-01-10","priya@gmail.com","9876543211"),
("24522", "Amit","patel","2003-02-01","amit@gmail.com","9876543212"),
("24523", "vikram","singh","2003-01-10","vikram@gmail.com","9876543213"),
("24524", "Ananya","shah","2003-12-15","ananya@gmail.com","9876543214"),
("24525", "Rana","Naidu","2003-07-17","rana@gmail.com","9876543215"),
("24526", "venkatesh","kumar","2002-01-1","venkatesh@gmail.com","9876543216"),
("24527", "Ramesh","bysani","2003-05-20","ramesh@gmail.com","9876543217"),
("24528", "krishna","Reddy","2003-08-09","krishna@gmail.com","9876543218"),
("24529", "vamsi","naidu","2003-11-12","vamsi@gmail.com","9876543219");
select * from students;
```



studentid	first_name	last_name	date_of_birth	email	phone_number
24520	Rajesh	kumar	2003-02-15	rajesh@gmail.com	9876543210
24521	priya	sharmar	2003-01-10	priya@gmail.com	9876543211
24522	Amit	patel	2003-02-01	amit@gmail.com	9876543212
24523	vikram	singh	2003-01-10	vikram@gmail.com	9876543213
24524	Ananya	shah	2003-12-15	ananya@gmail.com	9876543214
24525	Rana	Naidu	2003-07-17	rana@gmail.com	9876543215
24526	venkatesh	kumar	2002-01-01	venkatesh@gmail.com	9876543216
24527	Ramesh	bysani	2003-05-20	ramesh@gmail.com	9876543217
24528	krishna	Reddy	2003-08-09	krishna@gmail.com	9876543218
24529	vamsi	naidu	2003-11-12	vamsi@gmail.com	9876543219
NULL	NULL	NULL	NULL	NULL	NULL

Courses:

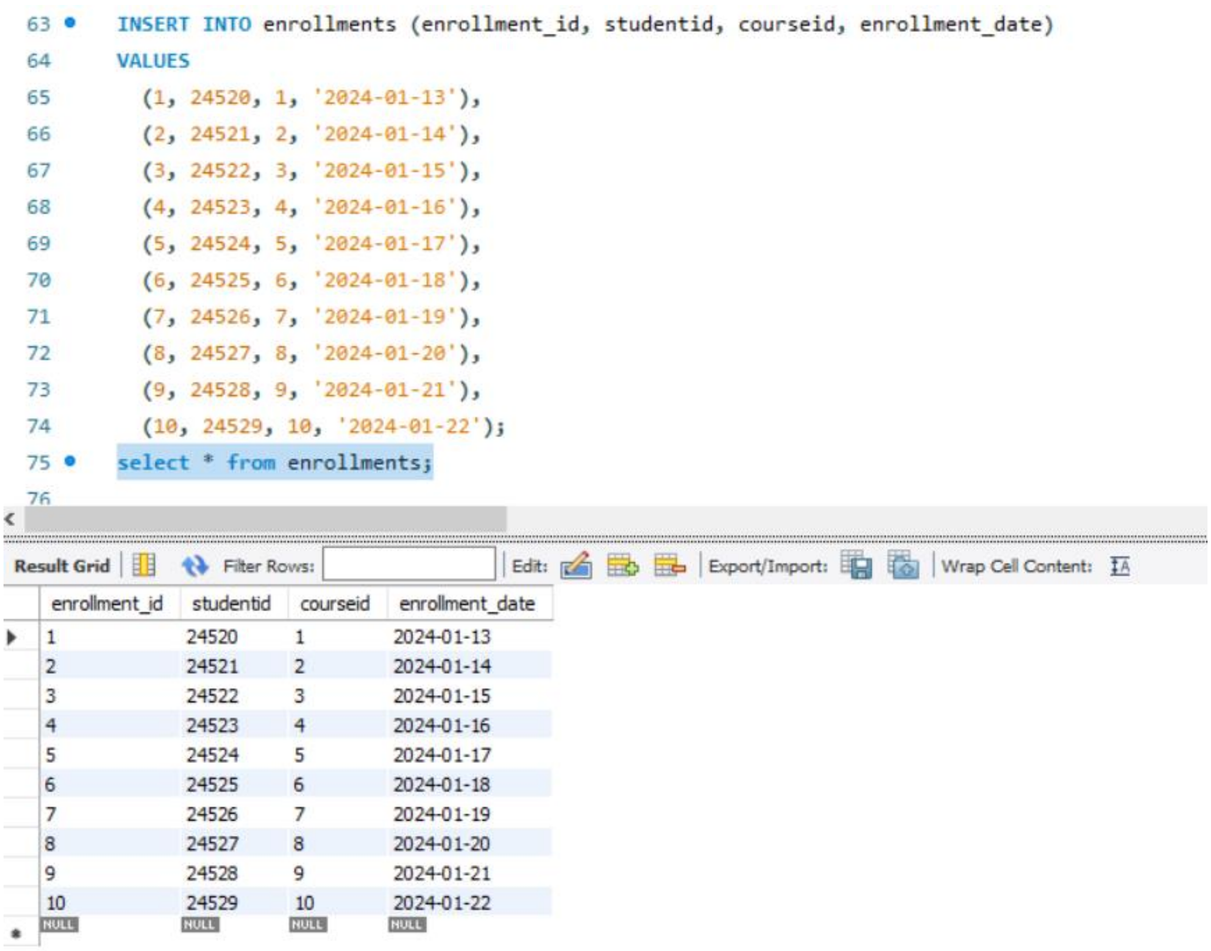


```
INSERT INTO courses (courseid, course_name, credits, teacher_id)
VALUES
  (1, 'Introduction to CS', 3, 101),
  (2, 'Data Structures', 4, 102),
  (3, 'Algorithms', 3, 103),
  (4, 'Database Management', 3, 104),
  (5, 'Web Development', 4, 105),
  (6, 'Machine Learning', 4, 106),
  (7, 'Network Security', 3, 107),
  (8, 'Software Engineering', 4, 108),
  (9, 'Mobile App Development', 3, 109),
  (10, 'Artificial Intelligence', 4, 110);

  select * from courses;
```

courseid	course_name	credits	teacher_id
1	Introduction to CS	3	101
2	Data Structures	4	102
3	Algorithms	3	103
4	Database Management	3	104
5	Web Development	4	105
6	Machine Learning	4	106
7	Network Security	3	107
8	Software Engineering	4	108
9	Mobile App Development	3	109
10	Artificial Intelligence	4	110
NULL	NULL	NULL	NULL

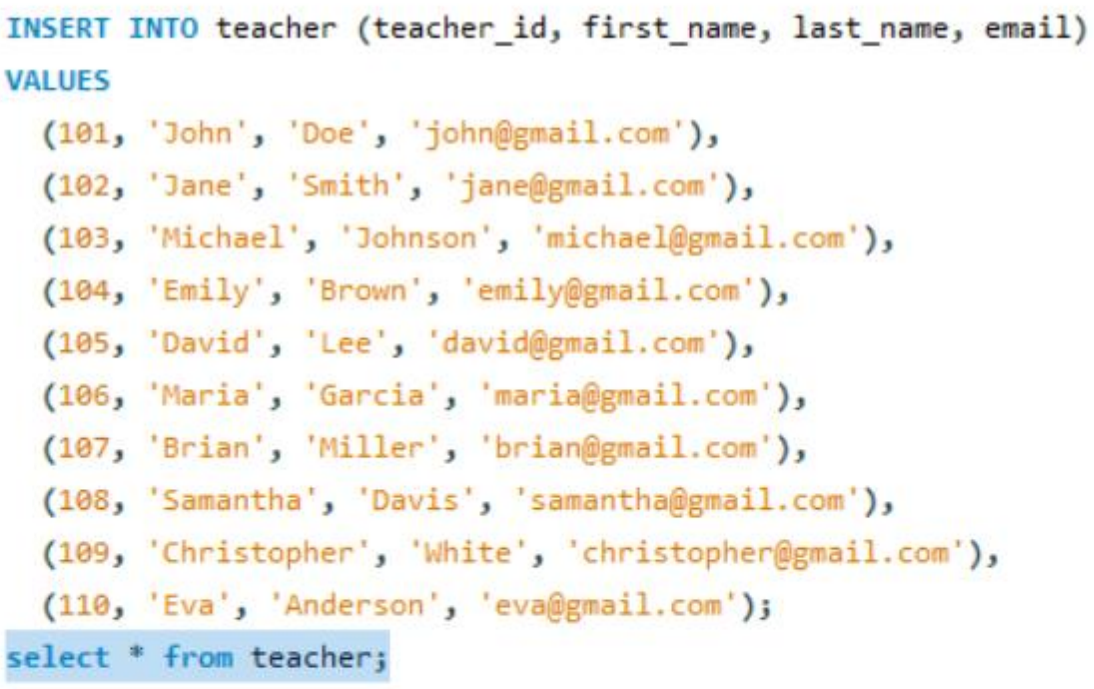
Enrollments:



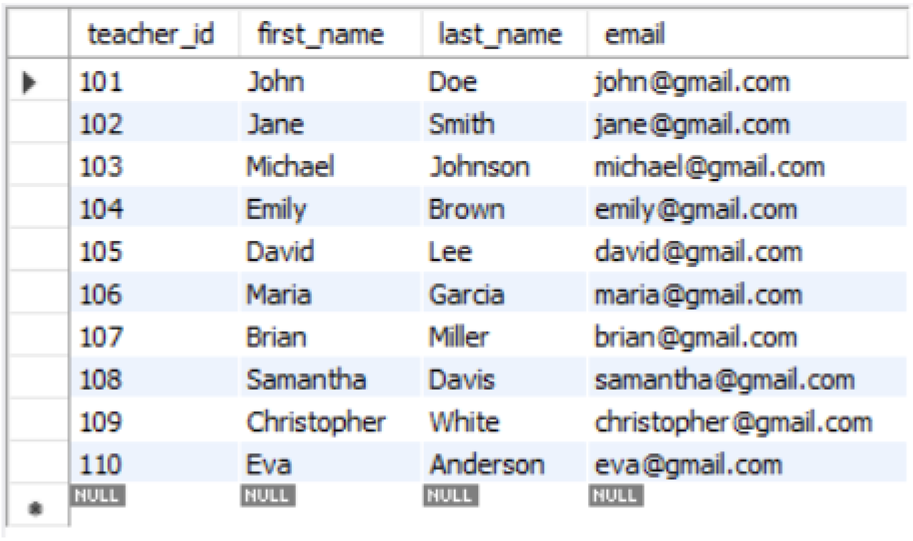
```
INSERT INTO enrollments (enrollment_id, studentid, courseid, enrollment_date)
VALUES
  (1, 24520, 1, '2024-01-13'),
  (2, 24521, 2, '2024-01-14'),
  (3, 24522, 3, '2024-01-15'),
  (4, 24523, 4, '2024-01-16'),
  (5, 24524, 5, '2024-01-17'),
  (6, 24525, 6, '2024-01-18'),
  (7, 24526, 7, '2024-01-19'),
  (8, 24527, 8, '2024-01-20'),
  (9, 24528, 9, '2024-01-21'),
  (10, 24529, 10, '2024-01-22');
select * from enrollments;
```

enrollment_id	studentid	courseid	enrollment_date
1	24520	1	2024-01-13
2	24521	2	2024-01-14
3	24522	3	2024-01-15
4	24523	4	2024-01-16
5	24524	5	2024-01-17
6	24525	6	2024-01-18
7	24526	7	2024-01-19
8	24527	8	2024-01-20
9	24528	9	2024-01-21
10	24529	10	2024-01-22
NULL	NULL	NULL	NULL

Teacher:

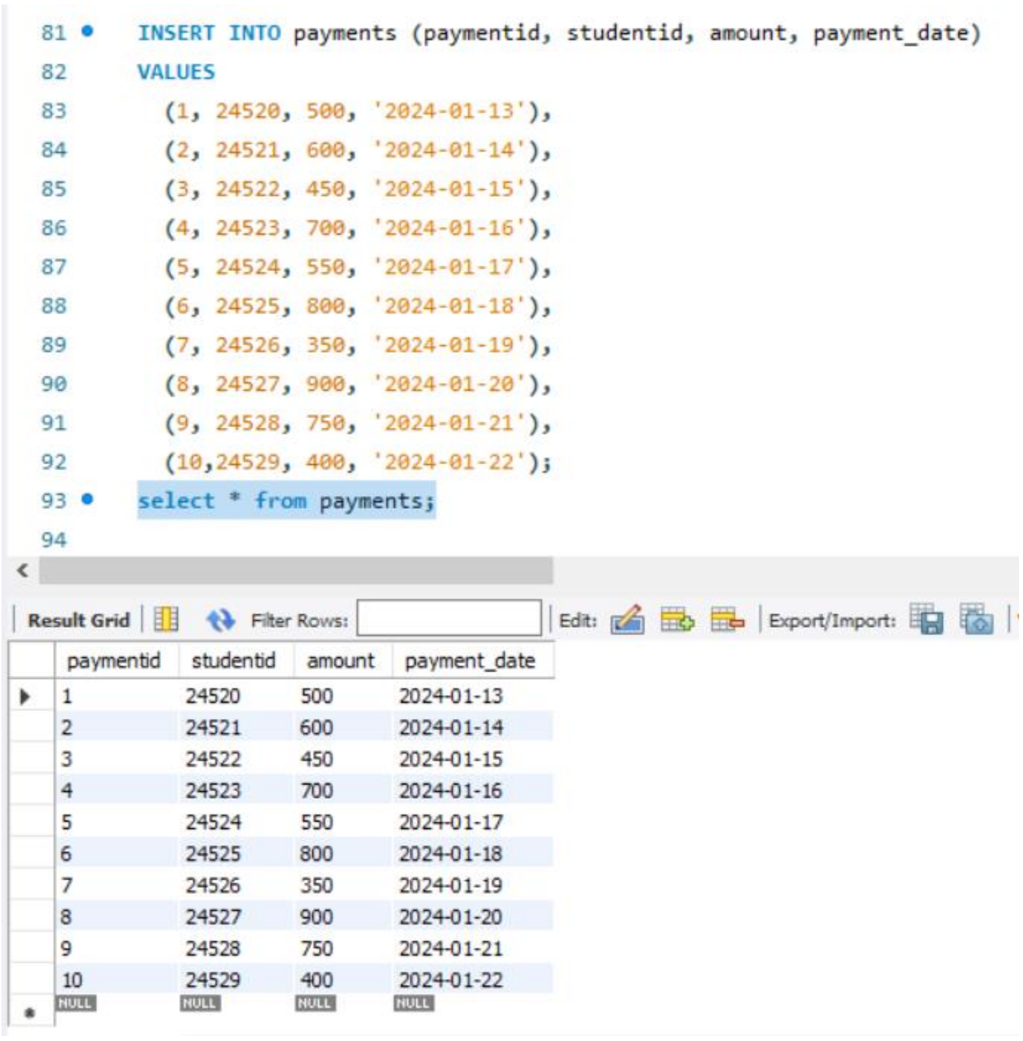


