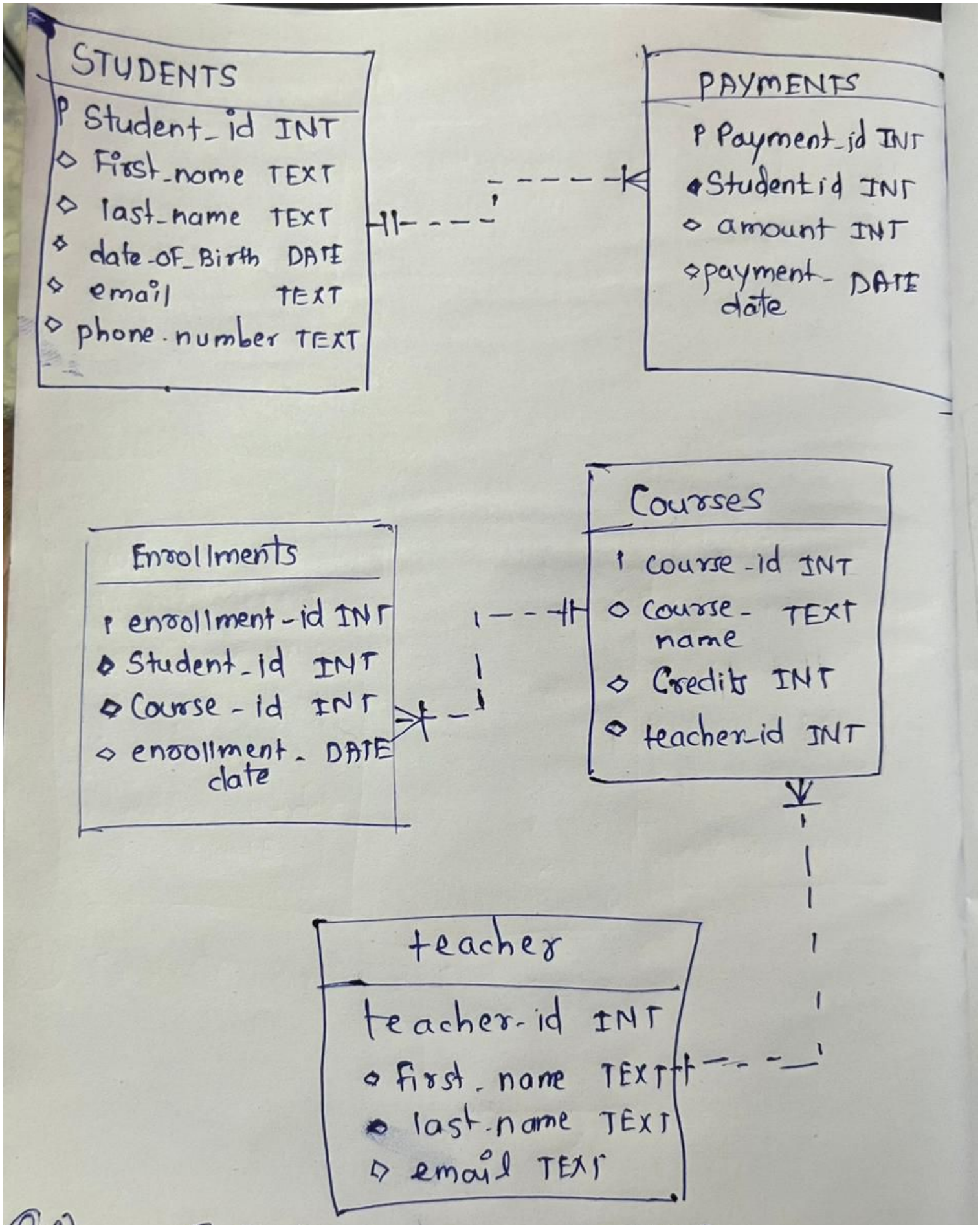
```
mysql> /* ASSIGNMENT II
   /*> TASK I:DATABASE DESIGN
   /*> 1.CREATE THE DATABASE NAMED "SSIDB"; */
mysql> CREATE DATABASE IF NOT EXISTS SSIDB;
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> use SSIDB;
Database changed
mysql> /* 2.2. Define the schema for the Students, Courses, Enrollments, Teacher, and Payments tables based on the provided schema. Write SQL scripts to create the ment
ioned tables with appropriate data types, constraints, and relationships.a. Students  b. Courses c. Enrollments d. Teacher e. Payments */
mysql> create table Students(student_id int,first_name text,last_name text,date_of_birth date,email text,phone_number text,primary key(student_id));
Query OK, 0 rows affected (0.16 sec)

mysql> desc Students;
+---------------+------+------+-----+---------+-------+
| Field         | Type | Null | Key | Default | Extra |
+---------------+------+------+-----+---------+-------+
| student_id    | int  | 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  | text | YES  |     | NULL    |       |
+---------------+------+------+-----+---------+-------+
6 rows in set (0.03 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
Query OK, 0 rows affected (0.04 sec)

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

mysql> /* Courses */
mysql> create table courses(course_id int,course_name text,credits int,teacher_id int,primary key(course_id), foreign key(teacher_id) references teacher(teacher_id));
Query OK, 0 rows affected (0.04 sec)

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

mysql> /*Enrollments*/
mysql> create table enrollments(enrollment_id int,student_id int,course_id int,enrollemnt_date date,primary key(enrollment_id),foreign key(student_id) references Studen
ts(student_id),foreign key(course_id) references courses(course_id));desc enrollments;
Query OK, 0 rows affected (0.06 sec)

+-----------------+------+------+-----+---------+-------+
| Field           | Type | Null | Key | Default | Extra |
+-----------------+------+------+-----+---------+-------+
| enrollment_id   | int  | NO   | PRI | NULL    |       |
| student_id      | int  | YES  | MUL | NULL    |       |
| course_id       | int  | YES  | MUL | NULL    |       |
| enrollemnt_date | date | YES  |     | NULL    |       |
+-----------------+------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql>
```



