GROUP MEMBER: NKURANGA BAZIGA CALEB 28845

ISHIMWE MUNEZA PRINCE 27816

IZERE SABIN PATIENCE 27969

DATABASE DEVELOPMENT WITH PL/SQL GROUP WORK

NO1: Create a table with appropriate columns and apply relevant constraints (e.g., primary key, foreign key, unique, not null).

Firstly before creating table we created database which we named it schooldb and then

We created three tables:

1. **Students** – contains details of students with a primary key and unique email constraint.
2. **Courses** – stores courses with a primary key and a check constraint for credit hours.
3. **Enrollments** – junction table linking students and courses with foreign keys.

Constraints Applied:

- `PRIMARY KEY` on all tables.
- `UNIQUE` on email in Students.
- `CHECK` to ensure credit hours > 0.
- `FOREIGN KEY` references between Students and Courses.

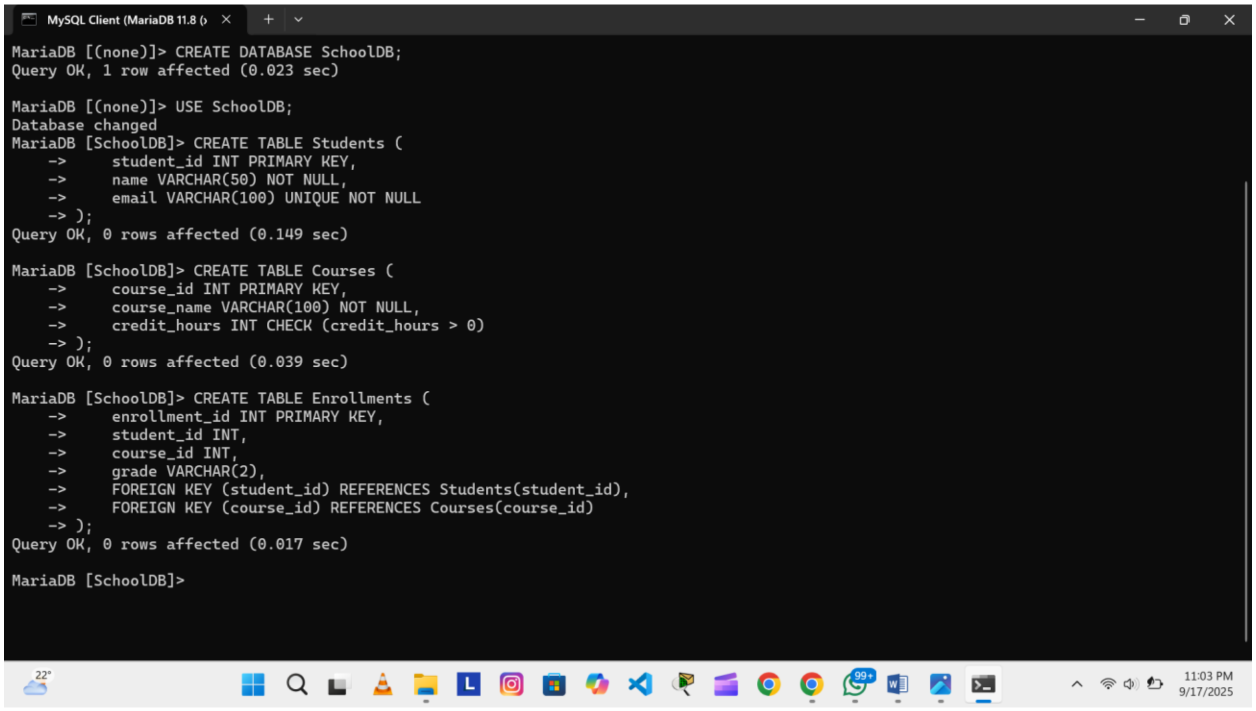
Inserting data :
after creating table we inserted data so that the query of joining will work smoothly
student table

