Тема лекції 4:

Формування SQL-запиту на вибірку даних

- Визначення запиту
- Усунення надлишковості вибраних даних
- Уточнення запиту за допомогою предикатів

Визначення SQL-запиту

Запит на вибірку даних – це команда або інструкція, яка дається СУБД, щоб вона вивела певну інформацію з таблиць бази даних

- Запити розглядаються як частина мови DML. Оскільки запит не змінює інформацію в таблицях, а просто виводить її користувачу, то будемо розглядати запити як самостійну категорію DQL.
- Усі запити на вибірку даних в SQL складаються з однієї команди (інструкції) – SELECT.

Загальний синтаксис інструкції SELECT

```
SELECT [ALL | DISTINCT]  {* |
    вираз-стовпець [AS псевдонім] [, ...]}
    FROM таблиця [, ...]
    [WHERE умова пошуку]
    [GROUP BY список стовпців групування]
    [HAVING умова пошуку]
    [ORDER BY умова сортування
                список стовпців сортування];
```

Основні аргументи інструкції SELECT

- `*` – вказує, що вибрано усі стовпці заданої таблиці або таблиць;
- `вираз-стовпець` – ім’я стовпця або вираз з декількох імен, з яких вибираються дані. Якщо включити декілька стовпців, то вони будуть вибиратись за вказаним порядком;
- `псевдонім` – ім’я або імена, які стануть заголовками стовпців у результаті запиту;
- `таблиця` – ім’я таблиці, з якої вибираються записи

Порядок **виконання** SQL-запиту

- **FROM** – СУБД вибирає таблицю з бази даних.
- **WHERE** – з таблиці вибираються записи, які відповідають умові пошуку, і відкидаються решта (фільтр записів).
- **GROUP BY** – створюються групи записів, відібраних оператором WHERE (якщо він є в SQL-виразі), і кожна група відповідає якому-небудь значенню стовпця групування. Стовпець групування може бути будь-яким стовпцем таблиці, заданій в операторі FROM, а не лише тими, які вказані у виразі SELECT.
- **HAVING** – обробляє кожну із створених груп записів, залишаючи лише ті з них, які задовольняють умові. Цей оператор використовується лише разом з оператором GROUP BY.
- **SELECT** – вибирає з таблиці, віртуально створеної в результаті застосування наведених операторів, лише вказані стовпці.
- **ORDER BY** – сортує записи таблиці. При цьому в умову сортування можна вказувати лише ті стовпці, які вказані в операторі SELECT.

Мінімальний синтаксис інструкції SELECT

SELECT стовпці FROM таблиця;

Фрази SELECT та FROM є обов’язковими в SQL-виразі.

Приклад 1. Вивести інформацію з таблиці Salers

```
SELECT snum, sname, city, comm
FROM  Salers;
```

- Зауважимо, що крапка з комою використовується у всіх інтерактивних командах SQL, щоб повідомити базі, що команда заповнена і готова виконуватись. У деяких системах індикатором кінця команди є похила риска вліво (\).

- Якщо необхідно вивести усі стовпці таблиці, то можна скористатись зірочкою (*):

```
SELECT * FROM Salers;
```

Приклад 2. Вивести лише деякі стовпці таблиці Salers

- Запит

  SELECT sname, comm FROM Salers;

  буде виводити імена продавців та отримувані ними проценти від продажу.

- Щоб **переупорядкувати стовпці**, необхідно задати їх у тому порядку, в якому хочете побачити.

