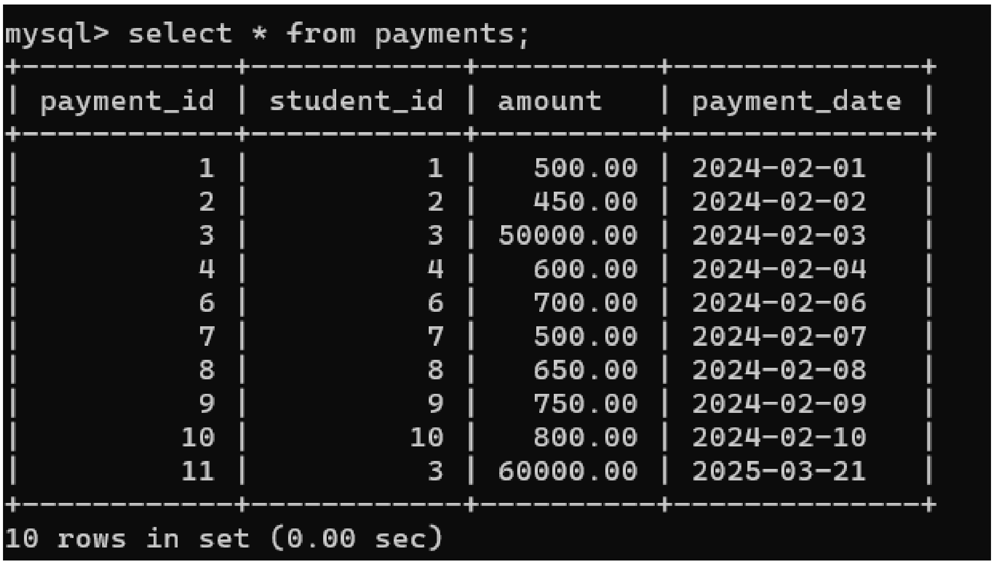
ASSIGNMENT – 1

STUDENT INFORMATION SYSTEM

J701-Deva R

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 * from payments;
+------------+------------+----------+--------------+
| payment_id | student_id | amount   | payment_date |
+------------+------------+----------+--------------+
|          1 |          1 |   500.00 | 2024-02-01   |
|          2 |          2 |   450.00 | 2024-02-02   |
|          3 |          3 | 50000.00 | 2024-02-03   |
|          4 |          4 |   600.00 | 2024-02-04   |
|          6 |          6 |   700.00 | 2024-02-06   |
|          7 |          7 |   500.00 | 2024-02-07   |
|          8 |          8 |   650.00 | 2024-02-08   |
|          9 |          9 |   750.00 | 2024-02-09   |
|         10 |         10 |   800.00 | 2024-02-10   |
|         11 |          3 | 60000.00 | 2025-03-21   |
+------------+------------+----------+--------------+
10 rows in set (0.00 sec)
```

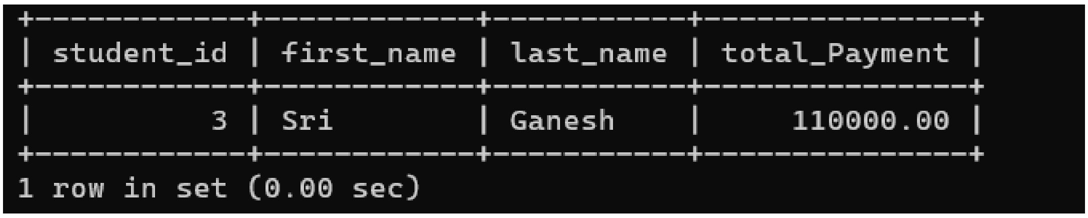
select s.student_id,s.first_name,s.last_name, sum(p.amount) as total_Payment

-> from students s join payments p on s.student_id=p.student_id

-> where s.student_id=3

-> group by s.student_id,s.first_name,s.last_name;

Output:



```
+------------+------------+-----------+---------------+
| student_id | first_name | last_name | total_Payment |
+------------+------------+-----------+---------------+
|          3 | Sri        | Ganesh    |     110000.00 |
+------------+------------+-----------+---------------+
1 row 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.