Student _id	Names	Email
1	Alice	alice@gmail.com
2	Bob	bob@gmail.com
3	Charlie	charlie@gmail.com
4	diana	diana@gmail.com

Course table

Course_id	Course_name	Credit_hours
101	Database systems	3
102	Computer networks	4
103	Operating systems	3
104	Artificial intelligence	4

Enrollment_id	Student_id	Course_id	Grade
1	1	101	A
2	1	102	B
3	2	103	A
4	3	101	C



```
MariaDB [(none)]> CREATE DATABASE SchoolDB;
Query OK, 1 row affected (0.023 sec)

MariaDB [(none)]> USE SchoolDB;
Database changed
MariaDB [SchoolDB]> CREATE TABLE Students (
    ->     student_id INT PRIMARY KEY,
    ->     name VARCHAR(50) NOT NULL,
    ->     email VARCHAR(100) UNIQUE NOT NULL
    -> );
Query OK, 0 rows affected (0.149 sec)

MariaDB [SchoolDB]> CREATE TABLE Courses (
    ->     course_id INT PRIMARY KEY,
    ->     course_name VARCHAR(100) NOT NULL,
    ->     credit_hours INT CHECK (credit_hours > 0)
    -> );
Query OK, 0 rows affected (0.039 sec)

MariaDB [SchoolDB]> CREATE TABLE Enrollments (
    ->     enrollment_id INT PRIMARY KEY,
    ->     student_id INT,
    ->     course_id INT,
    ->     grade VARCHAR(2),
    ->     FOREIGN KEY (student_id) REFERENCES Students(student_id),
    ->     FOREIGN KEY (course_id) REFERENCES Courses(course_id)
    -> );
Query OK, 0 rows affected (0.017 sec)

MariaDB [SchoolDB]>
```

2. Perform different types of joins (INNER, LEFT, RIGHT, FULL) using your table(s).

With an **INNER JOIN**, the database looks at both tables and only keeps the rows where there is a perfect match on both sides — so only students linked to courses through enrollments appear.

With a **LEFT JOIN**, the database takes **all the rows from the left table (Students)** first, then tries to match them with enrollments and courses. If a student has no match, the database still keeps them but fills the missing course and grade with `NULL`.

RIGHT JOIN
This join keeps **all courses** from the course list, even if nobody is enrolled in them. That's why the Artificial Intelligence course still shows up in the results, but without any student attached to it. Students not linked to a course won't appear here.

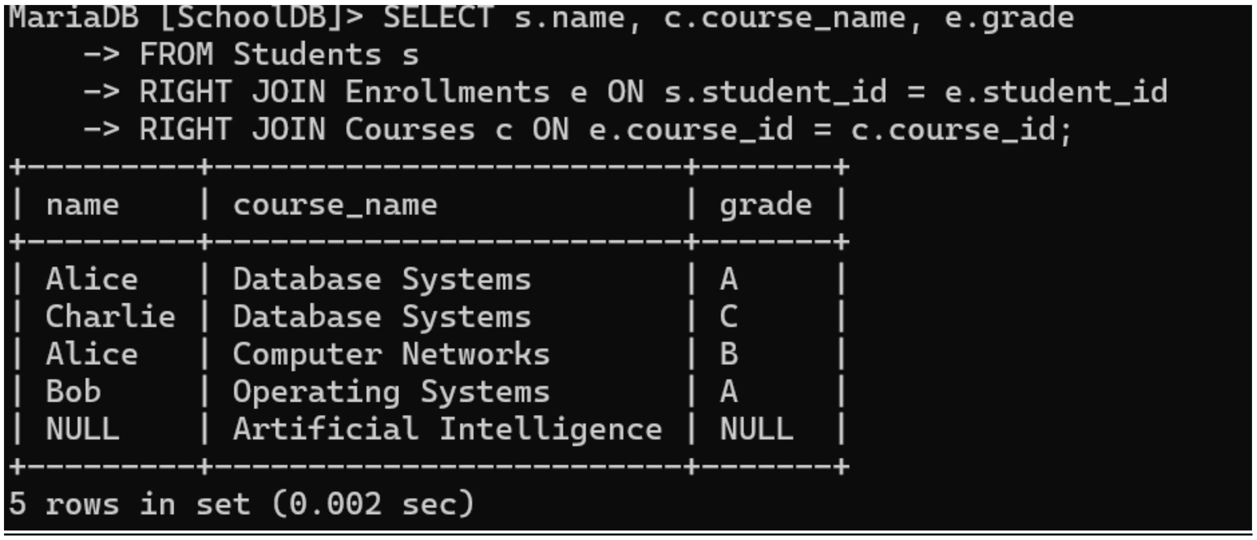
FULL JOIN
This join combines both sides — it shows **all students and all courses**, whether they match or

