МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»



**Лабораторна робота №10**

**З дисципліни «Організація баз даних та знань»**

***Виконала:*** *студентка групи КН-210*

*Кручковська Христина*

***Викладач:***

*Мельникова Н. І.*

Львів – 2020

**Тема:** Написання збережених процедур на мові SQL

**Мета:** Навчитися розробляти та виконувати збережені процедури та функції у MySQL.

Короткі теоретичні відомості.

Більшість СУБД підтримують використання збережених послідовностей команд для виконання часто повторюваних, однотипних дій над даними. Такі збережені процедури дозволяють спростити оброблення даних, а також підвищити безпеку при роботі з базою даних, оскільки в цьому випадку прикладні програми не потребують прямого доступу до таблиць, а отримують потрібну інформацію через процедури.

СУБД MySQL підтримує збережені процедури і збережені функції. Аналогічно до вбудованих функцій (типу COUNT), збережену функцію викликають з деякого виразу і вона повертає цьому виразу обчислене значення. Збережену процедуру викликають за допомогою команди CALL. Процедура повертає значення через вихідні параметри, або генерує набір даних, який передається у прикладну програму.

Синтаксис команд для створення збережених процедур описано нижче.

CREATE  
[DEFINER = { користувач | CURRENT\_USER }] FUNCTION назва\_функції ([параметри\_функції ...]) RETURNS тип  
[характеристика ...] тіло\_функції

CREATE  
[DEFINER = { користувач | CURRENT\_USER }]  
PROCEDURE назва\_процедури ([параметри\_процедури ...]) [характеристика ...] тіло\_процедури

Аргументи:

DEFINER  
Задає автора процедури чи функції. За замовчуванням – це CURRENT\_USER.

RETURNS

Вказує тип значення, яке повертає функція.

тіло\_функції, тіло\_процедури

Послідовність директив SQL. В тілі процедур і функцій можна оголошувати локальні змінні, використовувати директиви BEGIN ... END, CASE, цикли тощо. В тілі процедур також можна виконувати транзакії. Тіло функції обов’язково повинно містити команду RETURN і повертати значення.

параметри\_процедури:

[ IN | OUT | INOUT ] ім’я\_параметру тип  
Параметр, позначений як IN, передає значення у процедуру. OUT-параметр передає значення у точку виклику процедури. Параметр, позначений як INOUT, задається при виклику, може бути змінений всередині процедури і зчитаний після її завершення. Типом параметру може бути будь-який із типів даних, що підтримується MySQL.

параметри\_функції: ім’я\_параметру тип

У випадку функцій параметри використовують лише для передачі значень у функцію.

При створенні процедур і функцій можна вказувати їхні додаткові характеристики.

характеристика:

LANGUAGE SQL  
| [NOT] DETERMINISTIC  
| {CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA} | SQL SECURITY {DEFINER | INVOKER}  
| COMMENT 'короткий опис процедури'

DETERMINISTIC

Вказує на те, що процедура обробляє дані строго визначеним (детермінованим) чином. Тобто, залежно від вхідних даних, процедура повертає один і той самий результат. Недетерміновані процедури містять функції типу NOW() або RAND(), і результат їх виконання не можна передбачити. За замовчуванням всі процедури і функції є недетермінованими.

CONTAINS SQL | NO SQL

Вказує на те, що процедура містить (за замовчуванням), або не містить директиви SQL.

READS SQL DATA

Вказує на те, що процедура містить директиви, які тільки зчитують дані з таблиць.

MODIFIES SQL DATA

Вказує на те, що процедура містить директиви, які можуть змінювати дані в таблицях.

SQL SECURITY  
Задає рівень прав доступу, під яким буде виконуватись процедура. DEFINER – з правами автора процедури (задано за замовчуванням), INVOKER – з правами користувача, який викликає процедуру. Щоб запускати збережені процедури і функції, користувач повинен мати права EXECUTE.

При створенні процедур і функцій у командному рядку клієнта MySQL, потрібно перевизначити стандартний символ завершення вводу директив ";", щоб мати можливість ввести всі директиви процедури. Це робиться за допомогою команди DELIMITER. Наприклад,

DELIMITER |  
означає, що завершення вводу процедури буде позначатись символом "|".

Нижче наведено синтаксис додаткових директив MySQL, які дозволяють розробляти нескладні програми на мові SQL.

DECLARE назва\_змінної тип\_змінної [DEFAULT значення\_за\_замовчуванням]

Оголошення змінної заданого типу.

SET назва\_змінної = вираз Присвоєння змінній значення.

IF умова THEN директиви  
[ELSEIF умова THEN директиви] ... [ELSE директиви2]  
END IF

Умовний оператор. Якщо виконується вказана умова, то виконуються відповідні їй директиви, в протилежному випадку виконуються директиви2.

CASE вираз  
WHEN значення1 THEN директиви1  
[WHEN значення2 THEN директиви2] ... [ELSE директиви3]  
END CASE

Оператор умовного вибору. Якщо вираз приймає значення1, виконуються директиви1, якщо приймає значення2 – виконуються директиви2, і т.д. Якщо вираз не прийме жодного зі значень, виконуються директиви3.

