

Тема лекції 5:

Формування складних SQL-запитів

- ❑ Операції з'єднання таблиць
 - ❑ Використання теоретико-множинних операцій
-

Підзапити чи з'єднання таблиць ?

- ❑ Зв'язані підзапити подібні до з'єднань таблиць.
 - ❑ Ці два засоби SQL включають перевірку кожного рядка однієї таблиці з кожним рядком другої таблиці або псевдонімом.
 - ❑ Більшість операцій працює з використанням одного або другого засобу SQL.
 - ❑ Однак є і відмінності між ними:
 - Підзапити, наприклад, можуть використовувати агрегатну функцію у предикаті.
 - З'єднання таблиць можуть виводити рядки з обох таблиць, в той час коли результат підзапиту використовується лише у предикаті зовнішнього запиту.
 - ❑ Вибір засобу визначається поставленим завданням.
-

Операції з'єднання таблиць

- ❑ Операції з'єднання наборів записів повертають таблиці, записи в яких отримуються шляхом деякої комбінації записів таблиць, що з'єднуються.
 - ❑ Для цього використовується оператор JOIN (з'єднати).
 - ❑ Існує декілька видів з'єднання, яким відповідають певні ключові слова, які додаються до оператора JOIN.
-

Внутрішнє з'єднання (INNER JOIN)

- ❑ NATURAL JOIN - природне з'єднання
 - ❑ JOIN...ON - з'єднання за умовою
 - ❑ JOIN...USING - з'єднання за іменами стовпців
-

Природне з'єднання (NATURAL JOIN)

❑ Перевіряються на рівність усі одноіменні стовпці таблиці (з'єднання за рівністю).

❑ Синтаксис:

```
SELECT Таблиця1.*, Таблиця2.*  
FROM Таблиця1 NATURAL JOIN  
Таблиця2;
```

Умовне з'єднання (JOIN...ON), з'єднання за іменами стовпців (JOIN...USING)

- ❑ JOIN...ON використовується, коли потрібно з'єднати за іншими логічними умовами, не обов'язково за рівністю
 - ❑ JOIN...USING подібне на природне з'єднання. Відмінність полягає в тому, що можна вказати, які саме одноіменні стовпці повинні перевірятись.
-

Внутрішнє з'єднання (INNER JOIN)

- Приклад 1. Вивести імена продавців і замовників з однакових міст

- ```
SELECT Salers.sname, Salers.city,
Customers.cname
FROM Salers, Customers
WHERE Salers.city=Customers.city;
```

- ```
SELECT sname, Salers.city, cname  
FROM Salers INNER JOIN Customers  
ON Salers.city=Customers.city;
```

Внутрішнє з'єднання (INNER JOIN)

- Приклад 2. Вивести імена продавців, які обслуговують кожного замовника
 - `SELECT Customers.cname,Salers.sname
FROM Customers,Salers
WHERE Salers.snum=Customers.snum`
 - `SELECT Customers.cname,Salers.sname
FROM Customers JOIN Salers
ON Salers.snum=Customers.snum;`
-

Внутрішнє з'єднання (INNER JOIN) трьох таблиць

- Приклад 3.1. Знайти усі операції купівлі-продажу, в яких брали участь замовники, які знаходяться у інших містах, ніж продавці, які їх обслуговували

 - ```
SELECT onum,amt,odate,cname,sname
FROM Salers,Customers,Orders
WHERE Customers.city<>Salers.city
AND Orders.cnum=Customers.cnum
AND Orders.snum=Salers.snum;
```
-

# Внутрішнє з'єднання (INNER JOIN) трьох таблиць

---

- Приклад 3.2. Знайти усі операції купівлі-продажу, в яких брали участь замовники, які знаходяться у інших містах, ніж продавці, які їх обслуговували
  
  - ```
SELECT onum,amt,odate,cname,sname  
FROM Customers JOIN Orders  
ON Customers.cnum=Orders.cnum  
JOIN Salers  
ON Orders.snum=Salers.snum  
WHERE Customers.city<>Salers.city;
```
-

