МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота 7

з організації баз даних та знать

Виконав:

Студент групи КН-208

Воробель Адріан

Викладач:

Якимишин Х.М.

Львів – 2019р.

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 *).

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.

1. Показати опис заданого автомобіля.

```
SELECT * FROM cardescription WHERE car_id = 3;
```

id	car_engine	wheel_drive	sit_places	trunk	transmission	car_id
3	3.0	AWD	5	170	Automatic	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. Показати користувачів і їхні замовлення (ліве з'єднання таблиць).

```
SELECT userscontacts.u_name, userscontacts.u_surname,
orders.id AS orderID FROM userscontacts
LEFT JOIN orders ON userscontacts.user_id =
orders.user_id;
```

u_name	u_surname	orderID
Admin	Admin	NULL
Alexander	Votdzik	2
Alexander	Votdzik	6
Illarion	Leontov	3
Camil	Ramzan	4
Camil	Ramzan	7
Elvira	Catoshnik	5

3. Показати привід автомобілів з не ручною коробкою передач (натуральне з'єднання).

```
SELECT car.model, car.mark, cardescription.wheel_drive
FROM car INNER JOIN cardescription ON car.id =
cardescription.car_id
WHERE cardescription.transmission != "Manual";
```

model	mark	wheel_drive
A4	Audi	AWD
M5	BMW	AWD
C63	Mercedes-Benz	RWD
Urus	Lambotgini	AWD

4. Показати марку, модель автомобілів, крім пікапів та кросоверів, та назву їх дистрибютора (умовне з'єднання).

```
SELECT car.mark, car.model, distributor.distrib_name as
distributorID FROM car
INNER JOIN cardistributor ON car.id =
cardistributor.car_id
INNER JOIN distributor ON cardistributor.distrib_id =
distributor.id
WHERE car.car_type NOT IN ("Pickup","Crossover");
```

	mark	model	distributorID
▶	Honda	Civic Type R	HalychynaAuto
	Audi	A4	LvivDrive
	Mercedes-Benz	C63	LvivDrive
	Toyota	Mark 2	FayneAuto
	BMW	M5	FayneAuto

5. Показати марку, модель, привід та об’єм двигуна автомобілів, які не належать дистрибютору з id = 3 (підзапит).

```
SELECT car.mark, car.model, cardescription.car_engine,
cardescription.wheel_drive
FROM car INNER JOIN cardescription ON car.id =
cardescription.car_id
WHERE car.id IN (SELECT cardistributor.car_id FROM
cardistributor
WHERE cardistributor.distrib_id != 3) ORDER BY
wheel_drive;
```

	mark	model	car_engine	wheel_drive
▶	Lambotgini	Urus	3.5	AWD
	Audi	A4	2.1	AWD
	Honda	Civic Type R	1.6	FWD
	Chevrolet	El Camino	4.8	RWD
	Mercedes-Benz	C63	4.3	RWD

6. Визначити користувачів, які не замовили автомобілів.

```
SELECT userscontacts.u_name, userscontacts.u_surname FROM
userscontacts
WHERE NOT EXISTS
(SELECT * FROM orders WHERE userscontacts.user_id =
orders.user_id);
```

	u_name	u_surname
▶	Admin	Admin
	Elvira	Catoshnik

7. Визначити найстаршу машину із списку.

```
SELECT CONCAT(car.mark," ", car.model) as Car FROM car
WHERE car.product_date = (SELECT MIN(car.product_date)
FROM car);
```

	Car
▶	Chevrolet El Camino

Висновок.

В ході виконання лабораторної роботи я навчився використовувати ліве, умовне з’єднання, підзапити, декілька функцій мови sql із застосуванням у критеріях вибірки.