МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

ІНСТИТУТ ПІСЛЯДИПЛОМНОЇ ОСВІТИ

КАФЕДРА CИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ



ЗВІТ ДО ЛАБОРАТОРНОЇ РОБОТИ №7

**Підготувала:**

Студентка групи КН-209

Кульчицька Олена

**Викладач:**

Мельникова Н.І.

2020

Лабораторна робота №7

*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’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. 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*

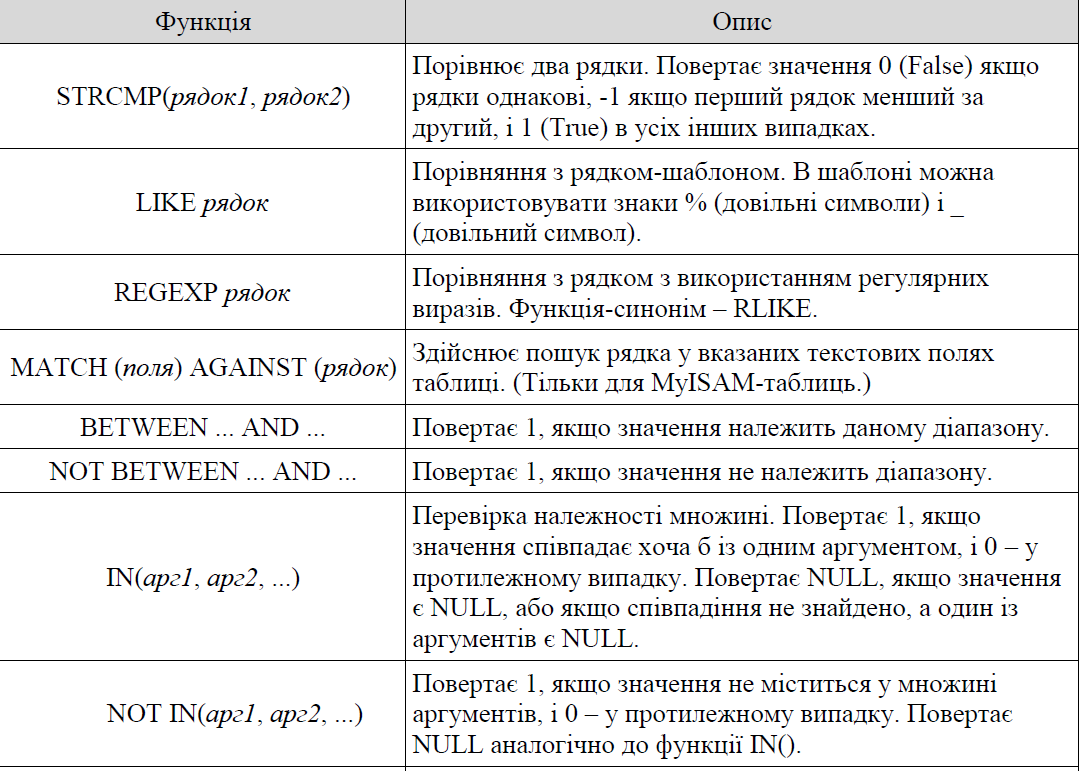
Вказує поля, за якими слід з’єднувати таблиці.

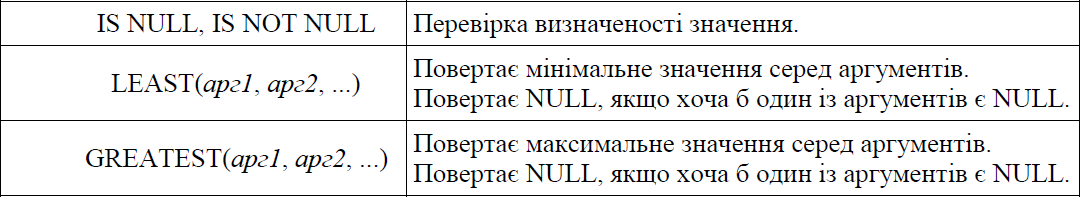
Замість ON можна також використовувати USING перелік\_спільних\_полів. В цьому випадку спільне поле буде відображене в результатах запиту лише один раз.

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.





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 WHERЕ. Повертає 1, якщо підзапит повертає хоча б одне визначене значення, і 0 – у протилежному випадку.

**Хід роботи.**

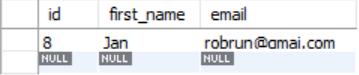
Для вивчення роботи директив вибору даних з таблиць розробимо та виконаємо такі запити над таблицями Customer, Order, Staff:

* Показати ім’я та емайл заданого користувача.
* Показати імена відвідувачів в порядку спадання.
* Показати відвідувачів, номери яких починаються з цифри 8.
* Показати відвідувачів та дати, час їхніх замовлень(ліве з’єднання).
* Показати відвідувачів , які зробили замовлення о 14:15:00(натуральне з’єднання).
* Показати відвідувачів та персонал,які обслуговували замовлення о 14:15:00 або 14:00:20(умовне з’єднання).
* Показати 4 замовлення відвідувачів, обслужених персоналом під індексом 2 або 3(підзапит).
* Визначимо відвідувачів, які не зробили жодного замовлення.
* Визначимо відвідувачів, номери яких не відповідають вимогам.

