Questions about profiling and optimizing queries in Neo4j 2.2

9 posts by 3 authors

bi...@levelstory.com Apr 8

Images are not displayed
Display images in this post - Always display images from bi...@levelstory.com- Always display images in Neo4j

I'm starting to profile my queries using the new 2.2 query profiler (which is great by the way!) and I had a couple of questions about what I am seeing.

Question 1

In a query like the following:

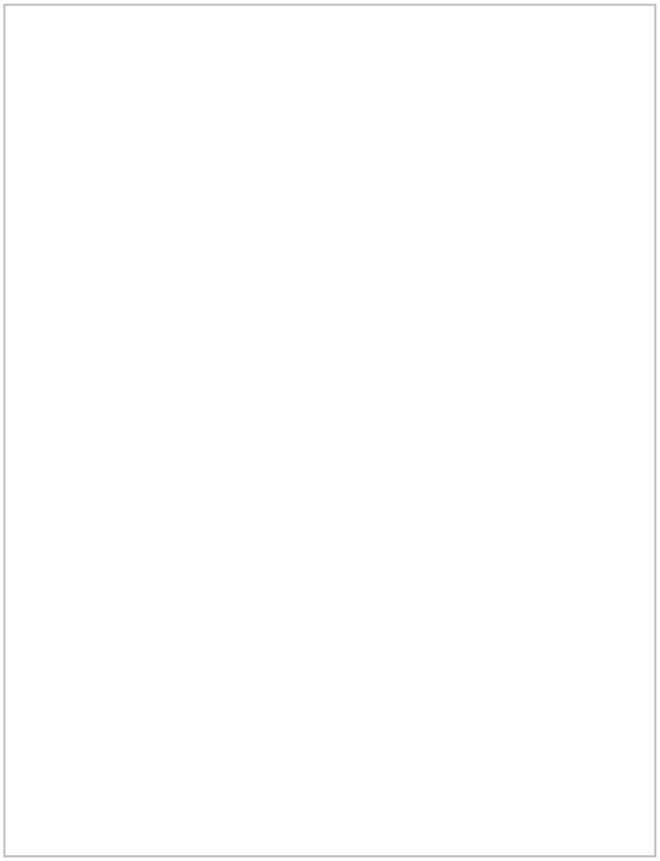
```
  MATCH (c:contact {teamid: uc.teamid})-[:CONTACT]->(cr:role)
  WHERE (ur.name IN ["Owners", "Admins"]) OR
        (ur.name = "Employees" AND NOT cr.name = "Clients") OR
        (cr:projectrole AND (uc)-[:CONTACT*1..3]->(:projectrole)<-
[:CONTACT*0..2]-(cr)) OR
        (c.id = uc.id)
  WITH DISTINCT c, {accessedby: uc.id, accessedrole:ur.name} AS r
```

Do all of the WHERE clauses get expanded immediately or do they get expanded in order? I thought they would only be expanded until the first TRUE term was found, but looking at the profile output I'm not sure. In the query that I profiled, the ur.name property was equal to "Owners". I assumed that this means that the other terms would be ignored but I see the following in the profile output.

If I modify the query to just use the where clause that returns true we get different results.

```
  MATCH (c:contact {teamid: uc.teamid})-[:CONTACT]->(cr:role)
  WHERE (ur.name IN ["Owners", "Admins"])
  WITH DISTINCT c, {accessedby: uc.id, accessedrole:ur.name} AS r
```

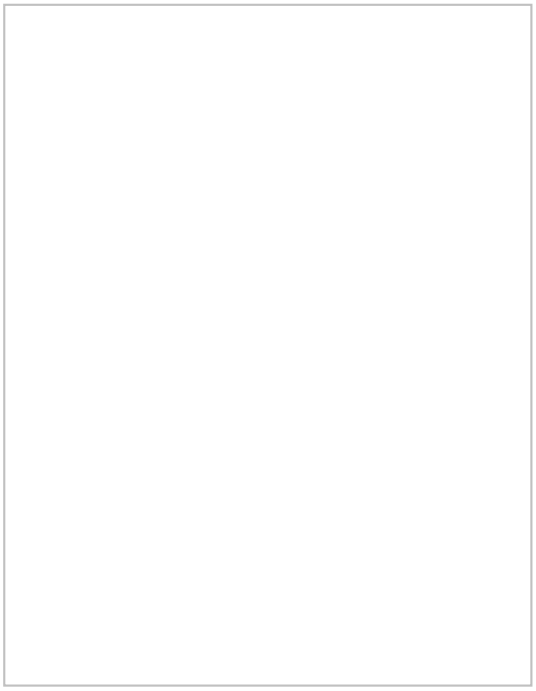
The profile output just shows this which is getting down to the same 1109 rows, but with only 5k db hits.



