МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Лабораторна робота №11

З дисципліни «Організація баз даних та знань»

Виконав:
студент групи КН-210
Бікєєв Андрій

Викладач:
Мельникова Н. І.

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.

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

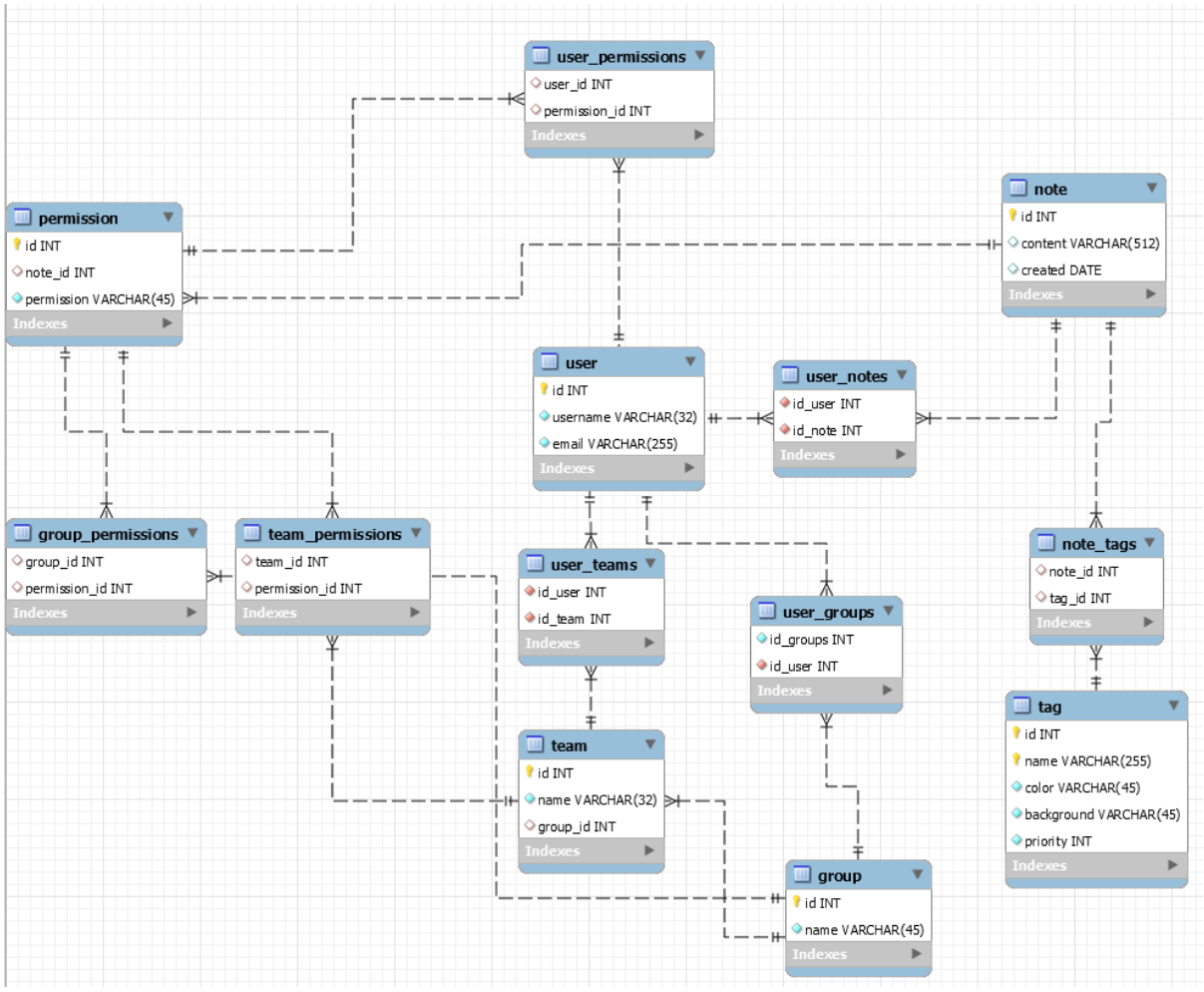


Рис 1. ER-діаграма бази даних

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

Розробимо транзакцію, яка буде вносити дані в таблицю tag. Транзакція буде відміняти всі зміни у таблицях при виникнені помилки.