```
INSERT INTO teacher (teacher_id, first_name, last_name, email)
VALUES
  (101, 'John', 'Doe', 'john@gmail.com'),
  (102, 'Jane', 'Smith', 'jane@gmail.com'),
  (103, 'Michael', 'Johnson', 'michael@gmail.com'),
  (104, 'Emily', 'Brown', 'emily@gmail.com'),
  (105, 'David', 'Lee', 'david@gmail.com'),
  (106, 'Maria', 'Garcia', 'maria@gmail.com'),
  (107, 'Brian', 'Miller', 'brian@gmail.com'),
  (108, 'Samantha', 'Davis', 'samantha@gmail.com'),
  (109, 'Christopher', 'White', 'christopher@gmail.com'),
  (110, 'Eva', 'Anderson', 'eva@gmail.com');
select * from teacher;
```



	teacher_id	first_name	last_name	email
▶	101	John	Doe	john@gmail.com
	102	Jane	Smith	jane@gmail.com
	103	Michael	Johnson	michael@gmail.com
	104	Emily	Brown	emily@gmail.com
	105	David	Lee	david@gmail.com
	106	Maria	Garcia	maria@gmail.com
	107	Brian	Miller	brian@gmail.com
	108	Samantha	Davis	samantha@gmail.com
	109	Christopher	White	christopher@gmail.com
	110	Eva	Anderson	eva@gmail.com
*	NULL	NULL	NULL	NULL

Payments:



```
INSERT INTO payments (paymentid, studentid, amount, payment_date)
VALUES
  (1, 24520, 500, '2024-01-13'),
  (2, 24521, 600, '2024-01-14'),
  (3, 24522, 450, '2024-01-15'),
  (4, 24523, 700, '2024-01-16'),
  (5, 24524, 550, '2024-01-17'),
  (6, 24525, 800, '2024-01-18'),
  (7, 24526, 350, '2024-01-19'),
  (8, 24527, 900, '2024-01-20'),
  (9, 24528, 750, '2024-01-21'),
  (10,24529, 400, '2024-01-22');
select * from payments;
```

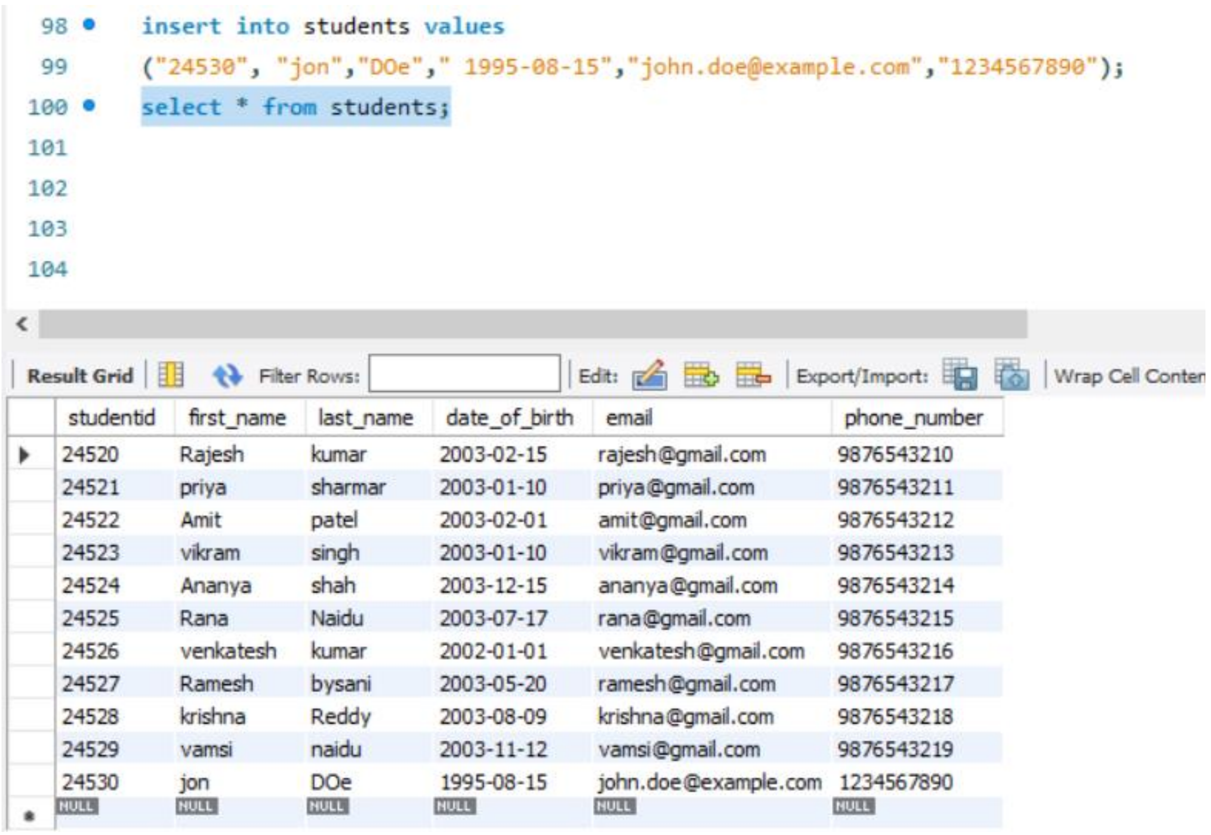
	paymentid	studentid	amount	payment_date
▶	1	24520	500	2024-01-13
	2	24521	600	2024-01-14
	3	24522	450	2024-01-15
	4	24523	700	2024-01-16
	5	24524	550	2024-01-17
	6	24525	800	2024-01-18
	7	24526	350	2024-01-19
	8	24527	900	2024-01-20
	9	24528	750	2024-01-21
	10	24529	400	2024-01-22
*	NULL	NULL	NULL	NULL

Task-2

1, Write an SQL query to insert a new student into the "Students" table with the following details:

a. First Name: John b. Last Name: Doe c. Date of Birth: 1995-08-15

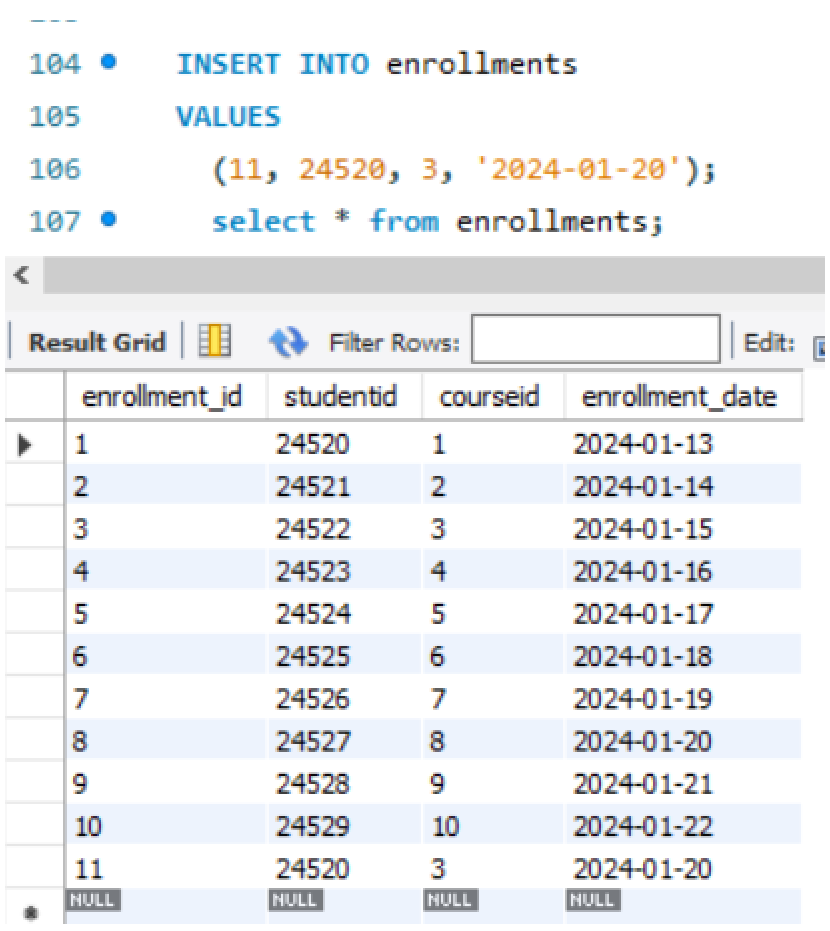
d. Email: john.doe@example.com e. Phone Number: 1234567890



```sql
insert into students values
("24530", "jon","DOe"," 1995-08-15","john.doe@example.com","1234567890");
select * from students;
```

studentid	first_name	last_name	date_of_birth	email	phone_number
24520	Rajesh	kumar	2003-02-15	rajesh@gmail.com	9876543210
24521	priya	sharmar	2003-01-10	priya@gmail.com	9876543211
24522	Amit	patel	2003-02-01	amit@gmail.com	9876543212
24523	vikram	singh	2003-01-10	vikram@gmail.com	9876543213
24524	Ananya	shah	2003-12-15	ananya@gmail.com	9876543214
24525	Rana	Naidu	2003-07-17	rana@gmail.com	9876543215
24526	venkatesh	kumar	2002-01-01	venkatesh@gmail.com	9876543216
24527	Ramesh	bysani	2003-05-20	ramesh@gmail.com	9876543217
24528	krishna	Reddy	2003-08-09	krishna@gmail.com	9876543218
24529	vamsi	naidu	2003-11-12	vamsi@gmail.com	9876543219
24530	jon	DOe	1995-08-15	john.doe@example.com	1234567890
NULL	NULL	NULL	NULL	NULL	NULL

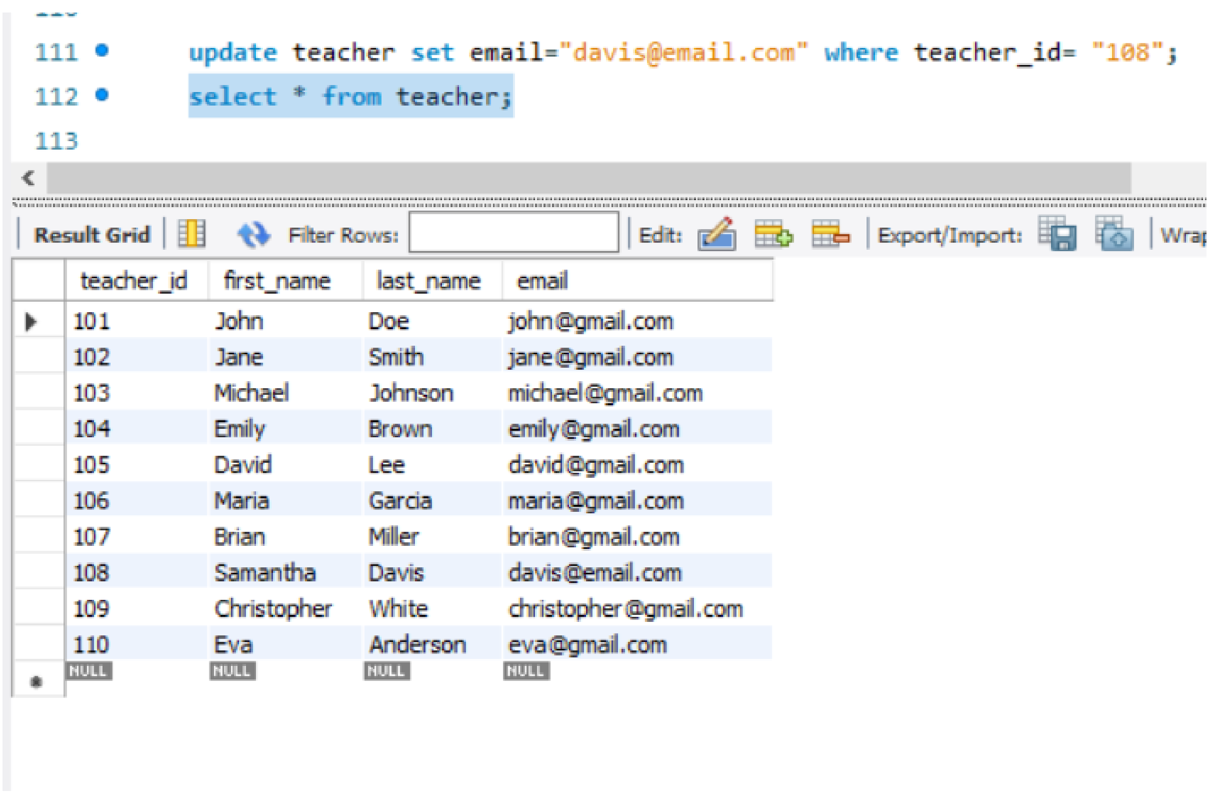
2, Write an SQL query to enroll a student in a course. Choose an existing student and course and insert a record into the "Enrollments" table with the enrollment date.