SELECT C.course_id, C.course_name, COUNT(E.student_id) AS student_count

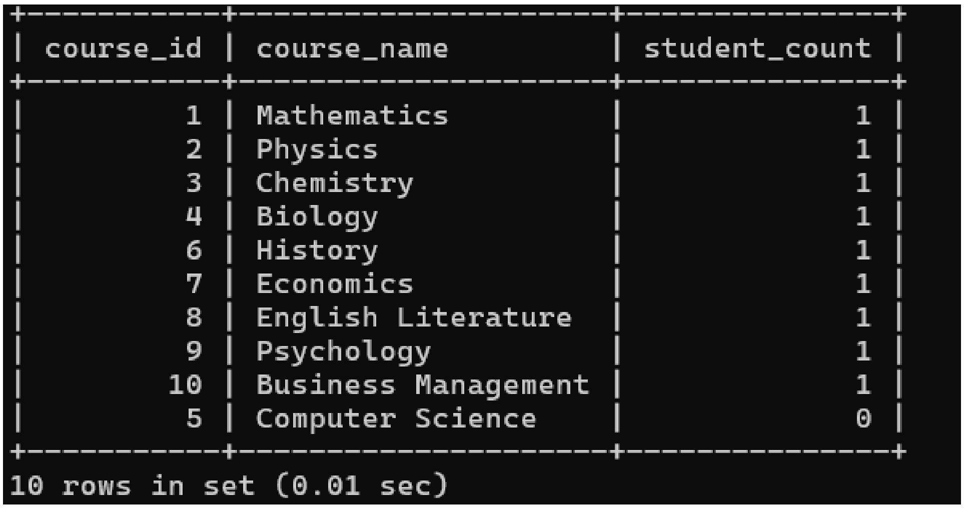
-> FROM Courses C

-> LEFT JOIN Enrollments E ON C.course_id = E.course_id

-> GROUP BY C.course_id, C.course_name

-> ORDER BY student_count DESC;

Output:



```
+-----------+---------------------+---------------+
| course_id | course_name         | student_count |
+-----------+---------------------+---------------+
|         1 | Mathematics         |             1 |
|         2 | Physics             |             1 |
|         3 | Chemistry           |             1 |
|         4 | Biology             |             1 |
|         6 | History             |             1 |
|         7 | Economics           |             1 |
|         8 | English Literature  |             1 |
|         9 | Psychology          |             1 |
|        10 | Business Management |             1 |
|         5 | Computer Science    |             0 |
+-----------+---------------------+---------------+
10 rows in set (0.01 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.

select s.first_name,s.last_name from students s

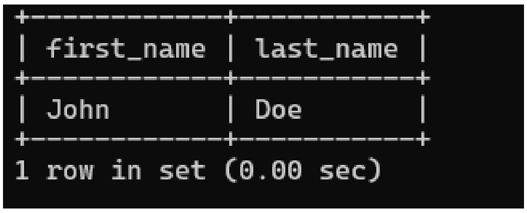
-> left join enrollments e

-> on s.student_id=e.student_id

-> where e.student_id is null

-> order by s.first_name,s.last_name;

Output:



```
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| John       | Doe       |
+------------+-----------+
1 row 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.

   select s.first_name,s.last_name,c.course_name from students s

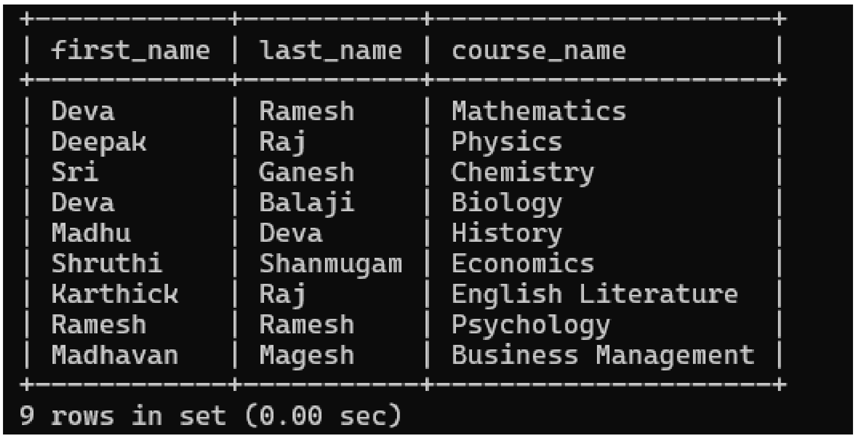
   -> join enrollments e

   -> on s.student_id=e.student_id

   -> join courses c

   -> on e.course_id=c.course_id;

