

Ten. Million. Questions. Let's celebrate [all we've done together](#).

Stack Overflow is a question and answer site for professional and enthusiast programmers. It's 100% free.

[Take the 2-minute tour](#) ×

Using Match with Multiple Clauses Causes Odd Results

{ USE STACK OVERFLOW TO
FIND THE BEST DEVELOPERS }

 stackoverflowcareers

asked 4 months ago

viewed 34 times

active 4 months ago

Blog

 Why Stack Overflow is a Good Workplace for Women

 **Love this site?**

Get the **weekly newsletter!**

- Top questions and answers
- Important announcements
- Unanswered questions

[Sign up for the newsletter](#)

see an [example newsletter](#)

Related

- 0 [Return all matching relationships in cypher](#)
- 3 [Neo4j Cypher: Find exact match to array Node property in WHERE clause](#)
- 0 [Order of match-clause affects result](#)
- 3 [Multiple labels in match painfully slow?](#)
- 1 [Neo4j graph showing multiple relationship between same nodes on top of each other](#)
- 1 [Incorrect results when matching multiple labels with Cypher](#)
- 0 [Matching, sorting and returning multiple nodes in Neo4j](#)
- 2 [Cypher matched results differ with "with" and without "with"](#)
- 0 [Cypher - Collecting relationships across different match clauses](#)
- 1 [Chaining to multiple parallel subgraph matches](#)

Hot Network Questions

 Did Arnold Rimmer kill the Red Dwarf crew?

▲
2

▼
★

I am writing a Cypher query in Neo4j 2.0.4 that attempts to get the total number of inbound and outbound relationships for a selected node. I can do this easily when I only use this query one-node-at-a-time, like so:

```
MATCH (g1:someIndex{name:"name1"})
MATCH g1-[r1]-()
RETURN count(r1);
//Returns 305

MATCH (g2:someIndex{name:"name2"})
MATCH g2-[r2]-()
RETURN count(r2);
//Returns 2334
```

But when I try to run the query with 2 nodes together (i.e. get the total number of relationships for both g1 and g2), I seem to get a *bizarre* result.

```
MATCH (g1:someIndex{name:"name1"}), (g2:someIndex{name:"name2"})
MATCH g1-[r1]-(), g2-[r2]-()
RETURN count(r1)+count(r2);
//Returns 1423740
```

For some reason, the number is much much greater than the total of 305+2334.

It seems like other Neo4j users have run into strange issues when using multiple `MATCH` clauses, so I read through Michael Hunger's explanation at <https://groups.google.com/d/msg/neo4j/7ePLU8y93h8/8jpuopsFEFsJ>, which advised Neo4j users to pipe the results of one match using `WITH` to avoid "identifier uniqueness". However, when I run the following query, it simply times out:

```
MATCH (g1:gene{name:"SV422_HUMAN"}), (g2:gene{name:"BRCA1_HUMAN"})
MATCH g1-[r1]-()
WITH r1
MATCH g2-[r2]-()
RETURN count(r1)+count(r2);
```


I suspect this query doesn't return because there's a lot of records returned by `r1`. In this case, how would I operate my "get-number-of-relationships" query on 2 nodes? Am I just using some incorrect syntax, or is there some fundamental issue with the logic of my "2 node at a time" query?

neo4j cypher

share improve this question

add a comment

asked Apr 3 at 15:59

 Shrey Gupta
2,444 ● 1 ● 21 ● 53

2 Answers

active oldest votes

▲

Your first problem is that you are returning a Cartesian product when you do this:

3
▼

```
MATCH (g1:someIndex{name:"name1"}), (g2:someIndex{name:"name2"})
MATCH g1-[r1]-(), g2-[r2]-()
RETURN count(r1)+count(r2);
```



If there are 305 instances of `r1` and 2334 instances of `r2`, you're returning $(305 * 2334) == 711870$ rows, and because you are summing this `(count(r1)+count(r2))` you're getting a total of $711870 + 711870 == 1423740$.

Your second problem is that you are not carrying over `g2` in the `WITH` clause of this query:

```
MATCH (g1:gene{name:"SV422_HUMAN"}), (g2:gene{name:"BRCA1_HUMAN"})
MATCH g1-[r1]-()
WITH r1
MATCH g2-[r2]-()
RETURN count(r1)+count(r2);
```

You match on `g2` in the first `MATCH` clause, but then you leave it behind when you only carry over `r1` in the `WITH` clause at line 3. Then, in line 4, when you match on `g2-[r2]-()` you are matching literally everything in your graph, because `g2` has been unbound.