Question 2

Why does an optional match query starting from a node collection result in an AllNodesScan? After getting a list of contacts, I then see if they have any phone numbers or addresses. Since they may not, I use an optional query. Since (c) is the collection of contacts (670 rows) and there are 1800 total phones, addresses, tags, and invites why is it doing an all nodes scan and coming up with 26k rows? Is there a better way to take advantage of optional queries?

```
OPTIONAL MATCH (c)-[:PHONE|ADDRESS|TAG|INVITE]->(x)
RETURN c, COLLECT(DISTINCT x) AS xs, r
```



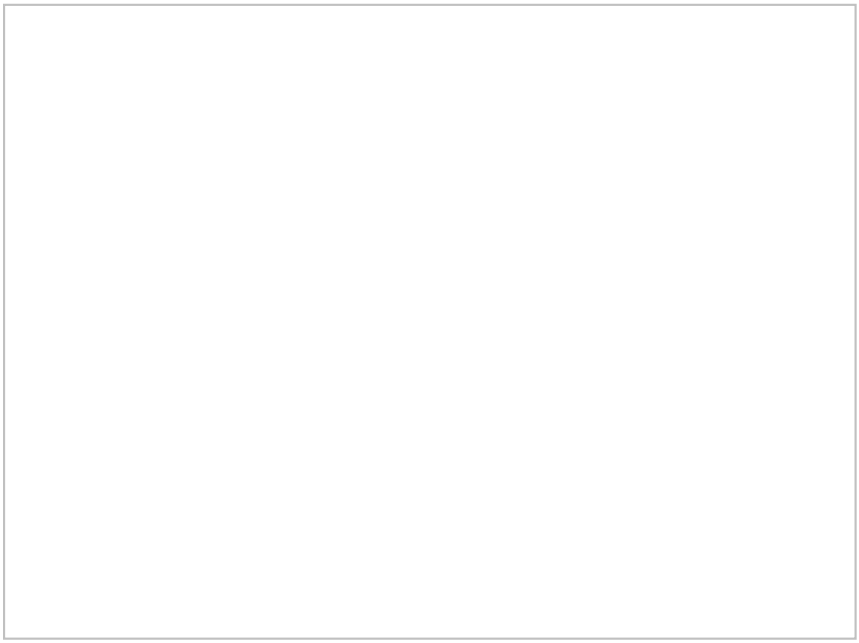
I tried adding a where clause to restrict x, but that just added in an extra 100k db hits and slowed everything down.

```
OPTIONAL MATCH (c)-[:PHONE|ADDRESS|TAG|INVITE]->(x)
WHERE x:phone OR x:address OR x:tag OR x:invite
RETURN c, COLLECT(DISTINCT x) AS xs, r
```

Question 3

Another thing I noticed is that the estimated row count ends up with some pretty funny numbers. Looking at the same previous query which returns a total of 670 rows, this is what I see in the profiler. Is this affecting the query planner adversely? If there are 670 rows in the c collection, and 1800 total rows in the x collection, what math led to there being an estimated 8 quintillion rows?

```
OPTIONAL MATCH (c)-[:PHONE|ADDRESS|TAG|INVITE]->(x)
RETURN c, COLLECT(DISTINCT x) AS xs, r
```



Thanks,
bill

Wes Freeman Apr 8

If the query you put in q2 is your whole query, you should really start with a MATCH for any OPTIONAL MATCH, even if it's not a big pattern... like:

```
MATCH (c:Contacts)
OPTIONAL MATCH (c)-[:PHONE|ADDRESS|TAG|INVITE]->(x)
RETURN c, COLLECT(DISTINCT x) AS xs, r
```

Wes Freeman Apr 8

I'm guessing q3 is something like len(all nodes) * (len(PHONE) + len(ADDRESS) + len(TAG) + len(INVITE)) * len(all nodes) -- since you have no labels on the end nodes of the pattern, it's hard to estimate.