Output:



```
+------------+-----------+---------------------+
| first_name | last_name | course_name         |
+------------+-----------+---------------------+
| Deva       | Ramesh    | Mathematics         |
| Deepak     | Raj       | Physics             |
| Sri        | Ganesh    | Chemistry           |
| Deva       | Balaji    | Biology             |
| Madhu      | Deva      | History             |
| Shruthi    | Shanmugam | Economics           |
| Karthick   | Raj       | English Literature  |
| Ramesh     | Ramesh    | Psychology          |
| Madhavan   | Magesh    | Business Management |
+------------+-----------+---------------------+
9 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.

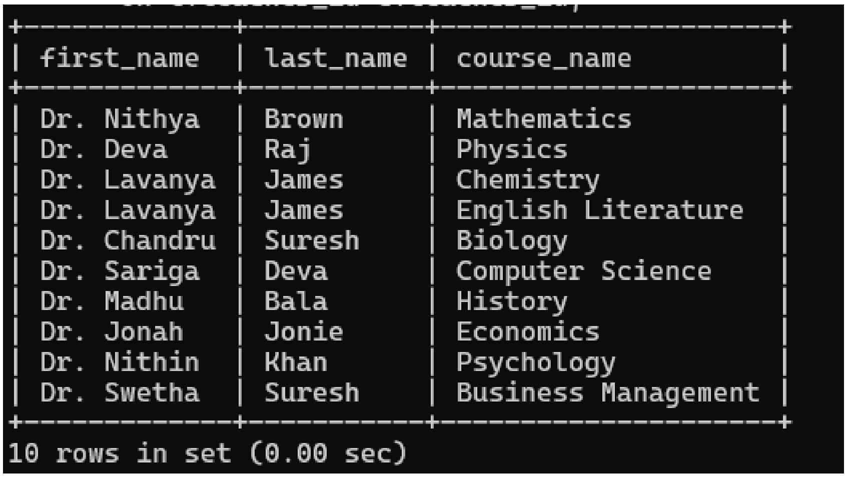
   Select t.first_name,t.last_name,c.course_name

   -> from teacher t

   -> join courses c

   -> on t.teacher_id=c.teacher_id;

Output:



```
+-------------+-----------+---------------------+
| first_name  | last_name | course_name         |
+-------------+-----------+---------------------+
| Dr. Nithya  | Brown     | Mathematics         |
| Dr. Deva    | Raj       | Physics             |
| Dr. Lavanya | James     | Chemistry           |
| Dr. Lavanya | James     | English Literature  |
| Dr. Chandru | Suresh    | Biology             |
| Dr. Sariga  | Deva      | Computer Science    |
| Dr. Madhu   | Bala      | History             |
| Dr. Jonah   | Jonie     | Economics           |
| Dr. Nithin  | Khan      | Psychology          |
| Dr. Swetha  | Suresh    | Business Management |
+-------------+-----------+---------------------+
10 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

select s.first_name,s.last_name,c.course_name,e.enrollment_date

-> from students s

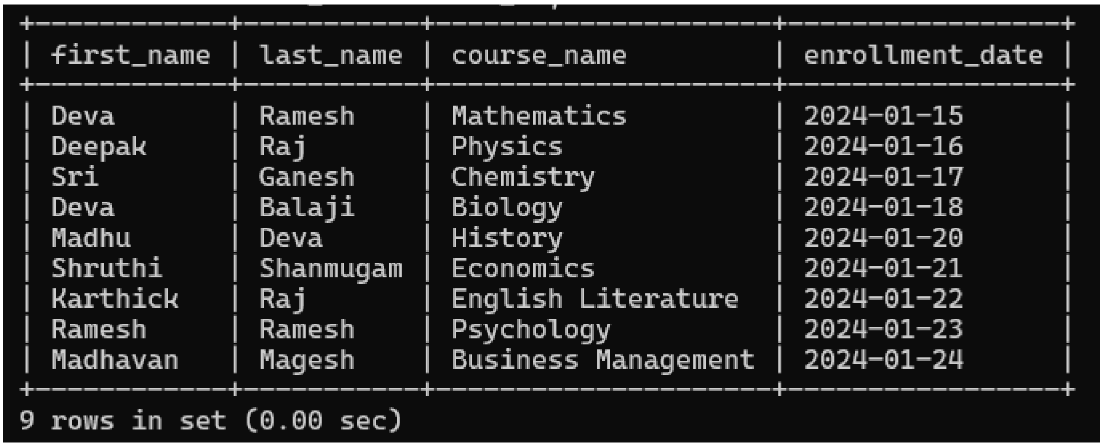
-> join enrollments e

-> on s.student_id=e.student_id

-> join courses c

-> on e.course_id=c.course_id;

