Re: Deadlocks

5 posts by 1 author

Michael Hunger Mar 27

Other recipients: matias...@brinqa.com

Hi Matias, please send questions like that to the google group.

A deadlock appears when it is about to acquire two locks in reverse order.

What you see is probably just a lock that is not released because the transaction which created it was not closed and is still running?

You can check via JMX if there is still an ongoing open transaction and open locks.

Also when you take a thread-dump you see what lock (incl. node-id) the thread is waiting for.

Cheers, Michael

On Fri, Mar 27, 2015 at 4:07 PM, Matias Burak <matias...@brinqa.com>wrote:

> Hi Michael, sorry I bother you again with another problem.
>
> Is it possible that Neo4j doesn't realize that it's in a deadlock? I'm using an old version of the db (1.9) and when running several transactions on different threads against the db sometimes the threads get locked and never are released and I don't see an exception, they are just stuck on deadlockGuardedWait().
> I expect deadlocks and I built a retry feature to try again, but I'm not getting any exceptions :((sometimes I am)
> Any idea?
>
> Thanks,
> MAtias.

Michael Hunger Mar 27

Other recipients: matias...@brinqa.com

> Am 27.03.2015 um 17:24 schrieb Matias Burak <matias...@brinqa.com>:
>
> Thanks for your response.
> Yeah, it seems that transactions might not be closed. Why would this happen? I mean... I'm doing my transactions on a Spring TransactionTemplate execute.

Good question, don't know where the leak would come from.
Do you perhaps execute cypher or index queries without exhausting the results or calling .close on them?

But your stack traces should give you a hint.

Do I need neo4j enterprise to check open transacctionn with JMX?

Yes

A thread-dump look like this:

"pool-11537-thread-1" prio=5 tid=0x00007f930b395800 nid=0x1a36f in Object.wait()
[0x000000013c924000]
   java.lang.Thread.State: WAITING (on object monitor)
        at java.lang.Object.wait(Native Method)
        - waiting on <0x00000006d6669c88> (a org.neo4j.kernel.impl.transaction.RWLock)
        at java.lang.Object.wait(Object.java:503)
        at org.neo4j.kernel.impl.transaction.RWLock.deadlockGuardedWait(RWLock.java:652)
        at org.neo4j.kernel.impl.transaction.RWLock.acquireWriteLock(RWLock.java:344)

nid=0x1a36f is the nid on the db? There's no nid 107375 (decimal for 0x1a36f)

- show quoted text -

Michael Hunger Mar 31

Other recipients: matias...@brinqa.com

So far it's deadlock detection is pretty good it uses a graph internally to do that (find cycles).

Do you think you could reproduce the situation somehow

Michael

Am 30.03.2015 um 15:44 schrieb Matias Burak <matias...@brinqa.com>:

Could it be possible that neo4j is not detecting a deadlock with more than 2 threads?
I mean this case: thread-1 needs something from thread-2 which needs something from thread-3 which needs something from thread-1.
- show quoted text -

Michael Hunger Mar 31

Other recipients: matias...@brinqa.com

Then that's something else.

Would you be able to create a minimal test-case that reproduces that?

Michael

Am 31.03.2015 um 15:26 schrieb Matias Burak <matias...@brinqa.com>:

Yes, I can. It happens every time I do a stress test. If I do the same process for only one user it works fine and it closes all the transactions. But doing it for 10 users it starts to do some long transactions (10 secs) in different threads and I start to see locks (i also see some deadlocks that i retry but locks are still there) and transactions are not closed.

- show quoted text -

Michael Hunger Apr 1

Other recipients: matias...@brinqa.com

reproduction of the deadlock :)

Michael

> Am 01.04.2015 um 18:21 schrieb Matias Burak <matias...@brinqa.com>:
>
> I have some a complex domain model, what'd you need from the test?
> - show quoted text -