Blocking Locks

by

Table of contents

1. Motivation

Every practicing Sybase DBA has had to kill of processes that are locking out other spids. This might be an overrunning batch process that's blocking online users, or a rogue query submitted by a developer (or individual unknown).

The conventional way to determine the process to kill is not difficult, but can be a little time consuming, involving correlating the output of `sp_who` and `sp_lock`. The *Blocking Locks* view feature simplifies this task by displaying a tree indicating how any given spid is locking another spid.

2. Demo

To see the feature in action, first setting up the following script (it just creates two tables with 500 rows in each; the fact that they are APL and DOL locking matters not as it happens):

```
use tempdb
go

create table apl (
 a numeric identity primary key
,b int
,c char(400) default 'x'
)
go
create table drl (
 a numeric identity  primary key
,b int
,c char(400) default 'x'
)
lock datarows
go

declare @a int
set @a=1
while @a <= 500 begin
   insert apl (b) select @a
   insert drl (b) select @a
   set @a=@a+1
end
```

In session #1, run the following:

```
begin tran
select *
  from drl holdlock
 where a between 20 and 30
```

In this session we're using `holdlock` to hold locks at isolation level 3 on 11 rows on the DRL table.

In a second session #2, run the following:

```
begin tran
```

```
select * from apl holdlock
  where a between 50 and 60

update drl
   set c = 'y'
 where a between 10 and 25
```

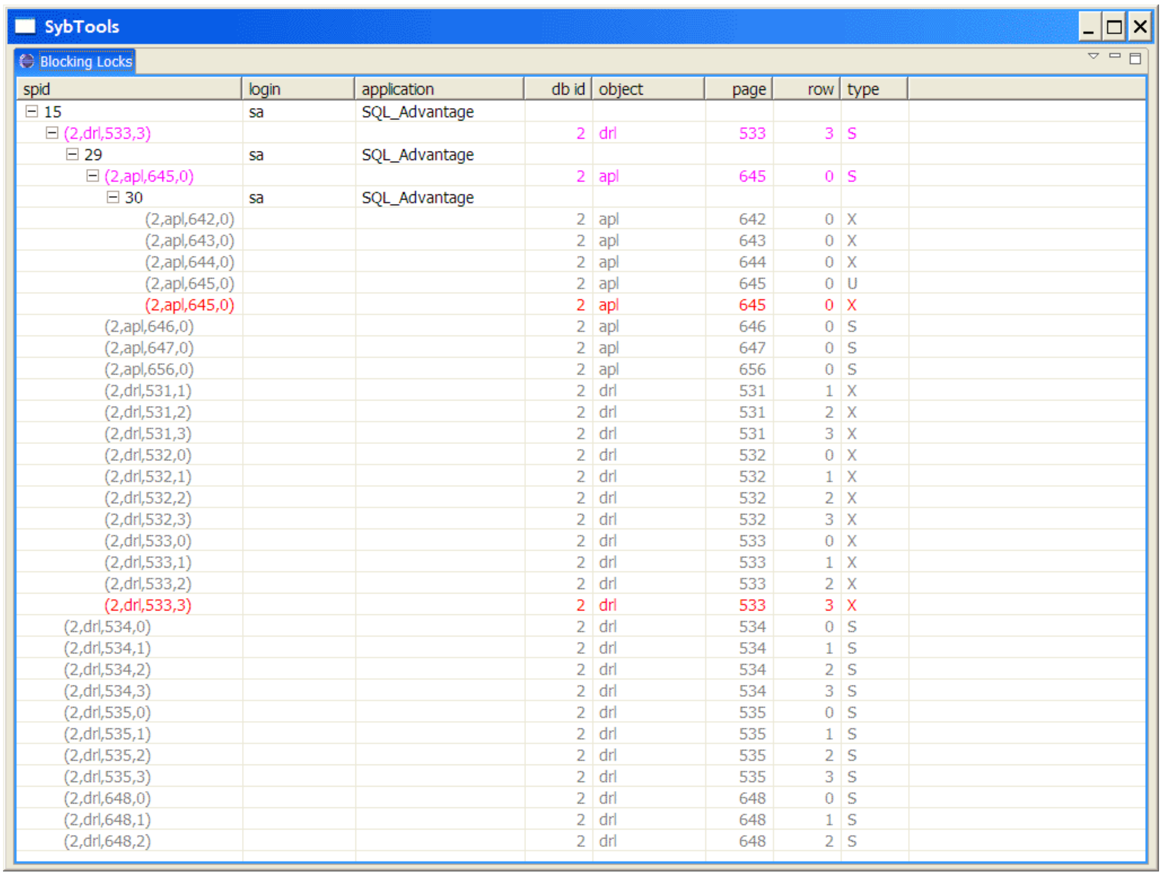
This session takes out shared locks on the APL table, and then attempts to update rows on the DRL table. Some of these succeed, but others will block because of the locks still held by session #1.

In a third session #3, run the following:

```
begin tran
update apl
   set c='z'
 where a between 40 and 55
```

In this session there may be some locks that are granted, but again it will then be blocked, this time by session #2.

Now run up the console (`sybtool`) and open up the *Blocking Locks* view:



spid	login	application	db id	object	page	row	type
⊟ 15	sa	SQL_Advantage					
⊟ (2,drl,533,3)			2	drl	533	3	S
⊟ 29	sa	SQL_Advantage					
⊟ (2,apl,645,0)			2	apl	645	0	S
⊟ 30	sa	SQL_Advantage					
(2,apl,642,0)			2	apl	642	0	X
(2,apl,643,0)			2	apl	643	0	X
(2,apl,644,0)			2	apl	644	0	X
(2,apl,645,0)			2	apl	645	0	U
(2,apl,645,0)			2	apl	645	0	X
(2,apl,646,0)			2	apl	646	0	S
(2,apl,647,0)			2	apl	647	0	S
(2,apl,656,0)			2	apl	656	0	S
(2,drl,531,1)			2	drl	531	1	X
(2,drl,531,2)			2	drl	531	2	X
(2,drl,531,3)			2	drl	531	3	X
(2,drl,532,0)			2	drl	532	0	X
(2,drl,532,1)			2	drl	532	1	X
(2,drl,532,2)			2	drl	532	2	X
(2,drl,532,3)			2	drl	532	3	X
(2,drl,533,0)			2	drl	533	0	X
(2,drl,533,1)			2	drl	533	1	X
(2,drl,533,2)			2	drl	533	2	X
(2,drl,533,3)			2	drl	533	3	X
(2,drl,534,0)			2	drl	534	0	S
(2,drl,534,1)			2	drl	534	1	S
(2,drl,534,2)			2	drl	534	2	S
(2,drl,534,3)			2	drl	534	3	S
(2,drl,535,0)			2	drl	535	0	S
(2,drl,535,1)			2	drl	535	1	S
(2,drl,535,2)			2	drl	535	2	S
(2,drl,535,3)			2	drl	535	3	S
(2,drl,648,0)			2	drl	648	0	S
(2,drl,648,1)			2	drl	648	1	S
(2,drl,648,2)			2	drl	648	2	S

In the above screenshot session #1 is spid 15 which is holding a lock (coloured magenta) that is blocking session #2 (spid 29). This in turn is holding a lock that is blocking session

#3 (spid 30).

3. Installation

The *Blocking Locks* view is bundled with the console so there are no separate installation steps.

TODO: in the future this functionality will be delivered as an Eclipse feature and so there will be separate installation steps.