**Лабораторна робота № 6**

**Процедури, що зберігаються. Тригери**

Мета роботи: дослідження властивостей та можливостей процедур, що зберігаються та тригерів.

Хід роботи:

Звіт містить:

1. Тему та мету лабораторної роботи
2. Скріпти самостійно створених процедур та тригерів до БД «Рейтинг» та результати їх виконання
3. Скріпти самостіно створених процедур та тригерів до БД за індивідуальним варіантом та результати їх виконання.

Побудувати до БД «Рейтинг» процедури, що зберігаються:

1. Розрахунок середнього балу студентів за період – без урахування перездач.

Лістинг:

```
CREATE PROCEDURE CalculateAverageGradeWithoutRetakes
AS
BEGIN
    SELECT Kod_student, AVG(Reiting) AS AvgGrade
    FROM Reiting
    WHERE Prisutn = 1
    GROUP BY Kod_student;
END
EXEC CalculateAverageGradeWithoutRetakes;
```

| | Kod_student | AvgGrade |
|---|---|---|
| 1 | 7 | 42 |
| 2 | 9 | 89 |
| 3 | 15 | 50 |

Рис. 1 Результат виконання роботи

2. Розрахунок середнього балу студентів за період – з урахування перездач.

Лістинг:

```
CREATE PROCEDURE CalculateAverageGradeWithRetakes
AS
BEGIN
    SELECT Kod_student, AVG(Reiting) AS AvgGrade
    FROM Reiting
    GROUP BY Kod_student;
END
EXEC CalculateAverageGradeWithRetakes
```

| | Kod_student | AvgGrade |
|---|---|---|
| 1 | 7 | 42 |
| 2 | 8 | 39 |
| 3 | 9 | 89 |
| 4 | 12 | 63 |
| 5 | 14 | 70 |
| 6 | 15 | 50 |
| 7 | 16 | 55 |
| 8 | 18 | 83 |
| 9 | 19 | 80 |
| 10 | 21 | 80 |
| 11 | 24 | 72 |
| 12 | 26 | 75 |
| 13 | 27 | 67 |
| 14 | 30 | 48 |
| 15 | 31 | 58 |
| 16 | 33 | 45 |
| 17 | 34 | 64 |
| 18 | 36 | 68 |
| 19 | 37 | 60 |

Рис. 2 Результат виконання роботи

3. Визначення студентів, що навчаються на 4 та 5.

Лістинг:

```
CREATE PROCEDURE GetStudentsWithGrades4And5
AS
BEGIN
    SELECT DISTINCT s.*
    FROM dbo_student s
    INNER JOIN Reiting r ON s.Kod_stud = r.Kod_student
    WHERE r.Reiting >= 74
END
EXEC GetStudentsWithGrades4And5
```

| | Kod_stud | Name_ini | Sname | Name | Fname | N_ingroup | Kod_group |
|---|---|---|---|---|---|---|---|
| 1 | 9 | Кузьменко А.В. | Кузьменко | Андрій | Васильович | 9 | с-53 |
| 2 | 18 | Лук'яненко Ю.А. | Лук'яненко | Юлія | Артемівна | 18 | с-53 |
| 3 | 19 | Міщенко Д.П. | Міщенко | Денис | Петрович | 19 | с-53 |
| 4 | 21 | Семененко І.О. | Семененко | Ігор | Олегович | 2 | с-54 |
| 5 | 26 | Павлюк Л.І. | Павлюк | Людмила | Ігорівна | 7 | с-54 |
| 6 | 38 | Іванова В.Є. | Іванова | Вікторія | Євгенівна | 19 | с-54 |

Рис. 3 Результат виконання роботи

4. Процедура виводить суму балів та її значення в національній системі та ECTS.

Лістинг:

```
CREATE PROCEDURE CalculateTotalGradeValues
    @studentCode INT
AS
BEGIN
    DECLARE @totalPoints INT
    DECLARE @nationalGrade NVARCHAR(10)
    DECLARE @ectsGrade NVARCHAR(10)

    SELECT @totalPoints = SUM(Reiting)
    FROM Reiting
    WHERE Kod_student = @studentCode;

    IF @totalPoints BETWEEN 60 AND 63
    BEGIN
        SET @nationalGrade = '3'
        SET @ectsGrade = 'E'
    END
    ELSE IF @totalPoints BETWEEN 64 AND 73
    BEGIN
        SET @nationalGrade = '3'
        SET @ectsGrade = 'D'
    END
    ELSE IF @totalPoints BETWEEN 74 AND 81
    BEGIN
        SET @nationalGrade = '4'
        SET @ectsGrade = 'C'
    END
    ELSE IF @totalPoints BETWEEN 82 AND 89
    BEGIN
        SET @nationalGrade = '4'
        SET @ectsGrade = 'B'
    END
    ELSE IF @totalPoints BETWEEN 90 AND 100
    BEGIN
        SET @nationalGrade = '5'
        SET @ectsGrade = 'A'
    END
    ELSE IF @totalPoints < 60
    BEGIN
        SET @nationalGrade = '2'
        SET @ectsGrade = 'F'
    END
    ELSE
    BEGIN
        SET @nationalGrade = NULL
        SET @ectsGrade = NULL
    END
    SELECT @totalPoints AS 'TotalPoints', @nationalGrade AS 'NationalGrade', @ectsGrade AS
'ECTSGrade'
END
EXEC CalculateTotalGradeValues @studentCode = 9
```