Wes
- show quoted text -

bi...@levelstory.com Apr 8

For q2, it's all part of the same query. I tried adding MATCH (c:contacts) in front of the optional match but that didn't have any effect. This is the query that I'm executing:

```
MATCH (c:contact {teamid: uc.teamid})-[:CONTACT]->(cr:role)
WHERE (ur.name IN ["Owners", "Admins"]) OR
      (ur.name = "Employees" AND NOT cr.name = "Clients") OR
      (cr:projectrole AND (uc)-[:CONTACT*1..3]->(:projectrole)<-
```

```
[:CONTACT*0..2]-(cr)) OR
        (c.id = uc.id)
  WITH DISTINCT c, {accessedby: uc.id, accessedrole: ur.name} AS r

  OPTIONAL MATCH (c)-[:PHONE|ADDRESS|TAG|INVITE]->(x)
  RETURN c, COLLECT(DISTINCT x) AS xs, r
```

For q3, I guess this is where I need help. How do optional matches actually work? In my query, I thought they would be rooted at my node collection (c) and then traverse outward from there. There are 670 nodes in (c) and a total of 1800 phones, addresses, tags, and invites associated with them. Why is the query looking at anything that is not directly connected to one of the nodes in (c)?

I opened up a separate question on just this point. It's not clear to me how to write performant OPTIONAL MATCH queries based on resulting rows from previous MATCH statements.

---

bi...@levelstory.com Apr 8

I meant MATCH (c) (to use the previous result), not MATCH (c:contacts) which would have matched all contacts.
- show quoted text -

Wes Freeman Apr 8

I get it, sorry. Your confusion is justified--it probably shouldn't do that. It must have decided an AllNodeScan would be easier than scanning the shorter list of c nodes, which is definitely a bad plan. Does it act the same without DISTINCT?

Wes

- 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.

---

bi...@levelstory.com Apr 8

Seems to act the same with or without DISTINCT. I also tried to break up the query so that the OPTIONAL MATCH is more specific but it still ends up doing a NodeByLabelScan (better than an AllNodesScan) but it should be doing an OptionalFilter on the existing nodes.
- show quoted text -

Michael Hunger Apr 9

Images are not displayed
Display images in this post - Always display images from Michael Hunger -Always display images in Neo4j

Can you share your full query and the full plans?
I don't know where uc comes from for instance

And the whole query is rewritten/analyzed

Regarding your question: the runtime eval of a where happens from left to right except if some parts can be

rewritten eg into index lookups

Michael

Von meinem iPhone gesendet
- show quoted text -

- show quoted text -

---

bi...@levelstory.com Apr 9

The full query is below. I uploaded the full plan to http://i.imgur.com/F7tA1eg.png.

```
  PROFILE
  MATCH (:user { id:"foobar" })-[:CONTACT]->(uc:contact)-[:CONTACT]->(ur:role)
  WHERE HAS(ur.teamid)
  WITH uc, ur

  MATCH (c:contact {teamid: uc.teamid})-[:CONTACT]->(cr:role)
  WHERE (ur.name IN ["Owners", "Admins"]) OR
        (ur.name = "Employees" AND NOT cr.name = "Clients") OR
        (cr:projectrole AND (uc)-[:CONTACT*1..3]->(:projectrole)<-
[:CONTACT*0..2]-(cr)) OR
        (c.id = uc.id)
  WITH DISTINCT c, {accessedby: uc.id, accessedrole:ur.name} AS r

  MATCH (c)-[:CONTACT]->(cr:role {teamid: c.teamid})
  WITH c, {
    accessedrole: r.accessedrole, accessedby: r.accessedby,
    role: cr.name
    } AS r

  OPTIONAL MATCH (c)-[:PHONE|ADDRESS|TAG|INVITE]->(x)
  WITH c, COLLECT(DISTINCT x) AS xs, r
  WITH c, {
    accessedrole: r.accessedrole, accessedby: r.accessedby, role: r.role,
    isinvited: HEAD([x in xs WHERE x.object = "invite" | {id: x.id, object:
x.object}]) IS NOT NULL,
    phones: [x IN xs WHERE x.object = "phone" AND r.accessedrole IN ['Owners',
'Admins', 'Employees'] |
      { id: x.id, object: x.object,number: x.number, description: x.description
}
      ],
    addresses: [x IN xs WHERE x.object = "address" AND r.accessedrole IN
['Owners', 'Admins', 'Employees'] |
      { id: x.id, object: x.object, street: x.street, locality: x.locality,
        region: x.region, postcode: x.postcode, country: x.country, description:
x.description, neighborhood: x.neighborhood, latlnglocation: x.latlnglocation
      }],
    tags: [x IN xs WHERE x.object = "tag" | {id:x.id, object: x.object,
name: x.name}]
    } AS r

  RETURN COUNT(c)
```