Q.3) E.R Diagram SSIDB

MySQL 8.0 Command Line Client

```
mysql> /*Payments */
mysql> create table Payments(payment_id int primary key,student_id int, amount int,payment_date date,foreign key(student_id) references Students(student_id));desc stude
nts;
Query OK, 0 rows affected (0.03 sec)

+--------------+------+------+-----+---------+-------+
| Field        | Type | Null | Key | Default | Extra |
+--------------+------+------+-----+---------+-------+
| student_id   | int  | 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 | text | YES  |     | NULL    |       |
+--------------+------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> /* 5 Insert at aleast 10 Sample records into each of the following table*/
mysql> /*Students*/
mysql> INSERT INTO students (student_id, first_name, last_name, date_of_birth, email, phone_number)
    -> VALUES(1, 'Aarav', 'Kumar', '1995-08-12', 'aarav.kumar@gmail.com', '9876543210'),(2, 'Ananya', 'Sharma', '1998-05-25', 'ananya.sharma@gmail.com', '8765432109'),(
3, 'Rahul', 'Patel', '1992-11-08', 'rahul.patel@gmail.com', '7654321098'),(4, 'Aisha', 'Singh', '1997-03-15', 'aisha.singh@gmail.com', '6543210987'),(5, 'Arjun', 'Verma
', '1990-12-30', 'arjun.verma@gmail.com', '5432109876'),(6, 'Sanya', 'Mishra', '1994-07-18', 'sanya.mishra@gmail.com', '4321098765'),(7, 'Vikram', 'Gupta', '1999-02-03'
, 'vikram.gupta@gmail.com', '3210987654'),(8, 'Jiya', 'Yadav', '1996-06-22', 'jiya.yadav@gmail.com', '2109876543'),(9, 'Kabir', 'Shah', '1993-09-10', 'kabir.shah@gmail.
com', '1098765432'),(10, 'Neha', 'Chopra', '2002-04-05', 'neha.chopra@gmail.com', '9876543210');
Query OK, 10 rows affected (0.35 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> Select * from Students;
+------------+------------+-----------+---------------+-------------------------+--------------+
| student_id | first_name | last_name | date_of_birth | email                   | phone_number |
+------------+------------+-----------+---------------+-------------------------+--------------+
|          1 | Aarav      | Kumar     | 1995-08-12    | aarav.kumar@gmail.com   | 9876543210   |
|          2 | Ananya     | Sharma    | 1998-05-25    | ananya.sharma@gmail.com | 8765432109   |
|          3 | Rahul      | Patel     | 1992-11-08    | rahul.patel@gmail.com   | 7654321098   |
|          4 | Aisha      | Singh     | 1997-03-15    | aisha.singh@gmail.com   | 6543210987   |
|          5 | Arjun      | Verma     | 1990-12-30    | arjun.verma@gmail.com   | 5432109876   |
|          6 | Sanya      | Mishra    | 1994-07-18    | sanya.mishra@gmail.com  | 4321098765   |
|          7 | Vikram     | Gupta     | 1999-02-03    | vikram.gupta@gmail.com  | 3210987654   |
|          8 | Jiya       | Yadav     | 1996-06-22    | jiya.yadav@gmail.com    | 2109876543   |
|          9 | Kabir      | Shah      | 1993-09-10    | kabir.shah@gmail.com    | 1098765432   |
|         10 | Neha       | Chopra    | 2002-04-05    | neha.chopra@gmail.com   | 9876543210   |
+------------+------------+-----------+---------------+-------------------------+--------------+
10 rows in set (0.03 sec)
```