[мітка:] LOOP директиви  
END LOOP

Оператор безумовного циклу. Вихід з циклу виконується командою LEAVE

REPEAT

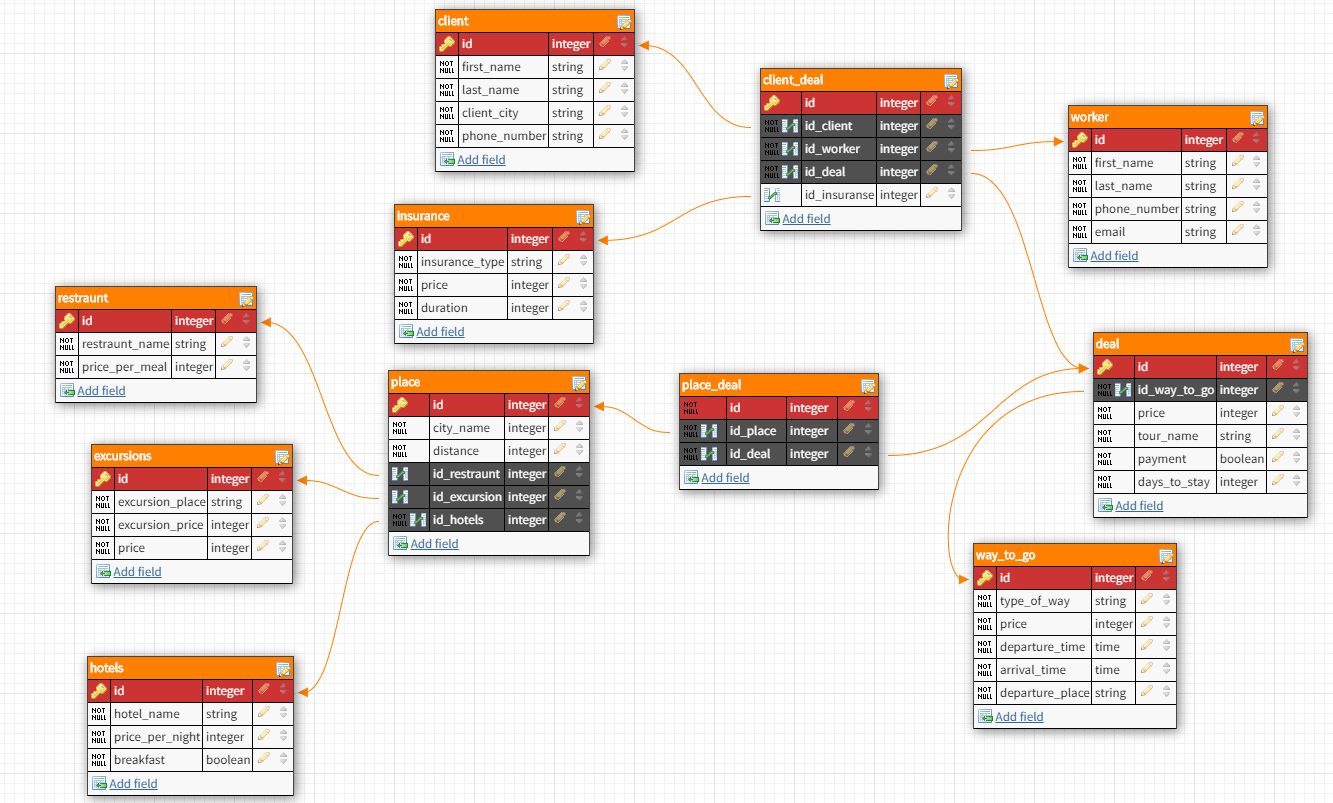
директиви UNTIL умова END REPEAT

WHILE умова DO директиви  
END WHILE

мітка.

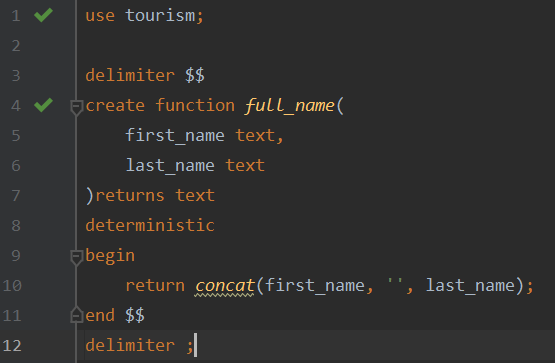
Оператори REPEAT і WHILE дозволяють організувати умовні цикли, які завершуються при виконанні деякої умови.