Отже, в таблиці permission є поле note_id, яке повинно вказувати на ідентифікатор замітки, до якої належить цей «дозвіл». У табилці note є 25 записів. Спробуємо здійснити транзакцію. Будемо додавати дані в таблицю, але одне з значень note_id виставимо 26, воно буде некоректне, тому що замітки з таким ідентифікатором в таблиці note немає. Виконання цьої транзакції повинно викликати помилку через некоректність вхідних даних.

Код транзакції з некоректними даними:

```
start transaction;
    insert into permission (permission, note_id)
        values("read", 11);
    insert into permission (permission, note_id)
        values("write", 26);
commit;
```

Результат виконання:

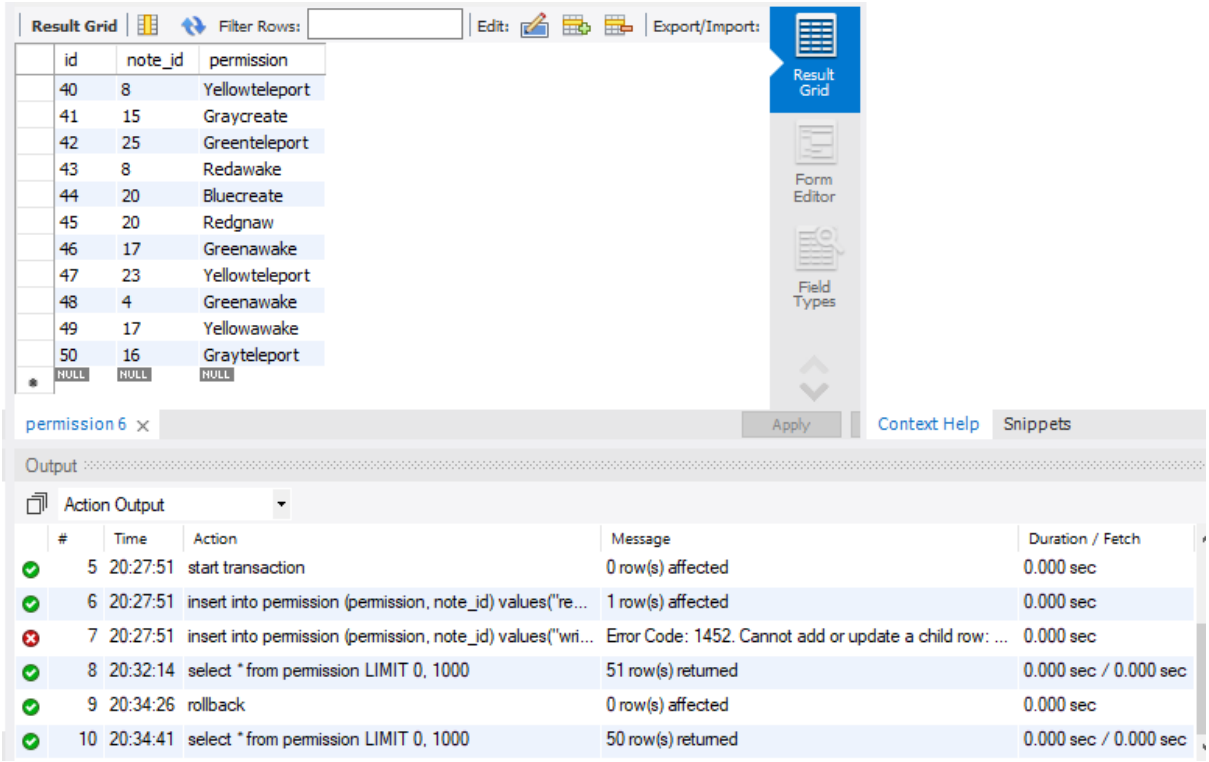
	#	Time	Action	Message	Duration / Fetch
✓	2	20:23:49	show tables	13 row(s) returned	0.000 sec / 0.000 sec
✓	3	20:23:56	select count(*) from permission LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
✓	4	20:24:07	select count(*) from note LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
✓	5	20:27:51	start transaction	0 row(s) affected	0.000 sec
✓	6	20:27:51	insert into permission (permission, note_id) values("re...	1 row(s) affected	0.000 sec
✗	7	20:27:51	insert into permission (permission, note_id) values("wri...	Error Code: 1452. Cannot add or update a child row: ...	0.000 sec

