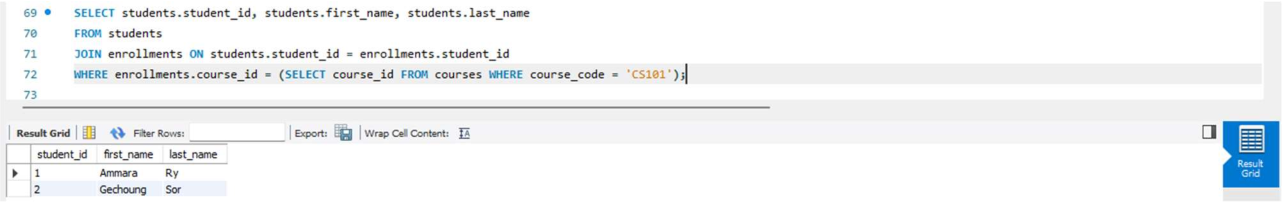
WCT II – Assignment

ITE (M2)

Name: Ry Kimchhay

1. Retrieve all students who enrolled in a specific course.



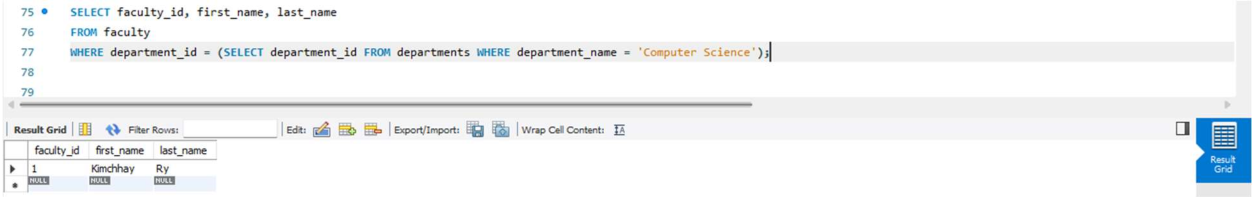
```sql
SELECT students.student_id, students.first_name, students.last_name
FROM students
JOIN enrollments ON students.student_id = enrollments.student_id
WHERE enrollments.course_id = (SELECT course_id FROM courses WHERE course_code = 'CS101');
```

student_id	first_name	last_name
1	Ammara	Ry
2	Gechoung	Sor

Explaination:

- This query finds students who are enrolled in the course "CS101".
- It joins the students table with the enrollments table using student_id.
- It filters the results to only show students who are in the course with code "CS101".

2. Find all faculty members in a particular department.



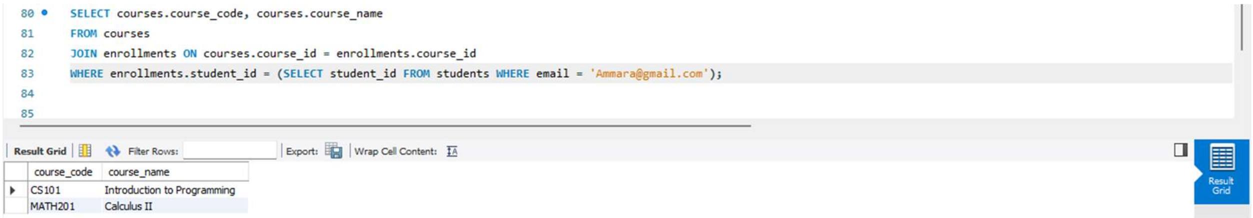
```sql
SELECT faculty_id, first_name, last_name
FROM faculty
WHERE department_id = (SELECT department_id FROM departments WHERE department_name = 'Computer Science');
```

faculty_id	first_name	last_name
1	Kimchhay	Ry
NULL	NULL	NULL

Explaination:

- This query finds all faculty members in the Computer Science department.
- It selects the faculty members where department_id matches the department_id of "Computer Science" in the departments table.