not. That means you'll see Diana (who has no course) and the Artificial Intelligence course (which has no student) alongside the normal enrollments.

So to sum up all four in one line:

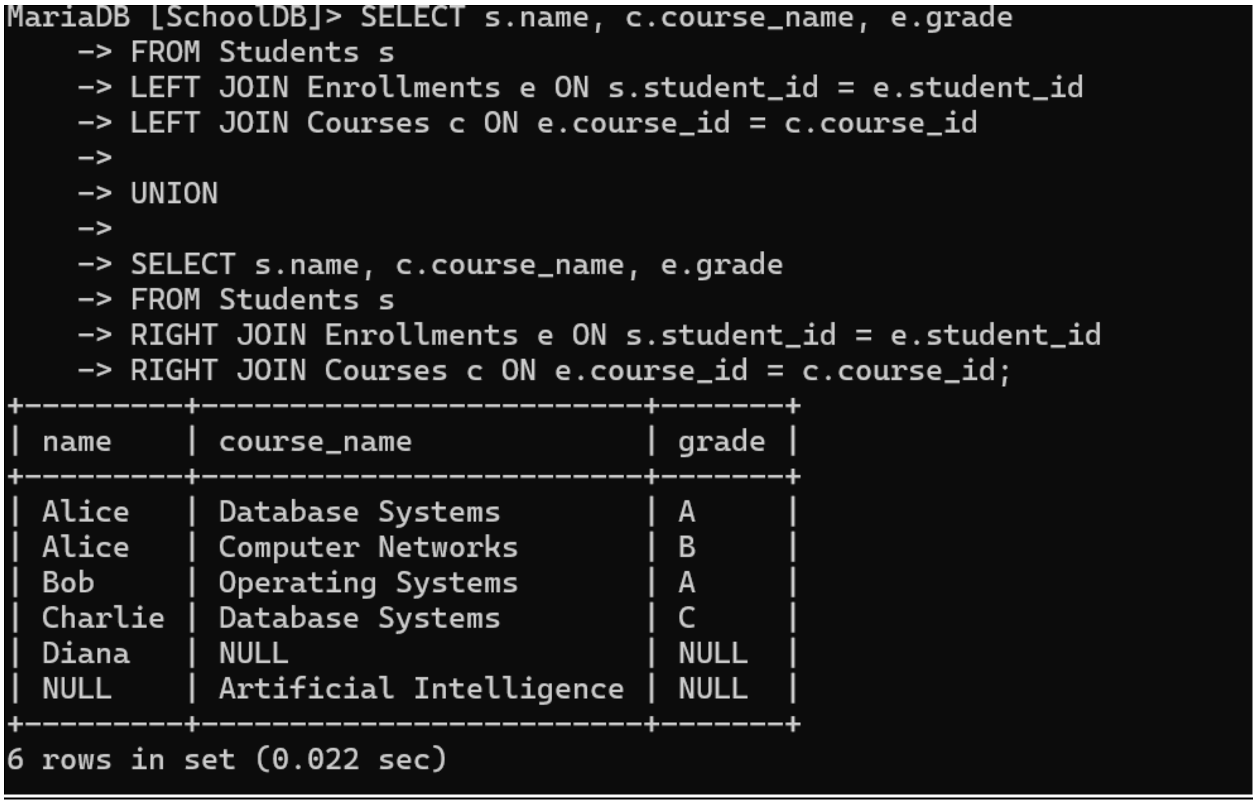
- **INNER JOIN** → Only matched student–course pairs.
- **LEFT JOIN** → All students, matched or not.
- **RIGHT JOIN** → All courses, matched or not.
- **FULL JOIN** → Everything from both sides, whether matched or not.