MySQL 8.0 Command Line Client

```
mysql> /*Teacher*/
mysql> INSERT INTO teacher (teacher_id, first_name, last_name, email)VALUES (301, 'Priya', 'Sharma', 'priya.sharma@gmail.com'),(302, 'Rahul', 'Verma', 'rahul.verma@gmai
l.com'),(303, 'Neha', 'Singh', 'neha.singh@gmail.com'),(304, 'Rajesh', 'Patil', 'rajesh.patil@gmail.com'),(305, 'Sneha', 'Gupta', 'sneha.gupta@gmail.com'),(306, 'Vikram
', 'Kumar', 'vikram.kumar@gmai.com'),(307, 'Anita', 'Chopra','anita.chopra@gmail.com'),(308,    'Amit','Mishra','amit.mishra@gmail.com'),(309, 'Shalini', 'Malhotra', 's
halini.malhotra@gmail.com'),(310, 'Ravi', 'Joshi', 'ravi.joshi@gmail.com');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM teacher;
+------------+------------+-----------+---------------------------+
| teacher_id | first_name | last_name | email                     |
+------------+------------+-----------+---------------------------+
|        301 | Priya      | Sharma    | priya.sharma@gmail.com    |
|        302 | Rahul      | Verma     | rahul.verma@gmail.com     |
|        303 | Neha       | Singh     | neha.singh@gmail.com      |
|        304 | Rajesh     | Patil     | rajesh.patil@gmail.com    |
|        305 | Sneha      | Gupta     | sneha.gupta@gmail.com     |
|        306 | Vikram     | Kumar     | vikram.kumar@gmai.com     |
|        307 | Anita      | Chopra    | anita.chopra@gmail.com    |
|        308 | Amit       | Mishra    | amit.mishra@gmail.com     |
|        309 | Shalini    | Malhotra  | shalini.malhotra@gmail.com|
|        310 | Ravi       | Joshi     | ravi.joshi@gmail.com      |
+------------+------------+-----------+---------------------------+
10 rows in set (0.00 sec)

mysql> /*Courses*/
mysql> INSERT INTO courses (course_id, course_name, credits, teacher_id)VALUES(201, 'Introduction to Computer Science', 3, 301),(202, 'Data Structures and Algorithms',
4, 302),(203, 'Database Management Systems', 3, 303),(204, 'Computer Networks', 4, 304),(205, 'Software Engineering', 3, 305),(206, 'Artificial Intelligence', 4, 306),(
207, 'Web Development', 3, 307),(208, 'Cybersecurity Fundamentals', 4, 308),(209, 'Operating Systems', 3, 309),(210, 'Machine Learning', 4, 310);
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 |
+-----------+---------------------------------+---------+------------+
|       201 | Introduction to Computer Science |       3 |        301 |
|       202 | Data Structures and Algorithms   |       4 |        302 |
|       203 | Database Management Systems      |       3 |        303 |
|       204 | Computer Networks                |       4 |        304 |
|       205 | Software Engineering             |       3 |        305 |
|       206 | Artificial Intelligence          |       4 |        306 |
|       207 | Web Development                  |       3 |        307 |
|       208 | Cybersecurity Fundamentals       |       4 |        308 |
```

MySQL 8.0 Command Line Client

```
mysql> /*Enrollments*/
mysql> INSERT INTO enrollments (enrollment_id, student_id, course_id, enrollemnt_date)
    -> VALUES(101,1,201,'2023-01-15'),(102,2,202,'2023-02-20'),(103,3,203,'2023-03-10'),(104,4,204,'2023-04-25'),(105,5,205,'2023-05-18'),(106, 6,206, '2023-06-30'),(10
7, 7,207, '2023-07-05'),(108, 8,208, '2023-08-12'),(109, 9,209, '2023-09-08'),(110, 10,210, '2023-10-22');
Query OK, 10 rows affected (0.00 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM enrollments;
+---------------+------------+-----------+-----------------+
| enrollment_id | student_id | course_id | enrollemnt_date |
+---------------+------------+-----------+-----------------+
|           101 |          1 |       201 | 2023-01-15      |
|           102 |          2 |       202 | 2023-02-20      |
|           103 |          3 |       203 | 2023-03-10      |
|           104 |          4 |       204 | 2023-04-25      |
|           105 |          5 |       205 | 2023-05-18      |
|           106 |          6 |       206 | 2023-06-30      |
|           107 |          7 |       207 | 2023-07-05      |
|           108 |          8 |       208 | 2023-08-12      |
|           109 |          9 |       209 | 2023-09-08      |
|           110 |         10 |       210 | 2023-10-22      |
+---------------+------------+-----------+-----------------+
10 rows in set (0.00 sec)

mysql> /*Payments*/
mysql> INSERT INTO payments (payment_id, student_id, amount, payment_date) VALUES (401,1, 5000, '2023-01-15'),(402, 2, 6000, '2023-02-20'),(403,3, 4500, '2023-03-10'),(
404,4, 5500, '2023-04-25'),(405,5, 7000, '2023-05-18'),(406,6, 6500, '2023-06-30'),(407,7, 8000, '2023-07-05'),(408,8, 7500, '2023-08-12'),(409, 9, 4800, '2023-09-08'),
(410,10, 5200, '2023-10-22');
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 |
+------------+------------+--------+--------------+
|        401 |          1 |   5000 | 2023-01-15   |
|        402 |          2 |   6000 | 2023-02-20   |
|        403 |          3 |   4500 | 2023-03-10   |
|        404 |          4 |   5500 | 2023-04-25   |
|        405 |          5 |   7000 | 2023-05-18   |
|        406 |          6 |   6500 | 2023-06-30   |
|        407 |          7 |   8000 | 2023-07-05   |
|        408 |          8 |   7500 | 2023-08-12   |
```