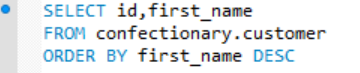
1. Визначимо ім’я та емайл відвідувача, айді якого = 8. Для цього виконуємо селекцію.



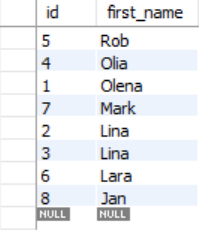
Результат запиту:



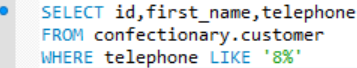
1. Для того,щоб вивести імені відвідувачів в порядку спадання, використовуємо команду ORDER BY.



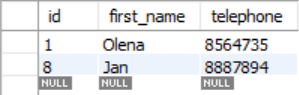
Результат запиту:



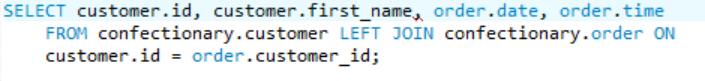
1. Для того, щоб вивести відвідувачів, номери телефонів яких починаються з цифри 8 використовуємо LIKE.

****

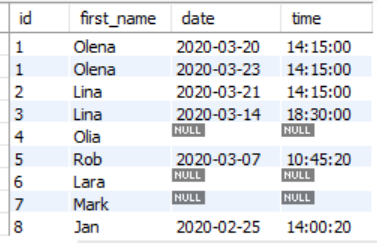
Результат запиту:



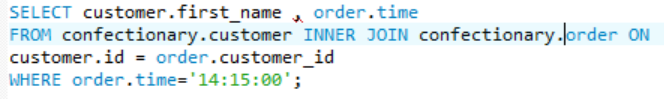
1. Виберемо всіх користувачів з датою та часом їхніх замовлень. Для цього потрібно виконати ліве з’єднання. Для відвідувачів, які не зробили жодного замовлення в результатах буде відображено порожні значення.

****

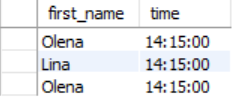
Результат запиту:

****

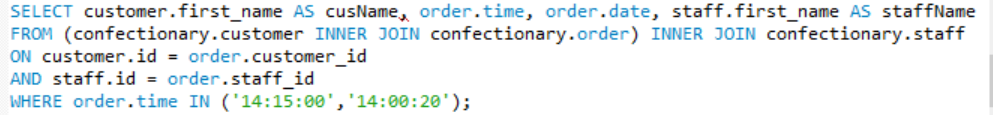
1. Виберемо відвідувачів, які зробили замовлення о 14:15:00. Для цього виконаємо умовне з’єднання таблиць Customer i Order за атрибутом *customer\_id*, використовуючи директиву INNER JOIN.



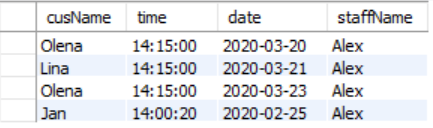
Результат запиту:



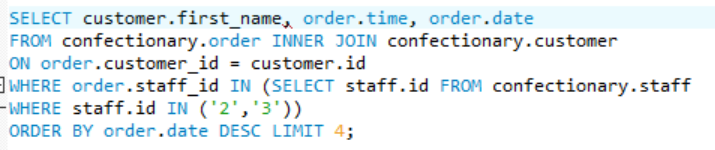
1. Виберемо відвідувачів та персонал,які обслуговували замовлення о 14:15:00 або 14:00:20. Для цього виконаємо умовне з’єднання таблиць Customer i Order за атрибутом *customer\_id*, та таблиці Staff використовуючи директиву INNER JOIN.



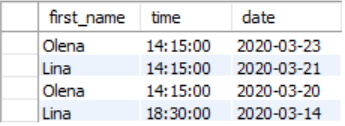
Результат запиту:



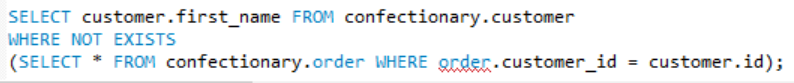
1. Виберемо 4 замовлення відвідувачів, обслужених персоналом під індексом 2 або 3. Для цього замість дирактиви JOIN використаємо підзапит в умові відбору.



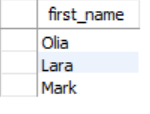
Результат запиту:



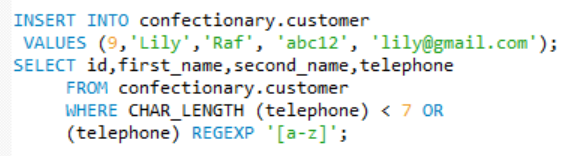
1. Визначимо відвідувачів, які не зробили жодного замовлення.



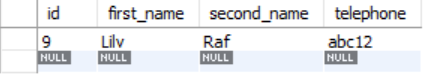
Результат запиту:



1. Визначимо відвідувачів, номери яких не відповідають вимогам(менші за 7 символів або містять букви).



Результат запиту:



**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.