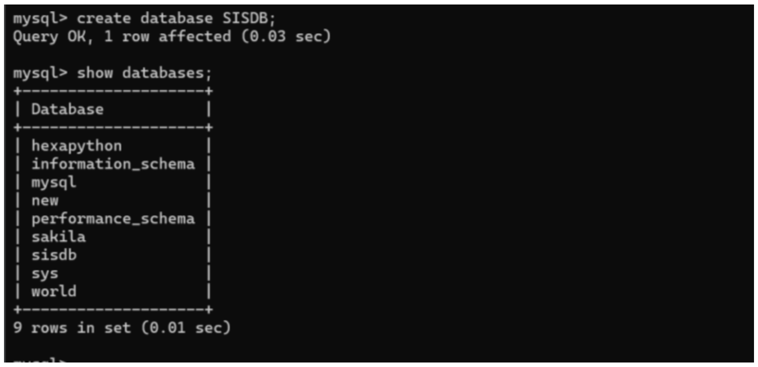
Assignment SQL - Student Information System

Name: Soundarya V

Batch: Python batch 2

TASK 1 – DATABASE DESIGN:

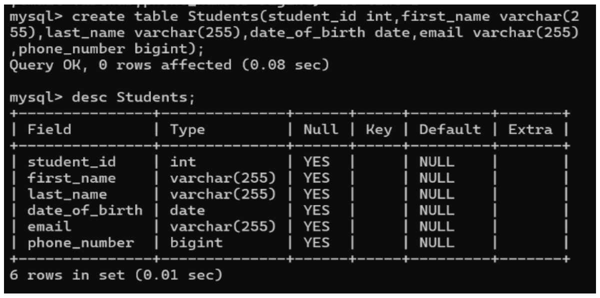
1. Create the database named "SISDB"



```
mysql> create database SISDB;
Query OK, 1 row affected (0.03 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| hexapython         |
| information_schema |
| mysql              |
| new                |
| performance_schema |
| sakila             |
| sisdb              |
| sys                |
| world              |
+--------------------+
9 rows in set (0.01 sec)
```

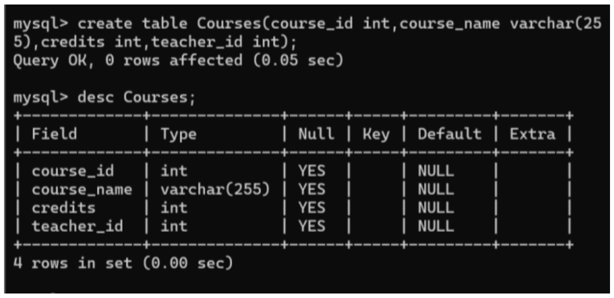
2. Define the schema for the Students, Courses, Enrollments, Teacher, and Payments tables based on the provided schema. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.
a. Students b. Courses c. Enrollments d. Teacher e. Payments

Table-Students:



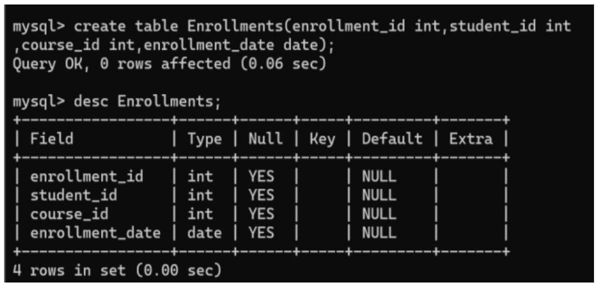
```
mysql> create table Students(student_id int,first_name varchar(2
55),last_name varchar(255),date_of_birth date,email varchar(255)
,phone_number bigint);
Query OK, 0 rows affected (0.08 sec)

mysql> desc Students;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| student_id    | int          | YES  |     | NULL    |       |
| first_name    | varchar(255) | YES  |     | NULL    |       |
| last_name     | varchar(255) | YES  |     | NULL    |       |
| date_of_birth | date         | YES  |     | NULL    |       |
| email         | varchar(255) | YES  |     | NULL    |       |
| phone_number  | bigint       | YES  |     | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

Table-Courses:



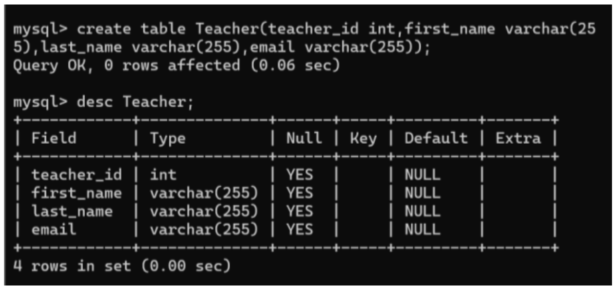
```
mysql> create table Courses(course_id int,course_name varchar(25
5),credits int,teacher_id int);
Query OK, 0 rows affected (0.05 sec)

mysql> desc Courses;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| course_id   | int          | YES  |     | NULL    |       |
| course_name | varchar(255) | YES  |     | NULL    |       |
| credits     | int          | YES  |     | NULL    |       |
| teacher_id  | int          | YES  |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Table-Enrollments:



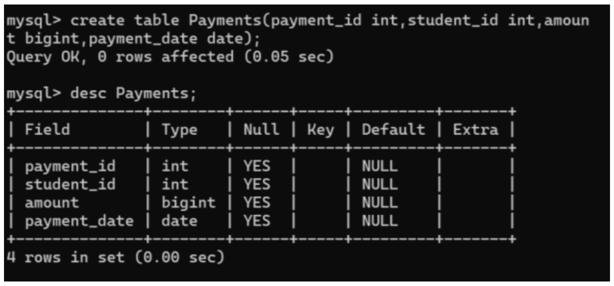
```
mysql> create table Enrollments(enrollment_id int,student_id int
,course_id int,enrollment_date date);
Query OK, 0 rows affected (0.06 sec)

mysql> desc Enrollments;
+-----------------+------+------+-----+---------+-------+
| Field           | Type | Null | Key | Default | Extra |
+-----------------+------+------+-----+---------+-------+
| enrollment_id   | int  | YES  |     | NULL    |       |
| student_id      | int  | YES  |     | NULL    |       |
| course_id       | int  | YES  |     | NULL    |       |
| enrollment_date | date | YES  |     | NULL    |       |
+-----------------+------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Table-Teachers:



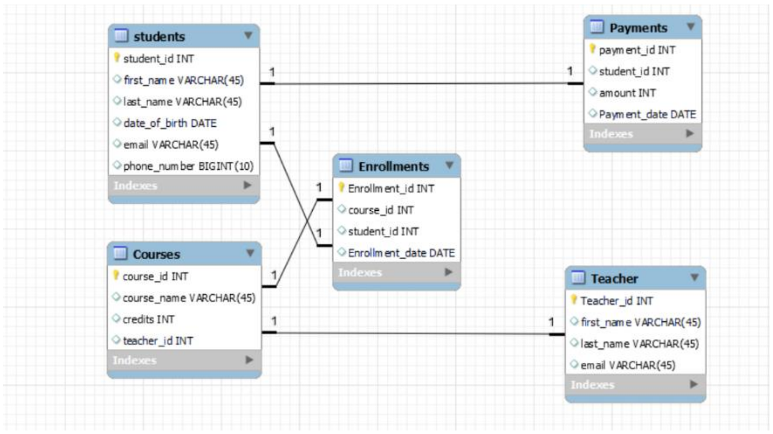
```
mysql> create table Teacher(teacher_id int,first_name varchar(25
5),last_name varchar(255),email varchar(255));
Query OK, 0 rows affected (0.06 sec)

mysql> desc Teacher;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| teacher_id | int          | YES  |     | NULL    |       |
| first_name | varchar(255) | YES  |     | NULL    |       |
| last_name  | varchar(255) | YES  |     | NULL    |       |
| email      | varchar(255) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Table-Payments:

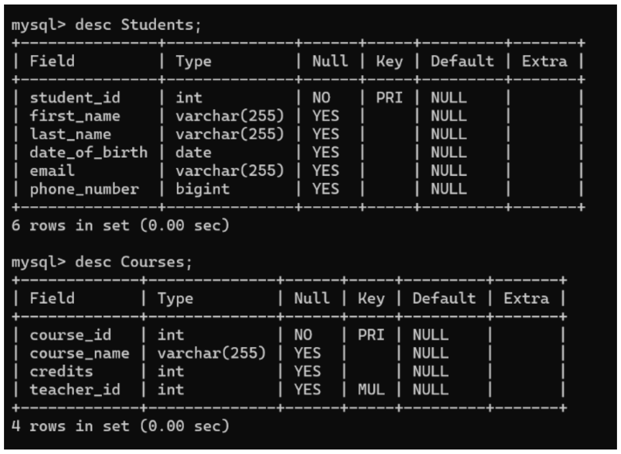


```
mysql> create table Payments(payment_id int,student_id int,amoun
t bigint,payment_date date);
Query OK, 0 rows affected (0.05 sec)

mysql> desc Payments;
+--------------+--------+------+-----+---------+-------+
| Field        | Type   | Null | Key | Default | Extra |
+--------------+--------+------+-----+---------+-------+
| payment_id   | int    | YES  |     | NULL    |       |
| student_id   | int    | YES  |     | NULL    |       |
| amount       | bigint | YES  |     | NULL    |       |
| payment_date | date   | YES  |     | NULL    |       |
+--------------+--------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

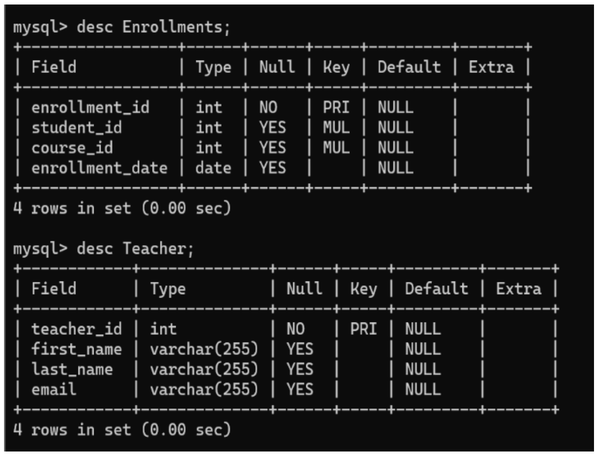
3.Create an ERD (Entity Relationship Diagram) for the database.



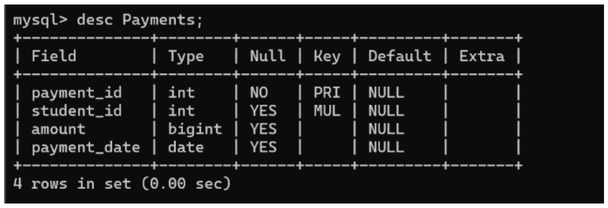
4.Create appropriate Primary Key and Foreign Key constraints for referential integrity