Усунення надлишковості вибраних даних

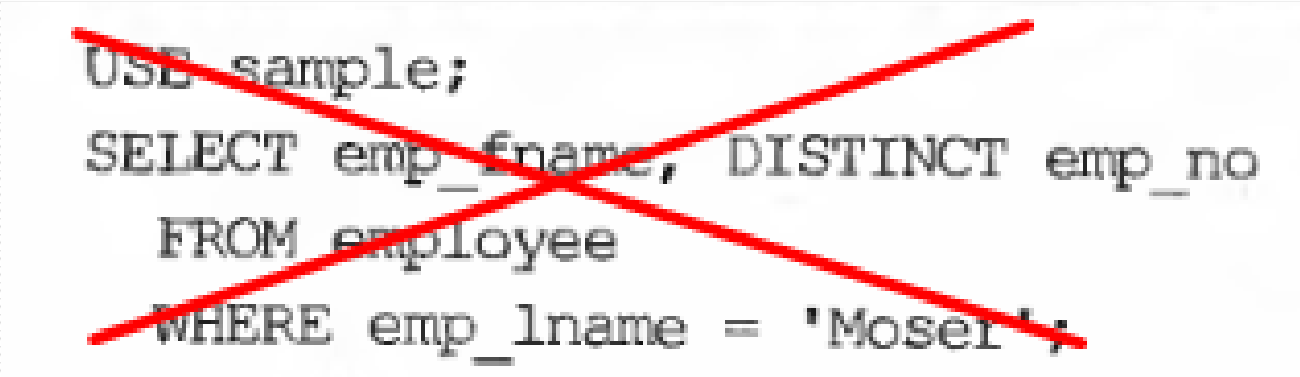
- Ключове слово DISTINCT (ВІДМІННІСТЬ) усуває повторювані значення з команди SELECT (***діє на весь перелік стовпців !)***:

  SELECT DISTINCT стовп_1, стовп_2

  FROM таблиця;
- DISTINCT слідкує за тим, які значення комбінації стовп_1 та стовп_2 були раніше, щоб вони не дублювались у результатній таблиці

Усунення надлишковості даних через DISTINCT

- DISTINCT може використовуватися в запиті лише один раз,
- при цьому має розміщатися перед всіма іменами стовпців, які виводяться у запиті



```
USE sample;
SELECT emp_fname, DISTINCT emp_no
   FROM employee
   WHERE emp_lname = 'Moser';
```

Приклад 3. Вивести номери продавців, які провели операції на поточний час

Ця інформація виводиться з таблиці Orders. Не потрібно знати, скільки операції провів кожен продавець, а потрібен тільки список продавців.

Тому можна ввести команду:

SELECT snum FROM Orders;

	snum
1	1007
2	1004
3	1001
4	1002
5	1007
6	1002
7	1001
8	1003
9	1002
10	1001

Приклад 3. Вивести номери продавців, які провели операції на поточний час

- Для отримання списку без дублікатів задається інструкція:

```
SELECT DISTINCT snum  FROM Orders;
```

В результаті виведуться тільки 5 номерів продавців.

	snum
1	1001
2	1002
3	1003
4	1004
5	1007

Фільтрування рядків в SQL-запитах

SELECT стовпці FROM таблиця
WHERE умова_пошуку;

При виконанні запиту логічний вираз у фразі WHERE застосовується до усіх рядків вихідної таблиці

Основні типи умов пошуку (предикатів)

- **порівняння** – порівнюються результати обчислення одного виразу з результатами обчислення іншого виразу;
- **діапазон** – перевіряється, чи попадає результат обчислення виразу у заданий діапазон значень;
- **належність до множини** – перевіряється, чи належить результат обчислення виразу до заданої множини значень;
- **відповідність шаблону** – перевіряється, чи відповідає деяке символьне значення заданому шаблону;
- **існування** – перевіряється чи існує хоча б один рядок, який задовольняє умові;
- **перевірка на невизначене значення** – перевіряється, чи містить заданий стовпець значення NULL.

