Temat: Transakcje, blokady i zakleszczenia.

Przykład 1

1. W bazie danych ***hurtownia_nazwisko*** zaprojektuj nową tabelę ***pracownicy***, która będzie obsługiwała transakcje. Pamiętaj o wymuszeniu silnika InnoDB. Aby widoczne były polskie znaki wpisz DEFAULT CHARSET=utf8.

id	imie	nazwisko	wiek
1	Jan	Nowak	24
2	Ewa	Werner	30
3	Marcin	Tracz	33
4	Łucja	Kowalska	26
5	Paweł	Nowak	23

2. Wyświetl zawartość tabeli *pracownicy*.
3. Zmodyfikuj dane w tabeli tak, aby wiek wszystkich pracowników o id większym od 2 zmienić na 18. Obejmij tą instrukcję transakcją.

   START TRANSACTION;
   UPDATE *pracownicy* SET wiek = 18 WHERE id > 2;

4. Sprawdź zawartość tabeli pracownicy.

```
mysql> SELECT * FROM pracownicy;
+----+--------+----------+------+
| id | imie   | nazwisko | wiek |
+----+--------+----------+------+
|  1 | Jan    | Nowak    |   24 |
|  2 | Ewa    | Werner   |   30 |
|  3 | Marcin | Tracz    |   18 |
|  4 | Łucja  | Kowalska |   18 |
|  5 | Paweł  | Nowak    |   18 |
+----+--------+----------+------+
```

5. Wycofaj zmianę, stosując słowo kluczowe **ROLLBACK**.
6. Sprawdź jeszcze raz zawartość tabeli pracownicy. Wiek pracowników powrócił do poprzedniej wartości.

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT * FROM pracownicy;
+----+--------+----------+------+
| id | imie   | nazwisko | wiek |
+----+--------+----------+------+
|  1 | Jan    | Nowak    |   24 |
|  2 | Ewa    | Werner   |   30 |
|  3 | Marcin | Tracz    |   33 |
|  4 | Łucja  | Kowalska |   26 |
|  5 | Paweł  | Nowak    |   23 |
+----+--------+----------+------+
```

Ćwiczenie 1

Wzorując się na przykładzie 1 obejmij transakcją instrukcję dopisywania nowego pracownika. Dopisz siebie i zatwierdź transakcję.

PUNKTY PRZYWRACANIA

Podczas wykonywania transakcji możemy określić również punkty zapisu, tzw. **SAVEPOINT**. Po słowie kluczowym SAVENPOINT używamy identyfikatora.

SAVEPOINT *nazwa_punktu_przywracania;*

Dzięki zdefiniowaniu identyfikatorów punktów zapisu możemy cofnąć zmiany przeprowadzane przez transakcję do określonego za pomocą identyfikatora punktu. Aby cofnąć zmiany do zdefiniowanego wcześniej punktu SAVENPOINT, używamy polecenia:

ROLLBACK TO SAVEPOINT *nazwa_punktu_przywracania;*

Aby usunąć punkt przywracania używamy polecenia:

RELEASE SAVEPOINT *nazwa_punktu_przywracania*;

KONTROLA TRANSAKCJI

Podczas wykonywania transakcji możliwy jest podział ich na mniejsze części za pomocą słowa kluczowego **SAVEPOINT**. Całość transakcji można nadal wycofać używając słowa kluczowego **ROLLBACK**.

Przykład 2

Obejmij transakcją dwie instrukcje: zmień imiona wszystkich pracowników o id>1 na Marcin i nazwiska na Zagłoba. Po każdej z nich zdefiniuj punkt przywracania.

1. Definiujemy pierwszy punkt przywracania:

   START TRANSACTION;
   UPDATE *pracownicy* SET imie = 'Marcin' WHERE id > 1;
   SAVEPOINT *punkt_pierwszy;*

2. Wyświetlamy zawartość tabeli:

id	imie	nazwisko	wiek
1	Jan	Nowak	24
2	Marcin	Werner	30
3	Marcin	Tracz	33
4	Marcin	Kowalska	26
5	Marcin	Nowak	23

3. Definiujemy drugi punkt przywracania:

   UPDATE *pracownicy* SET nazwisko = 'Zagłoba' WHERE id > 1;
   SAVEPOINT *punkt_drugi;*

4. Wyświetlamy zawartość tabeli:

id	imie	nazwisko	wiek
1	Jan	Nowak	24
2	Marcin	Zagłoba	30
3	Marcin	Zagłoba	33
4	Marcin	Zagłoba	26
5	Marcin	Zagłoba	23

5. Cofamy zmiany do poprzednio zdefiniowanego punktu przywracania:

   ROLLBACK TO SAVEPOINT *punkt_pierwszy*;

6. Wyświetlamy zawartość tabeli:

id	imie	nazwisko	wiek
1	Jan	Nowak	24
2	Marcin	Werner	30
3	Marcin	Tracz	33
4	Marcin	Kowalska	26
5	Marcin	Nowak	23

7. Całość transakcji można nadal wycofać używając słowa kluczowego **ROLLBACK**. Zrób to. Przekonasz się, że zawartość tabeli wróci do stanu przed rozpoczęciem transakcji.

Ćwiczenie 2

Wzorując się na przykładzie 2 obejmij transakcją instrukcje dopisania dwóch nowych pracowników (dwóch kolegów), zdefiniuj dwa punkty przywracania i spraw, aby tylko jeden z nich został dopisany (ten pierwszy).

BLOKADY

Zakleszczenia występują w bazie danych w chwili, gdy co najmniej dwie transakcje nałożą blokadę na dane, które mają być użyte przez inne transakcje. Wówczas transakcje czekają, aż każda z nich zostanie wykonana i dochodzi do zakleszczenia. Sytuacja ta spowodowana jest blokadami nakładanymi na dane w chwili ich modyfikacji.

Przykład 3

Aby przeanalizować powyższy przypadek posłużymy się dwoma oknami sesji numerowanymi **okno1** i **okno2**.

OKNO1 – rozpoczęcie pierwszej transakcji

