МІНІСТЕРСТВО ОСВІТИ І НАУКИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Кафедра систем штучного інтелекту

Лабораторна робота №11
З курсу “Організація баз даних та знань”

Виконав:
студент групи КН-210
Марій Павло

Викладач:
Мельникова Наталя Іванівна

Львів – 2020

Тема: Розробка та застосування транзакцій.

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

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

В ході роботи, потрібно продемонструвати успішне і неуспішне виконання транзакції.

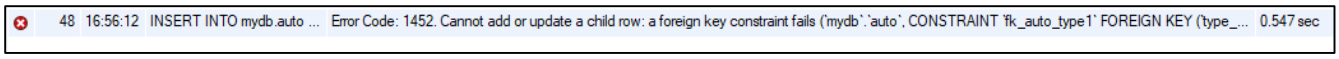
Розробимо транзакцію, яка буде вносити дані в таблицю auto. Транзакція буде відміняти всі зміни у таблицях при виникненні помилки чи іншої суперечливості.

Отже, в таблиці auto є 20 записів. Типів авто є 10. Спробуємо здійснити транзакцію. Будемо додавати дані в таблицю, але одне з значень type_id поставимо 11, що буде некоректним значенням. Повинна отримуватись помилка.

Код транзакції:

```
START TRANSACTION;
    INSERT INTO mydb.auto VALUE (21, 1.8, 180, 'Auto', 3, 8);
    INSERT INTO mydb.auto VALUE (22, 2.0, 200, 'Mechanic', 5, 12);
    INSERT INTO mydb.auto VALUE (23, 6.3, 580, 'Variator', 8, 18);
    INSERT INTO mydb.auto VALUE (24, 6.0, 400, 'T-tronic', 11, 15);
COMMIT;
```

Результат:



Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`mydb`.`auto`, CONSTRAINT `fk_auto_type1` FOREIGN KEY (`type_id`) REFERENCES `type` (`id`))

SELECT * FROM mydb.auto
WHERE id > 17;

Результат:

id	volume_engine	volume_power	transmission	type_id	model_id
18	4.2	450	Auto	7	18
19	4.9	600	T-tronic	10	18
20	1.6	110	T-tronic	9	20
21	1.8	180	Auto	3	8
22	2	200	Mechanic	5	12
23	6.3	580	Variator	8	18
NULL	NULL	NULL	NULL	NULL	NULL

Отримали помилку, бо типу з номером 11 немає, а ми хочемо внести дані, які пов’язані з цим типом.

Записи, які не викликали помилки, додались у таблицю. Проте я не хочу, щоб в таблицю вносились якісь дані, якщо принаймні один запит дає помилку. Тому виконую команду ROLLBACK і зміни, які зробила попередня транзакція, відміняються.

Результат виконання попереднього запиту після ROLLBACK:

id	volume_engine	volume_power	transmission	type_id	model_id
18	4.2	450	Auto	7	18
19	4.9	600	T-tronic	10	18
20	1.6	110	T-tronic	9	20
NULL	NULL	NULL	NULL	NULL	NULL

Тепер проведемо ту саму транзакцію, проте перед тим додамо в таблицю type 11-й тип кузова.

Код скрипта:

INSERT INTO mydb.type VALUE (11 ,'Ліфтбек', 'Деякий опис типу ліфтбек');

```
START TRANSACTION;
        INSERT INTO mydb.auto VALUE (21, 1.8, 180, 'Auto', 3, 8);
        INSERT INTO mydb.auto VALUE (22, 2.0, 200, 'Mechanic', 5, 12);
        INSERT INTO mydb.auto VALUE (23, 6.3, 580, 'Variator', 8, 18);
        INSERT INTO mydb.auto VALUE (24, 6.0, 400, 'T-tronic', 11, 15);
 COMMIT;
```

Результат:

✓	61	17:11:49	INSERT INTO mydb.type ...	1 row(s) affected
✓	62	17:11:49	START TRANSACTION	0 row(s) affected
✓	63	17:11:49	INSERT INTO mydb.auto ...	1 row(s) affected
✓	64	17:11:49	INSERT INTO mydb.auto ...	1 row(s) affected
✓	65	17:11:49	INSERT INTO mydb.auto ...	1 row(s) affected
✓	66	17:11:49	INSERT INTO mydb.auto ...	1 row(s) affected
✓	67	17:11:49	COMMIT	0 row(s) affected

Виберемо авто, id яких більше 17:

Result Grid | Filter Rows: | Edit: | Expo

id	volume_engine	volume_power	transmission	type_id	model_id
18	4.2	450	Auto	7	18
19	4.9	600	T-tronic	10	18
20	1.6	110	T-tronic	9	20
21	1.8	180	Auto	3	8
22	2	200	Mechanic	5	12
23	6.3	580	Variator	8	18
24	6	400	T-tronic	11	15
NULL	NULL	NULL	NULL	NULL	NULL

Як бачимо, транзакція пройшла успішно і всі команди виконались правильно.

Висновок: На цій лабораторній роботі я ознайомився із механізмом транзакцій у СУБД MySQL.