```
mysql> desc Students;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| student_id    | int          | NO   | PRI | NULL    |       |
| first_name    | varchar(255) | YES  |     | NULL    |       |
| last_name     | varchar(255) | YES  |     | NULL    |       |
| date_of_birth | date         | YES  |     | NULL    |       |
| email         | varchar(255) | YES  |     | NULL    |       |
| phone_number  | bigint       | YES  |     | NULL    |       |
+---------------+--------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> desc Courses;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| course_id   | int          | NO   | PRI | NULL    |       |
| course_name | varchar(255) | YES  |     | NULL    |       |
| credits     | int          | YES  |     | NULL    |       |
| teacher_id  | int          | YES  | MUL | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```



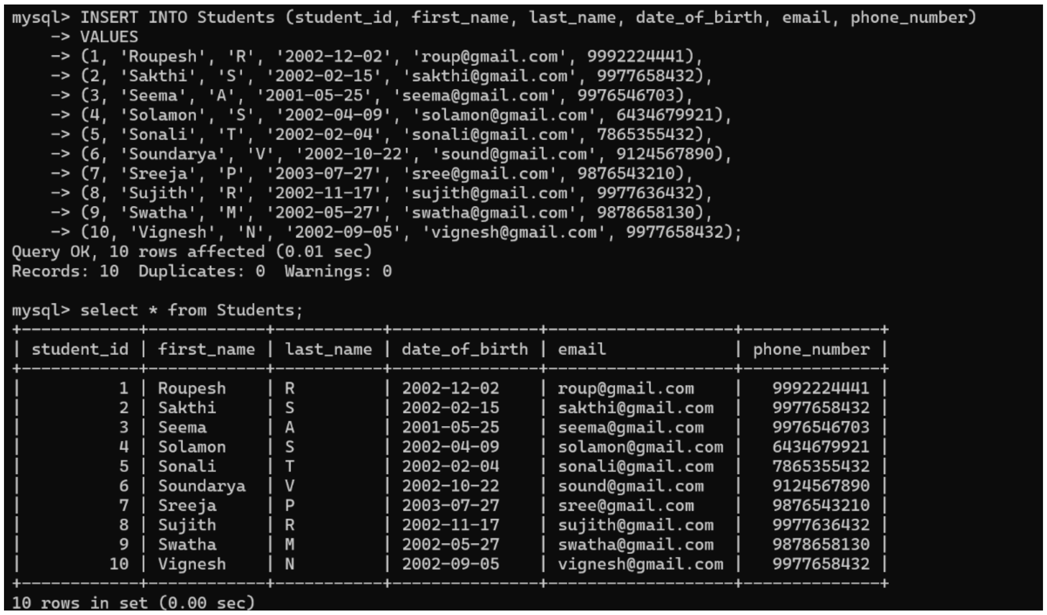
```
mysql> desc Enrollments;
+-----------------+------+------+-----+---------+-------+
| Field           | Type | Null | Key | Default | Extra |
+-----------------+------+------+-----+---------+-------+
| enrollment_id   | int  | NO   | PRI | NULL    |       |
| student_id      | int  | YES  | MUL | NULL    |       |
| course_id       | int  | YES  | MUL | NULL    |       |
| enrollment_date | date | YES  |     | NULL    |       |
+-----------------+------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc Teacher;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| teacher_id | int          | NO   | PRI | NULL    |       |
| first_name | varchar(255) | YES  |     | NULL    |       |
| last_name  | varchar(255) | YES  |     | NULL    |       |
| email      | varchar(255) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```



```
mysql> desc Payments;
+--------------+--------+------+-----+---------+-------+
| Field        | Type   | Null | Key | Default | Extra |
+--------------+--------+------+-----+---------+-------+
| payment_id   | int    | NO   | PRI | NULL    |       |
| student_id   | int    | YES  | MUL | NULL    |       |
| amount       | bigint | YES  |     | NULL    |       |
| payment_date | date   | YES  |     | NULL    |       |
+--------------+--------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

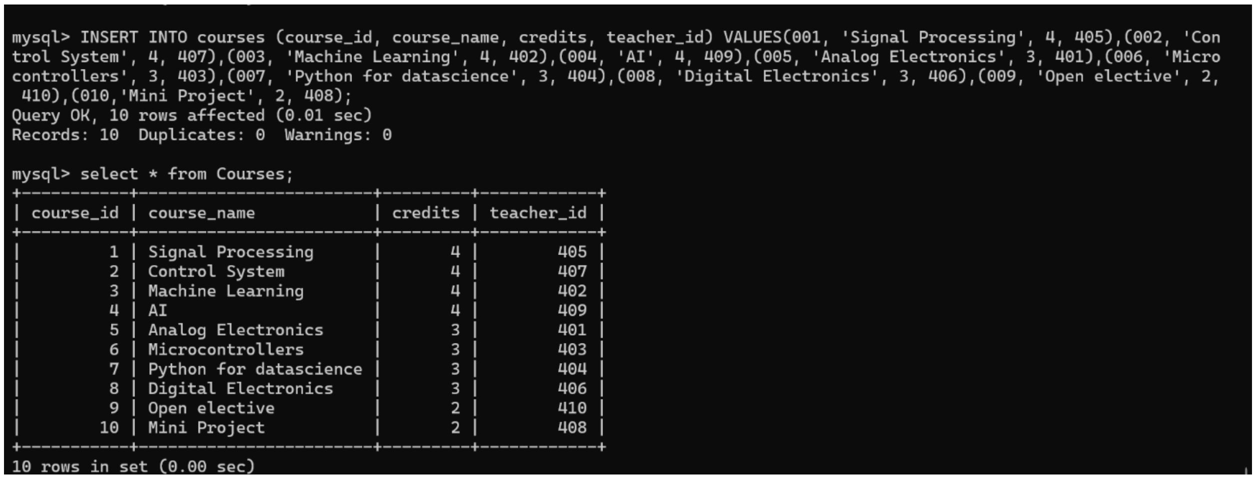
5.Insert at least 10 sample records into each of the following tables.

i.Students:



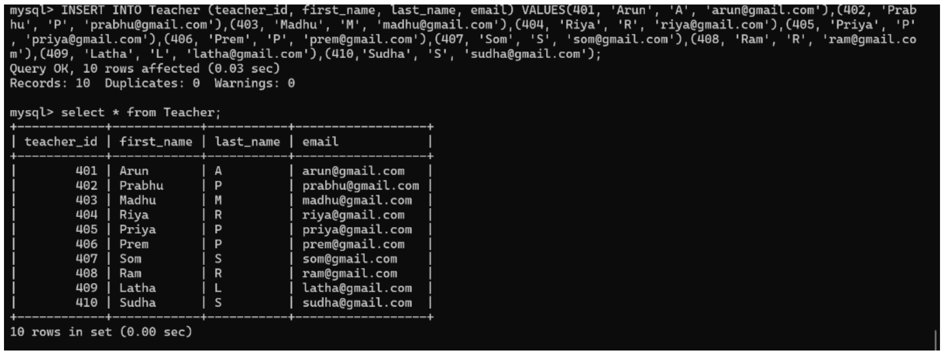
```
mysql> INSERT INTO Students (student_id, first_name, last_name, date_of_birth, email, phone_number)
    -> VALUES
    -> (1, 'Roupesh', 'R', '2002-12-02', 'roup@gmail.com', 9992224441),
    -> (2, 'Sakthi', 'S', '2002-02-15', 'sakthi@gmail.com', 9977658432),
    -> (3, 'Seema', 'A', '2001-05-25', 'seema@gmail.com', 9976546703),
    -> (4, 'Solamon', 'S', '2002-04-09', 'solamon@gmail.com', 6434679921),
    -> (5, 'Sonali', 'T', '2002-02-04', 'sonali@gmail.com', 7865355432),
    -> (6, 'Soundarya', 'V', '2002-10-22', 'sound@gmail.com', 9124567890),
    -> (7, 'Sreeja', 'P', '2003-07-27', 'sree@gmail.com', 9876543210),
    -> (8, 'Sujith', 'R', '2002-11-17', 'sujith@gmail.com', 9977636432),
    -> (9, 'Swatha', 'M', '2002-05-27', 'swatha@gmail.com', 9878658130),
    -> (10, 'Vignesh', 'N', '2002-09-05', 'vignesh@gmail.com', 9977658432);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Students;
+------------+------------+-----------+---------------+-------------------+--------------+
| student_id | first_name | last_name | date_of_birth | email             | phone_number |
+------------+------------+-----------+---------------+-------------------+--------------+
|          1 | Roupesh    | R         | 2002-12-02    | roup@gmail.com    |   9992224441 |
|          2 | Sakthi     | S         | 2002-02-15    | sakthi@gmail.com  |   9977658432 |
|          3 | Seema      | A         | 2001-05-25    | seema@gmail.com   |   9976546703 |
|          4 | Solamon    | S         | 2002-04-09    | solamon@gmail.com |   6434679921 |
|          5 | Sonali     | T         | 2002-02-04    | sonali@gmail.com  |   7865355432 |
|          6 | Soundarya  | V         | 2002-10-22    | sound@gmail.com   |   9124567890 |
|          7 | Sreeja     | P         | 2003-07-27    | sree@gmail.com    |   9876543210 |
|          8 | Sujith     | R         | 2002-11-17    | sujith@gmail.com  |   9977636432 |
|          9 | Swatha     | M         | 2002-05-27    | swatha@gmail.com  |   9878658130 |
|         10 | Vignesh    | N         | 2002-09-05    | vignesh@gmail.com |   9977658432 |
+------------+------------+-----------+---------------+-------------------+--------------+
10 rows in set (0.00 sec)
```

ii.Courses:



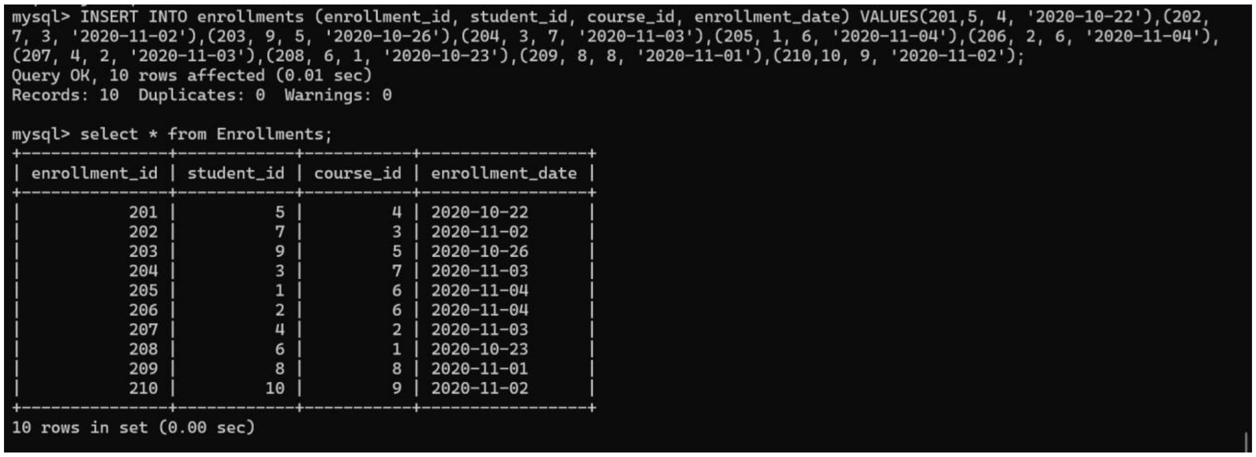
```
mysql> INSERT INTO courses (course_id, course_name, credits, teacher_id) VALUES(001, 'Signal Processing', 4, 405),(002, 'Con
trol System', 4, 407),(003, 'Machine Learning', 4, 402),(004, 'AI', 4, 409),(005, 'Analog Electronics', 3, 401),(006, 'Micro
controllers', 3, 403),(007, 'Python for datascience', 3, 404),(008, 'Digital Electronics', 3, 406),(009, 'Open elective', 2,
 410),(010,'Mini Project', 2, 408);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Courses;
+-----------+------------------------+---------+------------+
| course_id | course_name            | credits | teacher_id |
+-----------+------------------------+---------+------------+
|         1 | Signal Processing      |       4 |        405 |
|         2 | Control System         |       4 |        407 |
|         3 | Machine Learning       |       4 |        402 |
|         4 | AI                     |       4 |        409 |
|         5 | Analog Electronics     |       3 |        401 |
|         6 | Microcontrollers       |       3 |        403 |
|         7 | Python for datascience |       3 |        404 |
|         8 | Digital Electronics    |       3 |        406 |
|         9 | Open elective          |       2 |        410 |
|        10 | Mini Project           |       2 |        408 |
+-----------+------------------------+---------+------------+
10 rows in set (0.00 sec)
```

iii.Teacher:



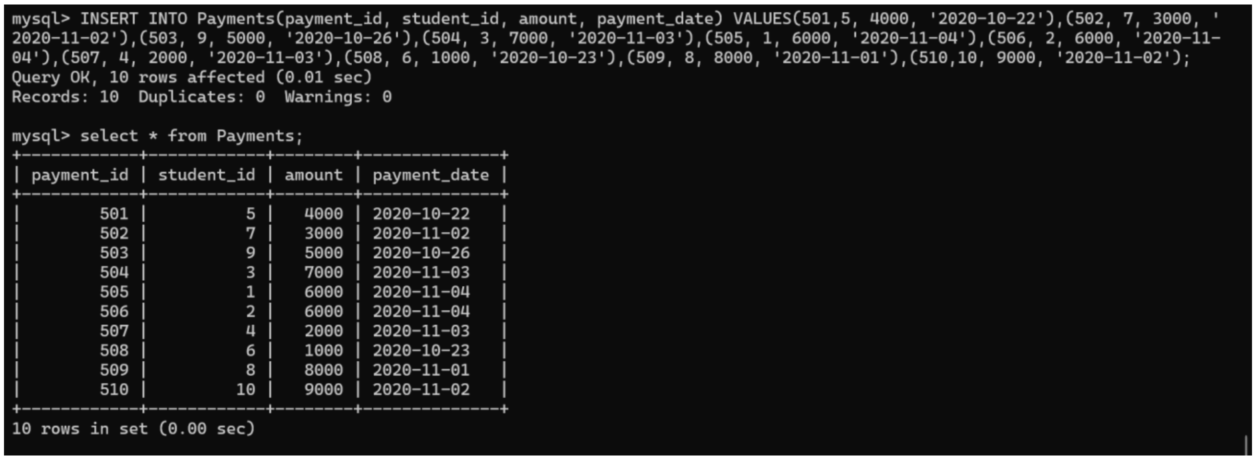
```
mysql> INSERT INTO Teacher (teacher_id, first_name, last_name, email) VALUES(401, 'Arun', 'A', 'arun@gmail.com'),(402, 'Prab
hu', 'P', 'prabhu@gmail.com'),(403, 'Madhu', 'M', 'madhu@gmail.com'),(404, 'Riya', 'R', 'riya@gmail.com'),(405, 'Priya', 'P'
, 'priya@gmail.com'),(406, 'Prem', 'P', 'prem@gmail.com'),(407, 'Som', 'S', 'som@gmail.com'),(408, 'Ram', 'R', 'ram@gmail.co
m'),(409, 'Latha', 'L', 'latha@gmail.com'),(410,'Sudha', 'S', 'sudha@gmail.com');
Query OK, 10 rows affected (0.03 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Teacher;
+------------+------------+-----------+------------------+
| teacher_id | first_name | last_name | email            |
+------------+------------+-----------+------------------+
|        401 | Arun       | A         | arun@gmail.com   |
|        402 | Prabhu     | P         | prabhu@gmail.com |
|        403 | Madhu      | M         | madhu@gmail.com  |
|        404 | Riya       | R         | riya@gmail.com   |
|        405 | Priya      | P         | priya@gmail.com  |
|        406 | Prem       | P         | prem@gmail.com   |
|        407 | Som        | S         | som@gmail.com    |
|        408 | Ram        | R         | ram@gmail.com    |
|        409 | Latha      | L         | latha@gmail.com  |
|        410 | Sudha      | S         | sudha@gmail.com  |
+------------+------------+-----------+------------------+
10 rows in set (0.00 sec)
```

iv.Enrollments:



```
mysql> INSERT INTO enrollments (enrollment_id, student_id, course_id, enrollment_date) VALUES(201,5, 4, '2020-10-22'),(202,
7, 3, '2020-11-02'),(203, 9, 5, '2020-10-26'),(204, 3, 7, '2020-11-03'),(205, 1, 6, '2020-11-04'),(206, 2, 6, '2020-11-04'),
(207, 4, 2, '2020-11-03'),(208, 6, 1, '2020-10-23'),(209, 8, 8, '2020-11-01'),(210,10, 9, '2020-11-02');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Enrollments;
+---------------+------------+-----------+-----------------+
| enrollment_id | student_id | course_id | enrollment_date |
+---------------+------------+-----------+-----------------+
|           201 |          5 |         4 | 2020-10-22      |
|           202 |          7 |         3 | 2020-11-02      |
|           203 |          9 |         5 | 2020-10-26      |
|           204 |          3 |         7 | 2020-11-03      |
|           205 |          1 |         6 | 2020-11-04      |
|           206 |          2 |         6 | 2020-11-04      |
|           207 |          4 |         2 | 2020-11-03      |
|           208 |          6 |         1 | 2020-10-23      |
|           209 |          8 |         8 | 2020-11-01      |
|           210 |         10 |         9 | 2020-11-02      |
+---------------+------------+-----------+-----------------+
10 rows in set (0.00 sec)
```