```
START TRANSACTION;
UPDATE pracownicy SET wiek = wiek + 10 WHERE id > 1;
```

OKNO2 – rozpoczęcie drugiej transakcji

```
START TRANSACTION;
UPDATE pracownicy SET wiek = 10 WHERE id > 1;
```

Jak prezentuje powyższy przykład, transakcja w drugim oknie została wstrzymana (napis *Query OK* … nie został wyświetlony), co oznacza, że zadziałał mechanizm blokujący dostęp do krotek (rekordów), na których aktualnie wykonywana jest już inna transakcja.
Aby odblokować drugie okno, musimy zakończyć transakcję w pierwszym oknie poleceniem **COMMIT** lub **ROLLBACK**. W naszym przykładzie w obu oknach (w kolejności **okno1**, **okno2**) zakończymy transakcję wycofaniem zmian. Po wpisaniu w pierwszym oknie polecenia **ROLLBACK** transakcja w oknie numer 2 zostanie odblokowana.
Powyższy przykład pozwolił na zilustrowanie mechanizmu blokowania.

Ćwiczenie 3

Wzorując się na przykładzie 3 obejmij transakcją instrukcje zmiany wieku (okno1) i usunięcia ostatniego pracownika (okno2).

ZAKLESZCZENIA

Nadeszła pora na prześledzenie, kiedy dochodzi do zakleszczenia. Do zakleszczenia dochodzi, gdy pierwsza transakcja nakłada blokadę na modyfikowane krotki (rekordy), w tym czasie druga transakcja chce wykonać operacje i czeka na zakończenie transakcji pierwszej, pierwsza transakcja również nie może dokonać zmian, ponieważ w drugiej kolejności krotki blokowane są przez transakcję drugą. W efekcie obie transakcje czekają na swoje zakończenie.

Przykład 4

1. TRANSAKCJA1 (zablokowanie krotek o id = 1). Rozpoczęcie pierwszej transakcji i nałożenie blokady na dane.

   ```
   START TRANSACTION;
   UPDATE pracownicy SET wiek = 55 WHERE id = 1;
   ```

   Wyświetlenie zawartości tabeli pracownicy.