```
INSERT INTO enrollments
VALUES
  (11, 24520, 3, '2024-01-20');
  select * from enrollments;
```

enrollment_id	studentid	courseid	enrollment_date
1	24520	1	2024-01-13
2	24521	2	2024-01-14
3	24522	3	2024-01-15
4	24523	4	2024-01-16
5	24524	5	2024-01-17
6	24525	6	2024-01-18
7	24526	7	2024-01-19
8	24527	8	2024-01-20
9	24528	9	2024-01-21
10	24529	10	2024-01-22
11	24520	3	2024-01-20
NULL	NULL	NULL	NULL

3, Update the email address of a specific teacher in the "Teacher" table. Choose any teacher and modify their email address

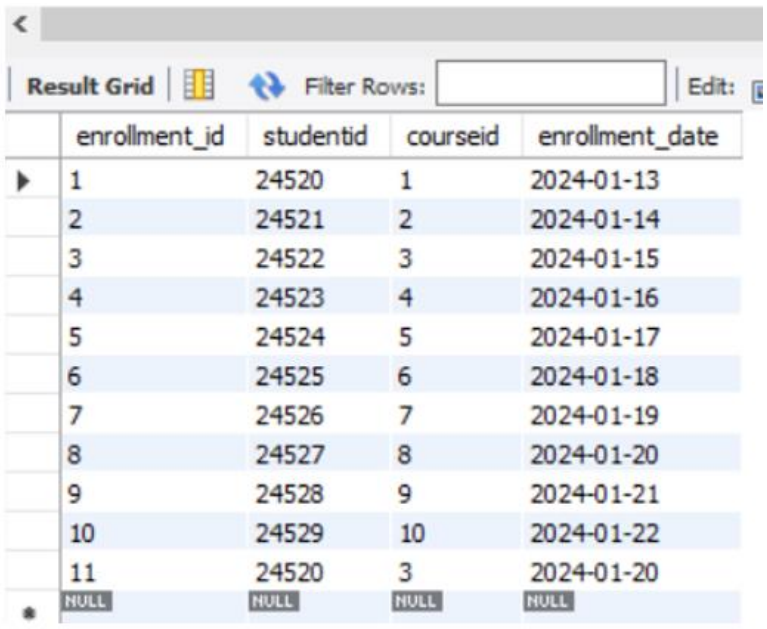


```
update teacher set email="davis@email.com" where teacher_id= "108";
select * from teacher;
```

teacher_id	first_name	last_name	email
101	John	Doe	john@gmail.com
102	Jane	Smith	jane@gmail.com
103	Michael	Johnson	michael@gmail.com
104	Emily	Brown	emily@gmail.com
105	David	Lee	david@gmail.com
106	Maria	Garcia	maria@gmail.com
107	Brian	Miller	brian@gmail.com
108	Samantha	Davis	davis@email.com
109	Christopher	White	christopher@gmail.com
110	Eva	Anderson	eva@gmail.com
NULL	NULL	NULL	NULL

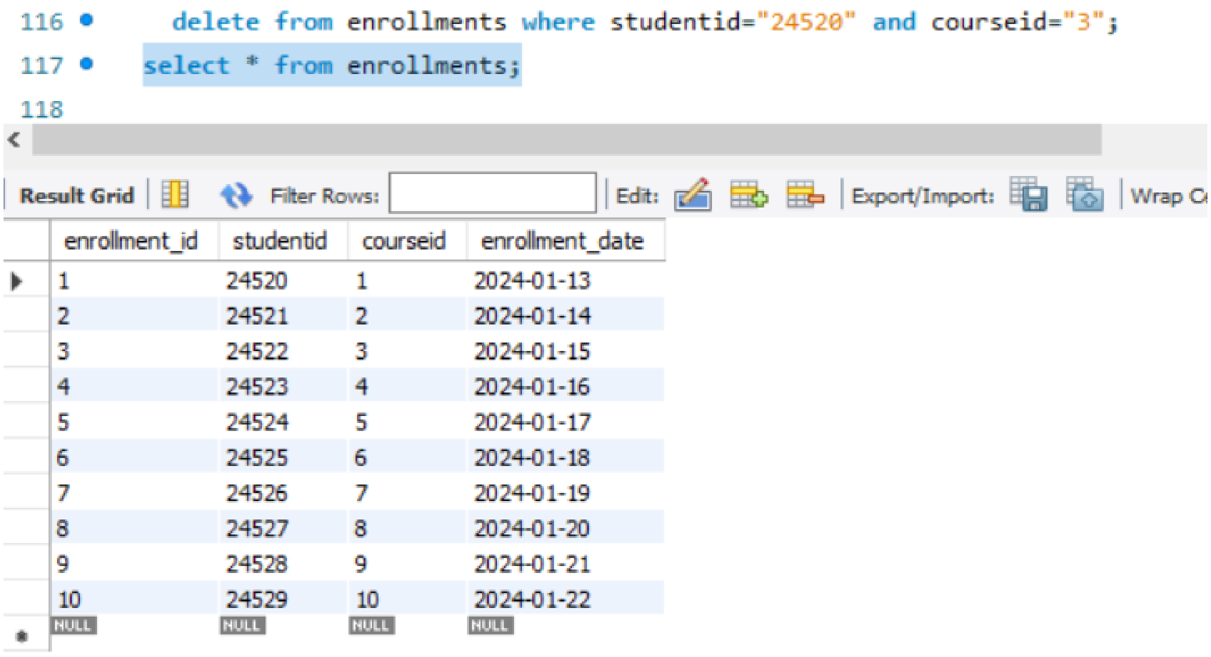
4, Write an SQL query to delete a specific enrollment record from the "Enrollments" table. Select an enrollment record based on the student and course

Before Deletion:



enrollment_id	studentid	courseid	enrollment_date
1	24520	1	2024-01-13
2	24521	2	2024-01-14
3	24522	3	2024-01-15
4	24523	4	2024-01-16
5	24524	5	2024-01-17
6	24525	6	2024-01-18
7	24526	7	2024-01-19
8	24527	8	2024-01-20
9	24528	9	2024-01-21
10	24529	10	2024-01-22
11	24520	3	2024-01-20
NULL	NULL	NULL	NULL

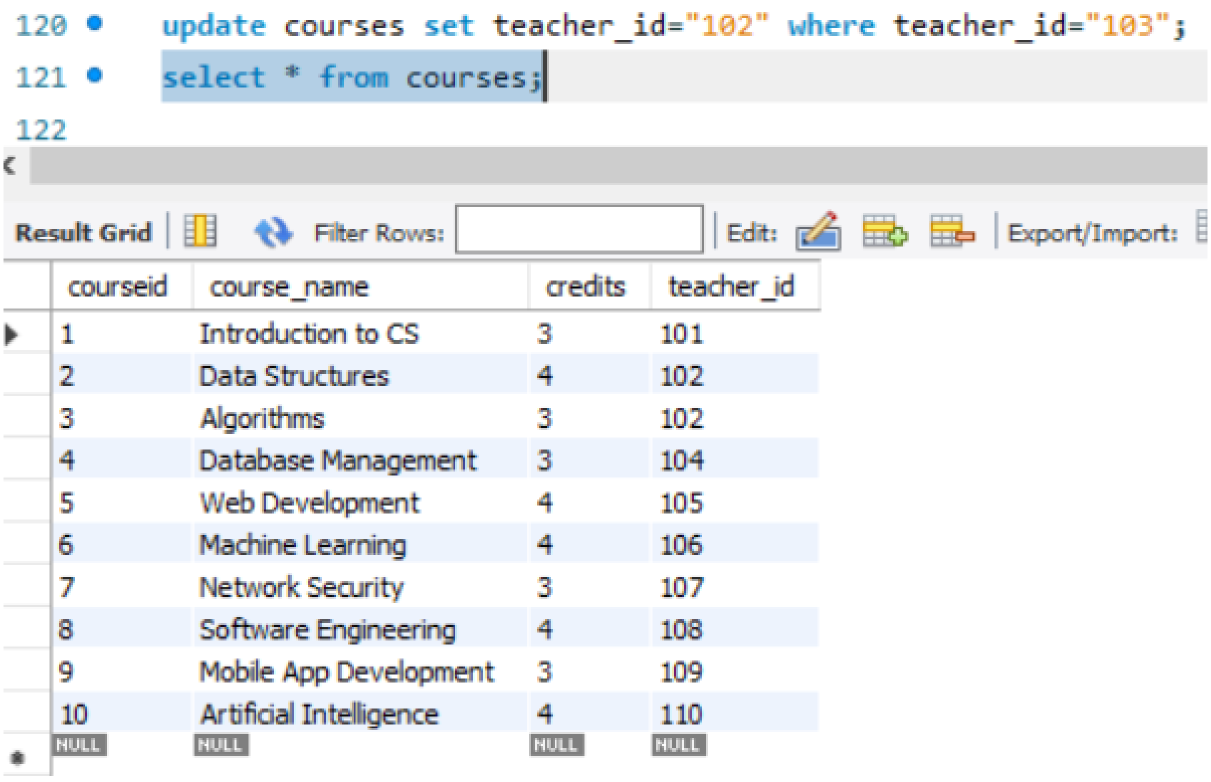
After Deletion:



```
delete from enrollments where studentid="24520" and courseid="3";
select * from enrollments;
```

enrollment_id	studentid	courseid	enrollment_date
1	24520	1	2024-01-13
2	24521	2	2024-01-14
3	24522	3	2024-01-15
4	24523	4	2024-01-16
5	24524	5	2024-01-17
6	24525	6	2024-01-18
7	24526	7	2024-01-19
8	24527	8	2024-01-20
9	24528	9	2024-01-21
10	24529	10	2024-01-22
NULL	NULL	NULL	NULL

5, Update the "Courses" table to assign a specific teacher to a course. Choose any course and teacher from the respective tables



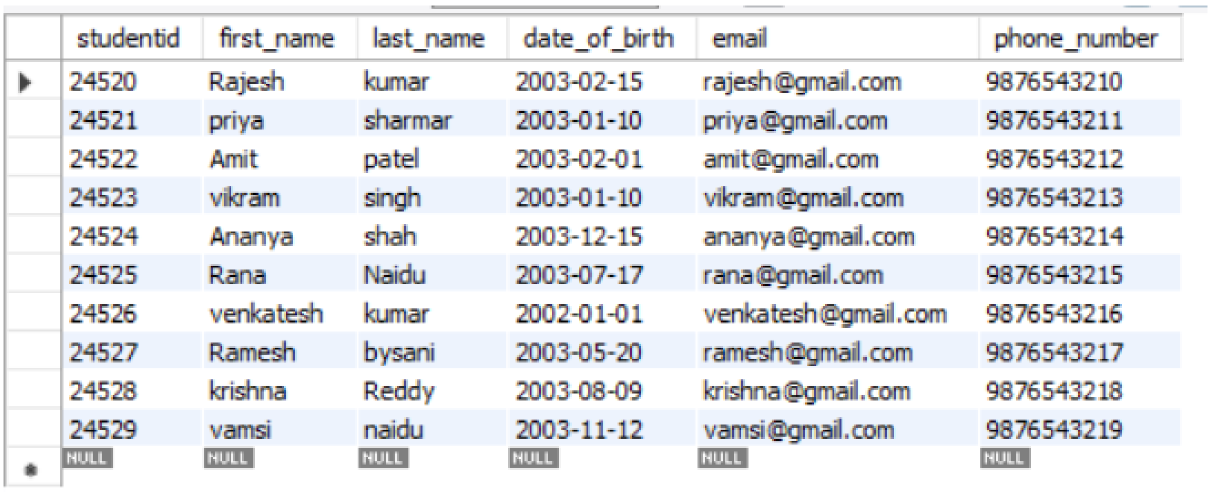
```
update courses set teacher_id="102" where teacher_id="103";
select * from courses;
```

courseid	course_name	credits	teacher_id
1	Introduction to CS	3	101
2	Data Structures	4	102
3	Algorithms	3	102
4	Database Management	3	104
5	Web Development	4	105
6	Machine Learning	4	106
7	Network Security	3	107
8	Software Engineering	4	108
9	Mobile App Development	3	109
10	Artificial Intelligence	4	110
NULL	NULL	NULL	NULL

6, Delete a specific student from the "Students" table and remove all their enrollment records from the "Enrollments" table. Be sure to maintain referential integrity.

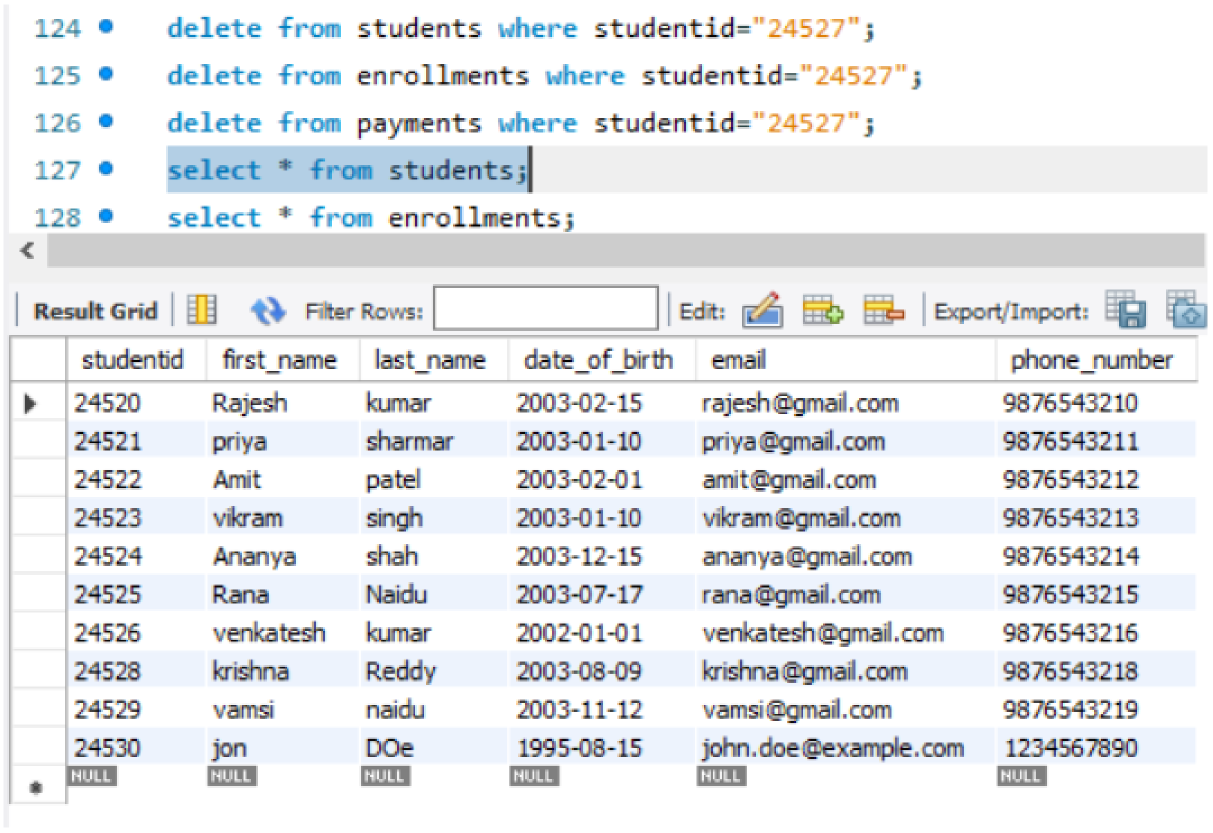
Students:

Before Deletion:



	studentid	first_name	last_name	date_of_birth	email	phone_number
▶	24520	Rajesh	kumar	2003-02-15	rajesh@gmail.com	9876543210
	24521	priya	sharmar	2003-01-10	priya@gmail.com	9876543211
	24522	Amit	patel	2003-02-01	amit@gmail.com	9876543212
	24523	vikram	singh	2003-01-10	vikram@gmail.com	9876543213
	24524	Ananya	shah	2003-12-15	ananya@gmail.com	9876543214
	24525	Rana	Naidu	2003-07-17	rana@gmail.com	9876543215
	24526	venkatesh	kumar	2002-01-01	venkatesh@gmail.com	9876543216
	24527	Ramesh	bysani	2003-05-20	ramesh@gmail.com	9876543217
	24528	krishna	Reddy	2003-08-09	krishna@gmail.com	9876543218
	24529	vamsi	naidu	2003-11-12	vamsi@gmail.com	9876543219
*	NULL	NULL	NULL	NULL	NULL	NULL

After Deletion:



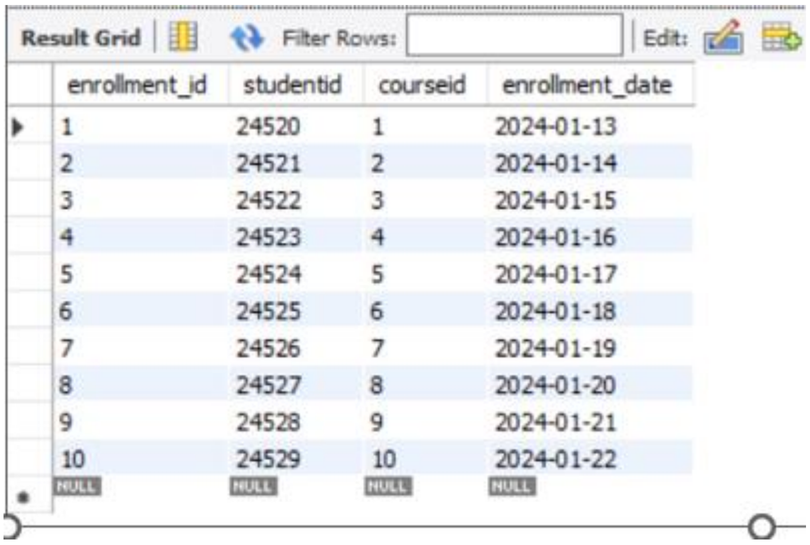
```
delete from students where studentid="24527";
delete from enrollments where studentid="24527";
delete from payments where studentid="24527";
select * from students;
select * from enrollments;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

	studentid	first_name	last_name	date_of_birth	email	phone_number
▶	24520	Rajesh	kumar	2003-02-15	rajesh@gmail.com	9876543210
	24521	priya	sharmar	2003-01-10	priya@gmail.com	9876543211
	24522	Amit	patel	2003-02-01	amit@gmail.com	9876543212
	24523	vikram	singh	2003-01-10	vikram@gmail.com	9876543213
	24524	Ananya	shah	2003-12-15	ananya@gmail.com	9876543214
	24525	Rana	Naidu	2003-07-17	rana@gmail.com	9876543215
	24526	venkatesh	kumar	2002-01-01	venkatesh@gmail.com	9876543216
	24528	krishna	Reddy	2003-08-09	krishna@gmail.com	9876543218
	24529	vamsi	naidu	2003-11-12	vamsi@gmail.com	9876543219
	24530	jon	DOe	1995-08-15	john.doe@example.com	1234567890
*	NULL	NULL	NULL	NULL	NULL	NULL

Enrollments:

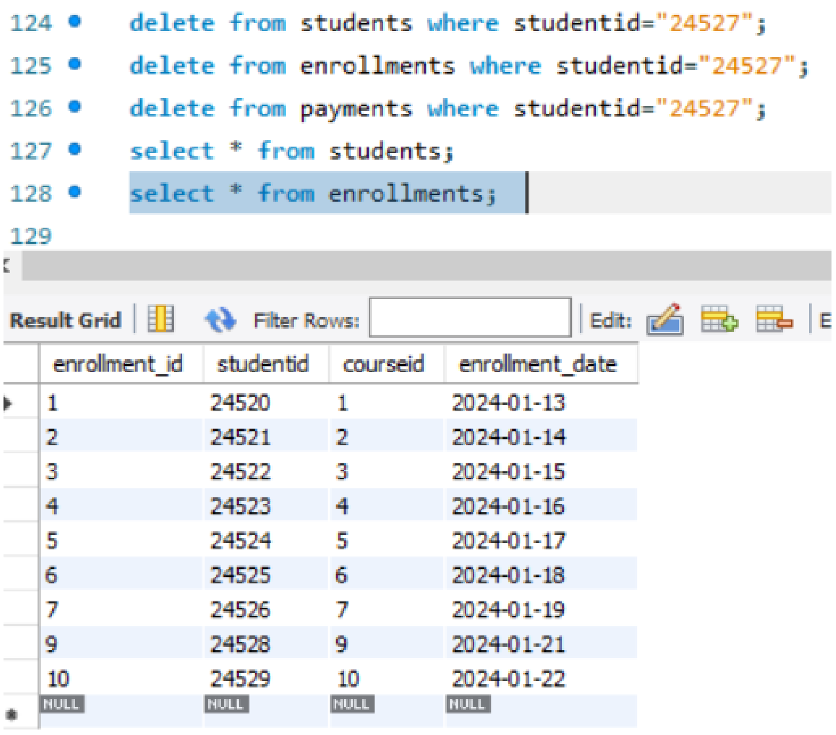
Before Deletion:



Result Grid | Filter Rows: | Edit:

enrollment_id	studentid	courseid	enrollment_date
1	24520	1	2024-01-13
2	24521	2	2024-01-14
3	24522	3	2024-01-15
4	24523	4	2024-01-16
5	24524	5	2024-01-17
6	24525	6	2024-01-18
7	24526	7	2024-01-19
8	24527	8	2024-01-20
9	24528	9	2024-01-21
10	24529	10	2024-01-22
NULL	NULL	NULL	NULL

After deletion:

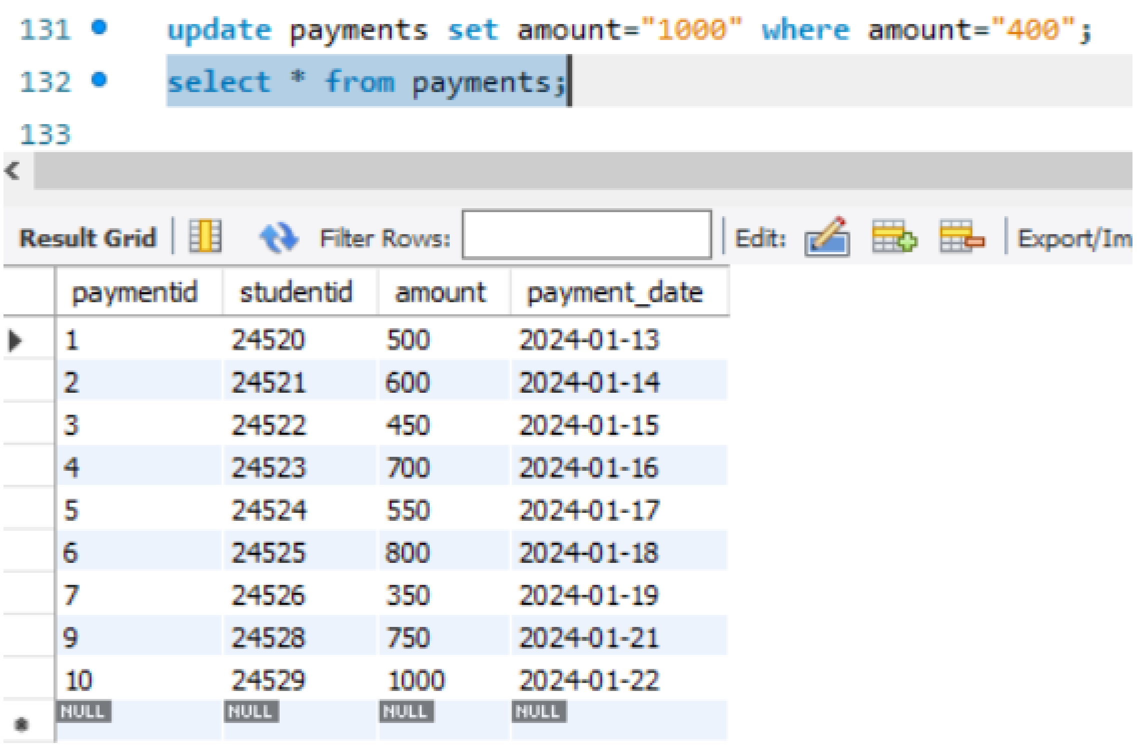


```sql
delete from students where studentid="24527";
delete from enrollments where studentid="24527";
delete from payments where studentid="24527";
select * from students;
select * from enrollments;
```

Result Grid | Filter Rows: | Edit:

enrollment_id	studentid	courseid	enrollment_date
1	24520	1	2024-01-13
2	24521	2	2024-01-14
3	24522	3	2024-01-15
4	24523	4	2024-01-16
5	24524	5	2024-01-17
6	24525	6	2024-01-18
7	24526	7	2024-01-19
9	24528	9	2024-01-21
10	24529	10	2024-01-22
NULL	NULL	NULL	NULL

7, Update the payment amount for a specific payment record in the "Payments" table. Choose any payment record and modify the payment amount

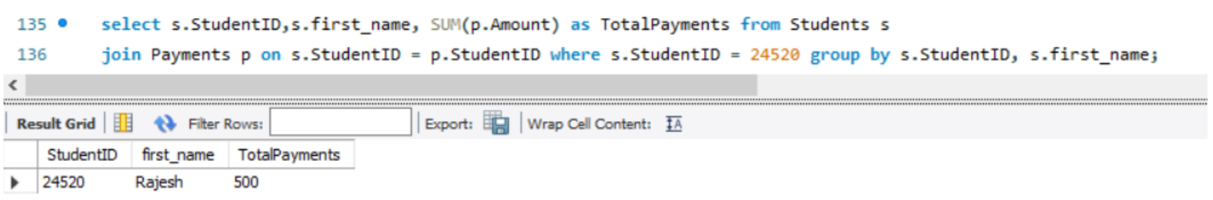


```
update payments set amount="1000" where amount="400";
select * from payments;
```

paymentid	studentid	amount	payment_date
1	24520	500	2024-01-13
2	24521	600	2024-01-14
3	24522	450	2024-01-15
4	24523	700	2024-01-16
5	24524	550	2024-01-17
6	24525	800	2024-01-18
7	24526	350	2024-01-19
9	24528	750	2024-01-21
10	24529	1000	2024-01-22
NULL	NULL	NULL	NULL