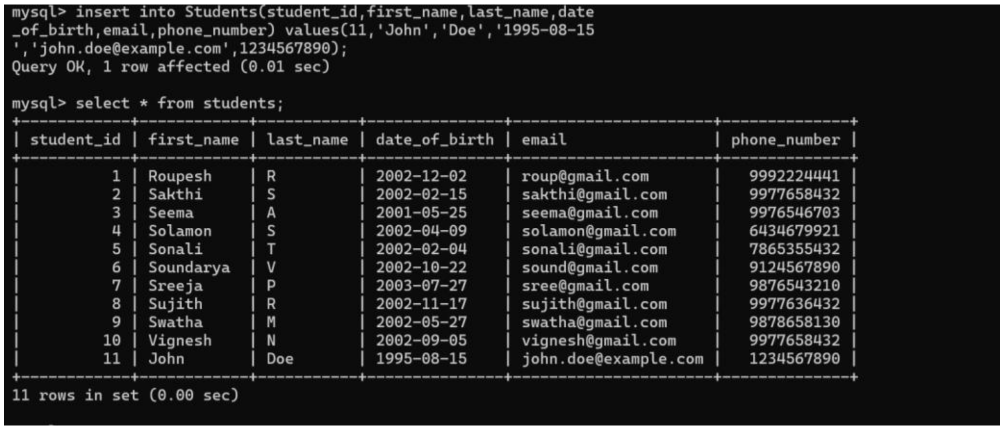
v.Payments:



```
mysql> INSERT INTO Payments(payment_id, student_id, amount, payment_date) VALUES(501,5, 4000, '2020-10-22'),(502, 7, 3000, '
2020-11-02'),(503, 9, 5000, '2020-10-26'),(504, 3, 7000, '2020-11-03'),(505, 1, 6000, '2020-11-04'),(506, 2, 6000, '2020-11-
04'),(507, 4, 2000, '2020-11-03'),(508, 6, 1000, '2020-10-23'),(509, 8, 8000, '2020-11-01'),(510,10, 9000, '2020-11-02');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> select * from Payments;
+------------+------------+--------+--------------+
| payment_id | student_id | amount | payment_date |
+------------+------------+--------+--------------+
|        501 |          5 |   4000 | 2020-10-22   |
|        502 |          7 |   3000 | 2020-11-02   |
|        503 |          9 |   5000 | 2020-10-26   |
|        504 |          3 |   7000 | 2020-11-03   |
|        505 |          1 |   6000 | 2020-11-04   |
|        506 |          2 |   6000 | 2020-11-04   |
|        507 |          4 |   2000 | 2020-11-03   |
|        508 |          6 |   1000 | 2020-10-23   |
|        509 |          8 |   8000 | 2020-11-01   |
|        510 |         10 |   9000 | 2020-11-02   |
+------------+------------+--------+--------------+
10 rows in set (0.00 sec)
```

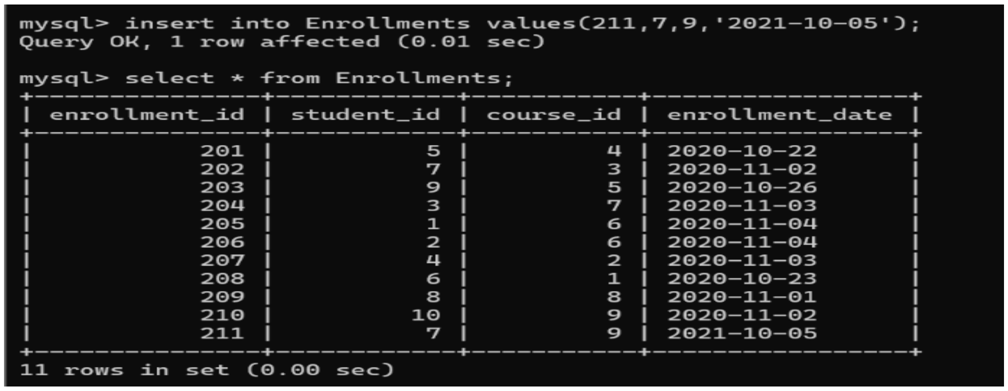
TASKS 2: SELECT, WHERE, BETWEEN, AND, LIKE:

1. Write an SQL query to insert a new student into the "Students" table with the following details: a. First Name: b. Last Name: Doe c. Date of Birth: 1995-08-15 d. Email: john.doe@example.com e. Phone Number: 1234567890



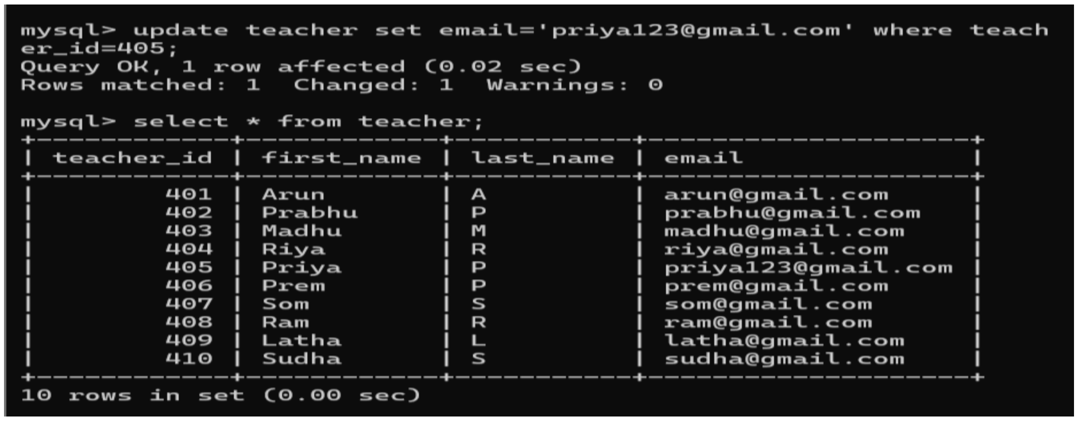
```
mysql> insert into Students(student_id,first_name,last_name,date
_of_birth,email,phone_number) values(11,'John','Doe','1995-08-15
','john.doe@example.com',1234567890);
Query OK, 1 row affected (0.01 sec)

mysql> select * from students;
+------------+------------+-----------+---------------+----------------------+--------------+
| student_id | first_name | last_name | date_of_birth | email                | phone_number |
+------------+------------+-----------+---------------+----------------------+--------------+
|          1 | Roupesh    | R         | 2002-12-02    | roup@gmail.com       |   9992224441 |
|          2 | Sakthi     | S         | 2002-02-15    | sakthi@gmail.com     |   9977658432 |
|          3 | Seema      | A         | 2001-05-25    | seema@gmail.com      |   9976546703 |
|          4 | Solamon    | S         | 2002-04-09    | solamon@gmail.com    |   6434679921 |
|          5 | Sonali     | T         | 2002-02-04    | sonali@gmail.com     |   7865355432 |
|          6 | Soundarya  | V         | 2002-10-22    | sound@gmail.com      |   9124567890 |
|          7 | Sreeja     | P         | 2003-07-27    | sree@gmail.com       |   9876543210 |
|          8 | Sujith     | R         | 2002-11-17    | sujith@gmail.com     |   9977636432 |
|          9 | Swatha     | M         | 2002-05-27    | swatha@gmail.com     |   9878658130 |
|         10 | Vignesh    | N         | 2002-09-05    | vignesh@gmail.com    |   9977658432 |
|         11 | John       | Doe       | 1995-08-15    | john.doe@example.com |   1234567890 |
+------------+------------+-----------+---------------+----------------------+--------------+
11 rows in set (0.00 sec)
```

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.



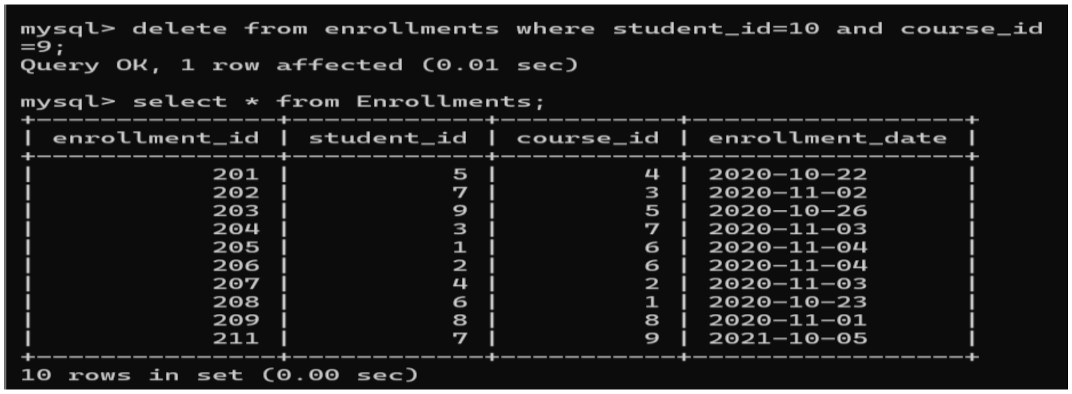
```
mysql> insert into Enrollments values(211,7,9,'2021-10-05');
Query OK, 1 row affected (0.01 sec)

mysql> select * from Enrollments;
+---------------+------------+-----------+-----------------+
| enrollment_id | student_id | course_id | enrollment_date |
+---------------+------------+-----------+-----------------+
|           201 |          5 |         4 | 2020-10-22      |
|           202 |          7 |         3 | 2020-11-02      |
|           203 |          9 |         5 | 2020-10-26      |
|           204 |          3 |         7 | 2020-11-03      |
|           205 |          1 |         6 | 2020-11-04      |
|           206 |          2 |         6 | 2020-11-04      |
|           207 |          4 |         2 | 2020-11-03      |
|           208 |          6 |         1 | 2020-10-23      |
|           209 |          8 |         8 | 2020-11-01      |
|           210 |         10 |         9 | 2020-11-02      |
|           211 |          7 |         9 | 2021-10-05      |
+---------------+------------+-----------+-----------------+
11 rows in set (0.00 sec)
```