**Хід роботи**

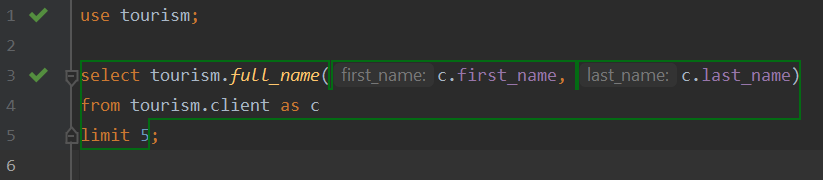


*Рис 1. Діаграма бд за моїм варіантом*

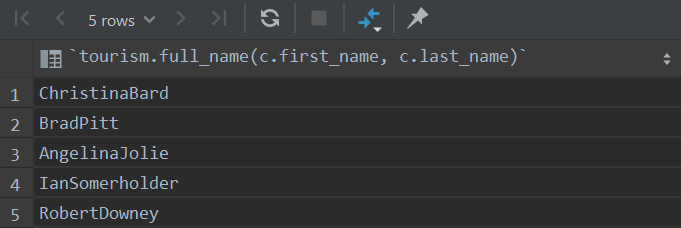
1. Функція конкатенації імені та прізвища:



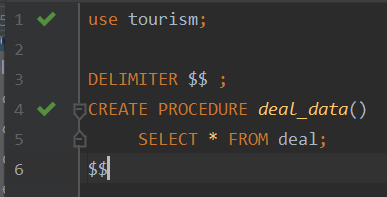
Перевіримо її роботу:



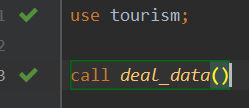
Результат виконання:



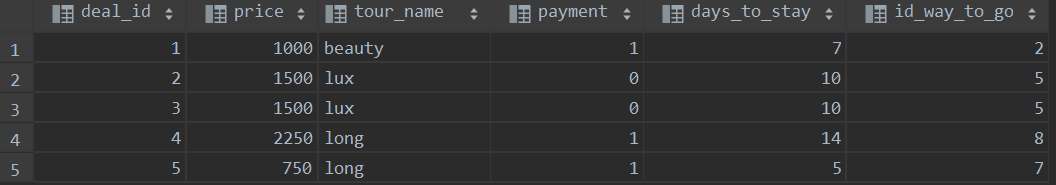
1. Процедура, виводу всіх даних таблиці deal:

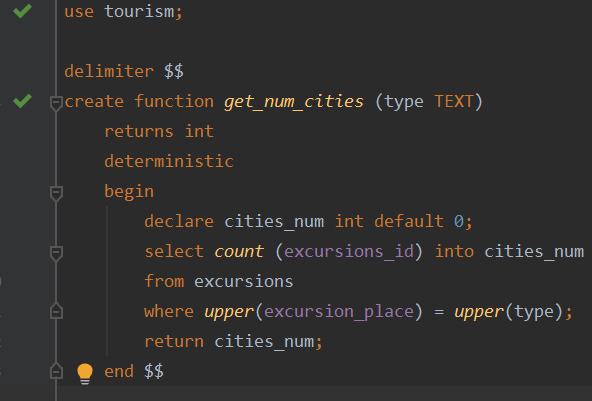


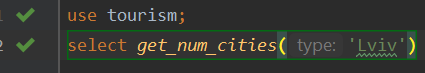
Перевіримо її роботу:



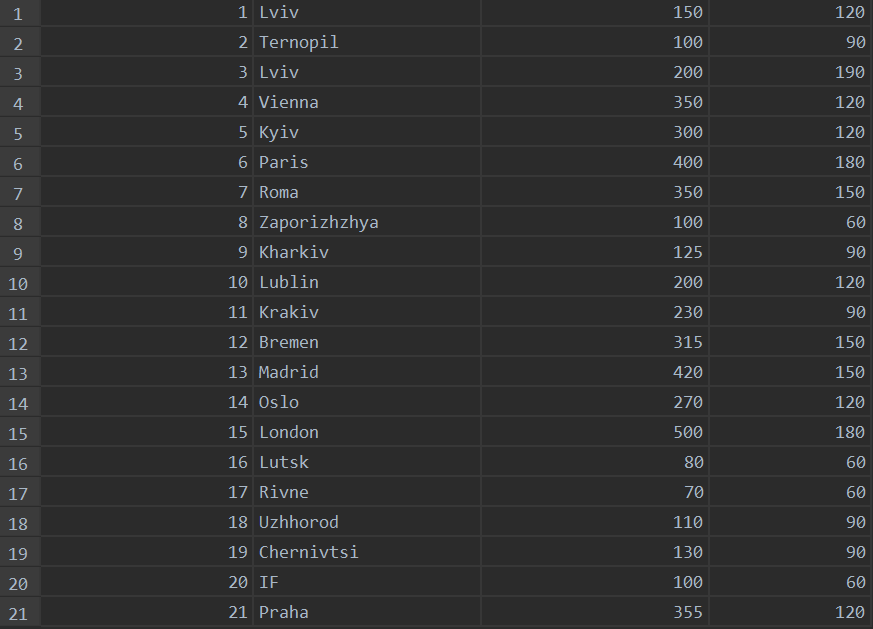
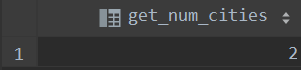
Результат виконання:



1. Функція, яка рахує кількість екскурсій у певному місті: 

Перевіряємо роботу: cre

Результат виконання:

Наша таблиця екскурсій, бачимо що Львів зустрічається лише двічі. 

**Висновок:** під час виконання даної лабораторної роботи я навчилася розробляти та використовувати збережені процедури і функції у СУБД MySQL.