Task-3

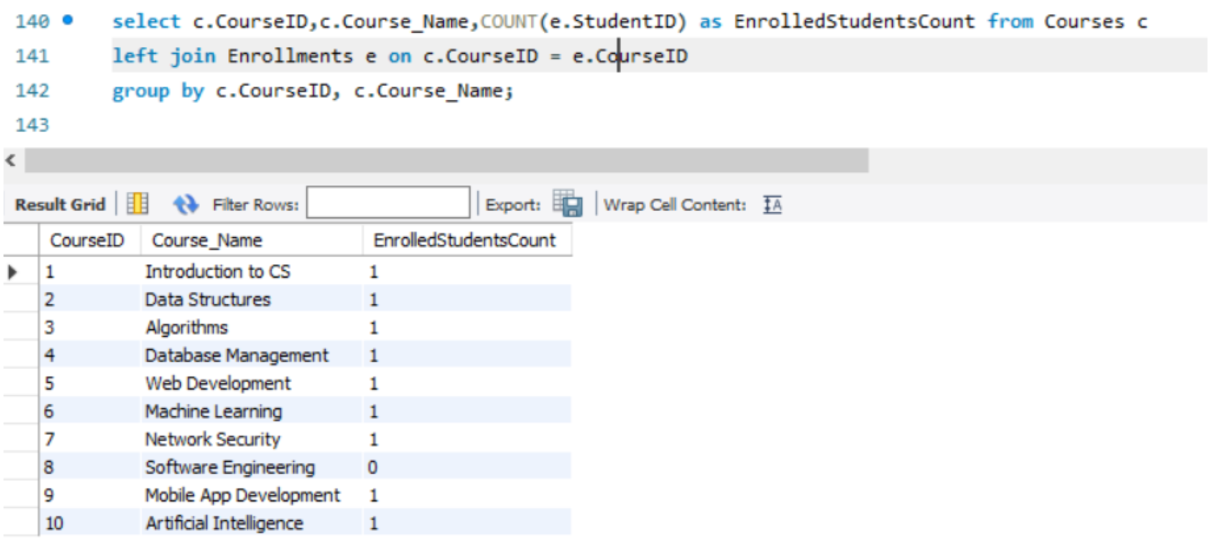
1, Write an SQL query to calculate the total payments made by a specific student. You will need to join the "Payments" table with the "Students" table based on the student's ID.



```
select s.StudentID,s.first_name, SUM(p.Amount) as TotalPayments from Students s
join Payments p on s.StudentID = p.StudentID where s.StudentID = 24520 group by s.StudentID, s.first_name;
```

StudentID	first_name	TotalPayments
24520	Rajesh	500

2, Write an SQL query to retrieve a list of courses along with the count of students enrolled in each course. Use a JOIN operation between the "Courses" table and the "Enrollments" table.

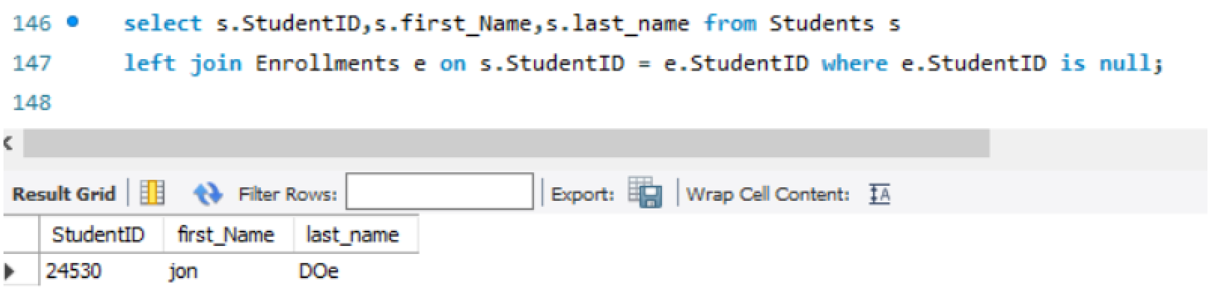


```
select c.CourseID,c.Course_Name,COUNT(e.StudentID) as EnrolledStudentsCount from Courses c
left join Enrollments e on c.CourseID = e.CourseID
group by c.CourseID, c.Course_Name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

CourseID	Course_Name	EnrolledStudentsCount
1	Introduction to CS	1
2	Data Structures	1
3	Algorithms	1
4	Database Management	1
5	Web Development	1
6	Machine Learning	1
7	Network Security	1
8	Software Engineering	0
9	Mobile App Development	1
10	Artificial Intelligence	1

3, Write an SQL query to find the names of students who have not enrolled in any course. Use a LEFT JOIN between the "Students" table and the "Enrollments" table to identify students without enrollments.

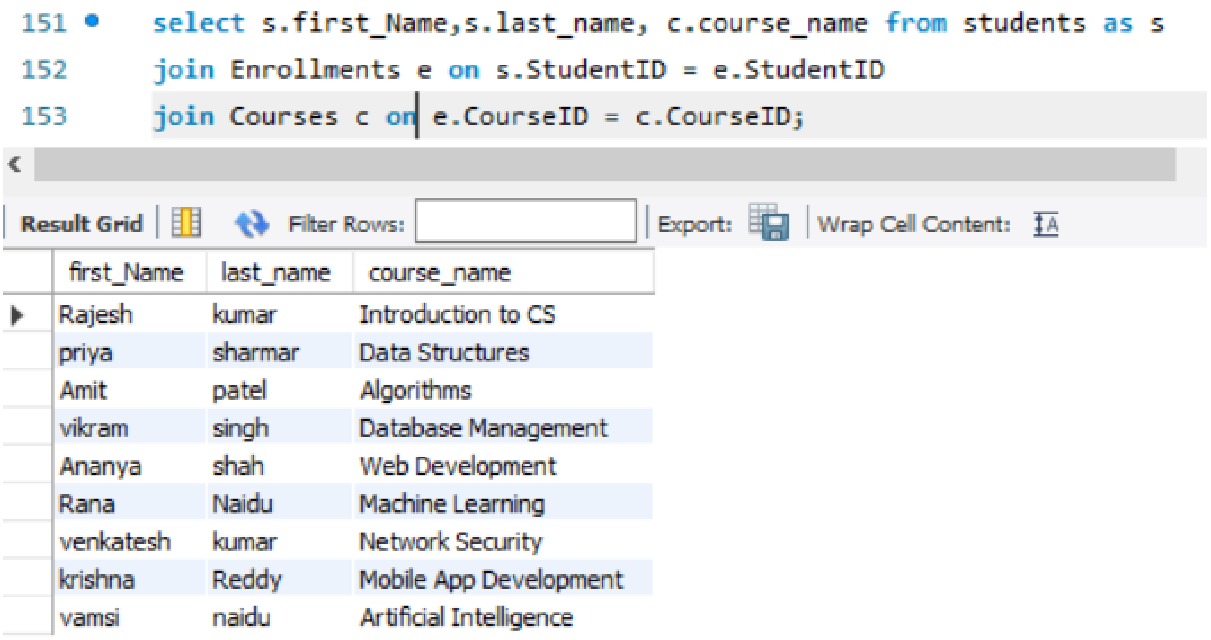


```
select s.StudentID,s.first_Name,s.last_name from Students s
left join Enrollments e on s.StudentID = e.StudentID where e.StudentID is null;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentID	first_Name	last_name
24530	jon	DOe

4, Write an SQL query to retrieve the first name, last name of students, and the names of the courses they are enrolled in. Use JOIN operations between the "Students" table and the "Enrollments" and"Courses" tables.

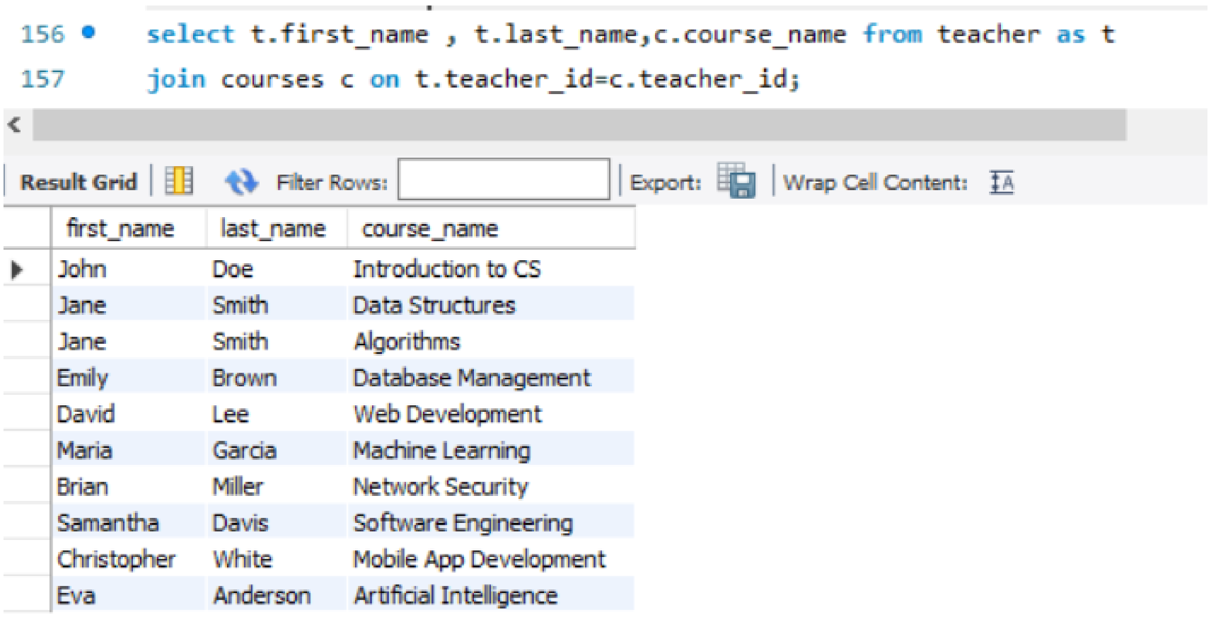


```
select s.first_Name,s.last_name, c.course_name from students as s
join Enrollments e on s.StudentID = e.StudentID
join Courses c on e.CourseID = c.CourseID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_Name	last_name	course_name
Rajesh	kumar	Introduction to CS
priya	sharmar	Data Structures
Amit	patel	Algorithms
vikram	singh	Database Management
Ananya	shah	Web Development
Rana	Naidu	Machine Learning
venkatesh	kumar	Network Security
krishna	Reddy	Mobile App Development
vamsi	naidu	Artificial Intelligence

5, Create a query to list the names of teachers and the courses they are assigned to. Join the "Teacher" table with the "Courses" table.

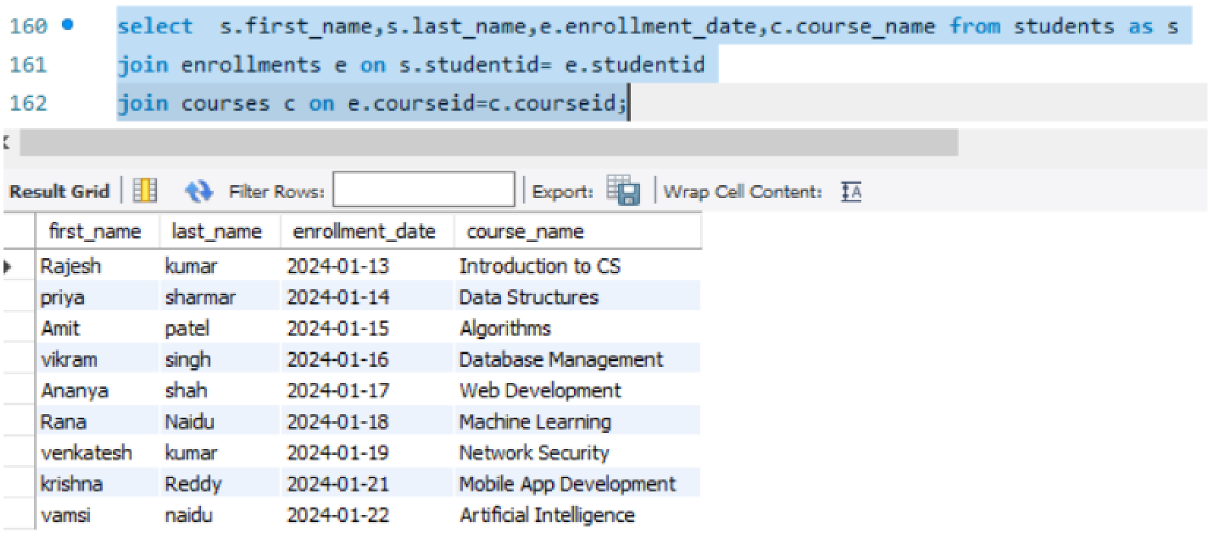


```
select t.first_name , t.last_name,c.course_name from teacher as t
join courses c on t.teacher_id=c.teacher_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	course_name
John	Doe	Introduction to CS
Jane	Smith	Data Structures
Jane	Smith	Algorithms
Emily	Brown	Database Management
David	Lee	Web Development
Maria	Garcia	Machine Learning
Brian	Miller	Network Security
Samantha	Davis	Software Engineering
Christopher	White	Mobile App Development
Eva	Anderson	Artificial Intelligence

6, Retrieve a list of students and their enrollment dates for a specific course. You'll need to join the "Students" table with the "Enrollments" and "Courses" tables.

