

数据库实验lab14

曹广杰

15352015 数据科学与计算机

授课教师：刘玉葆

Content

数据库实验lab14

Content

实验目的

实验内容

锁争夺

查看阻塞

设置超时

演示死锁

附录

实验目的

- 1.学会识别锁冲突
- 2.学会检查和处理死锁。

实验内容

锁争夺

锁的争夺，表示为使用锁对于将要修改的数据的权限的争夺，为了演示锁的争夺：

- 1.使用两个事务
- 2.二者申请同样的资源
- 3.至少有一方需要对资源进行修改，即申请X锁

首先运行一个事务：

```
1  set transaction isolation level repeatable read
2
3  begin tran
4  update courses
5  set hour=80
6  where cid='10001'
7  -- Now we get a x lock
```

此时该事务已经申请到了对于update内容的 `x` 锁，但是由于没有及时commit，导致lock资源一直都没有释放。

这时候笔者开启另一个事务，依然要求申请这个资源的 `x` 锁：

```
1 set transaction isolation level repeatable read
2 set lock timeout 2000
3 begin tran
4     select *
5     from courses
6     where cid = '10001'
7 commit tran
```

很显然，由于 `x` 锁的作用，该事务不能正常运行。实验结果见附录。

实验的现象显示该事务始终在运行而不能停止。

查看阻塞

如上，第二个事务由于信号量资源的阻塞，迟迟不能运行。为了查看事务之间的阻塞关系，这里使用一个命令：

```
1 | exec sp_who
```

可以查看到各个事务之间的阻塞关系，实验结果详见附录。

设置超时

如上，第二个事务由于信号量的阻塞，始终不能运行。为了解决这种永久等待问题，可以设置超时lock，此时第二个事务如下：

```
1 set transaction isolation level repeatable read
2 -- out time lock
3 set lock_timeout 2000
4 -- out time lock
5 begin tran
6     select *
7     from courses
8     where cid = '10001'
9 commit tran
```

此时即使该事务不能正常运行，在等待了20000ms之后也会停止，实验结果显示当前的事务已经被强制终止，实验结果详见附录。

演示死锁

死锁也需要两个事务的资源冲突才可以演示，需要强调的是，在SQL的处理中，是将超时但是没有运行的事务情况作为死锁的情况处理。

1. 首先运行一个事务使之申请到某条数据的 `x-lock`：

```

1 set transaction isolation level repeatable read
2
3 begin tran
4     waitfor delay '00:00:05'
5     update teachers
6     set salary = 4000
7     where tid = '20003125'
8 commit tran

```

此时该事务在尚未提交之前还一直保有着对于修改数据的lock，此时，立刻运行另一个相同的事务：

两个事务会申请同样的信号量，这无疑会导致冲突，所以后运行的事务因为超时缘故被强制终止，系统认为这种情况与死锁的情况表现相同，因此这种情况被作为死锁处理。实验结果详见附录。

附录

结果

消息

(1 行受影响)

消息 1205, 级别 13, 状态 51, 第 8 行

事务 (进程 ID 53) 与另一个进程被死锁在 锁 资源上, 并且已被选作死锁牺牲品。请重新运行该事务。

第1题实验结果

结果

消息

消息 1222, 级别 16, 状态 51, 第 4 行

已超过了锁请求超时时段。

锁超时

	spid	ecid	status	loginame	hostname	blk	dbname	cmd	request_id
16	16	0	background	sa		0	master	BRKR TASK	0
17	17	0	sleeping	sa		0	master	TASK MANAGER	0
18	18	0	sleeping	sa		0	master	TASK MANAGER	0
19	19	0	sleeping	sa		0	master	TASK MANAGER	0
20	20	0	sleeping	sa		0	master	TASK MANAGER	0
21	21	0	sleeping	sa		0	master	TASK MANAGER	0
22	22	0	sleeping	sa		0	master	TASK MANAGER	0
23	23	0	sleeping	sa		0	master	TASK MANAGER	0
24	24	0	sleeping	sa		0	master	TASK MANAGER	0
25	25	0	sleeping	sa		0	master	TASK MANAGER	0
26	26	0	sleeping	sa		0	master	TASK MANAGER	0
27	51	0	sleeping	DESK...	DESKT...	0	master	AWAITING COMMAND	0
28	52	0	sleeping	DESK...	DESKT...	0	School	AWAITING COMMAND	0
29	53	0	suspended	DESK...	DESKT...	52	School	SELECT	0
30	54	0	runnable	DESK...	DESKT...	0	School	SELECT	0

信号量阻塞