З'єднання таблиці з собою

- Приклад 4. Знайти усі пари замовників, які мають однаковий рейтинг

- Рішення 1 (з дублюванням):

```
SELECT a.cname, b.cname, a.rating  
FROM Customers a JOIN Customers b  
ON a.rating=b.rating;
```

З'єднання таблиці з собою

- Рішення 2 (з дублюванням):

```
SELECT a.cname, b.cname, a.rating  
FROM Customers a JOIN Customers b  
USING (a.rating,b.rating);
```

З'єднання таблиці з собою

- Рішення 3 (без дублювання):

```
SELECT a.cname, b.cname, a.rating  
      FROM Customers a, Customers b  
     WHERE a.rating=b.rating  
           AND a.cname<b.cname;
```

Або

```
SELECT a.cname, b.cname, a.rating  
      FROM Customers a JOIN Customers b  
     ON a.rating=b.rating  
     WHERE a.cname<b.cname;
```

Перехресне з'єднання (CROSS JOIN)

- ❑ Відповідає операції розширеного декартового добутку, тобто операції з'єднання двох таблиць, при якому кожний запис першої таблиці з'єднується з кожним записом другої таблиці.
- ❑ У деяких СУБД не використовується

```
SELECT <список стовпців>  
FROM Таблиця1, Таблиця2;
```

Або

```
SELECT <список стовпців>  
FROM Таблиця1 CROSS JOIN Таблиця2;
```

Відмінність внутрішнього і зовнішнього з'єднання

- ❑ З таблиці при внутрішньому з'єднанні відкидаються усі записи, для яких немає відповідних записів одночасно в обох таблицях.
 - ❑ При зовнішньому з'єднанні такі невідповідні записи повинні залишатись.
-

Зовнішні з'єднання

- ❑ LEFT OUTER JOIN – ліве з'єднання
 - ❑ RIGHT OUTER JOIN – праве з'єднання
 - ❑ FULL JOIN – повне з'єднання
 - ❑ UNION JOIN – об'єднане з'єднання
-

Ліве з'єднання (LEFT OUTER JOIN)

- При лівому зовнішньому з'єднанні невідповідні записи, які є в лівій таблиці (від оператора JOIN), зберігаються в результатній таблиці, а ті, які є в правій таблиці – відкидаються.

Results		Messages		
	vyd_tovaru	Cina	Data_pokupky	Kilkist
1	Джинси	600,00	2010-03-26 00:00:00.000	3
2	Штани	300,00	2010-03-20 00:00:00.000	1
3	Штани	400,00	2010-03-23 00:00:00.000	2
4	Сукні	1500,00	NULL	NULL
5	Футболки	200,00	NULL	NULL
6	Костюм	1500,00	2010-03-26 00:00:00.000	2
7	Джинси	500,00	NULL	NULL
8	Штани	200,00	2010-03-20 00:00:00.000	1
9	Мешти	600,00	NULL	NULL
10	Сорочка	200,00	2010-03-23 00:00:00.000	2
11	Кросівки	300,00	2010-03-30 00:00:00.000	1
12	Футболки	150,00	2010-03-28 00:00:00.000	4

Ліве з'єднання (LEFT OUTER JOIN)

□ Приклад 5. Схема БД:

- Proposition (ID_tov, cina, opys)
- Sklad(ID_tov, kilk).

При цьому в таблиці Sklad можуть міститись не всі товари, які пропонуються для продажу.

Ліве з'єднання (LEFT OUTER JOIN)

- Приклад 5 (продовж.) Виконати наступний запит: отримати список усіх товарів, які продаються, з вказанням їх кількості на складі:

```
SELECT Pr.ID_tov, Pr.opys, Sklad.kilk  
FROM Proposition Pr LEFT JOIN Sklad  
ON Pr.ID_tov=Sklad.ID_tov;
```

- В результатній таблиці у стовпці `kilk` будуть пусті (тобто `NULL`) значення тих товарів, яких нема на складі.
-

Ліве з'єднання (LEFT OUTER JOIN)

- Приклад 5 (продовж.). Альтернатива оператору LEFT JOIN:

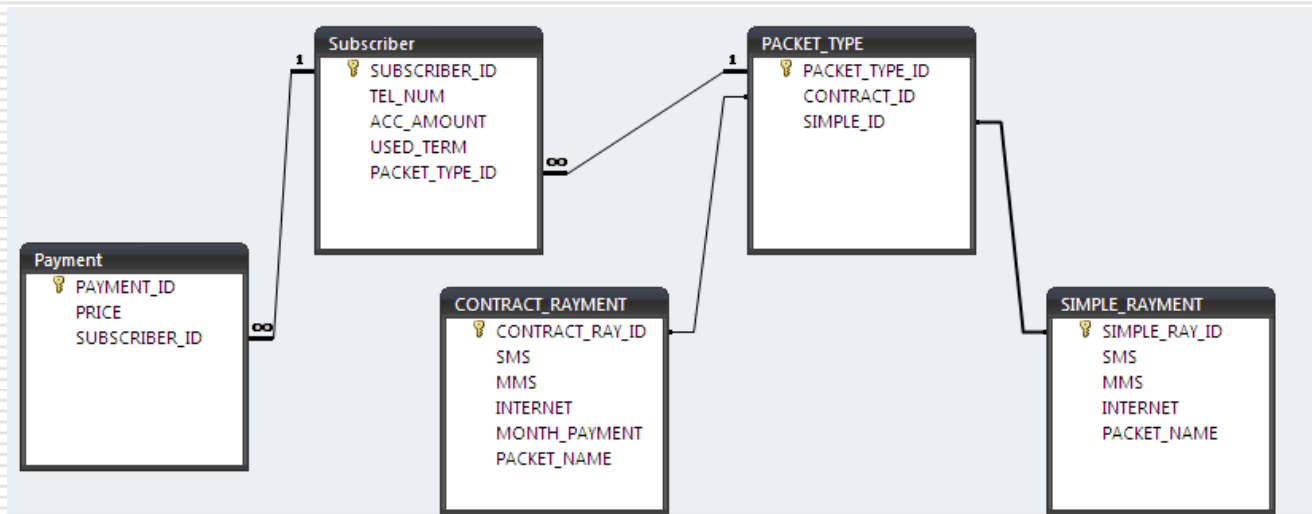
```
SELECT Pr.ID_tov, Pr.opys, Sklad.kilk  
FROM Proposition Pr, Sklad  
WHERE Pr.ID_tov=Sklad.ID_tov
```

```
UNION
```

```
SELECT Pr.ID_tov, Pr.opys, NULL  
FROM Proposition Pr  
WHERE Pr.ID_tov NOT IN  
      (SELECT Pr.ID_tov FROM Proposition Pr,  
                                                Sklad  
      WHERE Pr.ID_tov=Sklad.ID_tov)
```

Ліве з'єднання (LEFT OUTER JOIN)

□ Приклад 6. Діаграма БД:



□ Вивести номери телефонів та вартість платежів абонентів

Ліве з'єднання (LEFT OUTER JOIN)

□ Приклад 6 (продовж.)

```
SELECT Subscriber.Tel_num as Tel_Num,  
       Payment.Price as Payment_Price  
FROM Subscriber LEFT JOIN Payment ON  
       Subscriber.Subscriber_id =  
       Payment.Subscriber_id;
```

	Tel_Num	Payment_Price
1	0213458912	30.00
2	0214563290	15.00
3	0217452354	60.00
4	0217837378	5.00
5	0213452671	100.00
6	0213456732	NULL

Праве з'єднання (RIGHT OUTER JOIN)

- При правому зовнішньому з'єднанні невідповідні записи, які є у правій таблиці, зберігаються в результатній таблиці, а ті, які є в лівій таблиці – відкидаються
-

Праве з'єднання (RIGHT OUTER JOIN)

□ Приклад 7. Діаграма БД – попередня.