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



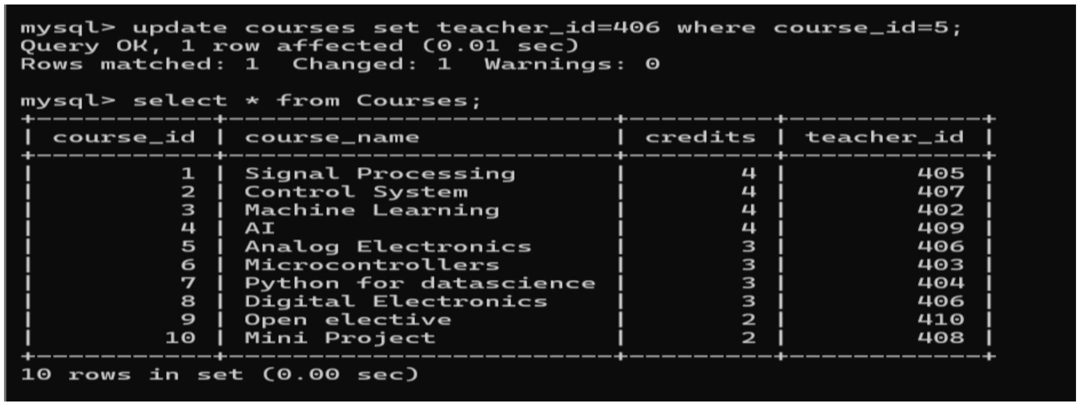
```
mysql> update teacher set email='priya123@gmail.com' where teach
er_id=405;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from teacher;
+------------+------------+-----------+---------------------+
| teacher_id | first_name | last_name | email               |
+------------+------------+-----------+---------------------+
|        401 | Arun       | A         | arun@gmail.com      |
|        402 | Prabhu     | P         | prabhu@gmail.com    |
|        403 | Madhu      | M         | madhu@gmail.com     |
|        404 | Riya       | R         | riya@gmail.com      |
|        405 | Priya      | P         | priya123@gmail.com  |
|        406 | Prem       | P         | prem@gmail.com      |
|        407 | Som        | S         | som@gmail.com       |
|        408 | Ram        | R         | ram@gmail.com       |
|        409 | Latha      | L         | latha@gmail.com     |
|        410 | Sudha      | S         | sudha@gmail.com     |
+------------+------------+-----------+---------------------+
10 rows in set (0.00 sec)
```

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.



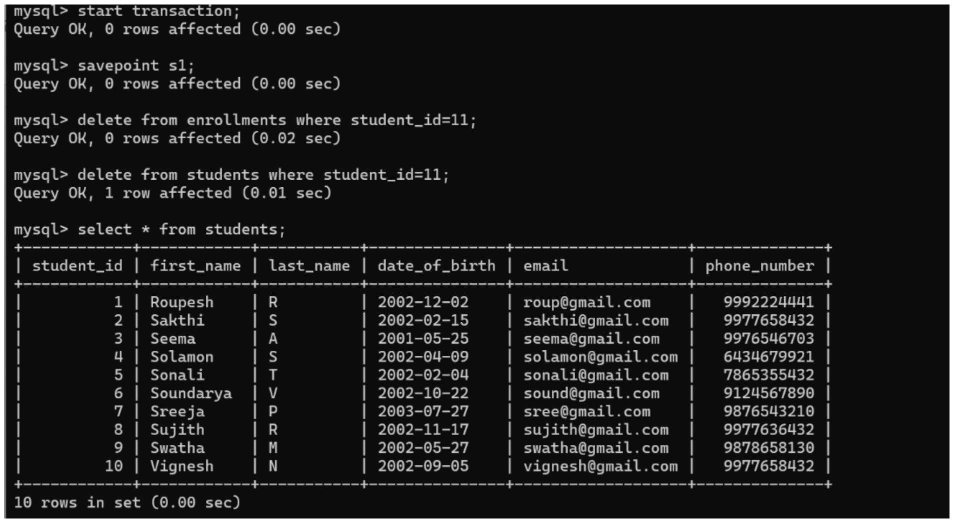
```
mysql> delete from enrollments where student_id=10 and course_id
=9;
Query OK, 1 row affected (0.01 sec)

mysql> select * from Enrollments;
+---------------+------------+-----------+-----------------+
| enrollment_id | student_id | course_id | enrollment_date |
+---------------+------------+-----------+-----------------+
|           201 |          5 |         4 | 2020-10-22      |
|           202 |          7 |         3 | 2020-11-02      |
|           203 |          9 |         5 | 2020-10-26      |
|           204 |          3 |         7 | 2020-11-03      |
|           205 |          1 |         6 | 2020-11-04      |
|           206 |          2 |         6 | 2020-11-04      |
|           207 |          4 |         2 | 2020-11-03      |
|           208 |          6 |         1 | 2020-10-23      |
|           209 |          8 |         8 | 2020-11-01      |
|           211 |          7 |         9 | 2021-10-05      |
+---------------+------------+-----------+-----------------+
10 rows in set (0.00 sec)
```

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



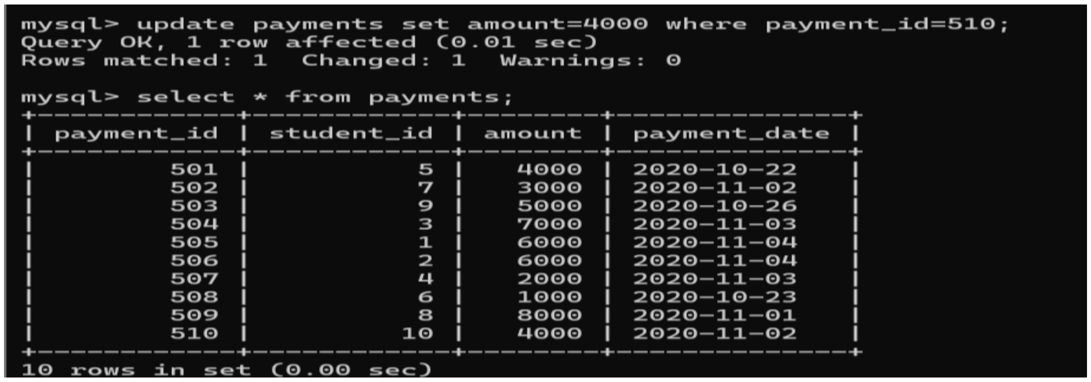
```
mysql> update courses set teacher_id=406 where course_id=5;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Courses;
+-----------+---------------------------+---------+------------+
| course_id | course_name               | credits | teacher_id |
+-----------+---------------------------+---------+------------+
|         1 | Signal Processing         |       4 |        405 |
|         2 | Control System            |       4 |        407 |
|         3 | Machine Learning          |       4 |        402 |
|         4 | AI                        |       4 |        409 |
|         5 | Analog Electronics        |       3 |        406 |
|         6 | Microcontrollers          |       3 |        403 |
|         7 | Python for datascience    |       3 |        404 |
|         8 | Digital Electronics       |       3 |        406 |
|         9 | Open elective             |       2 |        410 |
|        10 | Mini Project              |       2 |        408 |
+-----------+---------------------------+---------+------------+
10 rows in set (0.00 sec)
```

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.



```
mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> savepoint s1;
Query OK, 0 rows affected (0.00 sec)

mysql> delete from enrollments where student_id=11;
Query OK, 0 rows affected (0.02 sec)

mysql> delete from students where student_id=11;
Query OK, 1 row affected (0.01 sec)

mysql> select * from students;
+------------+------------+-----------+---------------+-------------------+--------------+
| student_id | first_name | last_name | date_of_birth | email             | phone_number |
+------------+------------+-----------+---------------+-------------------+--------------+
|          1 | Roupesh    | R         | 2002-12-02    | roup@gmail.com    |   9992224441 |
|          2 | Sakthi     | S         | 2002-02-15    | sakthi@gmail.com  |   9977658432 |
|          3 | Seema      | A         | 2001-05-25    | seema@gmail.com   |   9976546703 |
|          4 | Solamon    | S         | 2002-04-09    | solamon@gmail.com |   6434679921 |
|          5 | Sonali     | T         | 2002-02-04    | sonali@gmail.com  |   7865355432 |
|          6 | Soundarya  | V         | 2002-10-22    | sound@gmail.com   |   9124567890 |
|          7 | Sreeja     | P         | 2003-07-27    | sree@gmail.com    |   9876543210 |
|          8 | Sujith     | R         | 2002-11-17    | sujith@gmail.com  |   9977636432 |
|          9 | Swatha     | M         | 2002-05-27    | swatha@gmail.com  |   9878658130 |
|         10 | Vignesh    | N         | 2002-09-05    | vignesh@gmail.com |   9977658432 |
+------------+------------+-----------+---------------+-------------------+--------------+
10 rows in set (0.00 sec)
```