MySQL 8.0 Command Line Client

```
mysql> /*TASK 2:SELECT ,Where,Between,And,Like
   /*> 1.1. 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*/
mysql> Insert into students values(11,'John','Doe','1985-08-15','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 | Aarav      | Kumar     | 1995-08-12    | aarav.kumar@gmail.com  | 9876543210   |
|          2 | Ananya     | Sharma    | 1998-05-25    | ananya.sharma@gmail.com | 8765432109   |
|          3 | Rahul      | Patel     | 1992-11-08    | rahul.patel@gmail.com  | 7654321098   |
|          4 | Aisha      | Singh     | 1997-03-15    | aisha.singh@gmail.com  | 6543210987   |
|          5 | Arjun      | Verma     | 1990-12-30    | arjun.verma@gmail.com  | 5432109876   |
|          6 | Sanya      | Mishra    | 1994-07-18    | sanya.mishra@gmail.com | 4321098765   |
|          7 | Vikram     | Gupta     | 1999-02-03    | vikram.gupta@gmail.com | 3210987654   |
|          8 | Jiya       | Yadav     | 1996-06-22    | jiya.yadav@gmail.com   | 2109876543   |
|          9 | Kabir      | Shah      | 1993-09-10    | kabir.shah@gmail.com   | 1098765432   |
|         10 | Neha       | Chopra    | 2002-04-05    | neha.chopra@gmail.com  | 9876543210   |
|         11 | John       | Doe       | 1985-08-15    | john.doe@example.com   | 1234567890   |
+------------+------------+-----------+---------------+------------------------+--------------+
11 rows in set (0.00 sec)

mysql> /* 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 enrollme
nt date.*/
mysql> INSERT INTO Enrollments (enrollment_id,student_id, course_id, enrollemnt_date)
    -> VALUES (411,11, 202, '2023-12-10');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM enrollments;
+---------------+------------+-----------+-----------------+
| enrollment_id | student_id | course_id | enrollemnt_date |
+---------------+------------+-----------+-----------------+
|           101 |          1 |       201 | 2023-01-15      |
|           102 |          2 |       202 | 2023-02-20      |
|           103 |          3 |       203 | 2023-03-10      |
|           104 |          4 |       204 | 2023-04-25      |
|           105 |          5 |       205 | 2023-05-18      |
|           106 |          6 |       206 | 2023-06-30      |
|           107 |          7 |       207 | 2023-07-05      |
|           108 |          8 |       208 | 2023-08-12      |
|           109 |          9 |       209 | 2023-09-08      |
```

MySQL 8.0 Command Line Client

```
|           105 |           5 |       205 | 2023-05-18      |
|           106 |           6 |       206 | 2023-06-30      |
|           107 |           7 |       207 | 2023-07-05      |
|           108 |           8 |       208 | 2023-08-12      |
|           109 |           9 |       209 | 2023-09-08      |
|           110 |          10 |       210 | 2023-10-22      |
|           411 |          11 |       202 | 2023-12-10      |
+---------------+-------------+-----------+-----------------+
11 rows in set (0.00 sec)

mysql> /*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='priya.sharma@yahoo.com' where teacher_id=301;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> Select * from teacher where teacher_id=301;
+------------+------------+-----------+------------------------+
| teacher_id | first_name | last_name | email                  |
+------------+------------+-----------+------------------------+
|        301 | Priya      | Sharma    | priya.sharma@yahoo.com |
+------------+------------+-----------+------------------------+
1 row in set (0.00 sec)

mysql> /* 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=11 and course_id =202;
Query OK, 1 row affected (0.01 sec)

mysql> /*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=309 where course_id=210;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> Select * from Courses where course_id=210;
+-----------+------------------+---------+------------+
| course_id | course_name      | credits | teacher_id |
+-----------+------------------+---------+------------+
|       210 | Machine Learning |       4 |        309 |
+-----------+------------------+---------+------------+
1 row in set (0.00 sec)

mysql> /*6.Delete a specific student from the "Students" table and remove all their enrollment records from the "Enrollments" table. Be sure to maintain referential int
egrity.*/
mysql> DELETE FROM Enrollments WHERE student_id =10;
```

MySQL 8.0 Command Line Client

```
mysql> /*7 .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=9000 where payment_id=410;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM payments where payment_id=410;
+------------+------------+--------+--------------+
| payment_id | student_id | amount | payment_date |
+------------+------------+--------+--------------+
|        410 |         10 |   9000 | 2023-10-22   |
+------------+------------+--------+--------------+
1 row in set (0.00 sec)

mysql> /*Task 3. Aggregate functions, Having, Order By, GroupBy and Joins:*/
mysql>
```


