

Лабораторна робота №7  
з курсу “ОБДЗ”  
на тему:  
**“Запити на вибір даних з таблиць бази даних”**

**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.

**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'єднаних 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 можна отримати командою `SELECT FOUND_ROWS()`.

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

```
SELECT * FROM author INNER JOIN comment  
ON author.authorID = comment.authorID;
```

### 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 \*).

У таблиці нижче описано основні функції порівняння, які можна використовувати при формуванні складних критеріїв вибору.

Функція	Опис
STRCMP( <i>рядок1</i> , <i>рядок2</i> )	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 <i>рядок</i>	Порівняння з рядком-шаблоном. В шаблоні можна використовувати знаки % (довільні символи) і _ (довільний символ).
REGEXP <i>рядок</i>	Порівняння з рядком з використанням регулярних виразів. Функція-синонім – RLIKE.
MATCH ( <i>поля</i> ) AGAINST ( <i>рядок</i> )	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( <i>arg1</i> , <i>arg2</i> , ...)	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 співпадиння не знайдено, а один із аргументів є NULL.
NOT IN( <i>arg1</i> , <i>arg2</i> , ...)	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( <i>arg1</i> , <i>arg2</i> , ...)	Повертає мінімальне значення серед аргументів. Повертає NULL, якщо хоча б один із аргументів є NULL.
GREATEST( <i>arg1</i> , <i>arg2</i> , ...)	Повертає максимальне значення серед аргументів. Повертає NULL, якщо хоча б один із аргументів є NULL.

Для формування критеріїв вибору та підзапитів також використовують наступні оператори порівняння:

=

Оператор перевірки рівності двох виразів. Якщо відбувається порівняння двох не NULL значень, то повертає значення 1 (True) коли обидва вирази рівні, інакше результатом є значення 0 (False). Якщо хоча б один з виразів приймає значення NULL, то результатом є значення NULL.

<=>

Перевірка рівності виразів, яке враховує NULL значення. Повертає 1, якщо обидва вирази приймають значення NULL, або рівні значення. Повертає 0, якщо один із виразів приймає значення NULL, або значення виразів не рівні.

>, >=

Порівняння двох виразів. Результатом є 1, якщо ліве значення більше (більше рівне) ніж праве, інакше результатом є 0. Якщо хоча б один з виразів приймає значення NULL, то результатом теж стає NULL.

<, <=

Порівняння двох виразів. Результатом є 1, якщо ліве значення менше (менше рівне) ніж праве, інакше результатом є 0. Якщо хоча б один з виразів приймає значення NULL, то результатом теж є NULL.

!=, <>

Перевірка на не рівність. Результат набуває значення 1, якщо ліве значення менше або більше ніж праве, інакше результатом є 0. Якщо хоча б один з виразів приймає значення NULL, то результатом теж є NULL.

ALL, SOME, ANY

Оператори, які можна використовувати після операторів порівняння. Задають необхідність виконання оператора хоча б для одного (SOME, ANY) чи всіх (ALL) елементів, отриманих в результаті підзапиту. На відміну від функцій IN (), NOT IN () оператори не працюють зі списками значень.

[NOT] EXISTS

Оператор, який використовують після ключового слова WHERE. Повертає 1, якщо підзапит повертає хоча б одне визначене значення, і 0 – у протилежному випадку.

## Хід роботи.

Для вивчення роботи директив вибору даних з таблиць розробимо та виконаємо такі запити над таблицями Author, Role, Comment.

1. Показати пароль заданого користувача.
  2. Показати користувачів і їхні коментарі (ліве з'єднання таблиць).
  3. Показати перелік користувачів у групі Guests (натуральне з'єднання).
  4. Показати всі коментарі користувачів з груп Guests та SUGroup2 (умовне з'єднання).
  5. Показати останні 3 коментарі користувачів з груп Guests та SUGroup2 (підзапит).
  6. Визначити користувачів, які не написали жодного повідомлення.
  7. Визначити користувачів, паролі яких не відповідають вимогам безпеки (менші за 8 символів або не містять цифр).
1. Знайдемо пароль користувача з номером 9. Для цього слід використати функцію дешифрації AES\_DECRYPT, а в умові відбору вказати номер потрібного користувача.

```
SELECT AES_DECRYPT(password, 'key-key')  
FROM author WHERE authorID = 9;
```

Результат запиту:

←T→
<b>AES_DECRYPT(password,'key-key')</b>
su2pass

2. Виберемо всіх користувачів з їхніми коментарями. Для цього потрібно виконати ліве з'єднання. Для користувачів, які не написали жодного коментаря в результатах буде відображено порожні значення.

```
SELECT author.authorID, author.login, author.email,  
comment.text, comment.posted  
FROM author LEFT JOIN comment ON  
author.authorID = comment.authorID;
```

Результат запиту:

authorID	login	email	text	posted
2	admin	admin@admin.com	ОК, дуже добре	2009-03-04 11:12:33
3	user1	user1@gmail.com	Мені сподобалось :)	2009-03-05 23:21:22
4	user2	user2@gmail.com	NULL	NULL
5	user3	user3@gmail.com	NULL	NULL
6	guest1	guest1@mail.com	NULL	NULL
7	guest2	guest2@sco.com	Це дуже важливо!	2009-02-04 13:11:00
8	superuser1	spuser1@sp1.com	NULL	NULL
9	superuser2	spuser2@sp2.com	Перший коментар!	2009-03-03 12:23:12
9	superuser2	spuser2@sp2.com	Дякую за інформацію	2009-03-04 13:11:11
9	superuser2	spuser2@sp2.com	ОК	2009-03-04 00:00:00
10	user4	user3@gmail.com	NULL	NULL
11	user9	user9@mail.com	NULL	NULL

3. Виберемо користувачів з групи Guests. Для цього виконаємо умовне з'єднання таблиць Author і Role за атрибутом *roleID*, використовуючи директиву INNER JOIN.

```
SELECT author.login, role.rolename
FROM author INNER JOIN role ON role.roleID = author.roleID
WHERE role.rolename = "Guests";
```

Результат запити:

login	rolename
guest1	Guests
guest2	Guests

4. Виберемо всі коментарі користувачів з групи Guests та SUGroup2. Для цього виконаємо умовне з'єднання таблиць Author і Role за атрибутом *roleID*, та таблиці Comment використовуючи директиву INNER JOIN.

```
SELECT author.login, role.rolename, comment.text, comment.posted
FROM (author INNER JOIN role) INNER JOIN comment
ON role.roleID = author.roleID
AND comment.authorID = author.authorID
WHERE role.rolename IN ("Guests", "Sugroup2");
```

Результат запити:

login	rolename	text	posted
superuser2	SUGroup2	Перший коментар!	2009-03-03 12:23:12
superuser2	SUGroup2	Дякую за інформацію	2009-03-04 13:11:11
superuser2	SUGroup2	OK	2009-03-04 00:00:00
guest2	Guests	Це дуже важливо!	2009-02-04 13:11:00

5. Виберемо останні 3 коментарі користувачів з групи Guests та SUGroup2. Для цього замість директиви JOIN використаємо підзапит в умові відбору, який буде повертати номери потрібних груп.

```
SELECT author.login, comment.text, comment.posted
FROM author INNER JOIN comment
ON author.authorID = comment.authorID
WHERE author.roleID IN (SELECT role.roleID FROM role
WHERE role.rolename IN ("Guests", "Sugroup2"))
ORDER BY comment.posted DESC LIMIT 3;
```

Результат запити:

login	text	posted
superuser2	Дякую за інформацію	2009-03-04 13:11:11
superuser2	OK	2009-03-04 00:00:00
superuser2	Перший коментар!	2009-03-03 12:23:12

6. Визначимо користувачів, які не написали жодного повідомлення.

```
SELECT author.login FROM author
WHERE NOT EXISTS
(SELECT * FROM message WHERE message.authorID =
author.authorID);
```

Результат запиту:

login
user3
guest1
superuser1
user9

7. Визначимо користувачів, паролі яких не відповідають вимогам безпеки (менші за 8 символів або не містять цифр).

```
SELECT login, AES_DECRYPT(password, 'key-key') AS pass
FROM author
WHERE CHAR_LENGTH(AES_DECRYPT(password, 'key-key'))<8 OR
AES_DECRYPT(password, 'key-key') NOT REGEXP '[0-9]';
```

Результат запиту:

login	pass
admin	adminpass
user1	uuuupass
superuser1	su1pass
superuser2	su2pass
user4	user_pass
user9	user_pass

**Висновок:** на цій лабораторній роботі було вивчено методи вибору даних зі 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.