Приклад 4. Вивести імена та комісійні усіх продавців у Лондоні

```
SELECT sname, comm
  FROM Salers
  WHERE city='London';
```

- Коли задається у запиті Фраза WHERE, то СУБД проглядає усю таблицю по одному рядку і перевіряє кожен рядок чи виконується задана умова.

	sname	comm
1	Peel	0.12
2	Motika	0.15

Приклад 5. Вивести інформацію про усіх замовників з рейтингом 100

- Поле rating таблиці Customers призначене, щоб розділяти замовників на групи за певними критеріями.

```
SELECT *    FROM Customers
  WHERE rating=100;
```

	cnum	cname	city	rating	snum
1	2001	Hoffman	London	100	1001
2	2006	Clemens	London	100	1001
3	2007	Pereira	Rome	100	1004

Приклад 6. Вивести інформацію про усіх замовників з оцінкою вищою 200

- Використаємо предикат порівняння:

```
SELECT *  FROM Customers
  WHERE rating>200;
```

	cnum	cname	city	rating	snum
1	2004	Grass	Berlin	300	1002
2	2008	Cisneros	San Jose	300	1007

Приклад 7. Вивести інформацію про усіх замовників в місті San Jose, які мають рейтинг вище 200

- Використаємо предикати з операторами Буля:

```
SELECT *
  FROM Customers
  WHERE city='San Jose‘ AND rating>200;
```

	cnum	cname	city	rating	snum
1	2008	Cisneros	San Jose	300	1007

Приклад 8. Використання оператора NOT

- Оператор NOT використовується для інвертування булевих значень:

```
SELECT *    FROM Customers
  WHERE city='San Jose' OR NOT rating>200;
```

	cnum	cname	city	rating	snum
1	2001	Hoffman	London	100	1001
2	2002	Giovanni	Rome	200	1003
3	2003	Liu	San Jose	200	1002
4	2006	Clemens	London	100	1001
5	2007	Pereira	Rome	100	1004
6	2008	Cisneros	San Jose	300	1007

Приклад 9. Використання оператора NOT

- Оператор NOT стосується лише виразу, перед яким він стоїть. Тут NOT застосовується лише до виразу city='SanJose'.

```
SELECT *    FROM Customers
  WHERE NOT city='San Jose' OR rating>200;
```

	cnum	cname	city	rating	snum
1	2001	Hoffman	London	100	1001
2	2002	Giovanni	Rome	200	1003
3	2004	Grass	Berlin	300	1002
4	2006	Clemens	London	100	1001
5	2007	Pereira	Rome	100	1004
6	2008	Cisneros	San Jose	300	1007

Приклад 10. Використання оператора NOT

- Інший результат отримаємо при виконанні наступної інструкції. Тут SQL враховує дужки і обробляє вираз в дужках першим

```
SELECT *    FROM Customers
  WHERE NOT( city='San Jose' OR rating>200 );
```

	cnum	cname	city	rating	snum
1	2001	Hoffman	London	100	1001
2	2002	Giovanni	Rome	200	1003
3	2006	Clemens	London	100	1001
4	2007	Pereira	Rome	100	1004

Спеціальні SQL-предикати. Належність до множини.

- **IN** та **NOT IN** визначає список значень, в який може або не може входити дане значення стовпця

- Альтернативою є поєднання предикатів порівняння з логічною операцією OR.

Приклад 11. Вивести усіх замовників, яких обслуговують продавці з номерами 1001, 1007 і 1004.

```
SELECT *
  FROM Customers
  WHERE snum IN (1001,1007,1004);
```

	cnum	cname	city	rating	snum
1	2001	Hoffman	London	100	1001
2	2006	Clemens	London	100	1001
3	2007	Pereira	Rome	100	1004
4	2008	Cisneros	San Jose	300	1007

Приклад 12. Використання NOT IN (*сформулювати умову самостійно*)

```
SELECT *
  FROM Salers
  WHERE city NOT IN ('London','San Jose');
```

	snum	sname	city	comm
1	1003	Axelrod	New York	0.11
2	1007	Rifkin	Barcelona	0.10

Спеціальні SQL-предикати. Предикат діапазону.

- **BETWEEN** визначає діапазон значень, в який має попадати задане значення стовпця. Включає граничні значення у діапазон

WHERE стовп BETWEEN зн_1 AND зн_2;

- На відміну від оператора IN, оператор BETWEEN є чутливим до порядку, тобто першим має бути менше значення (як символьне так і числове).
- Має особливості роботи з символьними значеннями !!!

