**Лабораторна робота № 4-5**

**ВИКОРИСТАННЯ SELECT. ЗАПИТИ НА ВИБІРКУ ІЗ ГРУПОВИМИ ОПЕРАЦІЯМИ. ПІДЗАПИТИ У МОВІ SQL**

Мета роботи: дослідження властивостей DML-інструкції SELECT для групування даних та відображення проміжних підсумків.

Хід роботи:





SELECT SUM(NaSklade) AS TotalProducts

FROM tovar;

A screenshot of a computer

Description automatically generated

Рис. 1 Результат виконання



SELECT COUNT(\*) AS TotalEmployees

FROM sotrudnik;



Рис. 2 Результат виконання



SELECT COUNT(\*) AS TotalSuppliers

FROM postachalnik;



Рис. 3 Результат виконання



SELECT t.Nazva, SUM(zt.Kilkist) AS PurchasedQuantity

FROM zakaz z

INNER JOIN zakaz\_tovar zt ON z.id\_zakaz = zt.id\_zakaz

INNER JOIN tovar t ON zt.id\_tovar = t.id\_tovar

WHERE CONVERT(DATE, z.date\_rozm) BETWEEN '2017-06-01' AND '2017-06-30'

GROUP BY t.Nazva;

A screenshot of a computer

Description automatically generated

Рис. 4 Результат виконання



SELECT SUM(t.Price \* zt.Kilkist) AS TotalSpent

FROM zakaz z

INNER JOIN zakaz\_tovar zt ON z.id\_zakaz = zt.id\_zakaz

INNER JOIN tovar t ON zt.id\_tovar = t.id\_tovar

WHERE CONVERT(DATE, z.date\_rozm) BETWEEN '2017-06-01' AND '2017-06-30';



Рис. 5 Результат виконання



SELECT p.Nazva, SUM(t.Price \* zt.Kilkist) AS TotalSales

FROM zakaz\_tovar zt

INNER JOIN tovar t ON zt.id\_tovar = t.id\_tovar

INNER JOIN postachalnik p ON t.id\_postav = p.id\_postach

GROUP BY p.Nazva;

A screenshot of a computer

Description automatically generated

Рис. 6 Результат виконання



SELECT p.Nazva, COUNT(z.id\_zakaz) AS TotalOrders

FROM zakaz z

INNER JOIN zakaz\_tovar zt ON z.id\_zakaz = zt.id\_zakaz

INNER JOIN tovar t ON zt.id\_tovar = t.id\_tovar

INNER JOIN postachalnik p ON t.id\_postav = p.id\_postach

WHERE t.Nazva = 'Молоко'

GROUP BY p.Nazva;

A close up of a text

Description automatically generated

Рис. 7 Результат виконання



SELECT AVG(TotalSpent) AS AverageOrderAmount

FROM (

SELECT SUM(t.Price \* zt.Kilkist) AS TotalSpent

FROM zakaz\_tovar zt

INNER JOIN tovar t ON zt.id\_tovar = t.id\_tovar

GROUP BY zt.id\_zakaz

) AS OrderTotals;



Рис. 8 Результат виконання



SELECT SUM(t.Price \* zt.Kilkist) AS TotalOrderValue

FROM zakaz z

INNER JOIN zakaz\_tovar zt ON z.id\_zakaz = zt.id\_zakaz

INNER JOIN tovar t ON zt.id\_tovar = t.id\_tovar

INNER JOIN klient k ON z.id\_klient = k.id\_klient

WHERE k.City = 'Житомир';



Рис. 9 Результат виконання



SELECT p.Nazva, AVG(t.Price) AS AveragePrice

FROM tovar t

INNER JOIN postachalnik p ON t.id\_postav = p.id\_postach

GROUP BY p.Nazva;

A screenshot of a computer

Description automatically generated

Рис. 10 Результат виконання





SELECT Kod\_student, SUM(Reiting) AS Total\_Rating

FROM Reiting

GROUP BY Kod\_student;

A screenshot of a table

Description automatically generated

Рис. 11 Результат виконання



SELECT Kod\_group, COUNT(Kod\_stud) AS Students\_Count

FROM dbo\_student

GROUP BY Kod\_group;

A group of students and students

Description automatically generated

Рис. 12 Результат виконання



SELECT Kod\_group, COUNT(\*) AS Discipline\_Count

FROM dbo\_groups

GROUP BY Kod\_group;

A white grid with black text

Description automatically generated

Рис. 13 Результат виконання



SELECT Kod\_group, COUNT(\*) AS Classes\_Count

FROM Rozklad\_pids

GROUP BY Kod\_group;

A group of numbers and a square

Description automatically generated with medium confidence

Рис. 14 Результат виконання



SELECT Kod\_group, AVG(Reiting) AS Avg\_Rating

FROM Reiting r

JOIN dbo\_student s ON r.Kod\_student = s.Kod\_stud

GROUP BY Kod\_group;

