

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Український державний університет науки і технологій**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №6  
з дисципліни «Бази даних»**

**на тему: «Обертання даних. Модифікація даних»**

Виконав:

Студент гр. ПЗ2011

Кулик С.В.

Прийняв:

Іванов О.П.

Дніпро, 2023

**Тема.** **Обертання даних. Модифікація даних.**

**Мета.** Отримати навички написання запитів SQL, що реалізують обертання даних.

Ознайомитись з операціями модифікації даних.

**Скріншот діаграми бази даних із середовища SQL Server Management Studio**



**Файл “** **AdvancedQuerying.sql”**

use CinemaDB

go

-- 1.1.1. Ranking Window Functions

SELECT

id,

title,

ROW\_NUMBER() OVER (ORDER BY release\_date) AS row\_num,

RANK() OVER (ORDER BY release\_date) AS rank\_num,

DENSE\_RANK() OVER (ORDER BY release\_date) AS dense\_rank\_num,

NTILE(4) OVER (ORDER BY release\_date) AS ntile\_num

FROM

dbo.Film;

-- 1.1.2. Offset Window Functions

SELECT

id,

title,

LAG(release\_date) OVER (ORDER BY release\_date) AS previous\_release\_date,

LEAD(release\_date) OVER (ORDER BY release\_date) AS next\_release\_date,

FIRST\_VALUE(release\_date) OVER (ORDER BY release\_date) AS first\_release\_date,

LAST\_VALUE(release\_date) OVER (ORDER BY release\_date) AS last\_release\_date

FROM

dbo.Film;

-- 1.1.3. Aggregate Window Functions

SELECT

id,

title,

AVG(duration) OVER (PARTITION BY genre) AS avg\_duration\_per\_genre,

SUM(duration) OVER (PARTITION BY genre) AS total\_duration\_per\_genre

FROM

dbo.Film;

-- 1.2. PIVOT

SELECT \*

FROM (

SELECT genre, country, COUNT(\*) AS film\_count

FROM dbo.Film

GROUP BY genre, country

) AS SourceTable

PIVOT (

SUM(film\_count) FOR country IN ([United States], [New Zealand], [Other])

) AS PivotTable;

-- 1.3. UNPIVOT

SELECT genre, country, film\_count

FROM (

SELECT \*

FROM (

VALUES

('Drama', 100, 120),

('Action', 80, 90),

('Thriller', 50, 60)

) AS GenreTable(genre, [United States], [New Zealand])

) AS SourceTable

UNPIVOT (

film\_count FOR country IN ([United States], [New Zealand])

) AS UnpivotTable;

-- 1.4. GROUPING SETS

SELECT genre, country, COUNT(\*) AS film\_count

FROM dbo.Film

GROUP BY GROUPING SETS ((genre, country), (genre), (country), ());

-- 1.5. CUBE

SELECT genre, country, COUNT(\*) AS film\_count

FROM dbo.Film

GROUP BY CUBE (genre, country);

-- 1.6. ROLLUP

SELECT genre, country, COUNT(\*) AS film\_count

FROM dbo.Film

GROUP BY ROLLUP (genre, country);

-- 1.7. GROUPING()

SELECT

CASE

WHEN GROUPING(genre) = 1 THEN 'All Genres'

ELSE genre

END AS genre,

CASE

WHEN GROUPING(country) = 1 THEN 'All Countries'

ELSE country

END AS country,

COUNT(\*) AS film\_count

FROM dbo.Film

GROUP BY ROLLUP (genre, country);

-- 1.8. GROUPING\_ID()

SELECT

GROUPING\_ID(genre, country) AS group\_id,

CASE

WHEN GROUPING(genre) = 1 THEN 'All Genres'

ELSE genre

END AS genre,

CASE

WHEN GROUPING(country) = 1 THEN 'All Countries'

ELSE country

END AS country,

COUNT(\*) AS film\_count

FROM dbo.Film

GROUP BY ROLLUP (genre, country);