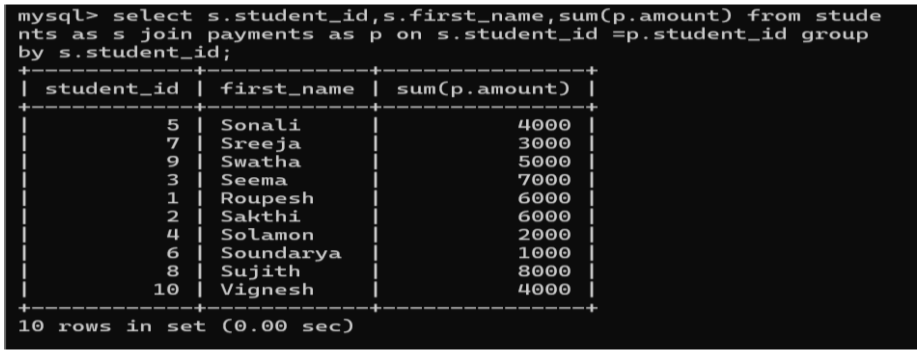
7. Update the payment amount for a specific payment record in the "Payments" table. Choose any payment record and modify the payment amount.



```
mysql> update payments set amount=4000 where payment_id=510;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from payments;
+------------+------------+--------+--------------+
| payment_id | student_id | amount | payment_date |
+------------+------------+--------+--------------+
|        501 |          5 |   4000 | 2020-10-22   |
|        502 |          7 |   3000 | 2020-11-02   |
|        503 |          9 |   5000 | 2020-10-26   |
|        504 |          3 |   7000 | 2020-11-03   |
|        505 |          1 |   6000 | 2020-11-04   |
|        506 |          2 |   6000 | 2020-11-04   |
|        507 |          4 |   2000 | 2020-11-03   |
|        508 |          6 |   1000 | 2020-10-23   |
|        509 |          8 |   8000 | 2020-11-01   |
|        510 |         10 |   4000 | 2020-11-02   |
+------------+------------+--------+--------------+
10 rows in set (0.00 sec)
```

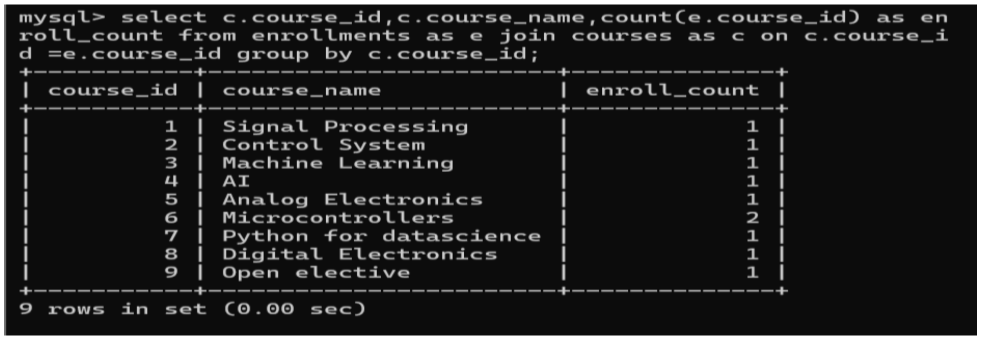
TASK 3 . AGGREGATE FUNCTIONS, HAVING, ORDER BY, GROUP BY AND JOINS:

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.



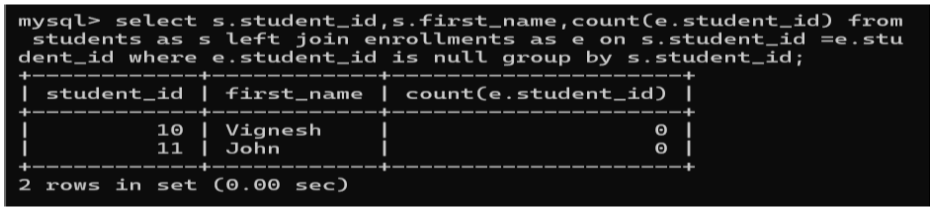
```
mysql> select s.student_id,s.first_name,sum(p.amount) from stude
nts as s join payments as p on s.student_id =p.student_id group
by s.student_id;
+------------+------------+---------------+
| student_id | first_name | sum(p.amount) |
+------------+------------+---------------+
|          5 | Sonali     |          4000 |
|          7 | Sreeja     |          3000 |
|          9 | Swatha     |          5000 |
|          3 | Seema      |          7000 |
|          1 | Roupesh    |          6000 |
|          2 | Sakthi     |          6000 |
|          4 | Solamon    |          2000 |
|          6 | Soundarya  |          1000 |
|          8 | Sujith     |          8000 |
|         10 | Vignesh    |          4000 |
+------------+------------+---------------+
10 rows in set (0.00 sec)
```

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.



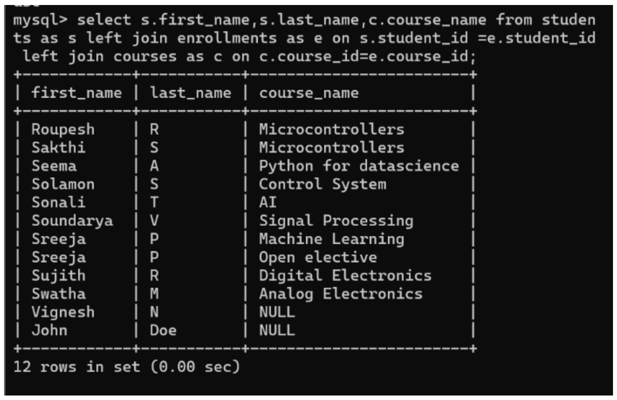
```
mysql> select c.course_id,c.course_name,count(e.course_id) as en
roll_count from enrollments as e join courses as c on c.course_i
d =e.course_id group by c.course_id;
+-----------+-------------------------+--------------+
| course_id | course_name             | enroll_count |
+-----------+-------------------------+--------------+
|         1 | Signal Processing       |            1 |
|         2 | Control System          |            1 |
|         3 | Machine Learning        |            1 |
|         4 | AI                      |            1 |
|         5 | Analog Electronics      |            1 |
|         6 | Microcontrollers        |            2 |
|         7 | Python for datascience  |            1 |
|         8 | Digital Electronics     |            1 |
|         9 | Open elective           |            1 |
+-----------+-------------------------+--------------+
9 rows in set (0.00 sec)
```

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.



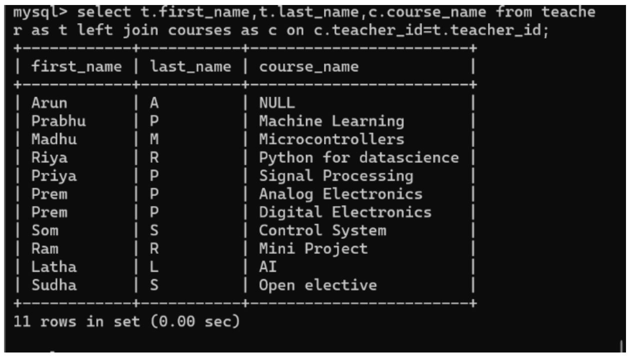
```
mysql> select s.student_id,s.first_name,count(e.student_id) from
 students as s left join enrollments as e on s.student_id =e.stu
dent_id where e.student_id is null group by s.student_id;
+------------+------------+---------------------+
| student_id | first_name | count(e.student_id) |
+------------+------------+---------------------+
|         10 | Vignesh    |                   0 |
|         11 | John       |                   0 |
+------------+------------+---------------------+
2 rows in set (0.00 sec)
```

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.



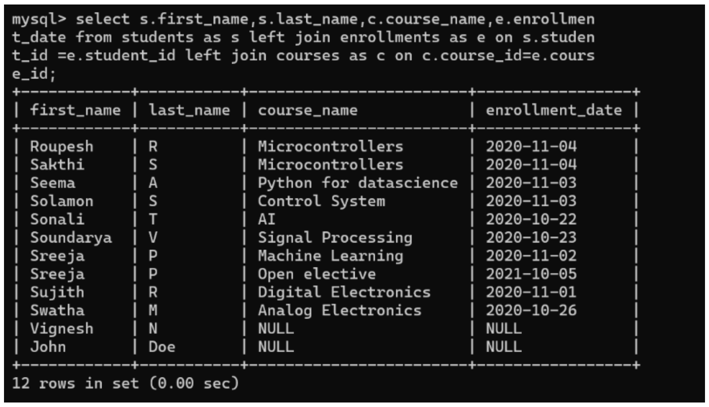
```
mysql> select s.first_name,s.last_name,c.course_name from studen
ts as s left join enrollments as e on s.student_id =e.student_id
 left join courses as c on c.course_id=e.course_id;
+------------+-----------+------------------------+
| first_name | last_name | course_name            |
+------------+-----------+------------------------+
| Roupesh    | R         | Microcontrollers       |
| Sakthi     | S         | Microcontrollers       |
| Seema      | A         | Python for datascience |
| Solamon    | S         | Control System         |
| Sonali     | T         | AI                     |
| Soundarya  | V         | Signal Processing      |
| Sreeja     | P         | Machine Learning       |
| Sreeja     | P         | Open elective          |
| Sujith     | R         | Digital Electronics    |
| Swatha     | M         | Analog Electronics     |
| Vignesh    | N         | NULL                   |
| John       | Doe       | NULL                   |
+------------+-----------+------------------------+
12 rows in set (0.00 sec)
```

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.



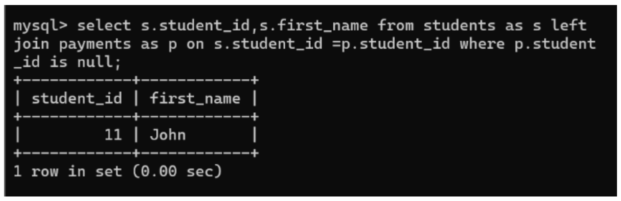
```
mysql> select t.first_name,t.last_name,c.course_name from teache
r as t left join courses as c on c.teacher_id=t.teacher_id;
+------------+-----------+------------------------+
| first_name | last_name | course_name            |
+------------+-----------+------------------------+
| Arun       | A         | NULL                   |
| Prabhu     | P         | Machine Learning       |
| Madhu      | M         | Microcontrollers       |
| Riya       | R         | Python for datascience |
| Priya      | P         | Signal Processing      |
| Prem       | P         | Analog Electronics     |
| Prem       | P         | Digital Electronics    |
| Som        | S         | Control System         |
| Ram        | R         | Mini Project           |
| Latha      | L         | AI                     |
| Sudha      | S         | Open elective          |
+------------+-----------+------------------------+
11 rows in set (0.00 sec)
```

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.