A group of numbers and a group of people

Description automatically generated with medium confidence

Рис. 15 Результат виконання



SELECT pp.K\_predmet, AVG(r.Reiting) AS Avg\_Rating

FROM Reiting r

JOIN Predmet\_plan pp ON pp.K\_predmet = pp.K\_predmet

GROUP BY pp.K\_predmet;

A screenshot of a computer

Description automatically generated

Рис. 16 Результат виконання



SELECT r.Kod\_student, pp.K\_predmet, r.Reiting AS Current\_Rating

FROM Reiting r

JOIN Predmet\_plan pp ON pp.K\_predmet = pp.K\_predmet;

A screenshot of a data

Description automatically generated

Рис. 17 Результат виконання



SELECT pp.K\_predmet, MIN(r.Reiting) AS Min\_Rating

FROM Reiting r

JOIN Predmet\_plan pp ON pp.K\_predmet = pp.K\_predmet

GROUP BY pp.K\_predmet;

A screenshot of a computer

Description automatically generated

Рис. 18 Результат виконання



SELECT pp.K\_predmet, MAX(r.Reiting) AS Min\_Rating

FROM Reiting r

JOIN Predmet\_plan pp ON pp.K\_predmet = pp.K\_predmet

GROUP BY pp.K\_predmet;

A screenshot of a data

Description automatically generated

Рис. 19 Результат виконання



SELECT K\_predm\_pl, COUNT(Zdacha\_type) AS Classes\_Count\_By\_Type

FROM Rozklad\_pids

GROUP BY K\_predm\_pl, Zdacha\_type;

A screenshot of a computer

Description automatically generated

Рис. 20 Результат виконання



SELECT K\_spets, COUNT(Kod\_group) AS Groups\_Count

FROM Navch\_plan

JOIN dbo\_groups ON Navch\_plan.K\_navch\_plan = dbo\_groups.K\_navch\_plan

GROUP BY K\_spets;

A white grid with black text and a white rectangle

Description automatically generated

Рис. 21 Результат виконання



DELETE FROM Reiting

WHERE Kod\_student IN (SELECT Kod\_stud FROM dbo\_student WHERE Sname = 'Vasiuta');

A screenshot of a table

Description automatically generated

Рис. 22 Результат до видалення

select \* from Reiting

WHERE Kod\_student = 109

A close up of a text

Description automatically generated

Рис. 23 Результат після видалення



A table of numbers and a few words

Description automatically generated with medium confidence

Рис. 24 До видалення

DELETE FROM Predmet\_plan

WHERE K\_predmet in (SELECT K\_predmet FROM predmet WHERE Nazva = 'DataBase');

A table of numbers and letters

Description automatically generated with medium confidence

Рис. 25 Після видалення



UPDATE Reiting

SET Reiting = Reiting \* 1.15

A screenshot of a table

Description automatically generated

Рис. 26 Результат виконання



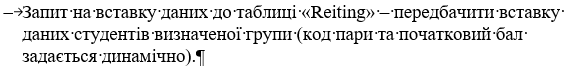
UPDATE Reiting

SET Reiting = Reiting \* 0.85

A screenshot of a table

Description automatically generated

Рис. 27 Результат виконання



DECLARE @K\_group varchar(10) = 'ϲ-53';

DECLARE @Reiting smallint = 76;

INSERT INTO Reiting (K\_zapis, Kod\_student, Reiting, Prisutn)

SELECT 36, s.Kod\_stud, @Reiting, 2

FROM dbo\_student s

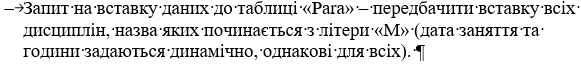
JOIN dbo\_groups g ON s.Kod\_group = g.Kod\_group

WHERE g.Kod\_group = @K\_group;

A table of numbers with black text

Description automatically generated

Рис. 28 Результат виконання



INSERT INTO Predmet\_plan(K\_predmet, K\_navch\_plan, Chas\_Lek, Cahs\_pr, Chas\_all, Chas\_Labor, Chas\_sem, Kilk\_modul, Cahs\_sam, Semestr, Shifr\_predmet, k\_fk)

SELECT p.K\_predmet, 17, @ALLHOURS / 2, @ALLHOURS / 2, @ALLHOURS, @ALLHOURS / 2, @ALLHOURS / 2, 5, null, 1, null, 4

FROM predmet p

WHERE Nazva LIKE 'M%';

A screenshot of a computer

Description automatically generated

Рис. 29 Результат виконання



UPDATE Predmet\_plan

SET Kilk\_modul = 5

WHERE K\_predmet = 388



Рис. 30 Результат виконання



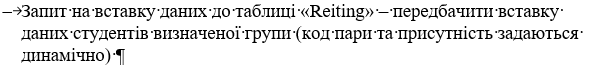
DELETE FROM dbo\_student

WHERE Kod\_group = 'ϲ-54'