Запит: вивести номер телефону та назву пакету абонентів

```
SELECT Subscriber.Tel_num, Packet.Name  
FROM Subscriber RIGHT JOIN Packet ON  
Packet.Packet_id=Subscriber.Packet_id;
```

	TEL_NUM	NAME
1	0213458912	класичний
2	0217837378	класичний
3	0217452354	найкращий
4	0213452671	найкращий
5	0214563290	бізнес
6	0213456732	бізнес
7	NULL	універсальний

Повне з'єднання (FULL JOIN)

□ Приклад 8. Діаграма БД – попередня.

Запит: вивести номер телефону та щомісячну плату для контрактних абонентів

```
SELECT Subscriber.Tel_num, Rayment.Month_pay  
FROM Packet INNER JOIN Rayment ON  
    Rayment.Rayment_id = Packet.Rayment_id  
    AND Packet.Type = 'contract'  
FULL JOIN Subscriber  
ON Subscriber.Packet_id = Packet.Packet_id;
```

	TEL_NUM	MONTH_PAY
1	0213458912	NULL
2	0214563290	100.00
3	0217452354	NULL
4	0217837378	NULL
5	0213452671	NULL
6	0213456732	100.00
7	NULL	150.00

Повне з'єднання (FULL JOIN)

- Приклад 9. База даних деякої компанії містить відомості про свої представництва, відділи та співробітників. Схема БД :
 - Predstavn(ID_pr, adress),
 - Viddil(ID_vid, ID_pr, nazva),
 - Spivrob(ID_spivr, ID_vid, name).
-

Повне з'єднання (FULL JOIN)

-
- Приклад 9 (продовж.) Переглянути усі представництва, відділи і усіх співробітників, незалежно від того, чи є в одних таблицях відповідні записи з інших:

```
SELECT *  
FROM Predstavn Pr FULL JOIN Viddil  
ON Pr.ID_pr=Viddil.ID_pr  
FULL JOIN Spivrob Sp  
ON Viddil.ID_vid= Sp.ID_vid;
```

Об'єднане з'єднання (UNION JOIN)

- ❑ При об'єднаному з'єднанні створюється віртуальна таблиця, яка містить усі стовпці двох вихідних таблиць.
 - ❑ При цьому стовпці з лівої вихідної таблиці містять усі свої записи, а в тих же записах в стовпцях з правої таблиці містяться значення NULL.
 - ❑ Аналогічно, стовпці з правої таблиці містять усі свої записи, а ці ж записи з лівої таблиці містять NULL.
 - ❑ Загальна кількість записів, які містяться в результатній таблиці рівна сумі кількості записів, які є в обох вихідних таблицях.
 - ❑ Як правило, результат об'єднаного з'єднання розглядається в якості проміжного при виконанні більш складного запиту.
-

Об'єднане з'єднання (UNION JOIN)

```
SELECT *  
FROM T1  
UNION JOIN T2;
```

T1

A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	c3
a4	b4	c4

T2

X	Y
x1	y1
x2	y2
x3	y3

Об'єднане з'єднання T1 та T2

A	B	C	X	Y
a1	b1	c1		
a2	b2	c2		
a3	b3	c3		
a4	b4	c4		
			x1	y1
			x2	y2
			x3	y3

Теоретико-множинні операції в SQL

- ❑ UNION – об'єднання наборів записів
 - ❑ INTERSECT – перетин наборів записів
 - ❑ EXCEPT – віднімання наборів записів
-

Об'єднання наборів записів (UNION)

Запит1 UNION Запит2;

- ❑ До набору рядків, які виводяться одним запитом додаються рядки, які виводяться другим запитом
 - ❑ Об'єднуються рядки двох чи більше таблиць з подібними структурами в одну таблицю
 - ❑ При об'єднанні в результатній таблиці залишаються лише різні рядки.
 - ❑ Щоб зберегти в результатній таблиці усі рядки, необхідно написати UNION ALL.
 - ❑ Коли використовується об'єднання більше ніж двох запитів, можна використовувати дужки для визначення порядку запитів.
-

Об'єднання наборів записів (UNION)

□ Приклад :

```
SELECT R.a1, R.a2 FROM R  
UNION  
SELECT S.b2, S.b1 FROM S ;
```

- Об'єднання наборів записів двох таблиць R і S є результатна таблиця, яка містить їх конкатенацію за виключенням кортежів-дублікатів
-

Правила використання оператора UNION

При об'єднанні рядків двох таблиць їх стовпці повинні бути сумісними.