Приклад 13. Вивести продавців з комісійними між .10 і .12

```
SELECT *
  FROM Salers
  WHERE comm BETWEEN 0.10 AND 0.12;
```

- Предикат BETWEEN включає граничні значення у діапазон

	snum	sname	city	comm
1	1001	Peel	London	0.12
2	1003	Axelrod	New York	0.11
3	1007	Rifkin	Barcelona	0.10

Приклад 14. Вивести продавців з комісійними між .10 і .12 (не включаючи граничних значень)

- В SQL не підтримується безпосереднього невключення границь оператора BETWEEN.
- Тому потрібно або вибирати правильно граничні значення, або написати запит наступного типу:

```
SELECT *    FROM Salers
  WHERE (comm BETWEEN 0.10 AND 0.12)
      AND NOT comm IN (0.10,0.12);
```

	snum	sname	city	comm
1	1003	Axelrod	New York	0.11

Приклад 15.

Вивести усіх замовників, чиї імена попали в заданий алфавітний діапазон (від A до G ???)

```
SELECT *
  FROM Customers
  WHERE cname BETWEEN 'A' AND `G';
```

Приклад 15. Вивести усіх замовників, чиї імена попали в заданий алфавітний діапазон (від A до G)

- В результаті виведуться записи з іменами Clemens та Cisneros.

	cnum	cname	city	rating	snum
1	2006	Clemens	London	100	1001
2	2008	Cisneros	San Jose	300	1007

Приклад 15. Вивести усіх замовників, чиї імена попали в заданий алфавітний діапазон (від A до G)

- Зауважимо, що замовники з іменами Grass і Giovanni будуть відсутніми в результаті виконання попереднього запиту.
- Це відбулось тому що, якщо стрічки мають не однакову довжину, то оператор BETWEEN в коротшу доставляє пропуски. А стрічка ‘G ’ є меншою в алфавітному порядку за ‘Giovanni’. Те ж саме з іменем Grass.
- Це необхідно пам’ятати, якщо використовувати предикат BETWEEN для виводу значень з алфавітних діапазонів.
- В нашому випадку, щоб вивелись імена на букву ‘G’, необхідно задати діапазон: BETWEEN ‘A’ AND ‘H’.

Або:

Приклад 15.

```
SELECT *
  FROM Customers
  WHERE cname like '[A-G]%';
```

	cnum	cname	city	rating	snum
1	2002	Giovanni	Rome	200	1003
2	2004	Grass	Berlin	300	1002
3	2006	Clemens	London	100	1001
4	2008	Cisneros	San Jose	300	1007

Спеціальні SQL-предикати. Предикат шаблона

- **LIKE** (подібний) та **NOT LIKE** (не подібний) застосовуються тільки до полів типу CHAR або VARCHAR, в яких вони знаходить підстрічки.
- Деякі СУБД є чутливими до регістру, наприклад, Oracle та DB2. А MS SQL Server і MySQL є нечутливими до регістру.

Спеціальні SQL-предикати. Предикат шаблона

- **LIKE** (подібний) та **NOT LIKE** (не подібний) в якості умови використовують групові символи (маски):
  - _ - заміняє одиничний символ (в MS Access використовується знак питання ?);
  - % - заміняє послідовність символів довільної довжини;
  - [acd] – заміняє собою перелік символів
  - [a-c] – заміняє собою діапазон символів
  - **^** – заперечення переліку або діапазону

Приклад 16. Використання шаблона

- Вивести інформацію про замовників, чиї імена починаються з букви ‘G’:

```
SELECT *
  FROM Customers
  WHERE cname LIKE ‘G%’;
```

	cnum	cname	city	rating	snum
1	2002	Giovanni	Rome	200	1003
2	2004	Grass	Berlin	300	1002

Приклад 17. Використання маски

- Припустимо, що ви не знаєте як правильно написати одного з продавців: Peal чи Peel. Тоді можна використати відому частину і маску:

```
SELECT *
  FROM Salers
  WHERE sname LIKE 'P_ _l%';
```

- Маска '%' в кінці стрічки потрібна в більшості реалізацій, якщо довжина поля sname більша, ніж кількість символів в імені Peel (тому що деякі інші значення sname довші за 4 символа).
- Таким чином поле sname зі значенням Peel має в кінці пробіли. Отже, символ 'l' не буде розглядатись як кінець стрічки, а символ '%' відповідає пробілам.
- Це робити необов'язково, якщо поле має тип VARCHAR.