A screenshot of a data

Description automatically generated

Рис. 31 Результат виконання



DECLARE @Kod\_group varchar(10) = 'ϲ-54'

DECLARE @Prisutn int = 0

INSERT INTO Reiting(K\_zapis, Kod\_student, Reiting, Prisutn)

SELECT 36, s.Kod\_stud, 66, @Prisutn

FROM dbo\_student s

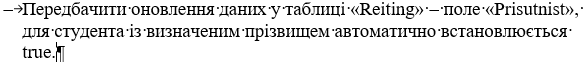
JOIN dbo\_groups g ON s.Kod\_group = g.Kod\_group

WHERE g.Kod\_group = @Kod\_group;

A white background with numbers

Description automatically generated

Рис. 32 Результат виконання



UPDATE Reiting

SET Prisutn = 1

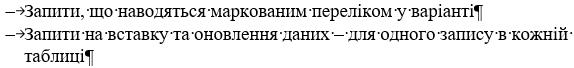
WHERE Kod\_student IN (SELECT Kod\_stud FROM dbo\_student WHERE Sname = 'Vasiuta');

A screenshot of a cell phone

Description automatically generated

Рис. 33 Результат виконання





INSERT INTO Departments (Department\_ID, Department\_Name) VALUES (4, 'Відділ кадрів');

INSERT INTO Employees (Employee\_ID, Last\_Name, First\_Name, Middle\_Name, Birth\_Year, Employment\_Date, Position, Salary, Department\_ID, Identification\_Code)

VALUES (4, 'Ковальчук', 'Марія', 'Василівна', 1987, '2023-06-12', 'HR менеджер', 2800.00, 4, '13579');

INSERT INTO Sick\_Leaves (Sick\_Leave\_ID, Employee\_ID, Start\_Date, End\_Date) VALUES (3, 1, '2024-04-01', '2024-04-05');

INSERT INTO Business\_Trips (Trip\_ID, Employee\_ID, Start\_Date, End\_Date) VALUES (3, 1, '2024-05-10', '2024-05-15');

UPDATE Departments SET Department\_Name = 'Відділ персоналу' WHERE Department\_ID = 4;

UPDATE Employees SET Position = 'HR спеціаліст' WHERE Employee\_ID = 4;

UPDATE Sick\_Leaves SET End\_Date = '2024-04-10' WHERE Sick\_Leave\_ID = 3;

UPDATE Business\_Trips SET End\_Date = '2024-05-20' WHERE Trip\_ID = 3;

A screenshot of a computer

Description automatically generated

Рис. 34 Результат виконання



INSERT INTO Employees (Employee\_ID, Last\_Name, First\_Name, Middle\_Name, Birth\_Year, Employment\_Date, Position, Salary, Department\_ID, Identification\_Code)

VALUES

(5, 'Семенов', 'Олексій', 'Павлович', 1982, '2024-04-01', 'Програміст', 3200.00, 1, '24680'),

(6, 'Ковальов', 'Іван', 'Олегович', 1995, '2024-03-15', 'Маркетолог', 2700.00, 2, '35791');

UPDATE Sick\_Leaves

SET End\_Date = '2024-04-15'

WHERE Employee\_ID IN (1, 2);

UPDATE Business\_Trips

SET End\_Date = '2024-05-25'

WHERE Employee\_ID = 2;

A screenshot of a computer

Description automatically generated

Рис. 35 Результат виконання



DECLARE @DepartmentID INT = 3;

DECLARE @BirthYearThreshold INT = 1990;

SELECT \*

FROM Employees

WHERE Department\_ID = @DepartmentID AND Birth\_Year < @BirthYearThreshold;

DECLARE @SalaryThreshold DECIMAL(10, 2) = 2500.00;

DECLARE @PositionToSearch VARCHAR(100) = 'Програміст';

SELECT \*

FROM Employees

WHERE Salary > @SalaryThreshold AND Position = @PositionToSearch;

DECLARE @StartDateFrom DATE = '2024-03-01';

DECLARE @StartDateTo DATE = '2024-03-31';

SELECT \*

FROM Sick\_Leaves

WHERE Start\_Date BETWEEN @StartDateFrom AND @StartDateTo;

DECLARE @EmployeeCountThreshold INT = 2;

SELECT Departments.Department\_Name, COUNT(\*) AS Employee\_Count

FROM Departments

JOIN Employees ON Departments.Department\_ID = Employees.Department\_ID

GROUP BY Departments.Department\_Name

HAVING COUNT(\*) < @EmployeeCountThreshold;

A screenshot of a computer

Description automatically generated

Рис. 36 Результат виконання

Висновок: дослідив властивості DML-інструкції SELECT для групування даних та відображення проміжних підсумків.