3. List all courses a particular student is enrolled in.



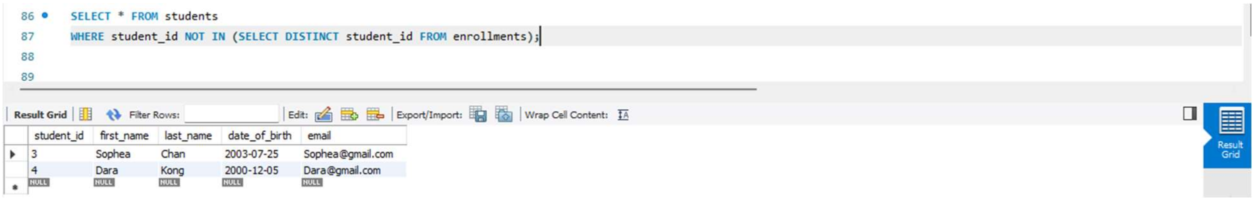
```sql
SELECT courses.course_code, courses.course_name
FROM courses
JOIN enrollments ON courses.course_id = enrollments.course_id
WHERE enrollments.student_id = (SELECT student_id FROM students WHERE email = 'Ammara@gmail.com');
```

course_code	course_name
CS101	Introduction to Programming
MATH201	Calculus II

Explaination:

- This query finds all courses that Ammara Ry is enrolled in.
- It joins the courses table with the enrollments table using course_id.
- It filters the results to show only courses where the student_id matches the one belonging to the student with the email "Ammara@gmail.com".

4. Retrieve students who have not enrolled in any course



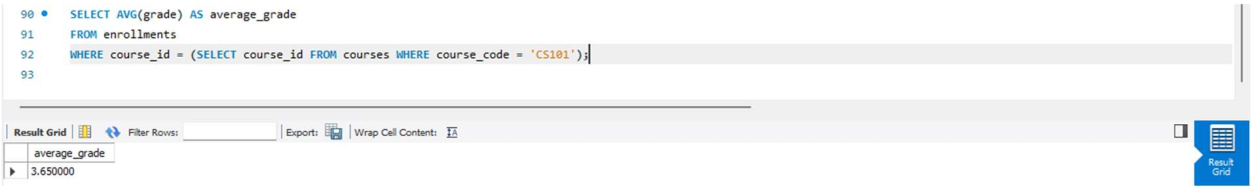
```sql
SELECT * FROM students
WHERE student_id NOT IN (SELECT DISTINCT student_id FROM enrollments);
```

student_id	first_name	last_name	date_of_birth	email
3	Sophea	Chan	2003-07-25	Sophea@gmail.com
4	Dara	Kong	2000-12-05	Dara@gmail.com
NULL	NULL	NULL	NULL	NULL

Explaination:

- This query finds students who haven't enrolled in any course.
- The subquery (SELECT DISTINCT student_id FROM enrollments) gets all students who have enrolled in at least one course.
- The main query selects students whose student_id is not in that list.

5. Find the average grade of students in a specific course.



```sql
SELECT AVG(grade) AS average_grade
FROM enrollments
WHERE course_id = (SELECT course_id FROM courses WHERE course_code = 'CS101');
```

```
average_grade
3.650000
```

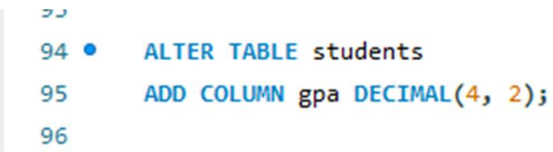
Explaination:

- This query calculates the average grade of students in CS101.
- The subquery finds the course_id of "CS101".
- The main query gets the average AVG() of all grades from students in that course.

