МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра систем штучного інтелекту

ЗВІТ № 11

з курсу “ОБДЗ”

на тему:

«Розробка та застосування транзакцій»

Виконала:

студентка групи КН-211

Лаврик Юліана

Викладач:

Якимишин Х.М.

Лабораторна робота № 11

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

Короткі теоретичні відомості

Транзакція – це сукупність директив SQL, які виконуються як єдине ціле з можливістю відміни результатів їх виконання. Зміни в таблицях записуються у базу даних лише після успішного виконання всіх директив транзакції. Інакше, всі зроблені зміни ігноруються. Це дозволяє уникати помилок при маніпулюванні великими обсягами записів, зберігати цілісність даних при помилках під час додавання, видалення, модифікації значень у різних таблицях і полях тощо. СУБД MySQL також підтримує глобальні розподілені транзакції, які виконуються на декількох базах даних, або на різних серверах баз даних (XA-транзакції).

Для організації транзакцій в MySQL використовують такі директиви, як SET autocommit, START TRANSACTION, COMMIT і ROLLBACK.

START TRANSACTION

Вказує на початок транзакції. Директива вимикає автоматичне збереження змін для всіх подальших запитів, поки не буде виконано команду COMMIT, або ROLLBACK.

COMMIT

Зберегти зміни, зроблені даною транзакцією.

ROLLBACK

Відмінити дану транзакцію і зроблені нею зміни у базі даних. Слід зауважити, що зміни у схемі бази даних не можна відмінити, тобто результат видалення, зміни або створення таблиці завжди зберігається.

SET autocommit=0

Вимикає автоматичне збереження змін для поточної сесії зв’язку з сервером БД. За замовчуванням, зміни зберігаються автоматично, тобто результат виконання запиту, який змінює таблицю, одразу записується на диск без можливості відміни операції.

AND CHAIN

Одразу після завершення даної транзакції розпочати виконання наступної.

RELEASE

Одразу після виконання даної транзакції завершити поточну сесію зв’язку з сервером.

Транзакції можна розбивати на окремі логічні частини, оголошуючи так звані точки збереження. Це дозволяє відміняти результати виконання не всієї транзакції, а лише тих запитів, які виконувались після оголошеної точки збереження (SAVEPOINT).

SAVEPOINT мітка

Оголошує точку збереження всередині транзакції та задає її назву.

ROLLBACK TO [SAVEPOINT] мітка

Відміняє результати виконання запитів, вказаних після даної точки збереження.

RELEASE SAVEPOINT мітка

Видаляє точку збереження.

Хід роботи

1.Відміна транзакції.

Транзакція складається з двох запитів на додавання нових інгредієнтів з постачальниками "Roshen" та "Aro". При цьому, постачальника "Aro" з id=5 в базі даних не існує, а отже, транзакція не виконується.

```
START TRANSACTION;
    INSERT INTO confectionary.ingredient VALUES
     (NULL, 'DarkChocolate','4kg','300', 4),
    (NULL, 'Sugar','10kg', '14.8', 5);
COMMIT;
```

Відповідь сервера :

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`confectionary`.`ingredient`, CONSTRAINT `supplier_id` FOREIGN KEY (`supplier_id`) REFERENCES `supplier` (`id`) ON DELETE CASCADE ON UPDATE CASCADE)

2.Успішна транзакція.

Транзакція складається з запитів на додавання таких же інгредієнтів з постачальниками "Roshen" та "Aro" після створення у таблиці Supplier постачальника "Aro".

```
INSERT INTO confectionary.supplier VALUES
(5, 'Aro', '(0352)56-11-20', 'aro@aro.com', 'sugar');
START TRANSACTION;
    INSERT INTO confectionary.ingredient VALUES
    (6, 'DarkChocolate','4kg','300', 4),
    (7, 'Sugar','10kg', '14.8', 5);
COMMIT;
```

Результат успішного додавання чотирьох користувачів у таблицю :

SELECT * FROM confectionary.ingredient;

id	name	quantity	price_for_unit	supplier_id
1	Flour	20kg	32.40	2
2	Milk	10l	0.00	1
3	Butter	5kg	222.45	1
4	Eggs	50units	1.74	3
5	Cacao	5kg	32.00	4
6	DarkChocolate	4kg	300.00	4
7	Sugar	10kg	14.80	5

3. Перед початком виконаємо наступну команду, для того, щоб відключити автоматичне збереження змін для поточної сесії зв’язку з сервером БД :

SET autocommit=0;

Переглянемо таблицю dish :

SELECT * FROM confectionary.dish;

id	name	kitchen_name	weight	kkal	price
1	Oreo	European	950g	347	450.00
2	Brownie	American	1kg	466	340.00
3	LemonPie	French	1,5kg	309	420.00
4	Strawberry cake "Fraisier"	French	750g	188	325.00
5	Baunti	England	600g	410	310.00
6	ApplePie	England	500g	360	300.00
7	PearPie	England	620g	452	410.00
8	ChocolatePie	American	531g	622	500.00

Створимо точку збереження, використовуючи наступний запит:

SAVEPOINT SP1;

Тепер виконаємо наступні запити:

UPDATE confectionary.dish SET price = 500.25 WHERE kkal = 347;

UPDATE confectionary.dish SET weight = '920g' WHERE id = 2;

На даний момент таблиця містить такі записи:

id	name	kitchen_name	weight	kkal	price
1	Oreo	European	950g	347	500.25
2	Brownie	American	920g	466	340.00
3	LemonPie	French	1,5kg	309	420.00
4	Strawberry cake "Fraisier"	French	750g	188	325.00
5	Baunti	England	600g	410	310.00
6	ApplePie	England	500g	360	300.00
7	PearPie	England	620g	452	410.00
8	ChocolatePie	American	531g	622	500.00

Створимо ще одну точку збереження :

SAVEPOINT SP2;

І виконаємо наступні запити:

DELETE FROM confectionary.dish WHERE kitchen_name = 'French' ;

DELETE FROM confectionary.dish WHERE id = 7 ;

DELETE FROM confectionary.dish WHERE id = 8;

Тепер таблиця містить такі записи:

id	name	kitchen_name	weight	kkal	price
1	Oreo	European	950g	347	500.25
2	Brownie	American	920g	466	340.00
5	Baunti	England	600g	410	310.00
6	ApplePie	England	500g	360	300.00

Тоді повернемося до точки збереження SP1 за допомогою команди:

ROLLBACK TO SP1;

Після виконання даного запиту, таблиця буде зберігати такі записи:

id	name	kitchen_name	weight	kkal	price
1	Oreo	European	950g	347	450.00
2	Brownie	American	1kg	466	340.00
3	LemonPie	French	1,5kg	309	420.00
4	Strawberry cake "Fraisier"	French	750g	188	325.00
5	Baunti	England	600g	410	310.00
6	ApplePie	England	500g	360	300.00
7	PearPie	England	620g	452	410.00
8	ChocolatePie	American	531g	622	500.00

Так повернулися до стану таблиці на момент створення точки збереження SP1.

Тепер, коли нам більше не потрібні ці точки збереження, можемо їх звільнити:

RELEASE SAVEPOINT SP1;

RELEASE SAVEPOINT SP2;

Висновок: під час виконання даної лабораторної роботи я навчилася використовувати механізм транзакцій у СУБД MySQL. Розробила SQL запити, які виконуються як єдине ціле в рамках однієї транзакції, використовуючи різні директиви для організації транзакцій.