RIGHT JOIN



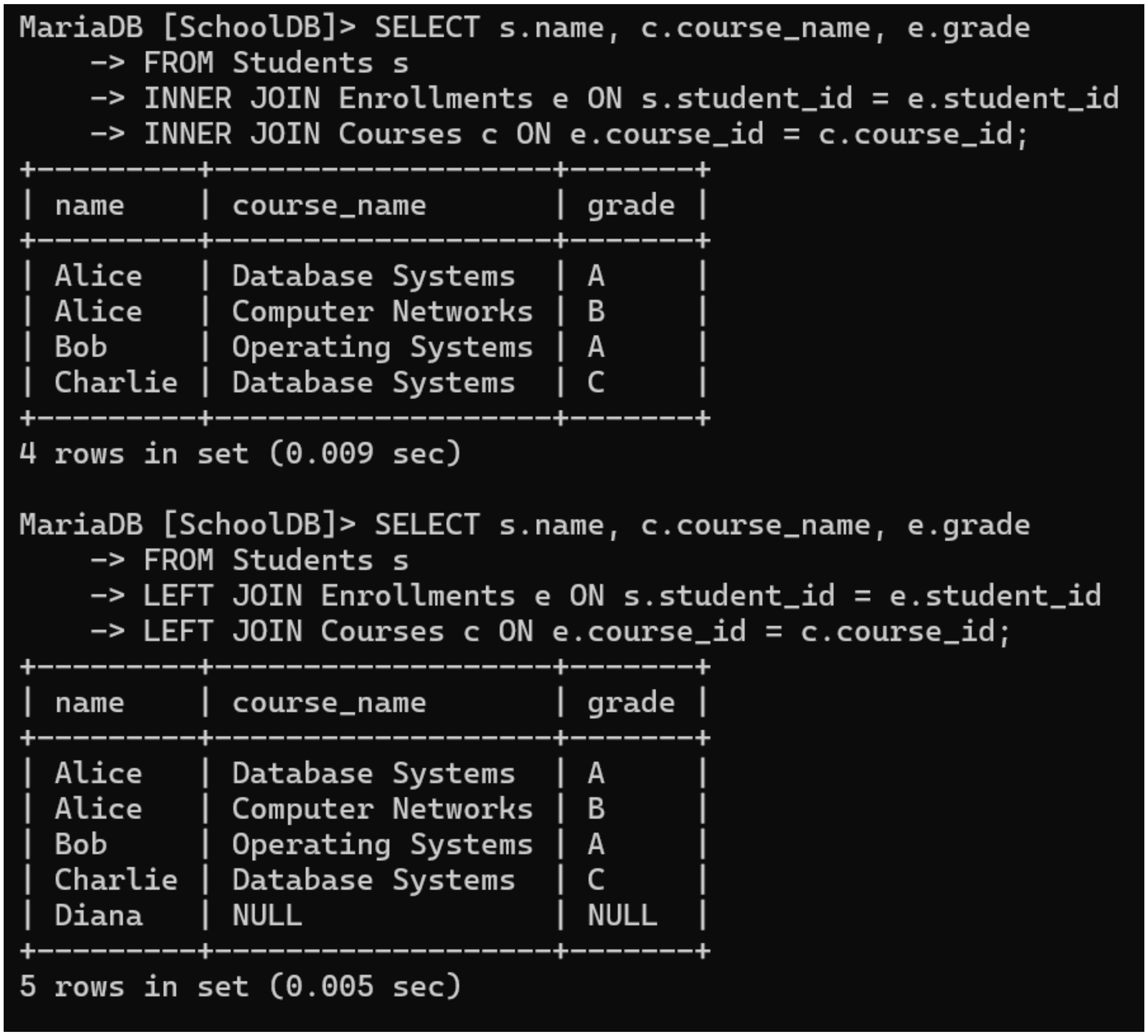
```
MariaDB [SchoolDB]> SELECT s.name, c.course_name, e.grade
    -> FROM Students s
    -> RIGHT JOIN Enrollments e ON s.student_id = e.student_id
    -> RIGHT JOIN Courses c ON e.course_id = c.course_id;
+---------+-------------------------+-------+
| name    | course_name             | grade |
+---------+-------------------------+-------+
| Alice   | Database Systems        | A     |
| Charlie | Database Systems        | C     |
| Alice   | Computer Networks       | B     |
| Bob     | Operating Systems       | A     |
| NULL    | Artificial Intelligence | NULL  |
+---------+-------------------------+-------+
5 rows in set (0.002 sec)
```