Bonus:

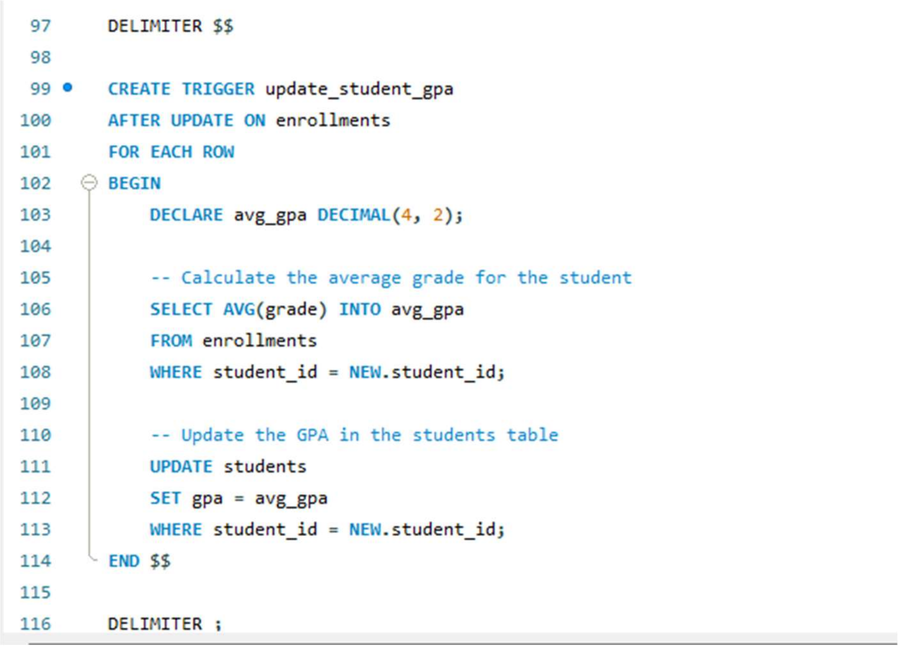
Implement a trigger to update a student's GPA when a grade is updated.

1. Add gpa column to the students table



```sql
ALTER TABLE students
ADD COLUMN gpa DECIMAL(4, 2);
```

2. Create the trigger

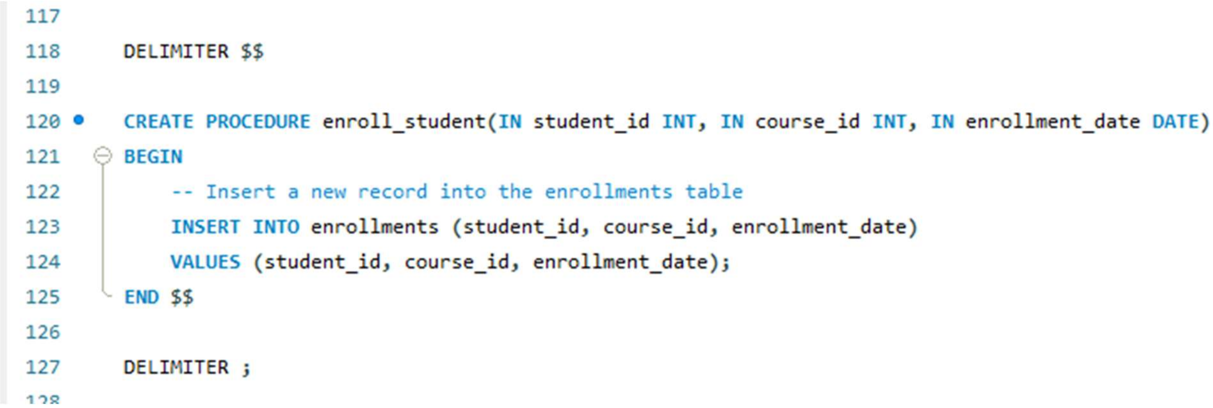


```
DELIMITER $$

CREATE TRIGGER update_student_gpa
AFTER UPDATE ON enrollments
FOR EACH ROW
BEGIN
    DECLARE avg_gpa DECIMAL(4, 2);

    -- Calculate the average grade for the student
    SELECT AVG(grade) INTO avg_gpa
    FROM enrollments
    WHERE student_id = NEW.student_id;

    -- Update the GPA in the students table
    UPDATE students
    SET gpa = avg_gpa
    WHERE student_id = NEW.student_id;
END $$

DELIMITER ;
```

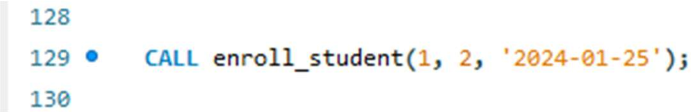
Explanation:

- The trigger is set to run after an update on the enrollments table.
- It calculates the average grade (avg_gpa) for the student whose enrollment was updated.
- After calculating the average, it updates the gpa column in the students table for that student.

