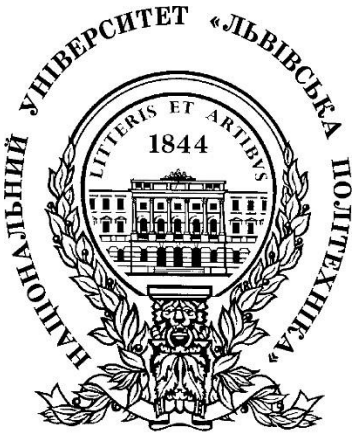
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра систем штучного інтелекту



ЗВІТ № 7 **з**

курсу “ОБДЗ”

на тему:

«Запити на вибір даних з таблиць бази даних»

Виконав:

студент групи КН-208

Фіняк М.В.

Викладач:

Якимишин Х.М.

Лабораторна робота № 7

Мета роботи: розробити SQL запити відбору даних з одиничних та з'єднаних таблиць, в тому числі з використанням підзапитів, натурального, умовного та лівого з'єднання, із застосуванням у критеріях вибірки функцій та операторів, в т. ч. LIKE, BETWEEN, IS NULL, IS NOT NULL, IN (…), NOT IN (…), ALL, SOME, ANY, EXISTS.

Короткі теоретичні відомості

Для вибирання даних з таблиць використовується директива SELECT, яка може містити інші директиви SELECT (підзапити, або вкладені запити) та директиви з'єднання таблиць.

Синтаксис :

```
SELECT
[ALL | DISTINCT | DISTINCTROW ]
[STRAIGHT_JOIN]
[SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
елемент_вибірки [, елемент_вибірки ...]
[FROM перелік_таблиць]
[WHERE умова_відбору]
[GROUP BY {ім'я_поля | синонім | позиція_поля}
 [ASC | DESC], ...]
 [HAVING умова_відбору]
 [ORDER BY {ім'я_поля | синонім | позиція_поля}
 [ASC | DESC], ...]
[LIMIT {к-сть_рядків [OFFSET зміщення]}
[PROCEDURE ім'я_процедури(аргументи)]
```

[INTO OUTFILE 'ім’я_файлу' опції_експорту

| INTO DUMPFILE 'ім’я_файлу'

| INTO змінна [, змінна]]

Параметри: **SELECT**

Вказує поля, константи та вирази, що будуть відображатися у результатах запиту. Директива вимагає чіткого дотримання порядку ключових слів FROM, WHERE і т.д.

елемент_вибірки

Вказує елемент, який буде включатися в результати відбору. Такими елементами можуть бути: ім’я поля, константа або вираз. Кожному елементу можна присвоїти ім’я- псевдонім, яке буде відображатись у результатах запиту. Для цього після назви елемента слід дописати AS псевдонім.

перелік_таблиць

Назви таблиць, з яких здійснюється вибір значень. Тут можна задавати синоніми назвам таблиць (ім’я_таблиці AS синонім), використовувати підзапити SELECT для формування таблиці з вказаним синонімом, з’єднувати декілька таблиць.

WHERE

Вказує критерії порівняння (або підзапити) для відбору рядків.

GROUP BY

Групує (і одночасно сортує) рядки за вказаними полями. Поля можна вказувати за іменами, синонімами або порядковими номерами в таблиці.

ORDER BY

Сортує рядки за вказаними полями. За замовчуванням – за зростанням значень (ASC).

HAVING

Дає можливість застосування до значень полів агрегатних функцій (COUNT, AVG, MIN, MAX тощо) при відборі чи групуванні рядків. Після слова WHERE ці функції не працюють, однак у всіх інших випадках слід використовувати саме WHERE.

LIMIT

Обмежує кількість рядків, повернутих в результаті запиту.

OFFSET

Вказує зміщення для LIMIT – з якого рядка в результатах запиту почати відбирати потрібну кількість рядків.

PROCEDURE

Задає назву збереженої процедури, яка повинна обробляти результат запиту.

INTO

Вказує місце, куди будуть збережені результати запиту. Це може бути як зовнішній файл, так і параметри чи змінні, визначені користувачем. Кількість змінних має бути рівна кількості полів у результаті.

DISTINCT | DISTINCTROW

Видалення з результату рядків-дублікатів. За замовчуванням вибираються всі рядки.

STRAIGHT_JOIN

Опція, яка строго задає порядок вибирання кортежів зі з’єднуваних таблиць в порядку переліку таблиць. (Оптимізатор запитів MySQL іноді змінює цей порядок.)

SQL_CACHE | SQL_NO_CACHE

Явним чином вмикає/вимикає зберігання результатів запиту у кеші запитів MySQL. За замовчуванням, кешування запитів залежить від системної змінної query_cache_type.