Full join



```
MariaDB [SchoolDB]> SELECT s.name, c.course_name, e.grade
    -> FROM Students s
    -> LEFT JOIN Enrollments e ON s.student_id = e.student_id
    -> LEFT JOIN Courses c ON e.course_id = c.course_id
    ->
    -> UNION
    ->
    -> SELECT s.name, c.course_name, e.grade
    -> FROM Students s
    -> RIGHT JOIN Enrollments e ON s.student_id = e.student_id
    -> RIGHT JOIN Courses c ON e.course_id = c.course_id;
+---------+-------------------------+-------+
| name    | course_name             | grade |
+---------+-------------------------+-------+
| Alice   | Database Systems        | A     |
| Alice   | Computer Networks       | B     |
| Bob     | Operating Systems       | A     |
| Charlie | Database Systems        | C     |
| Diana   | NULL                    | NULL  |
| NULL    | Artificial Intelligence | NULL  |
+---------+-------------------------+-------+
6 rows in set (0.022 sec)
```

INNERJOIN AND LEFT JOIN



```
MariaDB [SchoolDB]> SELECT s.name, c.course_name, e.grade
    -> FROM Students s
    -> INNER JOIN Enrollments e ON s.student_id = e.student_id
    -> INNER JOIN Courses c ON e.course_id = c.course_id;
+---------+-------------------+-------+
| name    | course_name       | grade |
+---------+-------------------+-------+
| Alice   | Database Systems  | A     |
| Alice   | Computer Networks | B     |
| Bob     | Operating Systems | A     |
| Charlie | Database Systems  | C     |
+---------+-------------------+-------+
4 rows in set (0.009 sec)

MariaDB [SchoolDB]> SELECT s.name, c.course_name, e.grade
    -> FROM Students s
    -> LEFT JOIN Enrollments e ON s.student_id = e.student_id
    -> LEFT JOIN Courses c ON e.course_id = c.course_id;
+---------+-------------------+-------+
| name    | course_name       | grade |
+---------+-------------------+-------+
| Alice   | Database Systems  | A     |
| Alice   | Computer Networks | B     |
| Bob     | Operating Systems | A     |
| Charlie | Database Systems  | C     |
| Diana   | NULL              | NULL  |
+---------+-------------------+-------+
5 rows in set (0.005 sec)
```

