Даалгавар 8. **ДААЛГАВАР 8 – 9 (100%) 20B1NUM0255** **У. Төрболд**

	employee_id	employee_name	last_updated
1	1	John Doe	2024-03-25
2	2	Jane Smith	2024-03-28
3	3	Michael Johnson	2024-03-27

```
CREATE TRIGGER updateEmployee
on employee
after update
as
begin
    update employee
    set last_updated = GETDATE()
    from employee
    INNER JOIN inserted i ON employee.employee_id = i.employee_id
    where employee.employee_id = i.employee_id
end;

DROP TRIGGER updateEmployee;

UPDATE employee
SET employee_name = 'UPDATED NAME'
WHERE employee_id = 1;

select * from employee
```

esults Messages

employee_id	employee_name	last_updated
1	UPDATED NAME	2024-03-29
2	Jane Smith	2024-03-28
3	Michael Johnson	2024-03-27

Даалгавар 9.

	employee_id	employee_name
1	1	John Doe
2	2	Jane Smith
3	3	Michael Johnson

	timecard_id	employee_id	hours_worked
1	1	1	8.50
2	2	1	7.00
3	3	2	5.50
4	4	2	6.00
5	5	3	7.50

```sql
CREATE PROCEDURE GetEmployeesHours
@EmployeeID INT
AS
BEGIN
    SELECT e.Employee_name,
    (
    Select Sum(hours_worked)
    FROM timecards
    Where employee_id = @EmployeeID
    ) as totalhours
    from employees e
    WHERE e.employee_id = @EmployeeID
End;

EXEC GetEmployeesHours @EmployeeID = 1
EXEC GetEmployeesHours @EmployeeID = 2;
EXEC GetEmployeesHours @EmployeeID = 3;
```

esults | Messages

Employee_name	totalhours
John Doe	15.50

Employee_name	totalhours
Jane Smith	11.50

Employee_name	totalhours
Michael Johnson	7.50

Даалгавар 10

	enrollmentid	studentid	courseid	score
1	1	1	1	85
2	2	1	2	90
3	3	2	1	95
4	4	2	3	88
5	5	3	2	75

	courseid	coursename
1	1	Mathematics
2	2	Science
3	3	History

	student_id	firstname	lastname
1	1	John	Doe
2	2	Jane	Smith
3	3	Michael	Johnson

```sql
CREATE PROCEDURE getAvarageScore
@StudentID INT
AS
BEGIN
    DECLARE @TotalCourse INT
    SELECT @TotalCourse = COUNT(*)
    FROM enrollment
    WHERE studentid = @StudentID

    SELECT s.student_id, s.firstname, s.lastname,
        CASE
            WHEN @TotalCourse = 0 THEN 0
            ELSE
                (
                    SELECT SUM(
                        CASE
                            WHEN e.score >= 90 THEN 4.0
                            WHEN e.score >= 80 THEN 3.0
                            WHEN e.score >= 70 THEN 2.0
                            ELSE 0.0
                        END
                    )
                    FROM enrollment e
                    WHERE e.studentid = s.student_id
                ) / CAST(@TotalCourse AS DECIMAL(5, 2))
        END AS GPA
    FROM student s
    WHERE s.student_id = @StudentID
END;

EXEC getAvarageScore @StudentID = 3;
EXEC getAvarageScore @StudentID = 2;
```

	student_id	firstname	lastname	GPA
1	1	John	Doe	3.500000

	student_id	firstname	lastname	GPA
1	2	Jane	Smith	3.500000

	student_id	firstname	lastname	GPA
1	3	Michael	Johnson	2.000000