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`noteworthy`.`permission`, CONSTRAINT `note's id` FOREIGN KEY (`note_id`) REFERENCES `note` (`id`))

Як видно на скріншоті, записи, які не викликали помилки, додались у таблицю.

id	note_id	permission
41	15	Graycreate
42	25	Greenteleport
43	8	Redawake
44	20	Bluecreate
45	20	Redgnaw
46	17	Greenawake
47	23	Yellowteleport
48	4	Greenawake
49	17	Yellowawake
50	16	Grayteleport
51	11	read
NULL	NULL	NULL

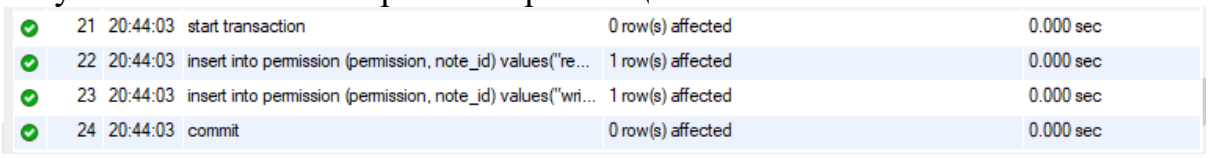
Виконаємо команду *ROLLBACK*, і зміни, які зробила попередня транзакція відміняються.

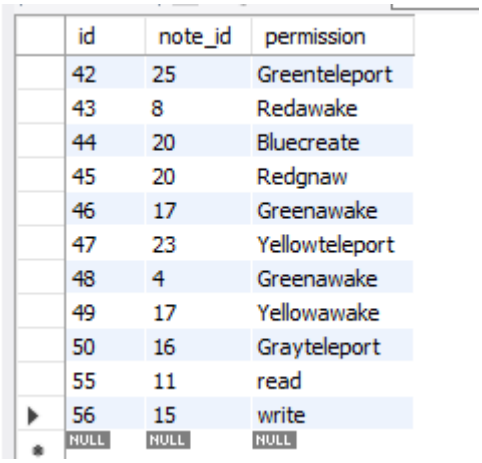


id	note_id	permission
40	8	Yellowteleport
41	15	Graycreate
42	25	Greenteleport
43	8	Redawake
44	20	Bluecreate
45	20	Redgnaw
46	17	Greenawake
47	23	Yellowteleport
48	4	Greenawake
49	17	Yellowawake
50	16	Grayteleport
NULL	NULL	NULL

Виконаємо ту саму транзакцію, але з коректними данними.

Результат виконання коректної транзакції:





id	note_id	permission
42	25	Greenteleport
43	8	Redawake
44	20	Bluecreate
45	20	Redgnaw
46	17	Greenawake
47	23	Yellowteleport
48	4	Greenawake
49	17	Yellowawake
50	16	Grayteleport
55	11	read
56	15	write
NULL	NULL	NULL

Висновок: на цій лабораторній роботі я ознайомився з механізмом транзакцій у СУБД MySQL.