

MySQL_REPEATABLE-READ事务隔离级别 && 幻读

关于mysql命令行中事务控制的语句见该文章

<http://my.oschina.net/xinxingegeya/blog/296459>

关于MVCC多版本控制

<http://my.oschina.net/xinxingegeya/blog/208821>

表结构

1	<code>create table t1 (</code>
2	<code> a int primary key,</code>
3	<code> b int not null</code>
4	<code>)</code>

REPEATABLE-READ可重复读（一）

这里打开两个mysql的命令行窗口，窗口A，即session1，窗口B，即session2。

session1

1	<code>mysql> begin ;</code>
2	<code>Query OK, 0 rows affected (0.00 sec)</code>
3	
4	<code>mysql> select * from t1;</code>
5	<code>+---+-----+</code>
6	<code> a b </code>
7	<code>+---+-----+</code>
8	<code> 51 3000 </code>
9	<code> 52 3000 </code>
10	<code> 53 3000 </code>
11	<code> 54 3000 </code>
12	<code>+---+-----+</code>
13	<code>4 rows in set (0.00 sec)</code>

以上sql只是显示的开启了事务，执行了sql查询。下面看session2的操作。

要注意这里的select操作是一般的快照读。根据MVCC多版本控制规则读取的数据行。

session2

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	
34	
35	
36	
37	
38	
39	
40	
41	
42	
43	
44	
45	
46	
47	
48	
49	
50	
51	
52	
53	
54	
55	
56	
57	
58	
59	
60	
61	
62	
63	
64	
65	
66	
67	
68	
69	
70	
71	
72	
73	
74	
75	
76	
77	
78	
79	
80	
81	
82	
83	
84	
85	
86	
87	
88	
89	
90	
91	
92	
93	
94	
95	
96	
97	
98	
99	
100	

```

1      mysql> begin ;
2      Query OK, 0 rows affected (0.00 sec)
3
4      mysql> insert into t1 values (55, 3000);
5      Query OK, 1 row affected (0.00 sec)
6
7      mysql> commit;
8      Query OK, 0 rows affected (0.03 sec)
9
10     mysql> select * from t1;
11     +-----+-----+
12     | a   | b     |
13     +-----+-----+
14     | 51  | 3000  |
15     | 52  | 3000  |
16     | 53  | 3000  |
17     | 54  | 3000  |
18     | 55  | 3000  |
19     +-----+-----+
20     5 rows in set (0.00 sec)

```

session2插入了一条数据，并显式的提交了事务。

session1

此时返回session1进行以下操作

```

1      mysql> select * from t1;
2      +-----+-----+
3      | a   | b     |
4      +-----+-----+
5      | 51  | 3000  |
6      | 52  | 3000  |
7      | 53  | 3000  |
8      | 54  | 3000  |
9      +-----+-----+
10     4 rows in set (0.00 sec)

```

此时，虽然在session2中插入了一条数据，并且提交了事务，但在**session1**中的查询和**session1**的上次查询还是同一个结果，这就是重复读。（也可以说是根据MVCC规则读取的数据行）。如果是在"READ-COMMITTED"级别下是可以读到a=55这条记录的（因为session2在刚才已经提交了事务）。

REPEATABLE-READ可重复读（二）

session1

2

```

1      mysql> begin;
2      Query OK, 0 rows affected (0.00 sec)
3
4      mysql> select * from t1 where a = 56;
5      +---+-----+
6      | a  | b      |
7      +---+-----+
8      | 56 | 7000  |
9
10     1 row in set (0.00 sec)

```

session1, 开启了一个事务, 查询a = 56 的记录。

```

1      mysql> begin;
2      Query OK, 0 rows affected (0.00 sec)
3
4      mysql> update t1 set b = 8000 where a = 56;
5      Query OK, 1 row affected (0.00 sec)
6      Rows matched: 1  Changed: 1  Warnings: 0
7
8      mysql> select * from t1 where a = 56;
9      +---+-----+
10     | a  | b      |
11     +---+-----+
12     | 56 | 8000  |
13     +---+-----+
14     1 row in set (0.00 sec)
15
16     mysql> commit
17     -> ;
18     Query OK, 0 rows affected (0.04 sec)

```

session2开启了事务, 更新a = 56 的记录, 同时查询a = 56 的记录, 可以看到在同一事务内重复读的效果。

session1

2

```

1      mysql> select * from t1 where a = 56;
2      +---+-----+
3      | a  | b      |
4      +---+-----+
5      | 56 | 7000  |
6      +---+-----+
7      1 row in set (0.00 sec)

