МІНІСТЕРСТВО ОСВІТИ І НАУКИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Інститут комп'ютерних наук та інформаційних технологій
Кафедра систем штучного інтелекту

Лабораторна робота №12

з дисципліни “ ОБДЗ ”

На тему: «Розробка »

Виконав:
ст. гр. КН-211
Ільків Андрій
Викладач:
Якимишин Х. М.

Львів – 2020

Мета роботи: Навчитися використовувати механізм транзакцій у СУБД MySQL. Розробити SQL запити, які виконуються як єдине ціле в рамках однієї транзакції.

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 видалені 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:

Потрібно розробити тригери, які виконуватимуть наступні дії.

1. Каскадне оновлення таблиці medicine при видаленні препарату з таблиці all_medicine.
2. Шифрування паролю користувача під час внесення в таблицю.
3. Тригер для таблиці customer_session, який буде фіксувати у таблиці customer дату останнього входу користувача в систему.

1. Каскадне оновлення таблиці studentsgroup при видаленні категорії з таблиці category. Діюче обмеження зовнішнього ключа при видалені препарату встановлює для користувача невизначену роль (значення NULL). Натомість, за допомогою тригера, групі потрібно присвоювати певну категорію за замовчуванням.

```
CREATE
    TRIGGER  category_delete
 BEFORE DELETE ON bass.Category FOR EACH ROW
    UPDATE bass.studentsgroup SET category_id = 4 WHERE
        category_id = old.category_id;
```

Перевіримо роботу тригера, видаливши роль з номером 2:

Так виглядає таблиця medicine перед видаленням:

group_id	teacher_id	lectionId	category_id
1	1	12	3
2	1	15	3
3	3	9	1
4	2	21	2

Тепер застосуємо тригер:

```
DELETE FROM category
WHERE
    category_id = 3;
```

Перевіряємо, чи з’явилися зміни:

group_id	teacher_id	lectionId	category_id
1	1	12	4
2	1	15	4
3	3	9	1
4	2	21	2

2. Створимо тригер, для шифрування паролю користувача під час внесення в таблицю.

```
CREATE
    TRIGGER  pass_enc
BEFORE INSERT ON student FOR EACH ROW
    SET new . password = HEX(AES_ENCRYPT(new.password, 'keey'));
```

Такий вигляд має таблиця student:

student_id	first_name	last_name	group_id	login	password	lastseen
1	Vitalii	Solyridze	1	vtko	123qwe	NULL
2	Viktor	Kachmaryk	4	bestlol	kiko	NULL
3	Melanii	Volskiy	3	amazoff	stronk23	NULL
4	Klasniy	Perec	2	uberli	987yh	NULL
5	Veri	Goodovich	2	da	da	NULL
6	Vova	Kupec	1	no	no	NULL

Тепер вставимо нового користувача:

```
insert into student (student_id,first_name, last_name,group_id, login, password) values (7, 'num', 'ber', 2, 'kekion', 'bloblob');
```

Перевіримо таблицю юзерів:

student_id	first_name	last_name	group_id	login	password
2	Viktor	Kachmaryk	4	bestlol	kiko
3	Melanii	Volskiy	3	amazoff	stronk23
4	Klasniy	Perec	2	uberli	987yh
5	Veri	Goodovich	2	da	da
6	Vova	Kupec	1	no	no
7	num	ber	2	kekion	66452B205068825AB6D9082CCE4571DD

3. Тригер для таблиці user_entry, який буде фіксувати у таблиці student дату останнього входу користувача в систему.

```
CREATE
    TRIGGER  last_seen
 AFTER INSERT ON user_entry FOR EACH ROW
    UPDATE student SET student.lastseen = DATE(new.entry_time) WHERE
        student_id = new.student_id;
```

Перевіримо таблицю користувачів:

student_id	first_name	last_name	group_id	login	password	lastseen
2	Viktor	Kachmaryk	4	bestlol	kiko	NULL
3	Melanii	Volskiy	3	amazoff	stronk23	NULL
4	Klasniy	Perec	2	uberli	987yh	NULL
5	Veri	Goodovich	2	da	da	NULL
6	Vova	Kupec	1	no	no	NULL
7	num	ber	2	kekion	66452B205068825AB6D9082CCE4571DD	NULL

Вставимо в таблицю customer_session нові значення:

```
insert into user_entry values (1, '20200520', 3),(2, '20200618', 5);
```

Тепер подивимось на таблицю користувачів:

student_id	first_name	last_name	group_id	login	password	lastseen
2	Viktor	Kachmaryk	4	bestlol	kiko	NULL
3	Melanii	Volskiy	3	amazoff	stronk23	2020-05-20 00:00:00
4	Klasniy	Perec	2	uberli	987yh	NULL
5	Veri	Goodovich	2	da	da	2020-06-18 00:00:00
6	Vova	Kupec	1	no	no	NULL
7	num	ber	2	kekion	66452B205068825AB6D9082CCE4571DD	NULL

Висновок: на цій лабораторній роботі було розглянуто тригери, їх призначення, створення та використання. Було розроблено тригери для таблиць entry_user, student та studentsgroup.