MySQL 8.0 Command Line Client

```
mysql> /*Task 3. Aggregate functions, Having, Order By, GroupBy 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,
    ->     s.last_name,
    ->     SUM(p.amount) AS total_payments
    -> FROM
    ->     Students s
    -> JOIN
    ->     Payments p ON s.student_id = p.student_id
    -> WHERE
    ->     s.student_id = 9;
+------------+------------+-----------+----------------+
| student_id | first_name | last_name | total_payments |
+------------+------------+-----------+----------------+
|          9 | Kabir      | Shah      |           4800 |
+------------+------------+-----------+----------------+
1 row in set (0.00 sec)

mysql> /*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.student_id) AS enrolled_students
    -> FROM
    ->     Courses c
    -> JOIN
    ->     Enrollments e ON c.course_id = e.course_id
    -> GROUP BY
    ->     c.course_id, c.course_name;
+-----------+---------------------------------+------------------+
| course_id | course_name                     | enrolled_students |
+-----------+---------------------------------+------------------+
|       201 | Introduction to Computer Science |                1 |
|       202 | Data Structures and Algorithms   |                1 |
|       203 | Database Management Systems      |                1 |
|       204 | Computer Networks                |                1 |
|       205 | Software Engineering             |                1 |
|       206 | Artificial Intelligence          |                1 |
|       207 | Web Development                  |                1 |
|       208 | Cybersecurity Fundamentals       |                1 |
```

MySQL 8.0 Command Line Client

```
8 rows in set (0.01 sec)

mysql> /* 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,
    ->     s.last_name
    -> FROM
    ->     Students s
    -> LEFT JOIN
    ->     Enrollments e ON s.student_id = e.student_id
    -> WHERE
    ->     e.student_id IS NULL;
+------------+------------+-----------+
| student_id | first_name | last_name |
+------------+------------+-----------+
|          9 | Kabir      | Shah      |
|         10 | Neha       | Chopra    |
|         11 | John       | Doe       |
+------------+------------+-----------+
3 rows in set (0.00 sec)

mysql> /* 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
    ->     Students s
    -> JOIN
    ->     Enrollments e ON s.student_id = e.student_id
    -> JOIN
    ->     Courses c ON e.course_id = c.course_id;
+------------+-----------+---------------------------------+
| first_name | last_name | course_name                     |
+------------+-----------+---------------------------------+
| Aarav      | Kumar     | Introduction to Computer Science |
| Ananya     | Sharma    | Data Structures and Algorithms   |
| Rahul      | Patel     | Database Management Systems      |
```