SQL_CALC_FOUND_ROWS

Вказує, що при виконанні запиту слід обчислити загальну кількість рядків в результаті, ігноруючи опцію обмеження LIMIT. Цю кількість рядків потім можа отримати командою SELECT FOUND_ROWS().

Для вибору записів зі з'єднаних таблиць використовується директива SELECT разом із директивами JOIN у переліку таблиць.

Параметри директиви:

INNER JOIN

Внутрішнє з'єднання. Результати вибору будуть містити тільки ті рядки, для яких існують один або більше відповідних рядків з іншої таблиці. В MySQL – є синонімом директиви CROSS JOIN. Слід зауважити, що вибір рядків директивою SELECT з кількох таблиць, вказаних через кому, є аналогічним до явного використання директиви INNER JOIN. В обох випадках MySQL формує декартовий добуток усіх кортежів, і з результату вибирає лише ті, для яких виконується умова відбору (порівняння) ON.

LEFT JOIN

Вказує на те, що результати вибору будуть містити всі рядки з таблиці, яка стоїть зліва від слова JOIN і тільки відповідні їм рядки з таблиці справа (ті, для яких виконується вказана умова). Якщо відповідний рядок відсутній, виводяться значення NULL.

RIGHT JOIN

Вказує на те, що результати вибору будуть містити всі рядки з таблиці, яка вказана справа від JOIN і тільки відповідні їм рядки з таблиці зліва. Для сумісності на практиці використовують в основному LEFT JOIN.

ON умова

Вказує поля, за якими слід з’єднувати таблиці. Замість ON можна також використовувати USING перелік_спільних_полів. В цьому випадку спільне поле буде відображене в результатах запиту лише один раз.

NATURAL JOIN

Еквівалент внутрішньому з’єднанню за всіма однаковими полями (з опцією USING *).

О***сновні функції порівняння, які можна використовувати при формуванні складних критеріїв вибору*** :

Функція	*Опис*
STRCMP(рядок1, рядок2)	Порівнює два рядки. Повертає значення 0 (False) якщо рядки однакові, -1 якщо перший рядок менший за другий, і 1 (True) в усіх інших випадках.
LIKE рядок	Порівняння з рядком-шаблоном. В шаблоні можна використовувати знаки % (довільні символи) і _ (довільний символ).
REGEXP рядок	Порівняння з рядком з використанням регулярних виразів. Функція-синонім – RLIKE.
MATCH (поля) AGAINST (рядок)	Здійснює пошук рядка у вказаних текстових полях таблиці. (Тільки для MyISAM-таблиць.)

BETWEEN ... AND ...	Повертає 1, якщо значення належить даному діапазону.
NOT BETWEEN ... AND ...	Повертає 1, якщо значення не належить діапазону.
IN(арг1, арг2, ...)	Перевірка належності множині. Повертає 1, якщо значення співпадає хоча б із одним аргументом, і 0 – у протилежному випадку. Повертає NULL, якщо значення є NULL, або якщо співпадіння не знайдено, а один із аргументів є NULL.
NOT IN(арг1, арг2, ...)	Повертає 1, якщо значення не міститься у множині аргументів, і 0 – у протилежному випадку. Повертає NULL
	аналогічно до функції IN().
IS NULL, IS NOT NULL	Перевірка визначеності значення.
LEAST(арг1, арг2, ...)	Повертає мінімальне значення серед аргументів. Повертає NULL, якщо хоча б один із аргументів є NULL
GREATEST(арг1, арг2, ...)	Повертає максимальне значення серед аргументів. Повертає NULL, якщо хоча б один із аргументів є NULL.

Хід роботи

Для вивчення роботи директив вибору даних з таблиць розробимо та виконаємо такі запити :

1. Знайти номер телефону користувача з номером 3.
2. Показати всі інгредієнти з їхніми постачальниками (ліве з’єднання таблиць).
3. Показати інгредієнти з постачальником ‘Molokiya’ (внутрішнє з’єднання).
4. Показати всі страви з інгредієнтами з постачальниками ‘Molokiya’ та ‘Hutorok’.
5. Вибрати останні 2 страви з інгредієнтами з постачальниками ‘Molokiya’ та ‘Hutorok’(орієнтуючись на алфавітний порядок назв інгредієнтів).
6. Визначити страву, яка не містить жодного з даних інгредієнтів.
7. Визначити неправильно вказані номери телефонів клієнтів.