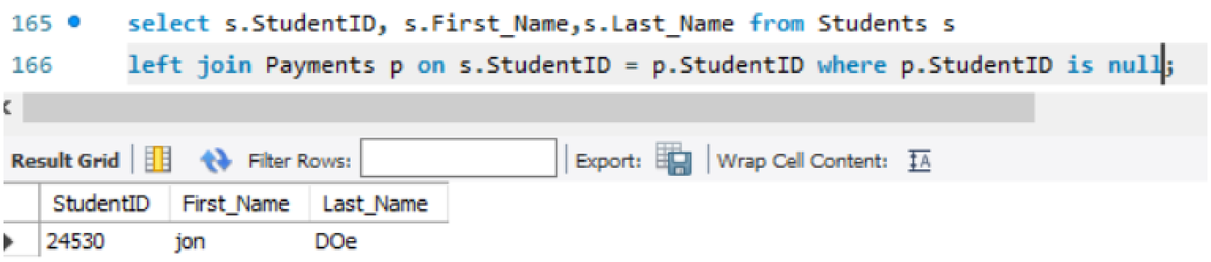


```sql
select  s.first_name,s.last_name,e.enrollment_date,c.course_name from students as s
join enrollments e on s.studentid= e.studentid
join courses c on e.courseid=c.courseid;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	enrollment_date	course_name
Rajesh	kumar	2024-01-13	Introduction to CS
priya	sharmar	2024-01-14	Data Structures
Amit	patel	2024-01-15	Algorithms
vikram	singh	2024-01-16	Database Management
Ananya	shah	2024-01-17	Web Development
Rana	Naidu	2024-01-18	Machine Learning
venkatesh	kumar	2024-01-19	Network Security
krishna	Reddy	2024-01-21	Mobile App Development
vamsi	naidu	2024-01-22	Artificial Intelligence

7, Find the names of students who have not made any payments. Use a LEFT JOIN between the "Students" table and the "Payments" table and filter for students with NULL payment records.

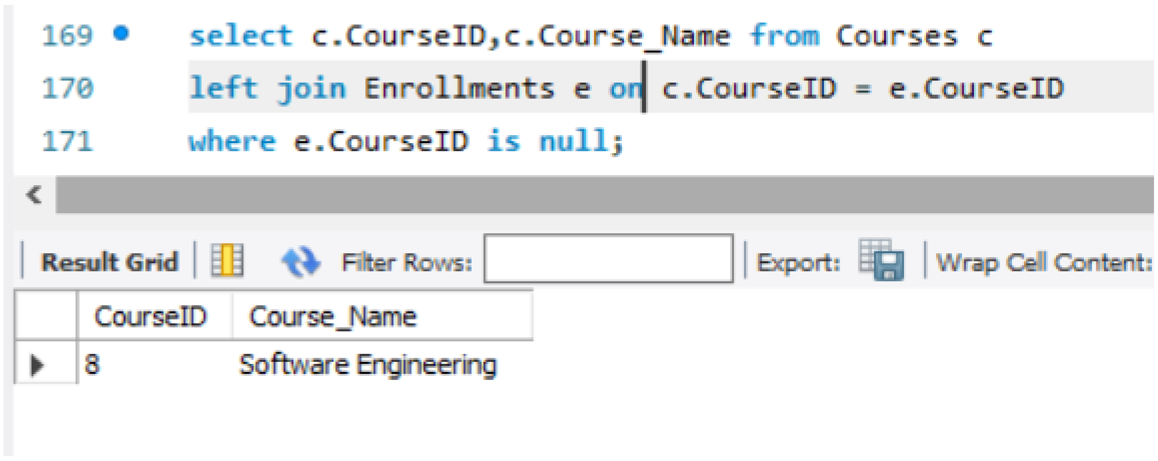


```sql
select s.StudentID, s.First_Name,s.Last_Name from Students s
left join Payments p on s.StudentID = p.StudentID where p.StudentID is null;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentID	First_Name	Last_Name
24530	jon	DOe

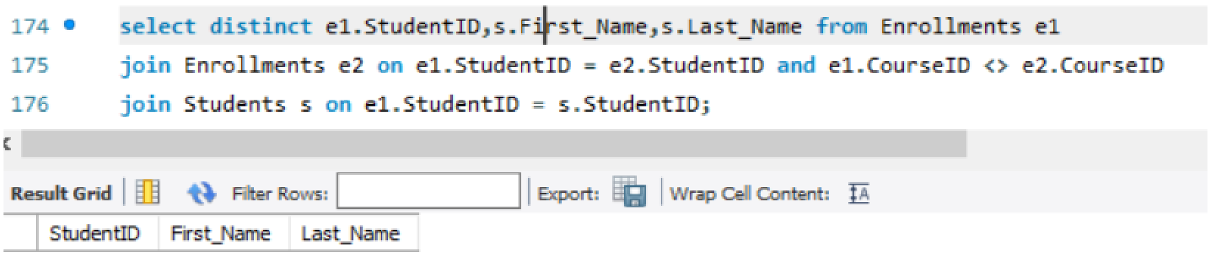
8, Write a query to identify courses that have no enrollments. You'll need to use a LEFT JOIN between the "Courses" table and the "Enrollments" table and filter for courses with NULL enrollment records



```
select c.CourseID,c.Course_Name from Courses c
left join Enrollments e on c.CourseID = e.CourseID
where e.CourseID is null;
```

CourseID	Course_Name
8	Software Engineering

9, Identify students who are enrolled in more than one course. Use a self-join on the "Enrollments" table to find students with multiple enrollment records.

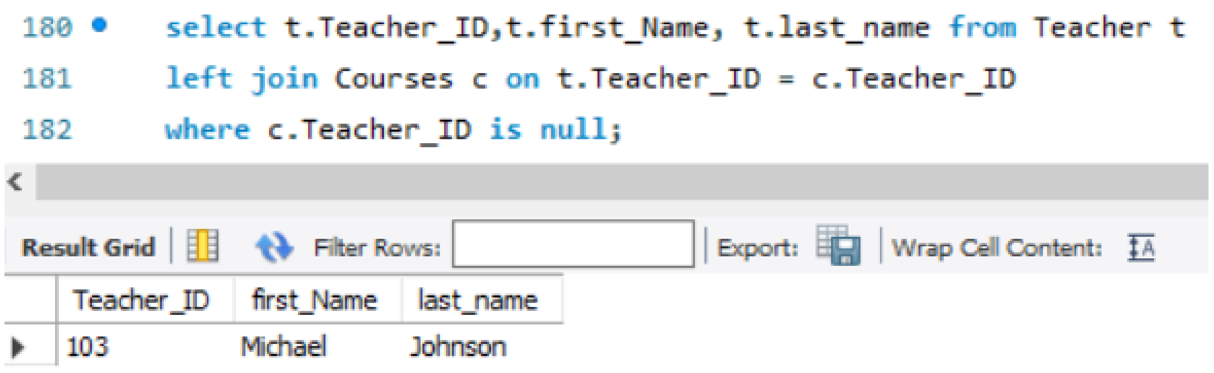


```
select distinct e1.StudentID,s.First_Name,s.Last_Name from Enrollments e1
join Enrollments e2 on e1.StudentID = e2.StudentID and e1.CourseID <> e2.CourseID
join Students s on e1.StudentID = s.StudentID;
```

StudentID	First_Name	Last_Name

Here it was not displaying any names because there are no students who have enrolled in the 2 courses

10, Find teachers who are not assigned to any courses. Use a LEFT JOIN between the "Teacher" table and the "Courses" table and filter for teachers with NULL course assignments.

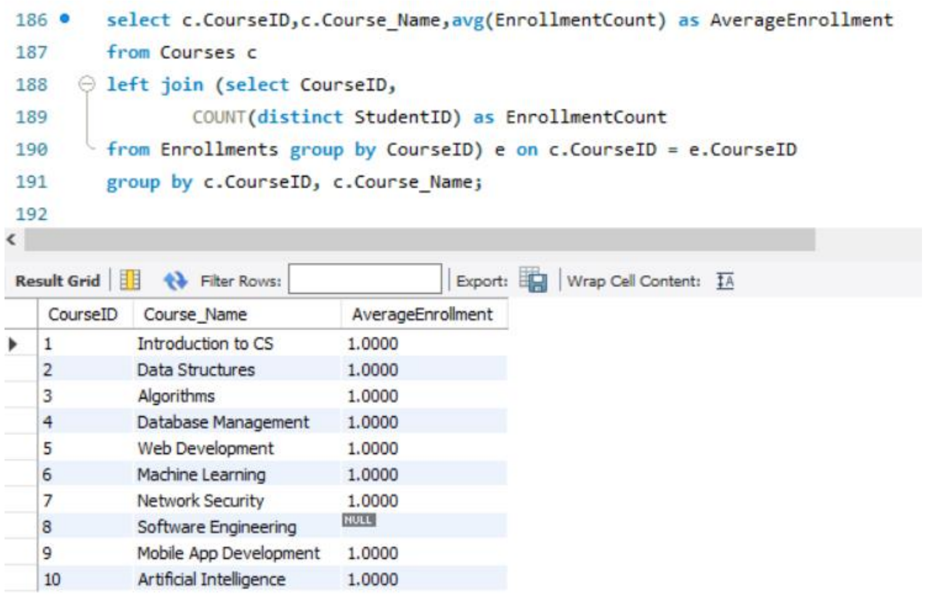


```sql
select t.Teacher_ID,t.first_Name, t.last_name from Teacher t
left join Courses c on t.Teacher_ID = c.Teacher_ID
where c.Teacher_ID is null;
```

Teacher_ID	first_Name	last_name
103	Michael	Johnson

Task -4

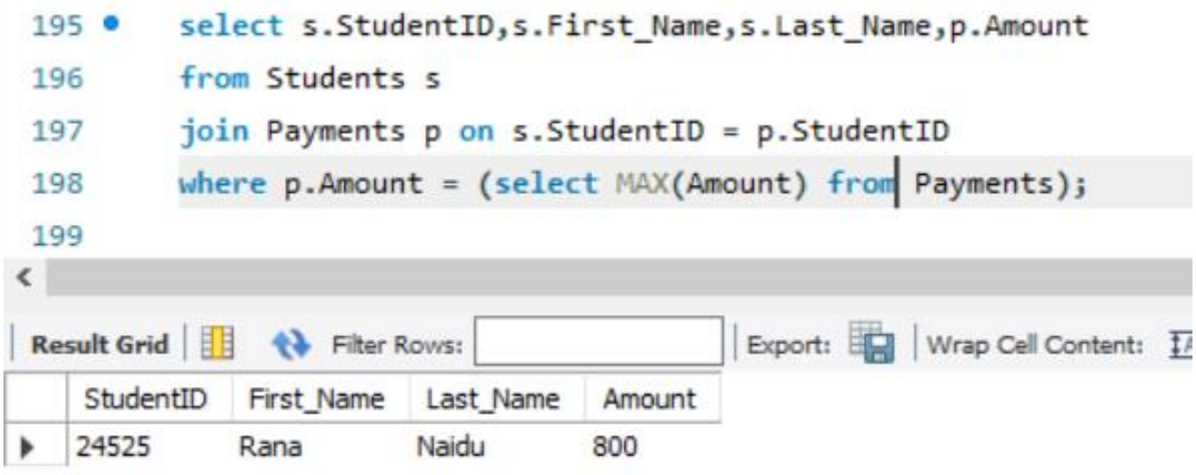
1, Write an SQL query to calculate the average number of students enrolled in each course. Use aggregate functions and subqueries to achieve this.



```sql
select c.CourseID,c.Course_Name,avg(EnrollmentCount) as AverageEnrollment
from Courses c
left join (select CourseID,
        COUNT(distinct StudentID) as EnrollmentCount
from Enrollments group by CourseID) e on c.CourseID = e.CourseID
group by c.CourseID, c.Course_Name;
```

CourseID	Course_Name	AverageEnrollment
1	Introduction to CS	1.0000
2	Data Structures	1.0000
3	Algorithms	1.0000
4	Database Management	1.0000
5	Web Development	1.0000
6	Machine Learning	1.0000
7	Network Security	1.0000
8	Software Engineering	NULL
9	Mobile App Development	1.0000
10	Artificial Intelligence	1.0000

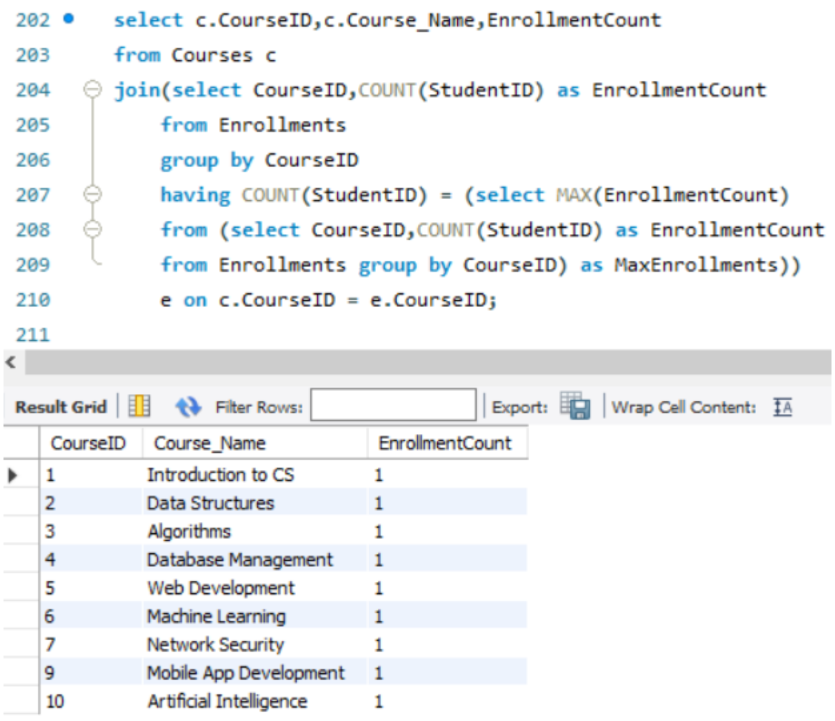
2, Identify the student(s) who made the highest payment. Use a subquery to find the maximum payment amount and then retrieve the student(s) associated with that amount.



```
select s.StudentID,s.First_Name,s.Last_Name,p.Amount
from Students s
join Payments p on s.StudentID = p.StudentID
where p.Amount = (select MAX(Amount) from Payments);
```