MySQL 8.0 Command Line Client

```
    -> JOIN
    ->     Courses c ON e.course_id = c.course_id;
+------------+-----------+---------------------------------+
| first_name | last_name | course_name                     |
+------------+-----------+---------------------------------+
| Aarav      | Kumar     | Introduction to Computer Science |
| Ananya     | Sharma    | Data Structures and Algorithms   |
| Rahul      | Patel     | Database Management Systems      |
| Aisha      | Singh     | Computer Networks                |
| Arjun      | Verma     | Software Engineering             |
| Sanya      | Mishra    | Artificial Intelligence          |
| Vikram     | Gupta     | Web Development                  |
| Jiya       | Yadav     | Cybersecurity Fundamentals       |
+------------+-----------+---------------------------------+
8 rows in set (0.00 sec)

mysql> /* 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 AS teacher_first_name,
    ->     t.last_name AS teacher_last_name,
    ->     c.course_name
    -> FROM
    ->     Teacher t
    -> JOIN
    ->     Courses c ON t.teacher_id = c.teacher_id;
+--------------------+-------------------+---------------------------------+
| teacher_first_name | teacher_last_name | course_name                     |
+--------------------+-------------------+---------------------------------+
| Priya              | Sharma            | Introduction to Computer Science |
| Rahul              | Verma             | Data Structures and Algorithms   |
| Neha               | Singh             | Database Management Systems      |
| Rajesh             | Patil             | Computer Networks                |
| Sneha              | Gupta             | Software Engineering             |
| Vikram             | Kumar             | Artificial Intelligence          |
| Anita              | Chopra            | Web Development                  |
| Amit               | Mishra            | Cybersecurity Fundamentals       |
| Shalini            | Malhotra          | Operating Systems                |
| Shalini            | Malhotra          | Machine Learning                 |
+--------------------+-------------------+---------------------------------+
10 rows in set (0.00 sec)

mysql> /* 6Retrieve 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 8.0 Command Line Client

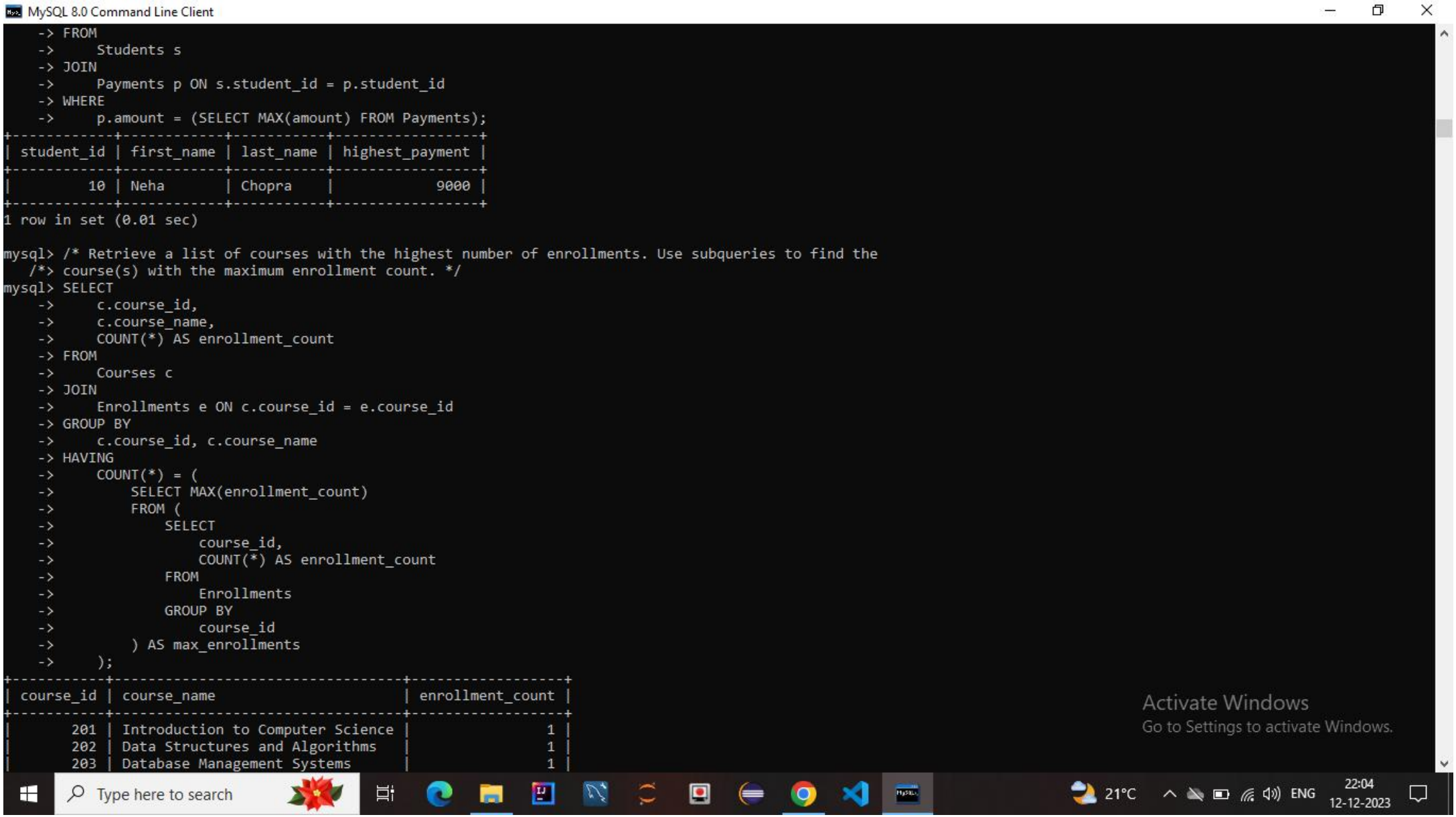
```
mysql> SELECT    s.first_name,s.last_name,e.enrollemnt_date FROM    Students s JOIN  Enrollments e ON s.student_id = e.student_id
               JOIN  Courses c ON e.course_id = c.course_id
      WHERE  c.course_name = 'Database Management Systems';
+------------+-----------+-----------------+
| first_name | last_name | enrollemnt_date |
+------------+-----------+-----------------+
| Rahul      | Patel     | 2023-03-10      |
+------------+-----------+-----------------+
1 row in set (0.00 sec)

mysql> /* 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.first_name,
    ->     s.last_name
    -> FROM
    ->     Students s
    -> LEFT JOIN
    ->     Payments p ON s.student_id = p.student_id
    -> WHERE
    ->     p.student_id IS NULL;
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| John       | Doe       |
+------------+-----------+
1 row in set (0.00 sec)

mysql> /* 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 c
    -> LEFT JOIN
    ->     Enrollments e ON c.course_id = e.course_id
    -> WHERE
    ->     e.course_id IS NULL;
+-----------+------------------+
| course_id | course_name      |
+-----------+------------------+
```

MySQL 8.0 Command Line Client

```
    ->     e.course_id IS NULL;
+-----------+-------------------+
| course_id | course_name       |
+-----------+-------------------+
|       209 | Operating Systems |
|       210 | Machine Learning  |
+-----------+-------------------+
2 rows in set (0.00 sec)

mysql> /* 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
    ->     e1.student_id,
    ->     s.first_name,
    ->     s.last_name,
    ->     COUNT(DISTINCT e1.course_id) AS enrolled_courses
    -> FROM
    ->     Enrollments e1
    -> JOIN
    ->     Students s ON e1.student_id = s.student_id
    -> GROUP BY
    ->     e1.student_id, s.first_name, s.last_name
    -> HAVING
    ->     COUNT(DISTINCT e1.course_id) > 1;
Empty set (0.00 sec)

mysql> /*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,
    ->     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 |
+------------+------------+-----------+
|        310 | Ravi       | Joshi     |
+------------+------------+-----------+
```

MySQL 8.0 Command Line Client