Виконання

1. Знайдемо номер телефону користувача з номером 3. Для цього слід в умові відбору вказати номер потрібного користувача.

```
SELECT id,telephone
    FROM restaurant.customer WHERE id = 3;
```

	telephone
▶	+380931073422

2. Виберемо всі інгредієнти з їхніми постачальниками. Для цього потрібно виконати ліве з'єднання.

```
SELECT ingredient.id, ingredient.name, supplier.name
  FROM restaurant.ingredient LEFT JOIN restaurant.supplier ON ingredient.supplier_id =
  supplier.id;
```

	id	name	name
▶	1	Flour	Hutorok
	2	Milk	Molokiya
	3	Butter	Molokiya
	4	Eggs	Kvochka

3. Виберемо інгредієнти з постачальником 'Molokiya'. Для цього виконаємо умовне з'єднання таблиць ingredient і supplier, використовуючи директиву INNER JOIN.

```
SELECT ingredient.name, supplier.name
  FROM ingredient INNER JOIN supplier ON supplier.id = ingredient.supplier_id
WHERE supplier.name = 'Molokiya';
```

	name	name
▶	Milk	Molokiya
	Butter	Molokiya

4. Виберемо всі страви з інгредієнтами з постачальниками 'Molokiya' та 'Hutorok'. Для цього виконаємо умовне з'єднання таблиць ingredient, supplier за id та supplier_id , та таблиці ingredient_dish2, використовуючи директиву INNER JOIN.

Таблиця ingredient_dish2 утворюється як результат запиту на виконання проекції :

```
CREATE VIEW ingredient_dish2
    AS SELECT DISTINCT ingredient_dish.ingredient_id, ingredient_dish.dish_id1, dish.name
    FROM ingredient_dish,dish
    WHERE ingredient_dish.dish_id1=dish.id;
SELECT * FROM ingredient_dish2;
```

ingredient_id	dish_id1	name
1	1	Oreo
2	1	Oreo
4	2	Brownie
3	4	Strawberry cake "Fraisier"

Результат запиту :

```
SELECT ingredient.name, supplier.name, ingredient_dish2.name
    FROM (ingredient INNER JOIN supplier) INNER JOIN ingredient_dish2
    ON supplier.id = ingredient.supplier_id
    AND ingredient_dish2.ingredient_id = ingredient.id
    WHERE supplier.name IN ('Molokiya', 'Hutorok');
```

name	name	name
Flour	Hutorok	Oreo
Milk	Molokiya	Oreo
Butter	Molokiya	Strawberry cake "Fraisier"

5. Виберемо останні 2 страви з інгредієнтами з постачальниками ‘Molokiya’ та

‘Hutorok’(орієнтуємось на алфавітний порядок назв інгредієнтів).

```
SELECT ingredient.name, ingredient_dish2.name
    FROM ingredient INNER JOIN ingredient_dish2
    ON ingredient.id = ingredient_dish2.ingredient_id
    WHERE ingredient.supplier_id IN (SELECT supplier.id FROM supplier
    WHERE supplier.name IN ('Molokiya', 'Hutorok'))
    ORDER BY ingredient.name DESC LIMIT 2;
```

name	name
Milk	Oreo
Flour	Oreo

6. Визначимо страву, яка не містить жодного з даних інгредієнтів.

```
SELECT dish.name FROM dish
    WHERE NOT EXISTS
   (SELECT * FROM ingredient_dish WHERE ingredient_dish.dish_id1 = dish.id);
```

	name
▶	LemonPie

7. Визначимо неправильно вказані номери телефонів клієнтів (менші за 13 символів або містять букви). Таблиця customer :

SELECT * FROM customer;

	id	first_name	last_name	email	telephone
▶	1	Yuliana	Lavryk	ylilav@gmail.com	+380931456091
	2	Iryna	Dosiak	iryna123@gmail.com	NULL
	3	Olena	Kulchytska	olena111@gmail.com	+380931073422
	4	Oleksandra	Dypko	olexandra17@gmail.com	+380675809127
	5	Anna	Kvitkova	anna31@gmail.com	+380976531098
	6	Igor	Melnyk	igor@gmail.com	+380731108
	7	Sergiy	Batruh	ser230@gmail.com	+38093ab22310
*	NULL	NULL	NULL	NULL	NULL

Результат запиту :

SELECT id,first_name,last_name,telephone
FROM customer
WHERE CHAR_LENGTH (telephone) < 13 OR
(telephone) REGEXP '[a-z]';

	id	first_name	last_name	telephone
▶	6	Igor	Melnyk	+380731108
	7	Sergiy	Batruh	+38093ab22310
*	NULL	NULL	NULL	NULL

Висновок: під час виконання даної лабораторної роботи було вивчено методи вибору даних з одиничних та з'єднаних таблиць БД засобами SQL та виконано запити до бази даних з використанням директив SELECT та JOIN, а також складних критеріїв в умові вибірки.