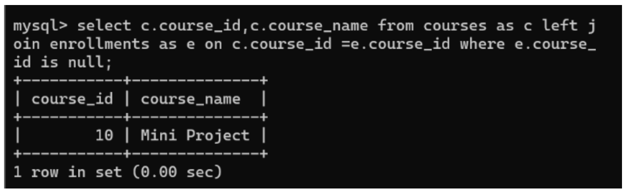
```
mysql> select s.first_name,s.last_name,c.course_name,e.enrollmen
t_date from students as s left join enrollments as e on s.studen
t_id =e.student_id left join courses as c on c.course_id=e.cours
e_id;
+------------+-----------+------------------------+-----------------+
| first_name | last_name | course_name            | enrollment_date |
+------------+-----------+------------------------+-----------------+
| Roupesh    | R         | Microcontrollers       | 2020-11-04      |
| Sakthi     | S         | Microcontrollers       | 2020-11-04      |
| Seema      | A         | Python for datascience | 2020-11-03      |
| Solamon    | S         | Control System         | 2020-11-03      |
| Sonali     | T         | AI                     | 2020-10-22      |
| Soundarya  | V         | Signal Processing      | 2020-10-23      |
| Sreeja     | P         | Machine Learning       | 2020-11-02      |
| Sreeja     | P         | Open elective          | 2021-10-05      |
| Sujith     | R         | Digital Electronics    | 2020-11-01      |
| Swatha     | M         | Analog Electronics     | 2020-10-26      |
| Vignesh    | N         | NULL                   | NULL            |
| John       | Doe       | NULL                   | NULL            |
+------------+-----------+------------------------+-----------------+
12 rows in set (0.00 sec)
```

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.



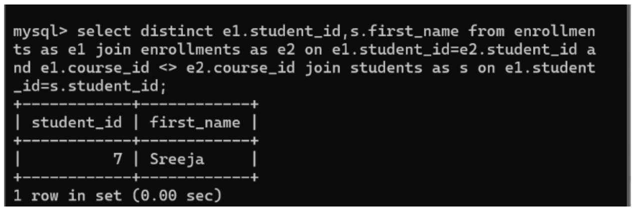
```
mysql> select s.student_id,s.first_name from students as s left
join payments as p on s.student_id =p.student_id where p.student
_id is null;
+------------+------------+
| student_id | first_name |
+------------+------------+
|         11 | John       |
+------------+------------+
1 row in set (0.00 sec)
```

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.



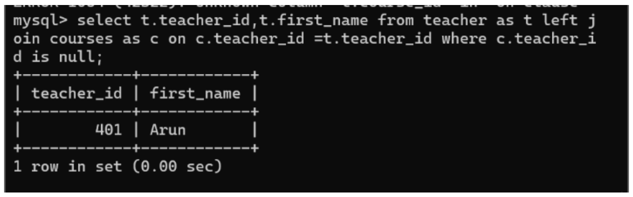
```
mysql> select c.course_id,c.course_name from courses as c left j
oin enrollments as e on c.course_id =e.course_id where e.course_
id is null;
+-----------+--------------+
| course_id | course_name  |
+-----------+--------------+
|        10 | Mini Project |
+-----------+--------------+
1 row in set (0.00 sec)
```

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.



```
mysql> select distinct e1.student_id,s.first_name from enrollmen
ts as e1 join enrollments as e2 on e1.student_id=e2.student_id a
nd e1.course_id <> e2.course_id join students as s on e1.student
_id=s.student_id;
+------------+------------+
| student_id | first_name |
+------------+------------+
|          7 | Sreeja     |
+------------+------------+
1 row in set (0.00 sec)
```

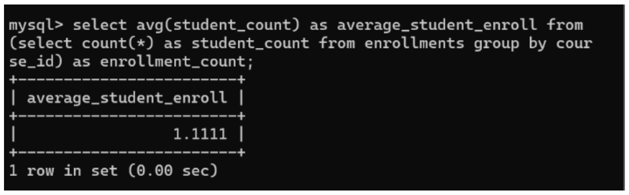
10. Find teachers who are not assigned to any courses. Use a LEFT JOIN between the "Teacher" table and the "Courses" table and filter for teachers with NULL course assignments.



```
mysql> select t.teacher_id,t.first_name from teacher as t left j
oin courses as c on c.teacher_id =t.teacher_id where c.teacher_i
d is null;
+------------+------------+
| teacher_id | first_name |
+------------+------------+
|        401 | Arun       |
+------------+------------+
1 row in set (0.00 sec)
```

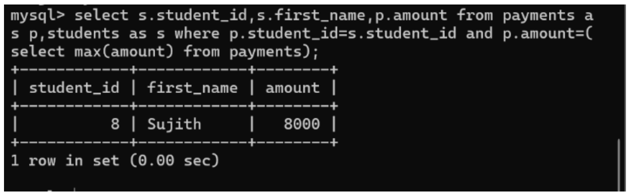
TASK 4. SUBQUERY AND ITS TYPE:

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



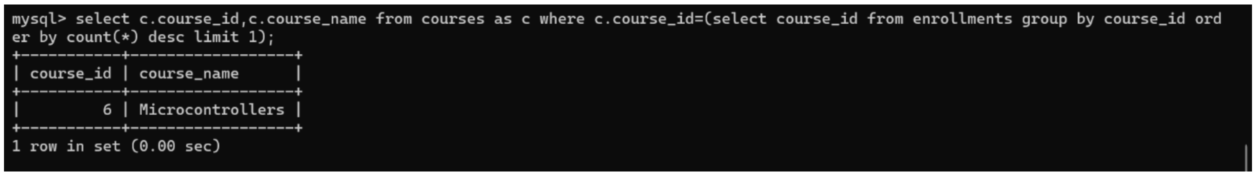
```
mysql> select avg(student_count) as average_student_enroll from
(select count(*) as student_count from enrollments group by cour
se_id) as enrollment_count;
+------------------------+
| average_student_enroll |
+------------------------+
|                 1.1111 |
+------------------------+
1 row in set (0.00 sec)
```

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.



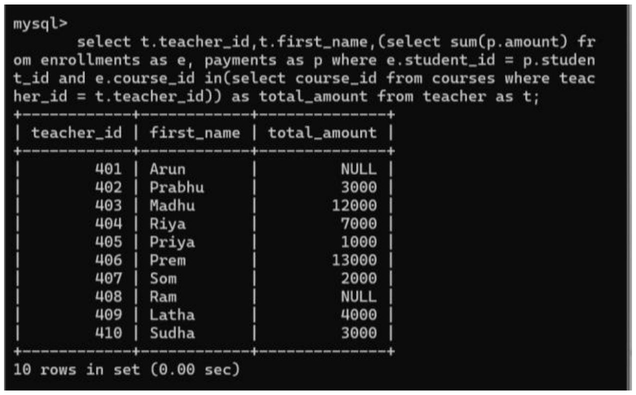
```
mysql> select s.student_id,s.first_name,p.amount from payments a
s p,students as s where p.student_id=s.student_id and p.amount=(
select max(amount) from payments);
+------------+------------+--------+
| student_id | first_name | amount |
+------------+------------+--------+
|          8 | Sujith     |   8000 |
+------------+------------+--------+
1 row in set (0.00 sec)
```

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



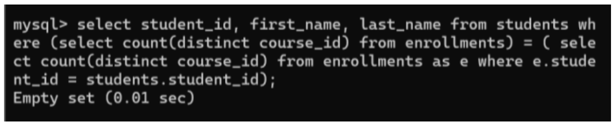
```
mysql> select c.course_id,c.course_name from courses as c where c.course_id=(select course_id from enrollments group by course_id ord
er by count(*) desc limit 1);
+-----------+------------------+
| course_id | course_name      |
+-----------+------------------+
|         6 | Microcontrollers |
+-----------+------------------+
1 row in set (0.00 sec)
```

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



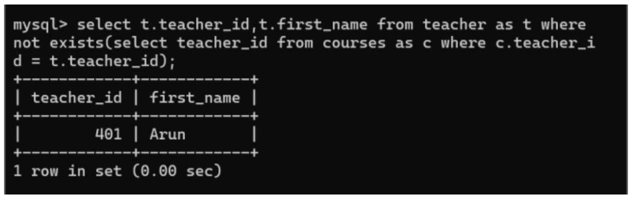
```
mysql>
       select t.teacher_id,t.first_name,(select sum(p.amount) fr
om enrollments as e, payments as p where e.student_id = p.studen
t_id and e.course_id in(select course_id from courses where teac
her_id = t.teacher_id)) as total_amount from teacher as t;
+------------+------------+--------------+
| teacher_id | first_name | total_amount |
+------------+------------+--------------+
|        401 | Arun       |         NULL |
|        402 | Prabhu     |         3000 |
|        403 | Madhu      |        12000 |
|        404 | Riya       |         7000 |
|        405 | Priya      |         1000 |
|        406 | Prem       |        13000 |
|        407 | Som        |         2000 |
|        408 | Ram        |         NULL |
|        409 | Latha      |         4000 |
|        410 | Sudha      |         3000 |
+------------+------------+--------------+
10 rows in set (0.00 sec)
```

