

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Кафедра «Системи штучного інтелекту»



Звіт

до лабораторної роботи №12

З дисципліни «Організація баз даних та знань»

Виконала:

студентка групи КН-209

Міхняк Софія

Прийняла:

Мельникова Н. І.

Львів-2020

Лабораторна робота №12

“Розробка та застосування тригерів”

Мета роботи: Розробити SQL запити, які моделюють роботу тригерів: каскадне знищення, зміна та доповнення записів у зв’язаних таблицях.

Короткі теоретичні відомості.

Тригер – це спеціальний вид користувацької процедури, який виконується автоматично при певних діях над таблицею, наприклад, при додаванні чи оновленні даних. Кожен тригер асоційований з конкретною таблицею і подією. Найчастіше тригери використовуються для перевірки коректності вводу нових даних та підтримки складних обмежень цілісності. Крім цього їх використовують для автоматичного обчислення значень полів таблиць, організації перевірок для захисту даних, збирання статистики доступу до таблиць баз даних чи реєстрації інших подій.

Для створення тригерів використовують директиву CREATE TRIGGER.

Синтаксис:

CREATE

[DEFINER = { *користувач* | CURRENT_USER }]

TRIGGER *ім'я_тригера* *час_виконання подія_виконання*

ON *назва_таблиці* FOR EACH ROW *тіло_тригера*

Аргументи:

DEFINER

Задає автора процедури чи функції. За замовчуванням – це CURRENT_USER.

ім'я_тригера

Ім'я тригера повинно бути унікальним в межах однієї бази даних.

час_виконання

Час виконання тригера відносно події виконання. BEFORE – виконати тіло тригера до виконання події, AFTER – виконати тіло тригера після події.

подія_виконання

Можлива подія – це внесення (INSERT), оновлення (UPDATE), або видалення (DELETE) рядка з таблиці. Один тригер може бути пов’язаний лише з однією подією. Команда AFTER INSERT, AFTER UPDATE, AFTER DELETE визначає виконання тіла тригера відповідно після внесення, оновлення, або видалення даних з таблиці. Команда BEFORE

INSERT, BEFORE UPDATE, BEFORE DELETE визначає виконання тіла тригера відповідно до внесення, оновлення, або видалення даних з таблиці.

ON *назва_таблиці*

Таблиця, або віртуальна таблиця (VIEW), для якої створюється даний тригер. При видаленні таблиці з бази даних, автоматично видаляються всі пов'язані з нею тригери.

FOR EACH ROW *тіло_тригера*

Задає набір SQL директив, які виконує тригер. Тригер викликається і виконується для кожного зміненого рядка. Директиви можуть об'єднуватись командами BEGIN ... END та містити спеціальні команди OLD та NEW для доступу до попереднього та нового значення поля у зміненому рядку відповідно. В тілі тригера дозволено викликати збережені процедури, але заборонено використовувати транзакції, оскільки тіло тригера автоматично виконується як одна транзакція.

NEW.*назва_поля*

Повертає нове значення поля для зміненого рядка. Працює лише при подіях INSERT та UPDATE. У тригерах, які виконуються перед (BEFORE) подією можна змінити нове значення поля командою SET

NEW.*назва_поля* = *значення*.

OLD.*назва_поля*

Повертає старе значення поля для зміненого рядка. Можна використовувати лише при подіях UPDATE та DELETE. Змінити старе значення поля не можливо.

Щоб видалити створений тригер з бази даних, потрібно виконати команду DROP TRIGGER *назва_тригера*.

Хід роботи.

Розробляємо три тригери:

1. Каскадне оновлення таблиці order_cocktail при видаленні з таблиці замовлень.
 2. Шифрування паролю користувача під час внесення в таблицю.
 3. Збереження замовлень в архіві перед видаленням замовлення.
-
1. Створюємо тригер, який при видаленні з таблиці consumer_order буде видаляти рядок order_cocktail з цим замовленням і тригер, який

після видалення буде додавати до архіву замовлень видалене замовлення.

Тригер каскадного оновлення таблиці order_cocktail при видаленні з таблиці замовлень:

```
drop trigger if exists before_order_delete;
```

```
DELIMITER $$
```

```
CREATE TRIGGER before_order_delete  
BEFORE DELETE  
ON consumer_order FOR EACH ROW  
BEGIN  
    delete from order_cocktail where order_id = OLD.id;  
END$$
```

```
DELIMITER ;
```

Тригер збереження замовлень в архіві перед видаленням замовлення:

```
drop trigger if exists after_order_delete;
```

```
DELIMITER $$
```

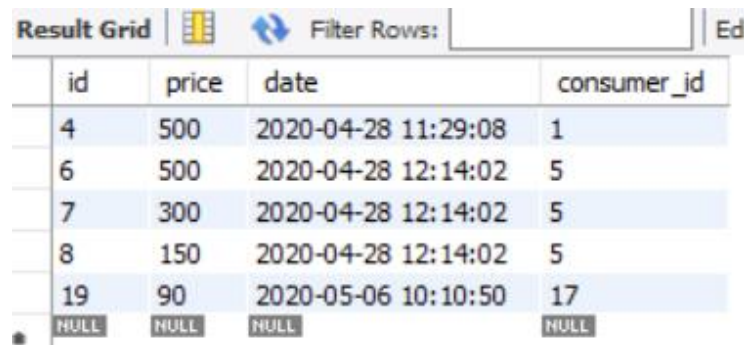
```
CREATE TRIGGER after_order_delete  
AFTER DELETE  
ON consumer_order FOR EACH ROW  
BEGIN  
    insert into order_archive(id, price, date, consumer_id)  
    value (OLD.id, OLD.price, OLD.date, OLD.consumer_id);  
END$$
```

```
DELIMITER ;
```