Create the stored procedure

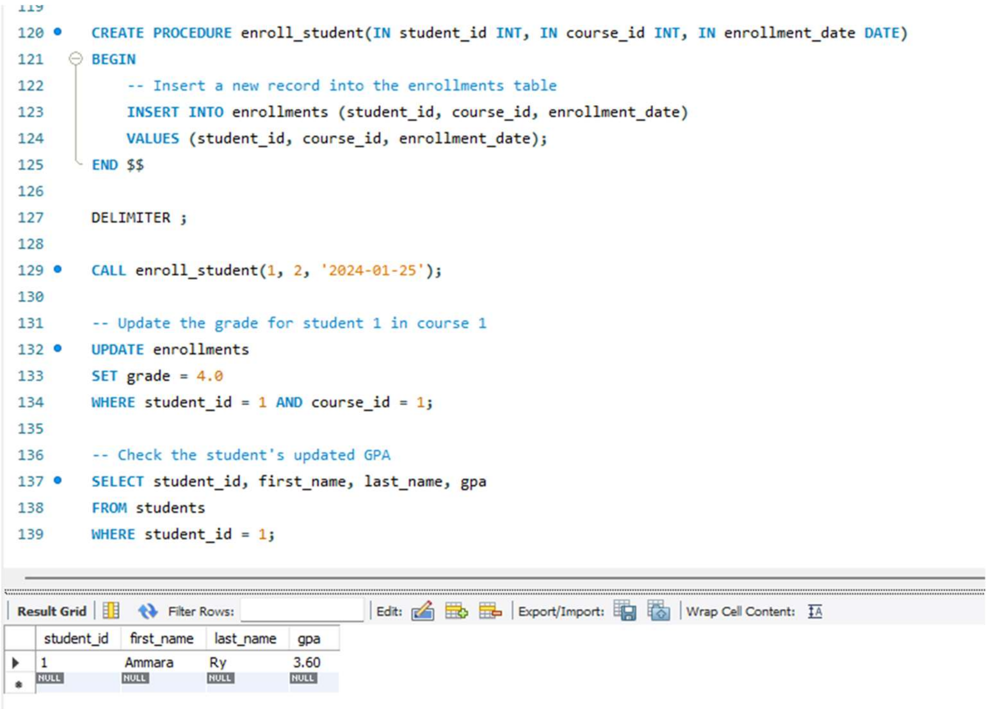


```

DELIMITER $$

CREATE PROCEDURE enroll_student(IN student_id INT, IN course_id INT, IN enrollment_date DATE)
BEGIN
    -- Insert a new record into the enrollments table
    INSERT INTO enrollments (student_id, course_id, enrollment_date)
    VALUES (student_id, course_id, enrollment_date);
END $$

DELIMITER ;
```



```sql

CALL enroll_student(1, 2, '2024-01-25');

```

Result:



```sql
CREATE PROCEDURE enroll_student(IN student_id INT, IN course_id INT, IN enrollment_date DATE)
BEGIN
    -- Insert a new record into the enrollments table
    INSERT INTO enrollments (student_id, course_id, enrollment_date)
    VALUES (student_id, course_id, enrollment_date);
END $$

DELIMITER ;

CALL enroll_student(1, 2, '2024-01-25');

-- Update the grade for student 1 in course 1
UPDATE enrollments
SET grade = 4.0
WHERE student_id = 1 AND course_id = 1;

-- Check the student's updated GPA
SELECT student_id, first_name, last_name, gpa
FROM students
WHERE student_id = 1;
```

student_id	first_name	last_name	gpa
1	Ammara	Ry	3.60
NULL	NULL	NULL	NULL