SYSTEMY BAZ DANYCH
SQL

TEMAT 25-07: Blokowanie tabel.

Autor dokumentu: Wojciech Galiński	sobota, 19 września 2015 r.	351203 Technik informatyk

ŹRÓDŁA WIEDZY: http://dev.mysql.com/doc/, http://webmaster.helion.pl/index.php/kurs-mysql.

Zagadnienia obowiązkowe

1. **Blokowanie tabel w tabelach z obsługą transakcji** – wyróżniamy:
   - *blokada gwarantująca możliwość odczytu danych (blokada współdzielona)* – zablokowanie możliwości zakładania blokady wyłączonej;
   - *blokada umożliwiająca zapis danych (blokada wyłączona)* – zablokowanie innym użytkownikom dostępu do modyfikowanych danych.

   Powyższe blokady są nakładane np. na tabele MySQL oparte o silnik „InnoDB”.
2. **Domyślne ustawienia izolowania danych podczas transakcji** – wyróżniamy:
   - *podczas odczytu danych* (a nie podczas trwania całej transakcji) wymagane jest nałożenie blokady współdzielonej na odczytywane dane (rekordy);
   - podczas transakcji zapisu danych wymagane jest nałożenie blokady wyłączonej na modyfikowane dane (rekordy).
3. **Ilustracja blokowania przez serwer MySQL tabel z obsługą transakcji** – przedstawia to poniższy przykład.

   PRZYKŁAD:

```
USE test;
CREATE TABLE tab ( id INT PRIMARY KEY AUTO_INCREMENT, k1 VARCHAR(20), k2 FLOAT )
  ENGINE=InnoDB DEFAULT CHARSET=utf8;
INSERT INTO tab (k1) VALUES ('aaa'), ('bbb');
SELECT * FROM tab;
```

SESJA 1	SESJA 2	SESJA 3

`START TRANSACTION;`
`UPDATE tab`
`  SET k1='abd' WHERE id=1;`
`SELECT * FROM tab;`		

	`SELECT * FROM tab;` `UPDATE tab SET k2=1 WHERE id=1;`	`SELECT * FROM tab;` `UPDATE tab SET k2=1 WHERE id=2;`
`COMMIT;` albo `ROLLBACK;`		
	`SELECT * FROM tab;`	

   Na koniec należy po sobie posprzątać. Oto instrukcje usuwające dane do ćwiczenia:

```
DROP TABLE tab;
```

4. **Blokowanie tabel w tabelach bez obsługi transakcji** – w tabelach, które nie obsługują transakcji (np. tabele MySQL oparte o silnik „MyISAM”) stosuje się następujące działania:
   - *„ręczne” blokowanie tabel* – realizujemy to za pomocą instrukcji:

     `LOCK TABLES` *`tabela_1`* ***`TRYB_BLOKOWANIA`***`, … ;`

     gdzie „TRYB_BLOKOWANIA” oznacza jedną z wartości:
     - „**READ**” – blokada potrzebna do odczytu danych (blokada współdzielona);
     - „**WRITE**” – blokada potrzebna do zapisu danych (blokada wyłączona);

     PRZYKŁADY:

```
LOCK TABLES tabela1 READ;
LOCK TABLES tabela2 WRITE;
LOCK TABLES tabela3 READ, tabela4 WRITE;
```

   - *„ręczne” odblokowanie tabel* – realizujemy to za pomocą instrukcji:

     `UNLOCK TABLES;`

5. **Ilustracja blokowania przez serwer MySQL tabel z obsługą transakcji** – przedstawia to poniższy przykład.

PRZYKŁAD:

```
USE test;
DELETE TABLE IF EXISTS tab;
CREATE TABLE tab
(
  id INT PRIMARY KEY AUTO_INCREMENT,
  k1 VARCHAR(20),
  k2 FLOAT
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
INSERT INTO tab (k1) VALUES ('aaa'), ('bbb');
SELECT * FROM tab;
```

SESJA 1	SESJA 2

`LOCK TABLES tab WRITE;`
`UPDATE tab SET k1='abc' WHERE id=1;`
`SELECT * FROM tab;`	

	`SELECT * FROM tab;`
`UNLOCK TABLES;`	
	`LOCK TABLES tab READ;` `SELECT * FROM tab;`
`SELECT * FROM tab;` `UPDATE tab SET k2=2 WHERE id=2;`	
	`UNLOCK TABLES;`
`SELECT * FROM tab;`	

```
DROP TABLE tab;  -- sprzątamy po sobie
```

6. **Zakleszczenie tabel** – ma miejsce, gdy 2 użytkowników jednocześnie modyfikuje dane w tej samej tabeli. Może się wtedy zdarzyć sytuacja, w której jeden użytkownik będzie czekać na zakończenie transakcji drugiego, podczas gdy drugi będzie czekać na zakończenie transakcji pierwszego. MySQL automatycznie rozpoznaje takie sytuacje i anuluje transakcję, która bezpośrednio doprowadziła do zakleszczenia.

PRZYKŁAD:

```
USE test;
CREATE TABLE tab ( id INT PRIMARY KEY AUTO_INCREMENT, k1 VARCHAR(20), k2 FLOAT )
  ENGINE=InnoDB DEFAULT CHARSET=utf8;
INSERT INTO tab (k1) VALUES ('aaa'), ('bbb');
SELECT * FROM tab;
```

	SESJA 1	SESJA 2

KROK 1	`START TRANSACTION;`
	`UPDATE tab SET k2=11 WHERE id=1;`
	`SELECT * FROM tab;`
		`START TRANSACTION;`
		`UPDATE tab SET k2=22 WHERE id=2;`
		`SELECT * FROM tab;`

KROK 2	`UPDATE tab SET k2=12 WHERE id=2;`	`UPDATE tab SET k2=21 WHERE id=1;`
KROK 3	`SELECT * FROM tab;`	`SELECT * FROM tab;`

```
DROP TABLE tab;  -- sprzątamy po sobie
```

Po wykonaniu powyższego ćwiczenia w sesji, która KROK 2 wykona jako druga, wystąpi błąd:

```
ERROR 1213 (40001): Deadlock found when trying to get lock; try restarting transaction
```

Stało się tak, ponieważ nastąpiło zakleszczenie tabel. Serwer MySQL rozwiązuje tą patową sytuację poprzez przerwanie transakcji w sesji, która do tego doprowadziła.

Pytania kontrolne

1. Po co i w jaki sposób blokuje się tabele?
2. W jaki sposób zapewniamy izolację danych w tabelach z obsługą transakcji?
3. Jakie są domyślne ustawienia izolowania danych podczas transakcji?
4. W jaki sposób zapewniamy izolację danych w tabelach bez obsługi transakcji?
5. Na czym polega zakleszczenie tabel i kiedy do niego dochodzi?

Zadania

1. Wykonaj przykład z punktu 3 – można robić to w 3-osobowych grupach.
2. Wykonaj przykład z punktu 3, zmieniając silnik tabeli z „InnoDB” na „MyISAM”.
3. Wykonaj przykład z punktu 5 – można robić to w 2-osobowych grupach.
4. Wykonaj przykład z punktu 6 – można robić to w 2-osobowych grupach.