StudentID	First_Name	Last_Name	Amount
24525	Rana	Naidu	800

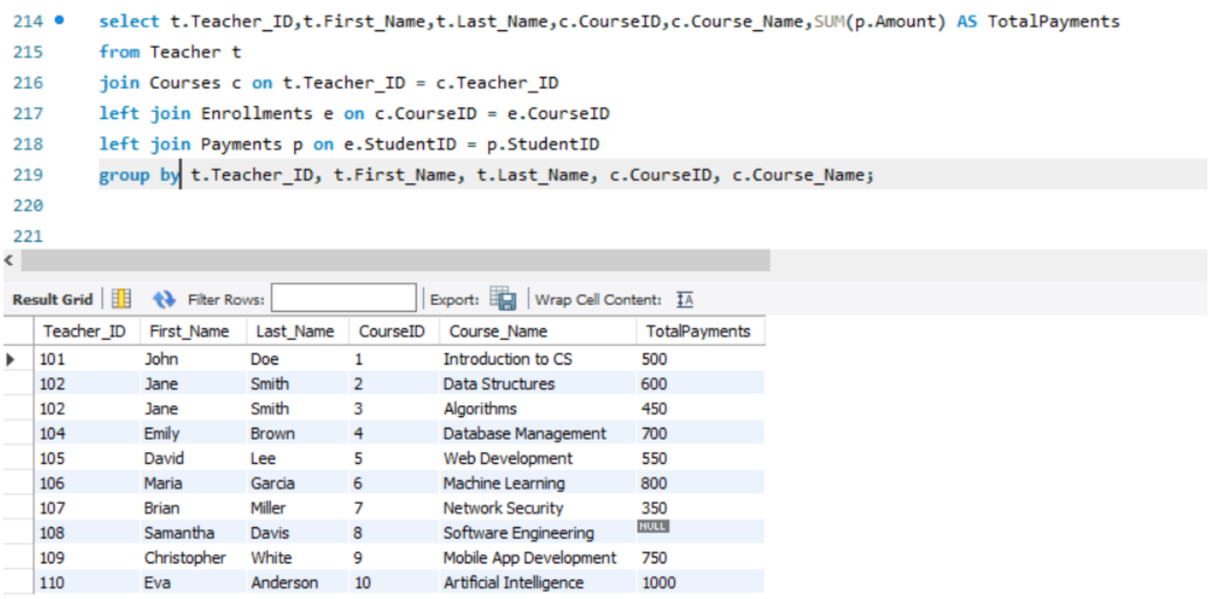
3, Retrieve a list of courses with the highest number of enrollments. Use subqueries to find the course(s) with the maximum enrollment count.



```
select c.CourseID,c.Course_Name,EnrollmentCount
from Courses c
join(select CourseID,COUNT(StudentID) as EnrollmentCount
    from Enrollments
    group by CourseID
    having COUNT(StudentID) = (select MAX(EnrollmentCount)
    from (select CourseID,COUNT(StudentID) as EnrollmentCount
    from Enrollments group by CourseID) as MaxEnrollments))
    e on c.CourseID = e.CourseID;
```

CourseID	Course_Name	EnrollmentCount
1	Introduction to CS	1
2	Data Structures	1
3	Algorithms	1
4	Database Management	1
5	Web Development	1
6	Machine Learning	1
7	Network Security	1
9	Mobile App Development	1
10	Artificial Intelligence	1

4, Calculate the total payments made to courses taught by each teacher. Use subqueries to sum payments for each teacher's courses.

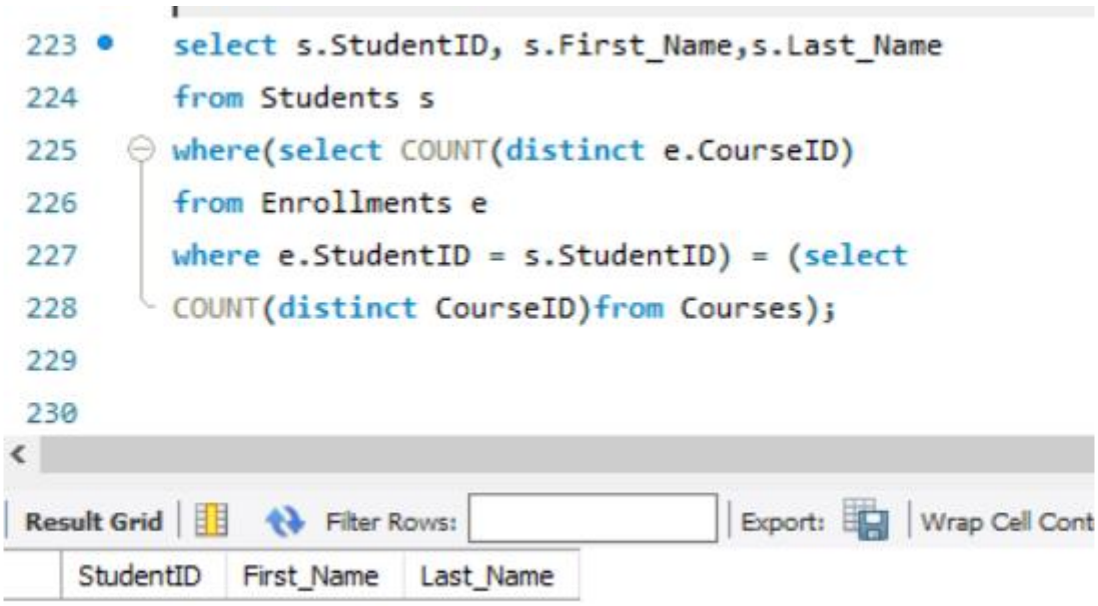


```
select t.Teacher_ID,t.First_Name,t.Last_Name,c.CourseID,c.Course_Name,SUM(p.Amount) AS TotalPayments
from Teacher t
join Courses c on t.Teacher_ID = c.Teacher_ID
left join Enrollments e on c.CourseID = e.CourseID
left join Payments p on e.StudentID = p.StudentID
group by t.Teacher_ID, t.First_Name, t.Last_Name, c.CourseID, c.Course_Name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Teacher_ID	First_Name	Last_Name	CourseID	Course_Name	TotalPayments
101	John	Doe	1	Introduction to CS	500
102	Jane	Smith	2	Data Structures	600
102	Jane	Smith	3	Algorithms	450
104	Emily	Brown	4	Database Management	700
105	David	Lee	5	Web Development	550
106	Maria	Garcia	6	Machine Learning	800
107	Brian	Miller	7	Network Security	350
108	Samantha	Davis	8	Software Engineering	NULL
109	Christopher	White	9	Mobile App Development	750
110	Eva	Anderson	10	Artificial Intelligence	1000

5, Identify students who are enrolled in all available courses. Use subqueries to compare a student's enrollments with the total number of courses



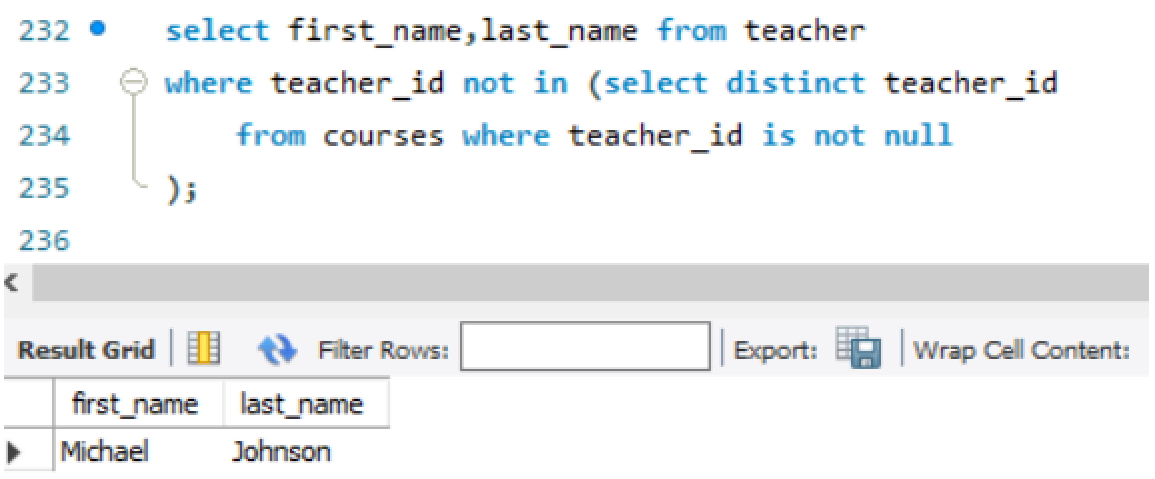
```
select s.StudentID, s.First_Name,s.Last_Name
from Students s
where(select COUNT(distinct e.CourseID)
from Enrollments e
where e.StudentID = s.StudentID) = (select
COUNT(distinct CourseID)from Courses);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

StudentID	First_Name	Last_Name

Here no student was enrolled in all courses

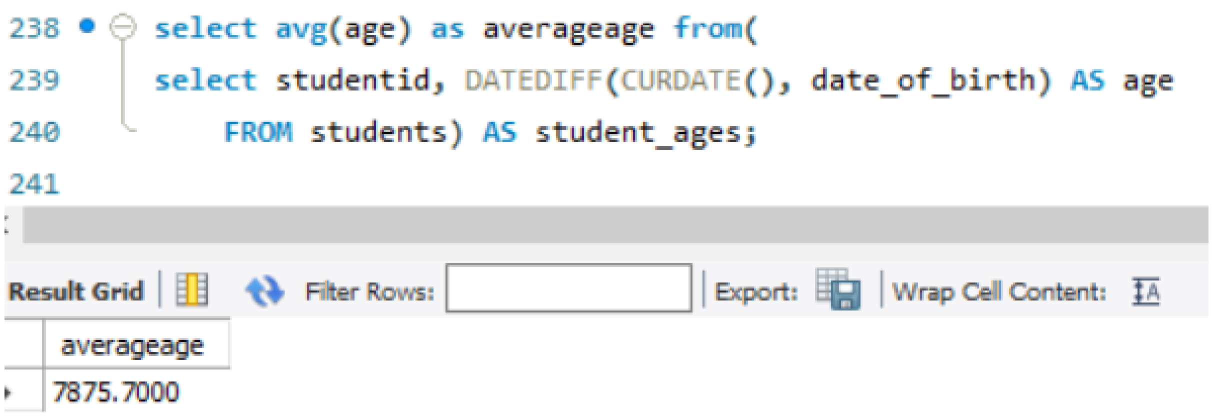
6, Retrieve the names of teachers who have not been assigned to any courses. Use subqueries to find teachers with no course assignments



```sql
select first_name,last_name from teacher
where teacher_id not in (select distinct teacher_id
    from courses where teacher_id is not null
);
```

```
first_name  last_name
Michael     Johnson
```

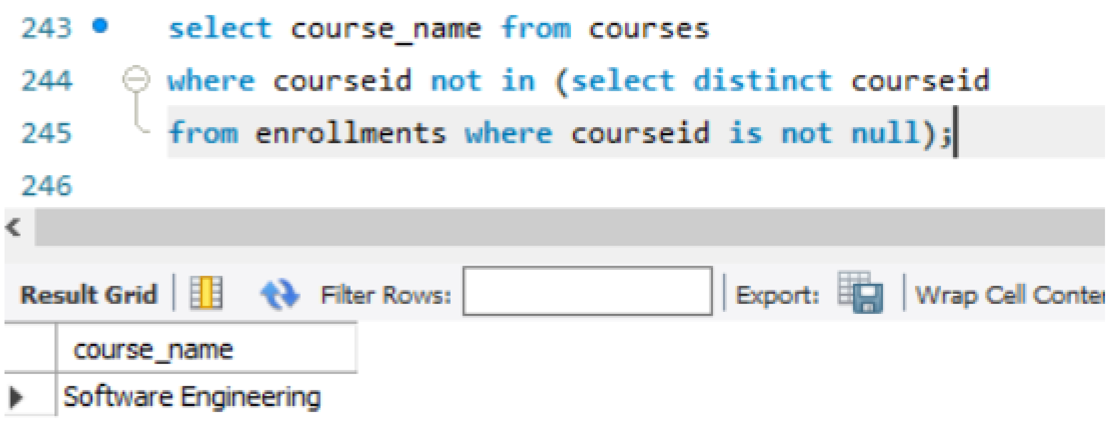
7, Calculate the average age of all students. Use subqueries to calculate the age of each student based on their date of birth.



```sql
select avg(age) as averageage from(
select studentid, DATEDIFF(CURDATE(), date_of_birth) AS age
    FROM students) AS student_ages;
```

```
averageage
7875.7000
```

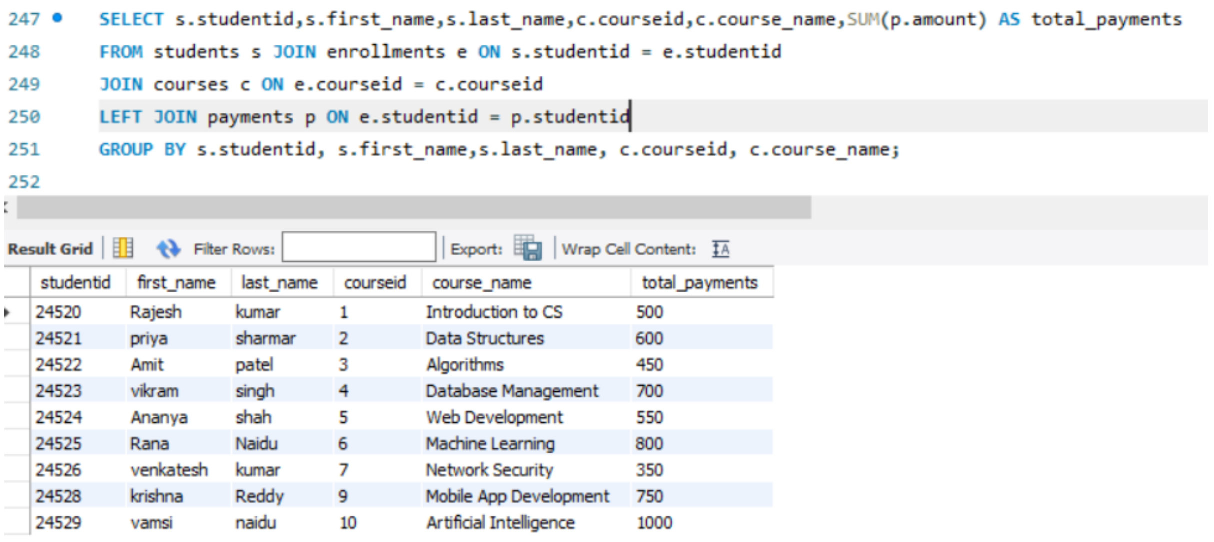
8, Identify courses with no enrollments. Use subqueries to find courses without enrollment records.



```sql
select course_name from courses
where courseid not in (select distinct courseid
from enrollments where courseid is not null);
```

```
course_name
Software Engineering
```

9, Calculate the total payments made by each student for each course they are enrolled in. Use subqueries and aggregate functions to sum payments.