```

和上次查询结果一致, 验证了重复读。还是要注意这里的select操作只是一般的快照读。其实不管session2 做什么操作, 这里的快照读都是重复读的。

此时, 如果session1提交该事务, 重新开启事务, 查询能查到session2中修改的结果

2

```

1      mysql> commit;
2      Query OK, 0 rows affected (0.00 sec)
3
4      mysql> select * from t1 where a = 56;
5      +----+-----+
6      | a  | b      |
7      +----+-----+
8      | 56 | 8000   |
9      +----+-----+
10     1 row in set (0.00 sec)

```

注：以上的重复读，虽然在当前事务中真的是重复读的现象，但到底来说是通过MVCC多版本控制实现的可重复读。

注：以上的重复读，虽然在当前事务中真的是重复读的现象，但到底来说是通过MVCC多版本控制实现的可重复读。

REPEATABLE-READ与幻读

session1

2

```

1      mysql> begin;
2      Query OK, 0 rows affected (0.00 sec)
3
4      mysql> select * from t1;
5      +----+-----+
6      | a  | b      |
7      +----+-----+
8      | 51 | 3000   |
9      | 52 | 3000   |
10     | 53 | 3000   |
11     | 54 | 3000   |
12     | 55 | 4000   |
13     | 56 | 8000   |
14     +----+-----+
15     6 rows in set (0.00 sec)

```

开启事务，select操作为快照读。

session2

2

```

1 mysql> begin;
2 Query OK, 0 rows affected (0.00 sec)
3
4 mysql> select * from t1;
5 +-----+-----+
6 | a | b |
7 +-----+-----+
8 | 51 | 3000 |
9 | 52 | 3000 |
10 | 53 | 3000 |
11 | 54 | 3000 |
12 | 55 | 4000 |
13 | 56 | 8000 |
14 +-----+-----+
15 6 rows in set (0.00 sec)
16
17 mysql> insert into t1 values (57, 1000);
18 Query OK, 1 row affected (0.00 sec)
19
20 mysql> select * from t1;
21 +-----+-----+
22 | a | b |
23 +-----+-----+
24 | 51 | 3000 |
25 | 52 | 3000 |
26 | 53 | 3000 |
27 | 54 | 3000 |
28 | 55 | 4000 |
29 | 56 | 8000 |
30 | 57 | 1000 |
31 +-----+-----+
32 7 rows in set (0.00 sec)
33
34 mysql> commit;
35 Query OK, 0 rows affected (0.11 sec)

```

31 session2 中做了一系列的操作，插入insert，这里其实是当前读（写入）。然后提交事务。
 32 session1

1	mysql> update t1 set b = b+1000;
2	Query OK, 7 rows affected (0.00 sec)
3	Rows matched: 7 Changed: 7 Warnings: 0
4	session1做更新操作，这里更新成功。如果session2 插入记录后，没有提交事务，这里更新
5	是要阻塞的，因为session2 插入记录持有那条记录的X锁。
6	mysql> select * from t1;
7	session1 整个会话的sql
8	+-----+-----+
9	a b
10	+-----+-----+
11	51 4000
12	52 4000
13	53 4000
14	54 4000
15	55 5000
16	56 9000
17	57 2000
	+-----+-----+
	7 rows in set (0.00 sec)

```

1      mysql> begin;
2
3      Query OK, 0 rows affected (0.00 sec)
4
5      mysql> select * from t1;
6      +-----+-----+
7      | a   | b     |
8      +-----+-----+
9      | 51  | 3000  |
10     | 52  | 3000  |
11     | 53  | 3000  |
12     | 54  | 3000  |
13     | 55  | 4000  |
14     | 56  | 8000  |
15     +-----+-----+
16     6 rows in set (0.00 sec)
17
18     mysql> select * from t1;
19     +-----+-----+
20     | a   | b     |
21     +-----+-----+
22     | 51  | 3000  |
23     | 52  | 3000  |
24     | 53  | 3000  |
25     | 54  | 3000  |
26     | 55  | 4000  |
27     | 56  | 8000  |
28     +-----+-----+
29     6 rows in set (0.00 sec)
30
31     mysql> update t1 set b = b+1000;
32     Query OK, 7 rows affected (0.00 sec)
33     Rows matched: 7  Changed: 7  Warnings: 0
34
35     mysql> select * from t1;
36     +-----+-----+
37     | a   | b     |
38     +-----+-----+
39     | 51  | 4000  |
40     | 52  | 4000  |
41     | 53  | 4000  |
42     | 54  | 4000  |
43     | 55  | 5000  |
44     | 56  | 9000  |
45     | 57  | 2000  |
46     +-----+-----+
47     7 rows in set (0.00 sec)
48     mysql>

```

以看到多出了一行，这算是幻读吗？其实不是。这是根据MVCC的select规则读出来的数据
 请详见<http://m54schib3000/xinxingegeya/blog/505675>