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



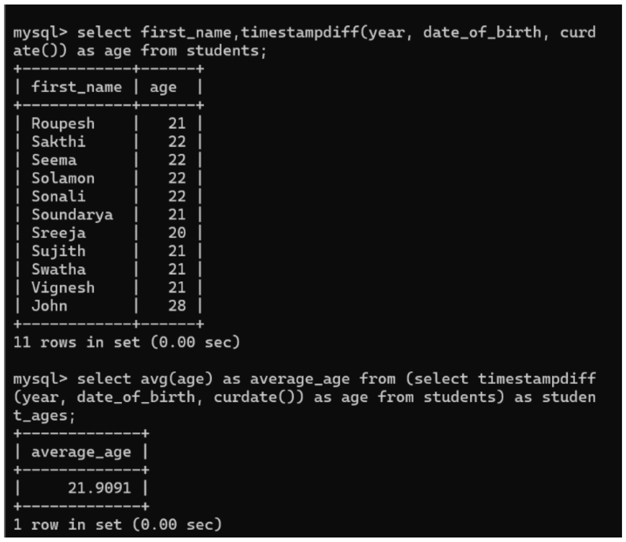
```
mysql> select student_id, first_name, last_name from students wh
ere (select count(distinct course_id) from enrollments) = ( sele
ct count(distinct course_id) from enrollments as e where e.stude
nt_id = students.student_id);
Empty set (0.01 sec)
```

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



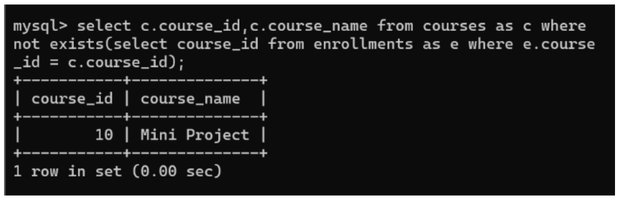
```
mysql> select t.teacher_id,t.first_name from teacher as t where
not exists(select teacher_id from courses as c where c.teacher_i
d = t.teacher_id);
+------------+------------+
| teacher_id | first_name |
+------------+------------+
|        401 | Arun       |
+------------+------------+
1 row in set (0.00 sec)
```

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



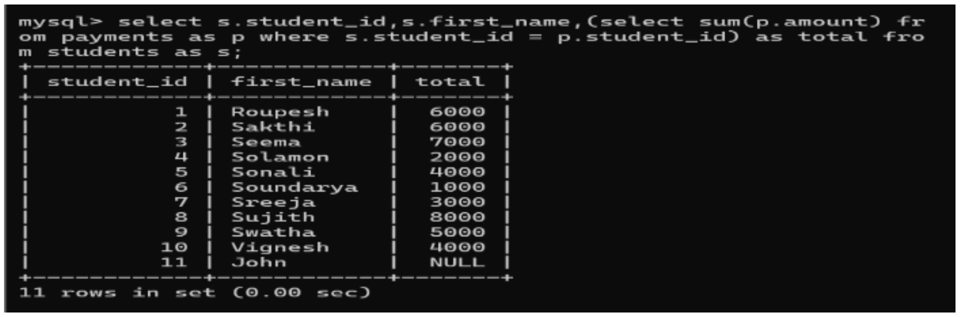
```
mysql> select first_name,timestampdiff(year, date_of_birth, curd
ate()) as age from students;
+------------+------+
| first_name | age  |
+------------+------+
| Roupesh    |   21 |
| Sakthi     |   22 |
| Seema      |   22 |
| Solamon    |   22 |
| Sonali     |   22 |
| Soundarya  |   21 |
| Sreeja     |   20 |
| Sujith     |   21 |
| Swatha     |   21 |
| Vignesh    |   21 |
| John       |   28 |
+------------+------+
11 rows in set (0.00 sec)

mysql> select avg(age) as average_age from (select timestampdiff
(year, date_of_birth, curdate()) as age from students) as studen
t_ages;
+-------------+
| average_age |
+-------------+
|     21.9091 |
+-------------+
1 row in set (0.00 sec)
```

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



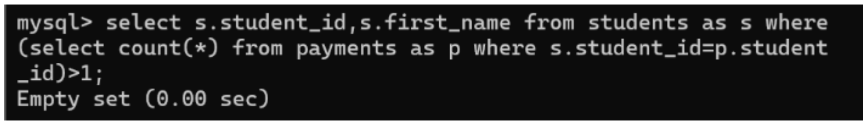
```
mysql> select c.course_id,c.course_name from courses as c where
not exists(select course_id from enrollments as e where e.course
_id = c.course_id);
+-----------+--------------+
| course_id | course_name  |
+-----------+--------------+
|        10 | Mini Project |
+-----------+--------------+
1 row in set (0.00 sec)
```

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



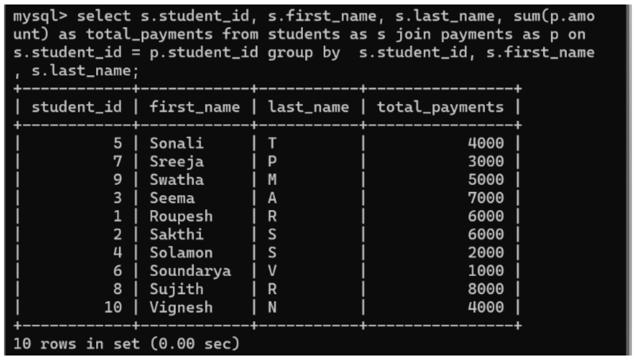
```
mysql> select s.student_id,s.first_name,(select sum(p.amount) fr
om payments as p where s.student_id = p.student_id) as total fro
m students as s;
+------------+------------+-------+
| student_id | first_name | total |
+------------+------------+-------+
|          1 | Roupesh    |  6000 |
|          2 | Sakthi     |  6000 |
|          3 | Seema      |  7000 |
|          4 | Solamon    |  2000 |
|          5 | Sonali     |  4000 |
|          6 | Soundarya  |  1000 |
|          7 | Sreeja     |  3000 |
|          8 | Sujith     |  8000 |
|          9 | Swatha     |  5000 |
|         10 | Vignesh    |  4000 |
|         11 | John       |  NULL |
+------------+------------+-------+
11 rows in set (0.00 sec)
```

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.



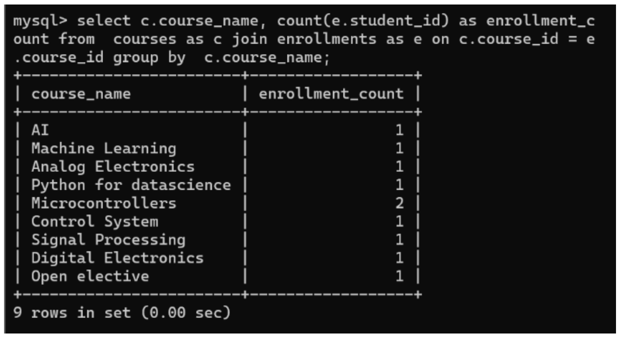
```
mysql> select s.student_id,s.first_name from students as s where
(select count(*) from payments as p where s.student_id=p.student
_id)>1;
Empty set (0.00 sec)
```

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.



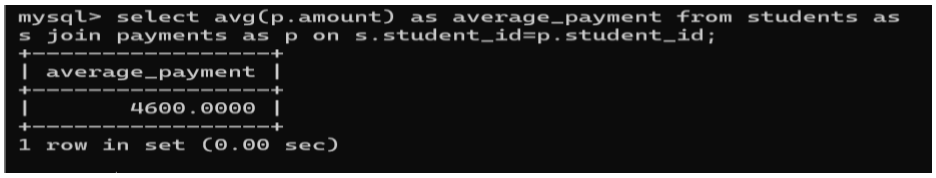
```
mysql> select s.student_id, s.first_name, s.last_name, sum(p.amo
unt) as total_payments from students as s join payments as p on
s.student_id = p.student_id group by  s.student_id, s.first_name
, s.last_name;
+------------+------------+-----------+----------------+
| student_id | first_name | last_name | total_payments |
+------------+------------+-----------+----------------+
|          5 | Sonali     | T         |           4000 |
|          7 | Sreeja     | P         |           3000 |
|          9 | Swatha     | M         |           5000 |
|          3 | Seema      | A         |           7000 |
|          1 | Roupesh    | R         |           6000 |
|          2 | Sakthi     | S         |           6000 |
|          4 | Solamon    | S         |           2000 |
|          6 | Soundarya  | V         |           1000 |
|          8 | Sujith     | R         |           8000 |
|         10 | Vignesh    | N         |           4000 |
+------------+------------+-----------+----------------+
10 rows in set (0.00 sec)
```

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.



```
mysql> select c.course_name, count(e.student_id) as enrollment_c
ount from  courses as c join enrollments as e on c.course_id = e
.course_id group by  c.course_name;
+------------------------+------------------+
| course_name            | enrollment_count |
+------------------------+------------------+
| AI                     |                1 |
| Machine Learning       |                1 |
| Analog Electronics     |                1 |
| Python for datascience |                1 |
| Microcontrollers       |                2 |
| Control System         |                1 |
| Signal Processing      |                1 |
| Digital Electronics    |                1 |
| Open elective          |                1 |
+------------------------+------------------+
9 rows in set (0.00 sec)
```

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.



```
mysql> select avg(p.amount) as average_payment from students as
s join payments as p on s.student_id=p.student_id;
+-----------------+
| average_payment |
+-----------------+
|       4600.0000 |
+-----------------+
1 row in set (0.00 sec)
```