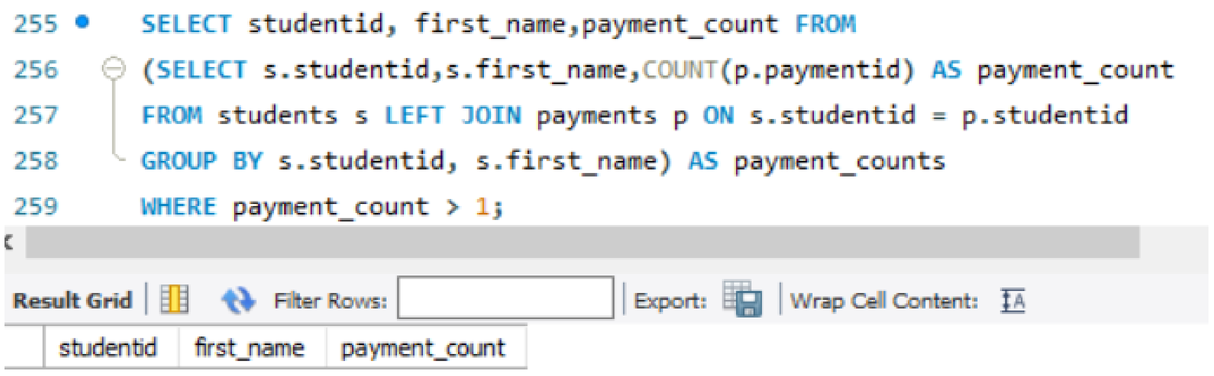


```sql
SELECT s.studentid,s.first_name,s.last_name,c.courseid,c.course_name,SUM(p.amount) AS total_payments
FROM students s JOIN enrollments e ON s.studentid = e.studentid
JOIN courses c ON e.courseid = c.courseid
LEFT JOIN payments p ON e.studentid = p.studentid
GROUP BY s.studentid, s.first_name,s.last_name, c.courseid, c.course_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

```
studentid  first_name  last_name  courseid  course_name             total_payments
24520      Rajesh      kumar      1         Introduction to CS      500
24521      priya       sharmar    2         Data Structures         600
24522      Amit        patel      3         Algorithms              450
24523      vikram      singh      4         Database Management     700
24524      Ananya      shah       5         Web Development         550
24525      Rana        Naidu      6         Machine Learning        800
24526      venkatesh   kumar      7         Network Security        350
24528      krishna     Reddy      9         Mobile App Development  750
24529      vamsi       naidu      10        Artificial Intelligence 1000
```

10, Identify students who have made more than one payment. Use subqueries and aggregate functions to count payments per student and filter for those with counts greater than one.

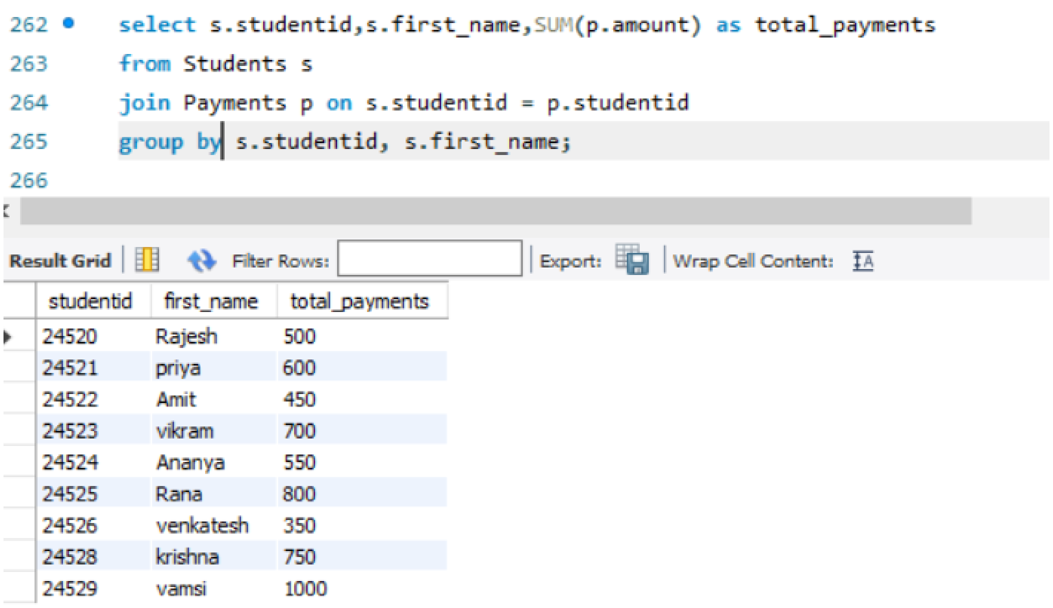


```sql
SELECT studentid, first_name,payment_count FROM
(SELECT s.studentid,s.first_name,COUNT(p.paymentid) AS payment_count
FROM students s LEFT JOIN payments p ON s.studentid = p.studentid
GROUP BY s.studentid, s.first_name) AS payment_counts
WHERE payment_count > 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

```
studentid  first_name  payment_count
```

11, Write an SQL query to calculate the total payments made by each student. Join the "Students" table with the "Payments" table and use GROUP BY to calculate the sum of payments for each student.

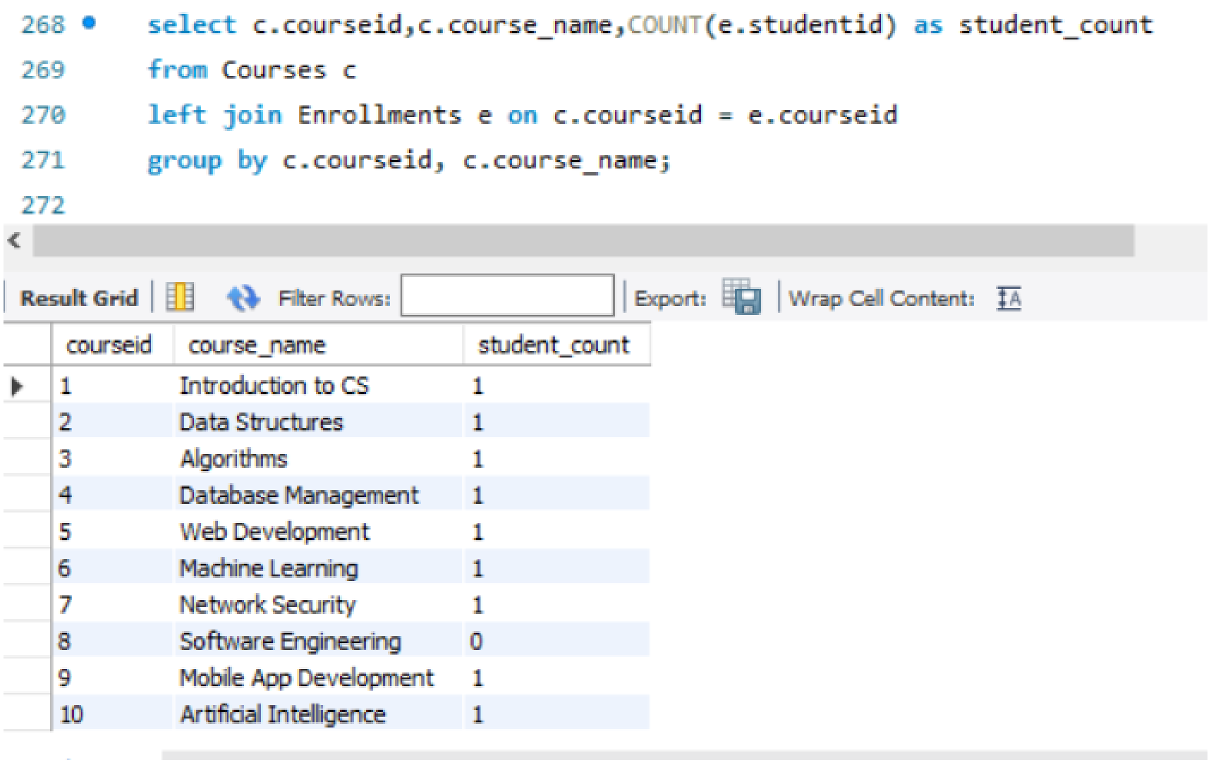


```
select s.studentid,s.first_name,SUM(p.amount) as total_payments
from Students s
join Payments p on s.studentid = p.studentid
group by s.studentid, s.first_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

studentid	first_name	total_payments
24520	Rajesh	500
24521	priya	600
24522	Amit	450
24523	vikram	700
24524	Ananya	550
24525	Rana	800
24526	venkatesh	350
24528	krishna	750
24529	vamsi	1000

12, Retrieve a list of course names along with the count of students enrolled in each course. Use JOIN operations between the "Courses" table and the "Enrollments" table and GROUP BY to count enrollments.

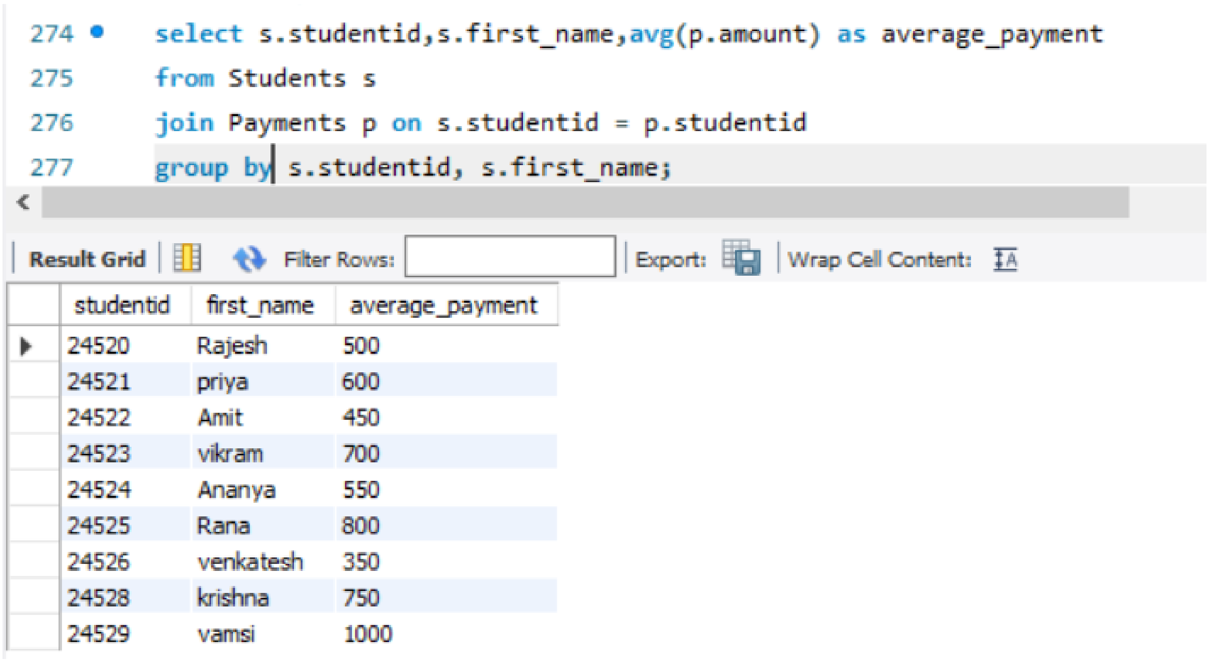


```
select c.courseid,c.course_name,COUNT(e.studentid) as student_count
from Courses c
left join Enrollments e on c.courseid = e.courseid
group by c.courseid, c.course_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

courseid	course_name	student_count
1	Introduction to CS	1
2	Data Structures	1
3	Algorithms	1
4	Database Management	1
5	Web Development	1
6	Machine Learning	1
7	Network Security	1
8	Software Engineering	0
9	Mobile App Development	1
10	Artificial Intelligence	1

13, Calculate the average payment amount made by students. Use JOIN operations between the"Students" table and the "Payments" table and GROUP BY to calculate the average



```sql
select s.studentid,s.first_name,avg(p.amount) as average_payment
from Students s
join Payments p on s.studentid = p.studentid
group by s.studentid, s.first_name;
```

studentid	first_name	average_payment
24520	Rajesh	500
24521	priya	600
24522	Amit	450
24523	vikram	700
24524	Ananya	550
24525	Rana	800
24526	venkatesh	350
24528	krishna	750
24529	vamsi	1000

Submitted By:

Devaki Akash