Перевірка роботи тригера

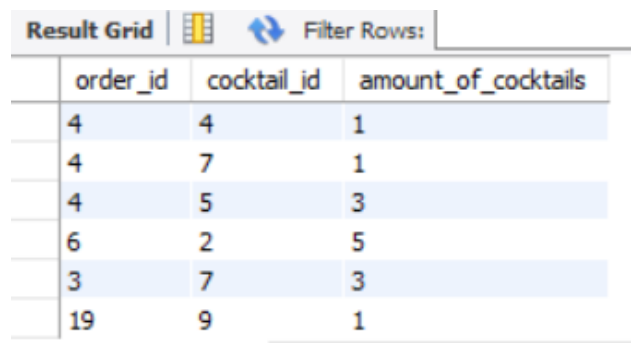
delete from consumer_order where id=17;

Таблиці consumer_order(рис.1) і order_cocktail(рис.2) до змін:



id	price	date	consumer_id
4	500	2020-04-28 11:29:08	1
6	500	2020-04-28 12:14:02	5
7	300	2020-04-28 12:14:02	5
8	150	2020-04-28 12:14:02	5
19	90	2020-05-06 10:10:50	17
NULL	NULL	NULL	NULL

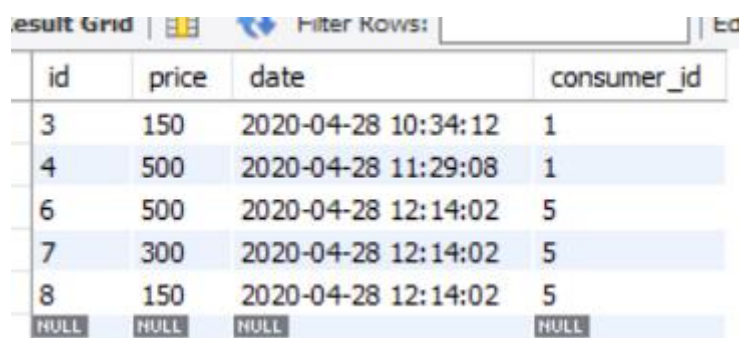
Рис.1 Таблиця consumer_order



order_id	cocktail_id	amount_of_cocktails
4	4	1
4	7	1
4	5	3
6	2	5
3	7	3
19	9	1

Рис.2 Таблиця order_cocktail

Таблиці після видалення зображені на рис.3-5. Заодно подала таблицю архіву замовлень



id	price	date	consumer_id
3	150	2020-04-28 10:34:12	1
4	500	2020-04-28 11:29:08	1
6	500	2020-04-28 12:14:02	5
7	300	2020-04-28 12:14:02	5
8	150	2020-04-28 12:14:02	5
NULL	NULL	NULL	NULL

Рис.3 Таблиця consumer_order

Result Grid Filter Rows:			
	order_id	cocktail_id	amount_of_cocktails
	8	7	5
	4	4	1
	4	7	1
	4	5	3
	6	2	5
	3	7	3

Рис.4 Таблиця order_cocktail

Result Grid Filter Rows: Ed				
	id	price	date	consumer_id
▶	19	90	2020-05-06 10:10:50	17
★	NULL	NULL	NULL	NULL

Рис.5 Таблиця order_archive

2. Шифрування паролю користувача під час внесення в таблицю.

Створюємо тригер, який буде шифрувати пароль під час внесення в таблицю.

DELIMITER \$\$

CREATE

TRIGGER user_password BEFORE INSERT ON drinmix.consumer FOR EACH ROW

BEGIN

SET NEW.password = AES_ENCRYPT(NEW.password, 'key-key');

END\$\$

DELIMITER ;

Перевірка роботи тригера

```
INSERT INTO consumer(name, email, password, gender, age) VALUE
('user60',
'user60@kml.com', 'u60pass', 'male', 27);
```

```
select *
from consumer;
```

Таблиця користувачів до вставлення:

Result Grid						
Filter Rows:				Edit:		
	id	name	email	password	gender	age
	13	user41	user41@kml.com	BLOB	male	33
	15	user23	user23@kml.com	BLOB	male	26
	16	user24	user24@kml.com	BLOB	male	26
	17	user533	user533@kml.com	BLOB	female	16
	18	user37	user37@kml.com	BLOB	male	43
	NULL	NULL	NULL	NULL	NULL	NULL

Таблиця після вставлення:

	id	name	email	password	gender	age
	15	user23	user23@kml.com	BLOB	male	26
	16	user24	user24@kml.com	BLOB	male	26
	17	user533	user533@kml.com	BLOB	female	16
	18	user37	user37@kml.com	BLOB	male	43
	37	user60	user60@kml.com	BLOB	male	27
	NULL	NULL	NULL	NULL	NULL	NULL

Висновок: виконавши дану лабораторну роботу навчилась створювати і використовувати тригери.