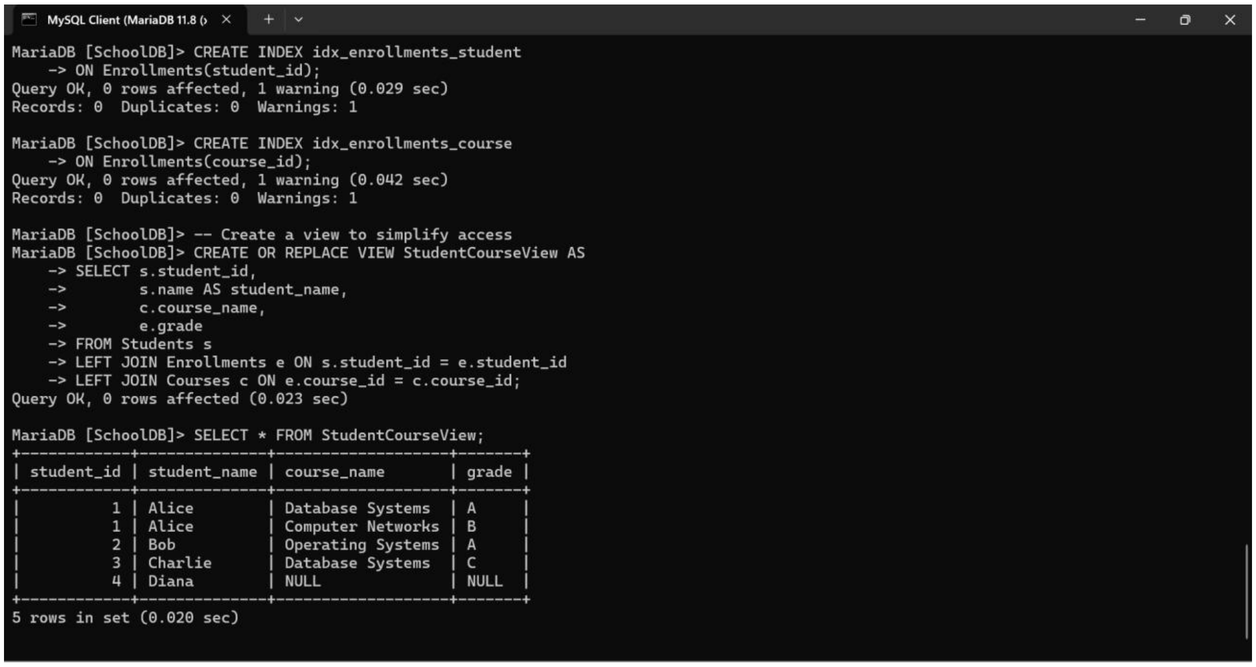
3. Create an index to optimize query performance.

- **Index:** It's like a shortcut in a book. Creating an index on `student_id` or `course_id` helps the database **find data faster** when searching or joining tables.

4. Create a view to simplify data access.

View: A view is like a **saved query**. Instead of writing a long join every time, you can create a view that shows **students, their courses, and grades** all together. Then you just ask the view for the information,

and it gives you the result.



```
MariaDB [SchoolDB]> CREATE INDEX idx_enrollments_student
    -> ON Enrollments(student_id);
Query OK, 0 rows affected, 1 warning (0.029 sec)
Records: 0  Duplicates: 0  Warnings: 1

MariaDB [SchoolDB]> CREATE INDEX idx_enrollments_course
    -> ON Enrollments(course_id);
Query OK, 0 rows affected, 1 warning (0.042 sec)
Records: 0  Duplicates: 0  Warnings: 1

MariaDB [SchoolDB]> -- Create a view to simplify access
MariaDB [SchoolDB]> CREATE OR REPLACE VIEW StudentCourseView AS
    -> SELECT s.student_id,
    ->        s.name AS student_name,
    ->        c.course_name,
    ->        e.grade
    -> FROM Students s
    -> LEFT JOIN Enrollments e ON s.student_id = e.student_id
    -> LEFT JOIN Courses c ON e.course_id = c.course_id;
Query OK, 0 rows affected (0.023 sec)

MariaDB [SchoolDB]> SELECT * FROM StudentCourseView;
+------------+--------------+-------------------+-------+
| student_id | student_name | course_name       | grade |
+------------+--------------+-------------------+-------+
|          1 | Alice        | Database Systems  | A     |
|          1 | Alice        | Computer Networks | B     |
|          2 | Bob          | Operating Systems | A     |
|          3 | Charlie      | Database Systems  | C     |
|          4 | Diana        | NULL              | NULL  |
+------------+--------------+-------------------+-------+
5 rows in set (0.020 sec)
```