Приклад 17. Використання маски переліку

```
SELECT *  FROM Customers
  WHERE cname like '[CP]%';
```

	cnum	cname	city	rating	snum
1	2006	Clemens	London	100	1001
2	2007	Pereira	Rome	100	1004
3	2008	Cisneros	San Jose	300	1007

```
SELECT *  FROM Customers
  WHERE cname like '[^CP]%';
```

	cnum	cname	city	rating	snum
1	2001	Hoffman	London	100	1001
2	2002	Giovanni	Rome	200	1003
3	2003	Liu	San Jose	200	1002
4	2004	Grass	Berlin	300	1002

Спеціальні SQL-предикати. Перевірка на значення NULL

- **IS NULL** застосовується для виявлення записів, в яких той чи інший стовпець має невідоме значення
- **IS NOT NULL** застосовується, коли необхідно виключити з результатів запису з NULL-значеннями

Приклад 18. Використання IS NULL

- Знайдемо усі записи в таблиці Customers з NULL значеннями у стовпці city:

```
SELECT *
  FROM Customers
  WHERE city IS NULL;
```

- Тут не буде виводу результатів, оскільки в даній таблиці нема NULL значень.

Приклад 19. Використання IS NOT NULL

- Необхідно виключити з результатів виведення записи з NULL-значеннями:

```
SELECT *
  FROM Customers
  WHERE city IS NOT NULL;
```

  В нашому випадку виведеться уся таблиця Customers.

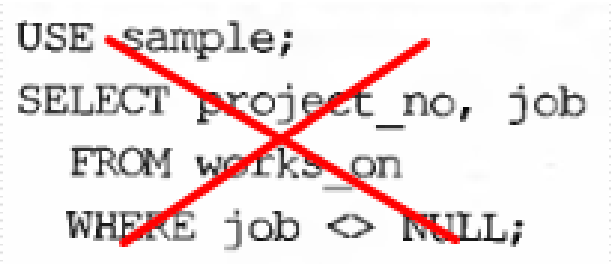
- Аналогічно можна ввести команду:

```
SELECT *
  FROM Customers
  WHERE NOT city IS NULL;
```

Використання NULL

- Будь-які порівняння зі значенням NULL дадуть значення *false*

(або неочікувані результати)



Тому:

<стовпець> **IS** [NOT] **NULL**

А також функція T-SQL:

ISNULL(<column>,’instead of NULL’)

Практичні завдання до БД «Операції купівлі-продажу»

- Напишіть запит, який може вивести усіх продавців у місті Barcelona з комісійними вищими .10.
- Напишіть запит, який би вивів усю інформацію про замовників з рейтингом 200.
- Напишіть запит, який би вивів усю інформацію про замовників з Риму.
- Напишіть запит, щоб його результат включав інформацію про усіх замовників з рейтингом більшим 100, якщо вони не знаходяться у Римі.
- Напишіть запит, який би вивів інформацію про усіх замовників, яких обслуговують продавці Serres, Axelrod, Rifkin.
- Напишіть запит, який би вивів усю інформацію про продавців з комісійними 12%-15%.
- Напишіть запит, який би вивів інформацію про усіх замовників, імена яких починаються з букв від ‘A’ до ‘C’.
- Напишіть запит, який би вивів інформацію про усіх замовників, імена яких починаються з букви ‘G’.
- Напишіть запит, який би виводів інформацію про операції купівлі-продажу, в яких брав участь замовник Giovanni.

Дякую за увагу

Опрацювати: Д.Петковіч «Microsoft SQL Server 2012. Руководство для начинающих» ***ст.149-163***