```
mysql> SELECT * FROM pracownicy;
+----+--------+----------+------+
| id | imie   | nazwisko | wiek |
+----+--------+----------+------+
|  1 | Jan    | Nowak    |   55 |
|  2 | Ewa    | Werner   |   30 |
|  3 | Marcin | Tracz    |   33 |
|  4 | Łucja  | Kowalska |   26 |
|  5 | Paweł  | Nowak    |   23 |
+----+--------+----------+------+
```

2. TRANSAKCJA2 (zablokowanie krotki o id = 2). Rozpoczęcie drugiej transakcji i nałożenie blokady na dane;

   ```
   START TRANSACTION;
   UPDATE pracownicy SET wiek = 60 WHERE id = 2;
   ```

   Wyświetlenie zawartości tabeli pracownicy.

3. TRANSAKCJA1 (próba zmiany krotki o id = 1 – oczekiwanie na zdjęcie blokady przez TRANSAKCJĘ2).

   UPDATE *pracownicy* SET wiek = 55 WHERE id = 2;

   Próba modyfikacji danych blokowanych przez transakcję drugą nie powiodła się, pojawił się komunikat: `ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transaction`

4. TRANSAKCJA2 (próba zmiany krotki o id = 1 – oczekiwanie na zdjęcie blokady przez TRANSAKCJĘ1, czyli zakleszczenie).

   UPDATE *pracownicy* SET wiek = 60 WHERE id = 1;

   Próba modyfikacji danych blokowanych przez transakcję pierwszą przez okno drugiej transakcji nie powiodła się. Jak ilustruje powyższy przykład, SQL wykrył zakleszczenie, a transakcja z tego powodu została wycofana.

BLOKOWANIE TABEL

W przypadku typów tabel nieobsługujących transakcji (dotyczy to również tabel typu MyISAM) jedynym sposobem zagwarantowania spójności modyfikowanych danych jest ich blokowanie. W takim przypadku przed rozpoczęciem modyfikacji danych należy jawnie zablokować docelowe tabele instrukcją:

LOCK TABLES *nazwa tabeli1 typ blokady*, *nazwa tabeli2 typ blokady*;

Tabele mogą zostać zablokowane do odczytu (typ blokady READ) — wtedy zakładana jest na nie blokada współdzielona — lub do zapisu (typ blokady WRITE) — wtedy zakładana jest na nie blokada wyłączna. Blokada współdzielona jest kompatybilna z innymi blokadami współdzielonymi, czyli po jej założeniu inni użytkownicy będą mogli odczytywać dane z zablokowanej w ten sposób tabeli.

Przykład 5

Okno1 — dwie tabele zostały zablokowane do odczytu:

LOCK TABLES *zamowienia* READ;

Okno2 — zawartość tabeli zablokowanej do odczytu może być odczytywana przez wszystkich użytkowników bazy danych:

SELECT * FROM *zamowienia*;

Natomiast próba zmodyfikowania struktury lub zawartości tabeli zablokowanej do odczytu będzie wstrzymana do momentu zdjęcia blokady.

Okno2 — próba usunięcia wiersza z zablokowanej tabeli — wykonanie instrukcji zostało wstrzymane:

```
DELETE FROM zamowienia
WHERE id=10;
```

Okno1 — zdjęcie blokady współdzielonej:

UNLOCK TABLES;

Okno2 — zdjęcie blokady współdzielonej spowodowało, że instrukcja DELETE została w końcu wykonana. Sprawdź to, wyświetlając zawartość tabeli *zamowienia*.

Ćwiczenie 4

Wzorując się na poprzednim przykładzie nałóż na tabelę *zamowienia* blokadę wyłączną i spróbuj odczytać zawartość tej tabeli.

Ćwiczenie 5

Zapisz do zeszytu temat lekcji i zrób notatkę z lekcji.

Ćwiczenie 6

Wzorując się na powyższych przykładach (1-5) wymyśl własny przykład na którym przećwiczysz transakcje, blokady, punkty przywracania i zakleszczenia. Zgłoś nauczycielowi wykonanie ćwiczeń.