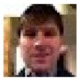
Spatial index returning non serialisable node, suggesting bad transaction handling on spatial indexes.

5 posts by 3 authors



Josef Karthauser Mar 31

I've got a problem with a spatial query. It's pretty simple:

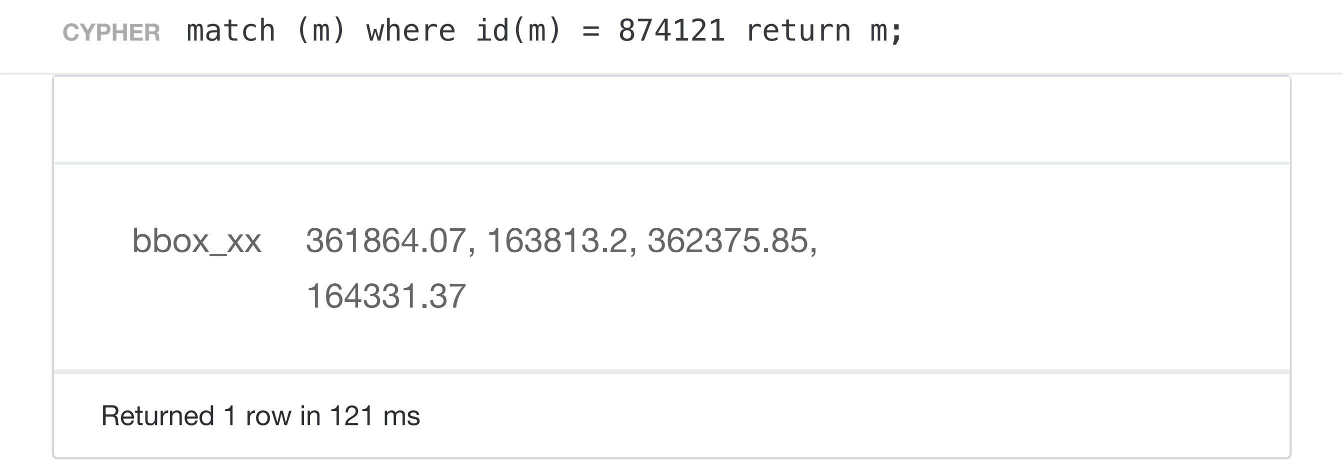
START n=node:topography('bbox:[357344.057555, 358504.481079, 162967.718391, 163797.476526]') RETURN n

It predominantly work, but in this region one of the nodes return fails to map:

Message: No primary SDN label exists .. (i.e one starting with _)

It turns out that the problem is that one of the index nodes is being returned by the cypher request instead an indexed node! Looks like the spatial index is corrupted!

The node in question is 874121, and it has these contents, which is a spatial index node:

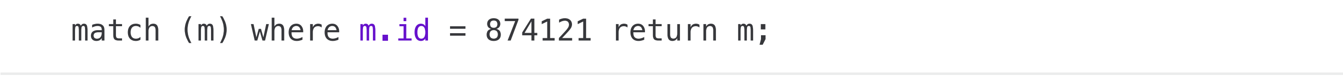


```
CYPHER match (m) where id(m) = 874121 return m;
```

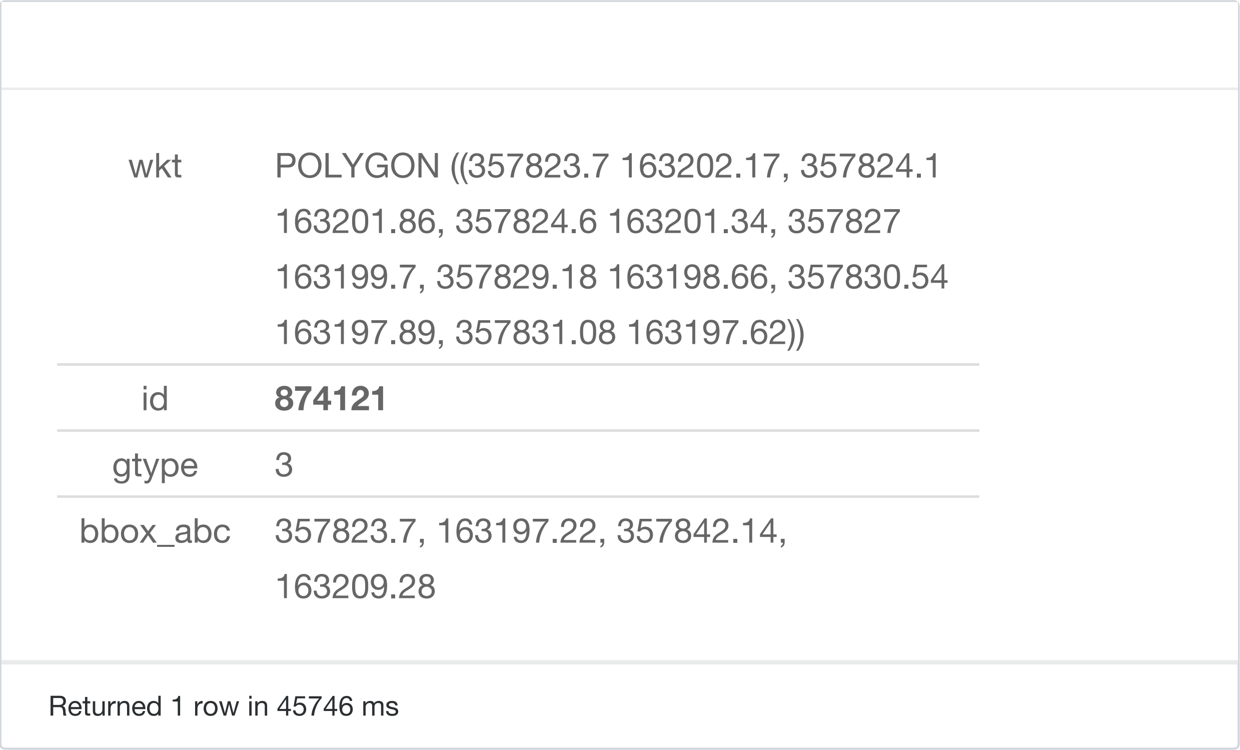
```
bbox_xx    361864.07, 163813.2, 362375.85,
           164331.37

Returned 1 row in 121 ms
```