Results | Messages

| | TotalPoints | NationalGrade | ECTSGrade |
|---|---|---|---|
| 1 | 89 | 4 | B |

Рис. 4 Результат виконання роботи

5. Тригер на вставку даних в таблицю студент – якщо код групи новий в таблицю додається група.

Лістинг:

```
CREATE TRIGGER AddNewGroupOnInsert
ON dbo_student
AFTER INSERT
AS
BEGIN
    INSERT INTO dbo_groups (Kod_group, K_navch_plan)
    SELECT DISTINCT i.Kod_group, 17
    FROM inserted i
    LEFT JOIN dbo_groups g ON i.Kod_group = g.Kod_group
    WHERE g.Kod_group IS NULL;
END
INSERT INTO dbo_student (Sname, Name, FName, N_ingroup, Kod_group)
VALUES ('Додо', 'Валерій', 'Петрович', 1, N'с-58');
```

Results Messages

| | Kod_group | Kod_men | Kod_zhum | K_navch_plan | kilk |
|---|---|---|---|---|---|
| 1 | с-53 | 2 | 3 | 13 | 19 |
| 2 | с-54 | 3 | 2 | 14 | 19 |
| 3 | с-55 | 4 | 3 | 17 | 0 |
| 4 | с-58 | NULL | NULL | 17 | NULL |

Рис. 5 Результат виконання роботи

6. Тригер на модифікацію даних з таблиці студенти якщо більше немає студентів в групі група знищується.

Лістинг:

```
CREATE TRIGGER DeleteEmptyGroupOnUpdate
ON dbo_student
AFTER DELETE
AS
BEGIN
    DELETE FROM dbo_groups
    WHERE Kod_group IN (
        SELECT g.Kod_group
        FROM dbo_groups g
        LEFT JOIN dbo_student s ON g.Kod_group = s.Kod_group
        GROUP BY g.Kod_group
        HAVING COUNT(s.Kod_stud) = 0
    );
END
```

| | Kod_group | Kod_men | Kod_zhum | K_navch_plan | kilk |
|---|---|---|---|---|---|
| 1 | с-53 | 2 | 3 | 13 | 19 |
| 2 | с-54 | 3 | 2 | 14 | 19 |
| 3 | с-55 | 4 | 3 | 17 | 0 |

Рис. 6 Результат виконання роботи

Для БД за індивідуальним завданням реалізувати:

1. Процедури, що зберігаються - не менше 3 (для запитів на пошук обов'язково).

Лістинг:

```
CREATE PROCEDURE FindEmployeeByLastName (@lastName VARCHAR(50))
AS
BEGIN
    SELECT * FROM Employees WHERE Last_Name = @lastName;
END;
EXEC FindEmployeeByLastName @lastName = N'Іванов';
```

Results Messages

| | Employee_ID | Last_Name | First_Name | Middle_Name | Birth_Year | Employment_Date | Position | Salary | Department_ID | Identification_Code |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Іванов | Петро | Олександрович | 1985 | 2020-01-15 | Програміст | 3000.00 | 1 | 12345 |

Рис. 7 Результат виконання роботи

Лістинг:

```
CREATE PROCEDURE FindEmployeesByDepartment (@departmentName VARCHAR(100))
AS
BEGIN
    SELECT * FROM Employees e JOIN Departments d ON e.Department_ID = d.Department_ID WHERE
d.Department_Name = @departmentName;
END;
EXEC FindEmployeesByDepartment @departmentName = N'Відділ розробки';
```

Results Messages

| | Employee_ID | Last_Name | First_Name | Middle_Name | Birth_Year | Employment_Date | Position | Salary | Department_ID | Identification_Code | Department_ID | Department_Name |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Іванов | Петро | Олександрович | 1985 | 2020-01-15 | Програміст | 3000.00 | 1 | 12345 | 1 | Відділ розробки |
| 2 | 5 | Семенов | Олексій | Павлович | 1982 | 2024-04-01 | Програміст | 3200.00 | 1 | 24680 | 1 | Відділ розробки |

Рис. 8 Результат виконання роботи

Лістинг:

```
CREATE PROCEDURE CalculateAverageSalaryByDepartment (@departmentName VARCHAR(100))
AS
BEGIN
    SELECT AVG(Salary) AS AverageSalary FROM Employees e JOIN Departments d ON e.Department_ID
= d.Department_ID WHERE d.Department_Name = @departmentName;
END;
EXEC CalculateAverageSalaryByDepartment @departmentName = N'Відділ маркетингу';
```