```
mysql> /*Task 4. Subquery and its type:*/
mysql> /* 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
    ->     c.course_id,
    ->     c.course_name,
    ->     AVG(num_students) AS average_students_enrolled
    -> FROM
    ->     Courses c
    -> JOIN (
    ->     SELECT
    ->         course_id,
    ->         COUNT(DISTINCT student_id) AS num_students
    ->     FROM
    ->         Enrollments
    ->     GROUP BY
    ->         course_id
    -> ) e ON c.course_id = e.course_id
    -> GROUP BY
    ->     c.course_id, c.course_name;
+-----------+----------------------------------+---------------------------+
| course_id | course_name                      | average_students_enrolled |
+-----------+----------------------------------+---------------------------+
|       201 | Introduction to Computer Science |                    1.0000 |
|       202 | Data Structures and Algorithms   |                    1.0000 |
|       203 | Database Management Systems      |                    1.0000 |
|       204 | Computer Networks                |                    1.0000 |
|       205 | Software Engineering             |                    1.0000 |
|       206 | Artificial Intelligence          |                    1.0000 |
|       207 | Web Development                  |                    1.0000 |
|       208 | Cybersecurity Fundamentals       |                    1.0000 |
+-----------+----------------------------------+---------------------------+
8 rows in set (0.01 sec)

mysql> /*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,
    ->     s.last_name,
    ->     p.amount AS highest_payment
    -> FROM
    ->     Students s
```


MySQL 8.0 Command Line Client
-> FROM
-> Students s
-> JOIN
-> Payments p ON s.student_id = p.student_id
-> WHERE
-> p.amount = (SELECT MAX(amount) FROM Payments);
| student_id | first_name | last_name | highest_payment |
| 10 | Neha | Chopra | 9000 |
1 row in set (0.01 sec)
mysql> /* 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,
-> COUNT(*) AS enrollment_count
-> FROM
-> Courses c
-> JOIN
-> Enrollments e ON c.course_id = e.course_id
-> GROUP BY
-> c.course_id, c.course_name
-> HAVING
-> COUNT(*) = (
-> SELECT MAX(enrollment_count)
-> FROM (
-> SELECT
-> course_id,
-> COUNT(*) AS enrollment_count
-> FROM
-> Enrollments
-> GROUP BY
-> course_id
->) AS max_enrollments
->);
course_id	course_name	enrollment_count
201	Introduction to Computer Science	1
202	Data Structures and Algorithms	1
203	Database Management Systems	1
Activate Windows
Go to Settings to activate Windows.
Type here to search
21°C
ENG
22:04
12-12-2023

MySQL 8.0 Command Line Client

```
|       202 | Data Structures and Algorithms   |               1 |
|       203 | Database Management Systems      |               1 |
|       204 | Computer Networks                |               1 |
|       205 | Software Engineering             |               1 |
|       206 | Artificial Intelligence          |               1 |
|       207 | Web Development                  |               1 |
|       208 | Cybersecurity Fundamentals       |               1 |
+-----------+----------------------------------+-----------------+
8 rows in set (0.00 sec)

mysql> /*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,
    ->     t.last_name,
    ->     SUM(p.amount) AS total_payments
    -> 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
    -> GROUP BY
    ->     t.teacher_id, t.first_name, t.last_name;
+------------+------------+-----------+----------------+
| teacher_id | first_name | last_name | total_payments |
+------------+------------+-----------+----------------+
|        301 | Priya      | Sharma    |           5000 |
|        302 | Rahul      | Verma     |           6000 |
|        303 | Neha       | Singh     |           4500 |
|        304 | Rajesh     | Patil     |           5500 |
|        305 | Sneha      | Gupta     |           7000 |
|        306 | Vikram     | Kumar     |           6500 |
|        307 | Anita      | Chopra    |           8000 |
|        308 | Amit       | Mishra    |           7500 |
+------------+------------+-----------+----------------+
8 rows in set (0.00 sec)

mysql> /*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 8.0 Command Line Client

```
8 rows in set (0.00 sec)

mysql> /*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
    ->     s.student_id,
    ->     s.first_name,
    ->     s.last_name
    -> FROM
    ->     Students s
    -> WHERE
    ->     (
    ->         SELECT COUNT(DISTINCT e.course_id)
    ->         FROM Enrollments e
    ->         ) = (
    ->         SELECT COUNT(DISTINCT e.course_id)
    ->         FROM Enrollments e
    ->         WHERE e.student_id = s.student_id
    ->     );
Empty set (0.00 sec)

mysql> /*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,
    ->     t.last_name
    -> FROM
    ->     Teacher t
    -> WHERE
    ->     t.teacher_id NOT IN (
    ->         SELECT DISTINCT c.teacher_id
    ->         FROM Courses c
    ->     );
+------------+------------+-----------+
| teacher_id | first_name | last_name |
+------------+------------+-----------+
|        310 | Ravi       | Joshi     |
+------------+------------+-----------+
1 row in set (0.00 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
1 row in set (0.00 sec)

mysql> /*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
    ->     AVG(student_age) AS average_age
    -> FROM (
    ->     SELECT
    ->         student_id,
    ->         TIMESTAMPDIFF(YEAR, date_of_birth, CURDATE()) AS student_age
    ->     FROM
    ->         Students
    -> ) AS student_ages;
+-------------+
| average_age |
+-------------+
|     28.2727 |
+-------------+
1 row in set (0.00 sec)

mysql> /* 8.Identify courses with no enrollments. Use subqueries to find courses without enrollment
   /*> records.*/
mysql> SELECT
    ->     c.course_id,
    ->     c.course_name
    -> FROM
    ->     Courses c
    -> WHERE
    ->     NOT EXISTS (
    ->         SELECT 1
    ->         FROM Enrollments e
    ->         WHERE e.course_id = c.course_id
    ->     );
+-----------+-------------------+
| course_id | course_name       |
+-----------+-------------------+
|       209 | Operating Systems |
|       210 | Machine Learning  |
+-----------+-------------------+
2 rows in set (0.00 sec)

mysql>
```