Output:



```
+------------+-----------+---------------------+-----------------+
| first_name | last_name | course_name         | enrollment_date |
+------------+-----------+---------------------+-----------------+
| Deva       | Ramesh    | Mathematics         | 2024-01-15      |
| Deepak     | Raj       | Physics             | 2024-01-16      |
| Sri        | Ganesh    | Chemistry           | 2024-01-17      |
| Deva       | Balaji    | Biology             | 2024-01-18      |
| Madhu      | Deva      | History             | 2024-01-20      |
| Shruthi    | Shanmugam | Economics           | 2024-01-21      |
| Karthick   | Raj       | English Literature  | 2024-01-22      |
| Ramesh     | Ramesh    | Psychology          | 2024-01-23      |
| Madhavan   | Magesh    | Business Management | 2024-01-24      |
+------------+-----------+---------------------+-----------------+
9 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.

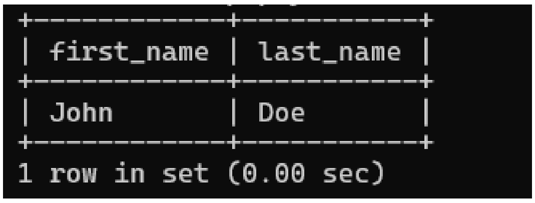
select s.first_name,s.last_name from students s

-> left join payments p

-> on s.student_id=p.student_id

-> where p.payment_id is null;

Output:

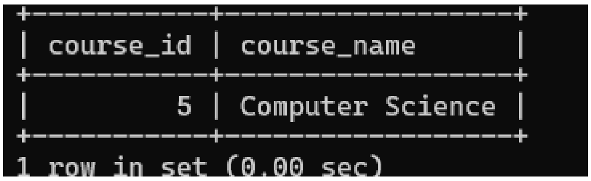


```
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| John       | Doe       |
+------------+-----------+
1 row in set (0.00 sec)
```

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.

select c.course_id,c.course_name from courses c
-> left join enrollments e
-> on e.course_id=c.course_id
-> where e.course_id is null;

Output:



```
+-----------+------------------+
| course_id | course_name      |
+-----------+------------------+
|         5 | Computer Science |
+-----------+------------------+
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

SELECT S.student_id, S.first_name, S.last_name, COUNT(E.course_id) AS total_courses

-> FROM Students S

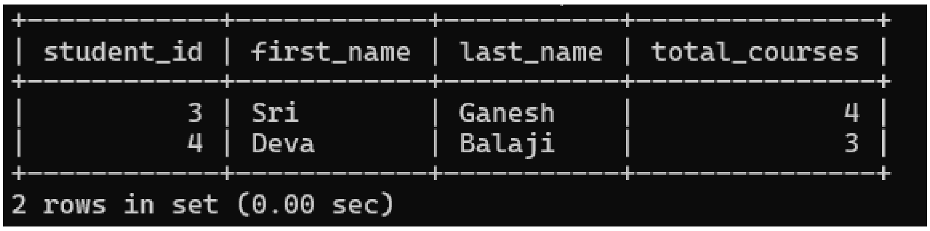
-> JOIN Enrollments E ON S.student_id = E.student_id

-> GROUP BY S.student_id, S.first_name, S.last_name

-> HAVING COUNT(E.course_id) > 1

-> ORDER BY total_courses DESC;

Output:

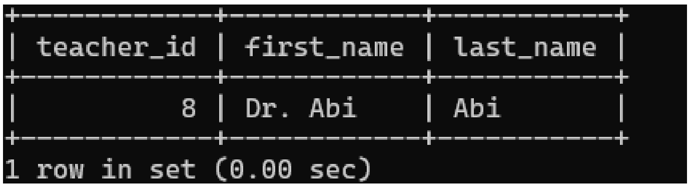


```
+------------+------------+-----------+---------------+
| student_id | first_name | last_name | total_courses |
+------------+------------+-----------+---------------+
|          3 | Sri        | Ganesh    |             4 |
|          4 | Deva       | Balaji    |             3 |
+------------+------------+-----------+---------------+
2 rows 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.

```
 SELECT T.teacher_id, T.first_name, T.last_name
    -> FROM Teacher T
    -> LEFT JOIN Courses C ON T.teacher_id = C.teacher_id
    -> WHERE C.course_id IS NULL
    -> ORDER BY T.last_name, T.first_name;
```

Output:



```
+------------+------------+-----------+
| teacher_id | first_name | last_name |
+------------+------------+-----------+
|          8 | Dr. Abi    | Abi       |
+------------+------------+-----------+
1 row in set (0.00 sec)
```