-- 1.9.1. INSERT VALUES

INSERT INTO dbo.Film (title, genre, duration, release\_date, country)

VALUES

('Interstellar', 'Science Fiction', 169, '2014-11-07', 'United States'),

('The Avengers', 'Action', 143, '2012-05-04', 'United States');

-- 1.9.2. INSERT SELECT

INSERT INTO s.Review (comment, rating, film\_id, users\_id)

SELECT

'A mind-bending film with stunning visuals and great performances',

9,

id,

5

FROM dbo.Film

WHERE genre = 'Science Fiction';

-- 1.9.4. SELECT INTO

SELECT

title,

genre,

duration,

release\_date,

country

INTO dbo.NewFilms

FROM dbo.Film

WHERE release\_date > '2022-01-01';

-- 1.10. $identity, @@identity, SCOPE\_IDENTITY(), IDENT\_CURRENT(), IDENTITY\_INSERT

-- Вставка даних у таблицю з ідентичністю (IDENTITY)

SET IDENTITY\_INSERT dbo.Film ON;

INSERT INTO dbo.Film (id, title, genre, duration, release\_date, country)

VALUES

(100, 'New Film', 'Drama', 120, '2023-01-01', 'United States');

SET IDENTITY\_INSERT dbo.Film OFF;

-- Отримання значення поточного identity

SELECT @@IDENTITY AS 'LastInsertedID';

-- Отримання значення identity для поточного обсягу

SELECT SCOPE\_IDENTITY() AS 'ScopeIdentity';

-- Отримання значення identity для конкретної таблиці

SELECT IDENT\_CURRENT('dbo.Film') AS 'IdentCurrent';

-- 1.11. CREATE SEQUENCE (MINVALUE, MAXVALUE, START WITH, INCREMENET BY, CACHE, NO CACHE, NEXT VALUE FOR)

-- Створення послідовності

CREATE SEQUENCE dbo.OrderNumbers

START WITH 1

INCREMENT BY 1

MINVALUE 1

MAXVALUE 1000

NO CACHE;

-- Використання послідовності для вставки значень

INSERT INTO dbo.Film (title, genre, duration, release\_date, country)

VALUES ('New Film', 'Drama', 120, '2023-01-01', 'United States');

-- Перегляд властивостей послідовності

SELECT \* FROM sys.sequences WHERE name = 'OrderNumbers';

-- Видалення залу з id = 5

DELETE FROM dbo.Hall WHERE id = 5;

-- Очищення таблиці Ticket

TRUNCATE TABLE s.Ticket;

-- Оновлення даних у таблиці Users

UPDATE dbo.Users SET password = 'newpassword' WHERE id = 1;

--1.17. INSERT ... OUTPUT

DECLARE @InsertedData TABLE (id INT, title NVARCHAR(100))

INSERT INTO dbo.Film (title, genre, duration, release\_date, country)

OUTPUT inserted.id, inserted.title INTO @InsertedData

VALUES ('New Movie', 'Drama', 120, '2023-01-01', 'United States')

SELECT \* FROM @InsertedData

--1.19. UPDATE ... OUTPUT

DECLARE @UpdatedData TABLE (id INT, old\_title NVARCHAR(100), new\_title NVARCHAR(100))

UPDATE dbo.Film

SET title = 'Updated Title'

OUTPUT deleted.id, deleted.title, inserted.title INTO @UpdatedData

WHERE genre = 'Drama'

SELECT \* FROM @UpdatedData

**Висновок**

У ході лабораторної роботи з обертання та модифікації даних в базі "Film" було успішно використано віконні функції, групування, та різноманітні операції вставки, оновлення, видалення та синхронізації. Оператори OUTPUT дозволили отримати та вивести змінені дані. Отримано практичні навички у написанні ефективних SQL-запитів, що є важливим у розробці оптимізованих операцій з базами даних.