Results Messages

| | AverageSalary |
|---|---|
| 1 | 2600.000000 |

Рис. 9 Результат виконання роботи

Лістинг:

```
CREATE TRIGGER AfterInsertEmployee
ON Employees
AFTER INSERT
AS
BEGIN
    PRINT 'Новий працівник був доданий!';
END;
INSERT INTO Employees (Employee_ID, Last_Name, First_Name, Middle_Name, Birth_Year,
Employment_Date, Position, Salary, Department_ID, Identification_Code)
VALUES (8, 'Сидоренко', 'Олександр', 'Іванович', 1990, '2024-05-06', 'Аналітик', 3000.00, 1,
'12345');
```

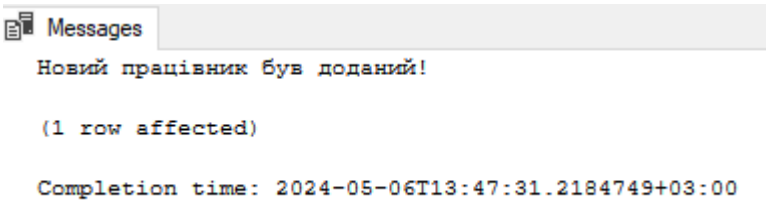


Рис. 10 Результат виконання роботи

Лістинг:

```
CREATE TRIGGER UpdatePositionOnDepartmentChange
ON Employees
AFTER UPDATE
AS
BEGIN
    IF EXISTS (
        SELECT * FROM inserted i
        JOIN deleted d ON i.Employee_ID = d.Employee_ID
        WHERE i.Department_ID = 4 AND d.Department_ID <> 4
    )
    BEGIN
        UPDATE Employees
        SET Position = 'HR менеджер'
        WHERE Employee_ID IN (SELECT Employee_ID FROM inserted);
    END
END;
UPDATE Employees
SET Department_ID = 4
WHERE Employee_ID = 1;
```

| | Employee_ID | Last_Name | First_Name | Middle_Name | Birth_Year | Employment_Date | Position | Salary | Department_ID | Identification_Code |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Іванов | Петро | Олександрович | 1985 | 2020-01-15 | HR менеджер | 3000.00 | 4 | 12345 |

Рис. 11 Результат виконання роботи

Лістинг:

```
CREATE TABLE EmployeeChanges (
    Change_ID INT PRIMARY KEY IDENTITY,
    Employee_ID INT,
    Change_Date DATETIME,
    Changed_Field VARCHAR(100),
    Old_Value VARCHAR(100),
    New_Value VARCHAR(100)
);

CREATE TRIGGER LogEmployeeChanges
ON Employees
AFTER UPDATE
AS
BEGIN
    INSERT INTO EmployeeChanges (Employee_ID, Change_Date, Changed_Field, Old_Value, New_Value)
    SELECT i.Employee_ID, GETDATE(), c.COLUMN_NAME, d.Position, i.Position
    FROM inserted i
    INNER JOIN deleted d ON i.Employee_ID = d.Employee_ID
    CROSS JOIN INFORMATION_SCHEMA.COLUMNS c
    WHERE c.TABLE_NAME = 'Employees' AND c.COLUMN_NAME NOT IN ('Employee_ID',
'Employment_Date')
    AND i.Position <> d.Position;
END;

UPDATE Employees
SET Position = 'Системний адміністратор'
WHERE Employee_ID = 1;

select * from EmployeeChanges
```

Results | Messages

| | Change_ID | Employee_ID | Change_Date | Changed_Field | Old_Value | New_Value |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 2024-05-06 13:57:26.650 | Last_Name | HR менеджер | Системний адміністратор |
| 2 | 2 | 1 | 2024-05-06 13:57:26.650 | First_Name | HR менеджер | Системний адміністратор |
| 3 | 3 | 1 | 2024-05-06 13:57:26.650 | Middle_Name | HR менеджер | Системний адміністратор |
| 4 | 4 | 1 | 2024-05-06 13:57:26.650 | Birth_Year | HR менеджер | Системний адміністратор |
| 5 | 5 | 1 | 2024-05-06 13:57:26.650 | Position | HR менеджер | Системний адміністратор |
| 6 | 6 | 1 | 2024-05-06 13:57:26.650 | Salary | HR менеджер | Системний адміністратор |
| 7 | 7 | 1 | 2024-05-06 13:57:26.650 | Department_ID | HR менеджер | Системний адміністратор |
| 8 | 8 | 1 | 2024-05-06 13:57:26.650 | Identification_Code | HR менеджер | Системний адміністратор |

Рис. 12 Результат виконання роботи

Висновок: дослідив властивості та можливості процедур, що зберігаються та тригерів.