It's being returned because it's referred to within the spatial index as being a referenced node:



```
match (m) where m.id = 874121 return m;
```



wkt	POLYGON ((357823.7 163202.17, 357824.1 163201.86, 357824.6 163201.34, 357827 163199.7, 357829.18 163198.66, 357830.54 163197.89, 357831.08 163197.62))
id	**874121**
gtype	3
bbox_abc	357823.7, 163197.22, 357842.14, 163209.28

Returned 1 row in 45746 ms

That shouldn’t happen! A spatial index should not index itself!

The only way I can think that it happened was due to the ‘out of files in spatial index’ problem I reported a couple of days ago.

There, I was importing 100k of spatially indexes polygons, and the import was blowing up with ‘out of file handles’ every 8k or so. So, I modified the import to check to see whether the node was already in the database or not, so I could rerun the import to carry on where it left off. So, in that way, ignoring the crashes and rerunning the import several times, I eventually ended up with all the spatial nodes in the database.

So, how did we end up with a spatial index referencing itself? That’s pretty serious.

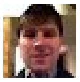
I can only assume that something like the following happened. In the lead up to an ‘out of file handles crash’ a spatially indexed node was added to the database with id 874121, and then added to the spatial index. During the crash, restart, transaction unwind node 874121 was wound back, but the spatial node referencing it was not. Then, during the next run a new node with id 874121 was created, but this was an index node, not a data node.

That sounds crazy, but plausible. But, if true suggests then the transaction protection isn’t absolute. Running out of file handles is a likely outcome and transactions should protect against corruption in this scenario, right? Why isn’t the spatial index also getting wound back after a transaction failure?

This is with neo4j-2.1.6 and neo4j-spatial 0.13

Thanks,
Joe
—

Dr Josef Karthauser, Technical Director, Wansdyke Telecom CIC
mob: +44 (7703) 596893; web: www.wansdyketele.com; skype: josefkarthauser



Josef Karthauser Mar 31

On 31 Mar 2015, at 08:33, Dr Josef Karthauser <joe.kar...@wansdyketele.com> wrote:

> [cut]
>
> I can only assume that something like the following happened. In the lead up to an ‘out of file handles crash’ a spatially indexed node was added to the database with id 874121, and then added to the spatial index. During the crash, restart, transaction unwind node 874121 was wound back, but the spatial node referencing it was not. Then, during the next run a new node with id 874121 was created, but this was an index node, not a data node.
>
> That sounds crazy, but plausible. But, if true suggests then the transaction protection isn’t absolute. Running out of file handles is a likely outcome and transactions should protect against corruption in this scenario, right? Why isn’t the spatial index also getting wound back after a transaction failure?
>
> This is with neo4j-2.1.6 and neo4j-spatial 0.13

This is the extent of the problem:

```
match (:ReferenceNode)--(m {layer: "topography"})
match m-[*]->(n) where has(n.id)
with n as referencing
match q<-[*]-(m) where id(q) = referencing.id
return count(q)
```

```
114

Returned 1 row in 1285 ms
```

That looks like 114 cross links within the index to me.

Is my interpretation right? I’m minded to just delete all of these ‘referencing’ nodes, and hopefully that will fix the index.

Joe

Michael Hunger Mar 31

The index links to the domain nodes via the id-property.

This is weird as both should happen in the same transaction at least in embedded.

Do you think you can reproduce that issue?

Michael

> - show quoted text -
> - show quoted text -
> --
> You received this message because you are subscribed to the Google Groups "Neo4j" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to neo4j+un...@googlegroups.com.

For more options, visit https://groups.google.com/d/optout.

Josef Karthauser Mar 31

It is strange isn't it. I am using an embedded database.

I removed all the bad spatial linkages, and then checked to see that all of my domain objects were otherwise indexed correctly, and they were. So we have to explain how these self referencing nodes happened.

The only story that makes sense to me, and granted I don't know how the transaction code works so it might be impossible, is that the domain objects got rolled back when the database crashed, but the spatial index nodes didn't.

I can try again with a fresh database and see whether it's reproducible easily enough. I'll try and do that tomorrow and report back.

Cheers,
Joe
- show quoted text -

Michael Hunger Apr 2

> Am 31.03.2015 um 23:14 schrieb Josef Karthauser <j...@wansdyketele.com>:
>
> It is strange isn't it. I am using an embedded database.
>
> I removed all the bad spatial linkages, and then checked to see that all of my domain objects were otherwise indexed correctly, and they were. So we have to explain how these self referencing nodes happened.
>
> The only story that makes sense to me, and granted I don't know how the transaction code works so it might be impossible, is that the domain objects got rolled back when the database crashed, but the spatial index nodes didn't.
>
> I can try again with a fresh database and see whether it's reproducible easily enough. I'll try and do that tomorrow and report back.

That would be great, thanks so much !

Michael
- show quoted text -