МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут ІКНІ

Кафедра Систем Штучного Інтелекту

Організація баз даних та знань

Лабораторна робота

Виконав:

Кіндрат В.Р.

КН-209

Викладач:

Мельникова Н. І.

Львів – 2020

Лабораторна робота №10

з курсу “ОБДЗ”

на тему: “Написання збережених процедур на мові SQL”

Мета роботи: 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.

Хід роботи

Напишемо функцію, яка буде по імені працівника підраховувати середню заробітню плату за один робочий день.

```
DELIMITER $
CREATE FUNCTION get_salary_by_day(name VARCHAR(25))
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE salary_by_day INT DEFAULT 0;
    SELECT salary / work_days INTO salary_by_day
    FROM worker
    WHERE first_name = name;
RETURN salary_by_day;
END$
DELIMITER ;
```

```
mysql> DELIMITER $
mysql> CREATE FUNCTION get_salary_by_day(name VARCHAR(25))
    -> RETURNS INT
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE salary_by_day INT DEFAULT 0;
    -> SELECT salary / work_days INTO salary_by_day
    -> FROM worker
    -> WHERE first_name = name;
    -> RETURN salary_by_day;
    -> END$
Query OK, 0 rows affected (0.11 sec)

mysql> DELIMITER ;
```

Для перевірки правильності виконання функції створимо працівника «Vova», в якого буде 15 робочих днів, та зарплата 15000:

```
mysql> INSERT INTO worker (ID, first_name,second_name, post, work_days, salary
)
    -> VALUES ('1','Vova','Kindrat', 'cleaner', '15', '15000');
Query OK, 1 row affected (0.24 sec)
```

Отже, зарплата за день повинна бути 1000:

```
mysql> SELECT get_salary_by_day('Vova');
+-----------------------------+
| get_salary_by_day('Vova') |
+-----------------------------+
|                        1000 |
+-----------------------------+
1 row in set (0.02 sec)

mysql> _
```

Як бачимо, функція працює коректно.

Створимо процедуру, яка буде підраховувати загальну суму витрачену на замовлення номерів в готелі від певної дати до теперішньої.

```
DELIMITER $
CREATE PROCEDURE summary (IN id INT, IN date DATE)
BEGIN
    DECLARE error VARCHAR(25);
    SET error =  'incorrect date';
    IF (date < CURDATE()) THEN
    BEGIN
        SELECT SUM(price) as total_price
        FROM `order`
        WHERE client_id = id
        AND entry_date BETWEEN date AND CURDATE();
    END;
    ELSE SELECT error;
    END IF;
END$
    DELIMITER ;
```

```
mysql> DELIMITER $
mysql> CREATE PROCEDURE summary (IN id INT, IN date DATE)
    -> BEGIN
    -> DECLARE error VARCHAR(25);
    -> SET error =  'incorrect date';
    -> IF (date < CURDATE()) THEN
    -> BEGIN
    -> SELECT SUM(price) as total_price
    -> FROM `order`
    -> WHERE client_id = id
    -> AND entry_date BETWEEN date AND CURDATE();
    -> END;
    -> ELSE SELECT error;
    -> END IF;
    -> END$
Query OK, 0 rows affected (0.11 sec)

mysql> DELIMITER ;
```

Запишемо деякі дані в таблицю для перевірки:

```
mysql> INSERT INTO `order` ()
    -> VALUES (1,1,1,'14-06-22','14-06-26',1,5200,0,'ukrainian');
Query OK, 1 row affected (0.05 sec)

mysql> INSERT INTO `order` ()
    -> VALUES (2,1,1,'16-03-12','16-03-16',1,5200,0,'ukrainian');
Query OK, 1 row affected (0.07 sec)

mysql> INSERT INTO `order` ()
    -> VALUES (3,2,2,'15-11-04','15-11-14',4,23000,0,'ukrainian');
Query OK, 1 row affected (0.24 sec)

mysql> INSERT INTO `order` ()
    -> VALUES (4,2,1,'18-01-02','18-01-04',`,4600,0,'ukrainian');
    `> ;
    `> ^C
mysql> INSERT INTO `order` ()
    -> VALUES (4,2,1,'18-01-02','18-01-04',1,4600,0,'ukrainian');
Query OK, 1 row affected (0.07 sec)

mysql> SELECT * FROM `order`;
+----+---------+-----------+------------+---------------+-----------------+-------+----------------+------------+
| id | room_id | client_id | entry_date | eviction_date | number_of_people | price | services_price | language   |
+----+---------+-----------+------------+---------------+-----------------+-------+----------------+------------+
|  1 |       1 |         1 | 2014-06-22 | 2014-06-26    |               1 |  5200 |              0 | ukrainian |
|  2 |       1 |         1 | 2016-03-12 | 2016-03-16    |               1 |  5200 |              0 | ukrainian |
|  3 |       2 |         2 | 2015-11-04 | 2015-11-14    |               4 | 23000 |              0 | ukrainian |
|  4 |       2 |         1 | 2018-01-02 | 2018-01-04    |               1 |  4600 |              0 | ukrainian |
+----+---------+-----------+------------+---------------+-----------------+-------+----------------+------------+
```

Перевіримо роботу процедури від 14-01-01:

```
mysql> CALL summary(1,'14-01-01');
+--------------+
| total_price |
+--------------+
|        15000 |
+--------------+
1 row in set (0.02 sec)
```

Також впишемо дату, аби не всі замовлення входили в вибраний проміжок часу 160-01-01:

```
mysql> CALL summary(1,'16-01-01');
+--------------+
| total_price |
+--------------+
|         9800 |
+--------------+
1 row in set (0.00 sec)
```

Висновок:

Під час виконання лабораторної роботи я навчився розробляти та виконувати збережені процедури та функції у MySQL