- ❑ Це означає, що кожний запит повинен вказувати однакову кількість стовпців і в однаковому порядку.
- ❑ Типи полів повинні бути теж сумісні.
- ❑ Однак, імена відповідних стовпців та їх розміри можуть бути різними.
- ❑ Щоб об'єднати рядки таблиць з несумісними (по типу даних) стовпцями, необхідно застосувати функцію перетворення типу даних CAST().

Упорядкування об'єднання наборів записів

- ❑ Використовується фраза:
ORDER BY <номер стовпця>
 - ❑ Можна упорядковувати об'єднання за декількома полями, одне всередині іншого, і вказати ASC або DESC для кожного.
-

Перетин наборів записів (INTERSECT)

Запит1 INTERSECT Запит2;

- ❑ Перетин наборів рядків повертає таблицю, рядки якої містяться одночасно у двох наборах
 - ❑ При перетині в результатній таблиці залишаються лише різні записи.
 - ❑ Щоб зберегти в ній усі записи, необхідно написати INTERSECT ALL.
-

Перетин наборів записів (INTERSECT)

- Приклад 8. Дві технічні системи містять однакові компоненти. Необхідно отримати список компонент, які входять в одну і в другу системи:

```
SELECT * FROM Система1  
INTERSECT  
SELECT * FROM Система2;
```

- Припускається, що таблиці Система1 і Система2 мають однакову структуру.
-

Перетин наборів записів (INTERSECT)

- Приклад 8 (продовж.). Якщо структури таблиць Система1 і Система2 відрізняються, але є стовпці, наприклад, ID_comp і Type_comp, які мають однакові тип і довжину, то можна застосувати наступний запит:

```
SELECT S1.ID_comp, S1.Type_comp  
      FROM Система1 AS S1  
INTERSECT  
SELECT S2.ID_comp, S2.Type_comp  
      FROM Система2 AS S2;
```

Віднімання наборів записів (EXCEPT)

Запит1 EXCEPT Запит2;

- ❑ Віднімання наборів рядків повертає таблицю, рядки якої містяться в одному наборі за виключенням тих, які містяться в другому наборі.
-

Віднімання наборів записів (EXCEPT)

- Приклад 9. Таблиці Clients та Contacts мають однотипні поля name та adress. Щоб дізнатись, чи усі клієнти записані у списку контактів, можна використати наступний запит:

```
SELECT Clients.name, Clients.adress  
FROM Clients  
EXCEPT
```

```
SELECT Contacts.name, Contacts.adress  
FROM Contacts;
```

- В результаті цього запиту виведуться ті рядки, яких немає в таблиці Contacts.
 - Якщо ж запит поверне порожню таблицю, то це означає, що усі клієнти представлені в таблиці Contacts.
-

Правила використання операторів UNION, EXCEPT і INTERSECT

Реалізація SQL Server

- ❑ Оператор INTERSECT має більший пріоритет, ніж EXCEPT.
 - ❑ Перший запит може містити фразу INTO, яка створює таблицю, в якій буде зберігатись результатний набір. Фразу INTO можна використовувати лише в першому запиті. Якщо фраза INTO буде вказана в будь-якому іншому місці, то SQL Server поверне повідомлення про помилку.
-

Правила використання операторів UNION, EXCEPT і INTERSECT

Реалізація SQL Server

- ❑ Фразу ORDER BY можна вказувати лише вкінці інструкції. Її не можна використовувати всередині окремих запитів, які складають інструкцію.
 - ❑ Фрази GROUP BY і HAVING можна використовувати лише всередині окремих запитів; їх не можна використовувати для того, щоб вплинути на результатний набір.
 - ❑ Оператори UNION, EXCEPT та INTERSECT не можна використовувати разом з інструкцією INSERT.
-

Дякую за увагу