MySQL 8.0 Command Line Client

```
mysql> /*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,
    ->     s.last_name,
    ->     e.course_id,
    ->     c.course_name,
    ->     COALESCE(SUM(p.amount), 0) AS total_payments
    -> FROM
    ->     Students s
    -> JOIN
    ->     Enrollments e ON s.student_id = e.student_id
    -> JOIN
    ->     Courses c ON e.course_id = c.course_id
    -> LEFT JOIN
    ->     Payments p ON s.student_id = p.student_id
    -> GROUP BY
    ->     s.student_id, s.first_name, s.last_name, e.course_id, c.course_name;
+------------+------------+-----------+-----------+---------------------------------+----------------+
| student_id | first_name | last_name | course_id | course_name                     | total_payments |
+------------+------------+-----------+-----------+---------------------------------+----------------+
|          1 | Aarav      | Kumar     |       201 | Introduction to Computer Science |           5000 |
|          2 | Ananya     | Sharma    |       202 | Data Structures and Algorithms   |           6000 |
|          3 | Rahul      | Patel     |       203 | Database Management Systems      |           4500 |
|          4 | Aisha      | Singh     |       204 | Computer Networks                |           5500 |
|          5 | Arjun      | Verma     |       205 | Software Engineering             |           7000 |
|          6 | Sanya      | Mishra    |       206 | Artificial Intelligence          |           6500 |
|          7 | Vikram     | Gupta     |       207 | Web Development                  |           8000 |
|          8 | Jiya       | Yadav     |       208 | Cybersecurity Fundamentals       |           7500 |
+------------+------------+-----------+-----------+---------------------------------+----------------+
8 rows in set (0.00 sec)

mysql> /*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,
    ->     s.last_name
    -> FROM
    ->     Students s
    -> JOIN (
    ->     SELECT
```

MySQL 8.0 Command Line Client

```
    -> FROM
    ->     Students s
    -> JOIN (
    ->     SELECT
    ->         student_id,
    ->         COUNT(*) AS payment_count
    ->     FROM
    ->         Payments
    ->     GROUP BY
    ->         student_id
    ->     HAVING
    ->         COUNT(*) > 1
    -> ) AS payment_counts ON s.student_id = payment_counts.student_id;
Empty set (0.00 sec)

mysql> /* 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,
    ->     COALESCE(SUM(p.amount), 0) 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 | Aarav      | Kumar     |           5000 |
|          2 | Ananya     | Sharma    |           6000 |
|          3 | Rahul      | Patel     |           4500 |
|          4 | Aisha      | Singh     |           5500 |
|          5 | Arjun      | Verma     |           7000 |
|          6 | Sanya      | Mishra    |           6500 |
|          7 | Vikram     | Gupta     |           8000 |
|          8 | Jiya       | Yadav     |           7500 |
|          9 | Kabir      | Shah      |           4800 |
|         10 | Neha       | Chopra    |           9000 |
|         11 | John       | Doe       |              0 |
+------------+------------+-----------+----------------+
```

MySQL 8.0 Command Line Client

```
mysql> /*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 enrolled_students_count
    -> FROM
    ->     Courses c
    -> LEFT JOIN
    ->     Enrollments e ON c.course_id = e.course_id
    -> GROUP BY
    ->     c.course_name;
+---------------------------------+-------------------------+
| course_name                     | enrolled_students_count |
+---------------------------------+-------------------------+
| Introduction to Computer Science |                       1 |
| Data Structures and Algorithms   |                       1 |
| Database Management Systems      |                       1 |
| Computer Networks                |                       1 |
| Software Engineering             |                       1 |
| Artificial Intelligence          |                       1 |
| Web Development                  |                       1 |
| Cybersecurity Fundamentals       |                       1 |
| Operating Systems                |                       0 |
| Machine Learning                 |                       0 |
+---------------------------------+-------------------------+
10 rows in set (0.00 sec)

mysql> /*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
    ->     s.student_id,
    ->     s.first_name,
    ->     s.last_name,
    ->     AVG(p.amount) AS average_payment_amount
    -> FROM
    ->     Students s
    -> 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 | average_payment_amount |
```