Let me walk through a solution with the movie dataset that ships with the Neo4j browser, as you have not provided sample data. Let's say I want to get the total count of relationships attached to Tom Hanks and Hugo Weaving.

As separate queries:

```
MATCH (:Person {name:'Tom Hanks'})-[r]-()
RETURN COUNT(r)
```

=> 13

```
MATCH (:Person {name:'Hugo Weaving'})-[r]-()
RETURN COUNT(r)
```

=> 5

If I try to do it your way, I'll get $(13 * 5) * 2 == 90$, which is incorrect:

```
MATCH (:Person {name:'Tom Hanks'})-[r1]-(),
      (:Person {name:'Hugo Weaving'})-[r2]-()
RETURN COUNT(r1) + COUNT(r2)
```

=> 90

Again, this is because I've matched on all combinations of `r1` and `r2`, of which there are 65 ($13 * 5 == 65$) and then summed this to arrive at a total of 90 ($65 + 65 == 90$).

The solution is to use `DISTINCT`:

```
MATCH (:Person {name:'Tom Hanks'})-[r1]-(),
      (:Person {name:'Hugo Weaving'})-[r2]-()
RETURN COUNT(DISTINCT r1) + COUNT(DISTINCT r2)
```

=> 18

Clearly, the `DISTINCT` modifier only counts the distinct instances of each entity.

You can also accomplish this with `WITH` if you wanted:

```
MATCH (:Person {name:'Tom Hanks'})-[r]-()
WITH COUNT(r) AS r1
MATCH (:Person {name:'Hugo Weaving'})-[r]-()
RETURN r1 + COUNT(r)
```

=> 18

TL;DR - Beware of Cartesian products. `DISTINCT` is your friend:

```
MATCH (:someIndex{name:"name1"})-[r1]-(),
      (:someIndex{name:"name2"})-[r2]-()
RETURN COUNT(DISTINCT r1) + COUNT(DISTINCT r2);
```

[share](#) [improve this answer](#)

answered Apr 3 at 16:40



Nicole White

3,059 ● 4 ● 15

Thank you Nicole! I had no clue that my current query was using a cross product; that makes a lot more sense now. – [Shrey Gupta](#) Apr 3 at 16:42

[add a comment](#)

- What are some techniques to avoid touching a wet tent wall?
- Remove a language from the "Preferred language order"
- Capture output without redirection and leave it on terminal too
- Could an autotrophic civilisation develop, or will evolving life forms always eat each other?
- The Hiding Rules Contradict Themselves. What Can I Make Of This?
- Was フリスマスケーキ used metaphorically about women?
- Scraping High-Res image tiles from the MoMA website
- How do I calculate approximate equity liquidity?
- Seven overlapping circles
- Noun form of "extant"?
- Preventing a Print expression from generating a new cell each time it executed
- How can I grant a non-Super User group permission to force check-in of articles?
- Where did the nickname of 'Bones' for McCoy come from?
- Why do tenured professors still publish in pay-walled venues?
- Make subscript size smaller (always)
- Do companies only pay dividends if they are in profit?
- If static methods can't be overridden, how its working here (For Java)?
- Should you always minimize cognitive load
- Religion After the Discovery of a Multiverse
- Carrying a handgun in other countries with a US concealed carry permit
- As a professional photographer, how can one handle a wedding photographer being rude or demanding in a way that will compromise overall image quality?
- Why moving fan seems transparent?
- How do I know when my sauce is reduced enough?

Work on work you love. From home.



stackoverflowcareers

▲ The explosion of results you're seeing can be easily explained:

3
▼

```
MATCH (g1:someIndex{name:"name1"}), (g2:someIndex{name:"name2"})
MATCH g1-[r1]-(), g2-[r2]-()
RETURN count(r1)+count(r2);
//Returns 1423740
```

In the 2nd line every combination of any relationship from `g1` is combined with any relationship of `g2`, this explains the number since $1423740 = 305 * 2334 * 2$. So you're evaluating basically a cross product here.

The right way to calculate the sum of all relationships for `name1` and `name2` is:

```
MATCH (g:someIndex)-[r]-()
WHERE g.name in ["name1", "name2"]
RETURN count(r)
```

share improve this answer

answered Apr 3 at 16:40



[Stefan Armbruster](#)

23.1k ● 2 ● 28 ● 51

Thank you Stefan! This makes sense now. +1 – [Shrey Gupta](#) Apr 3 at 20:45

add a comment

Your Answer

B *I*

Sign up or [log in](#)

Sign up using Google

Sign up using Facebook

Sign up using Stack Exchange

Post as a guest

Name

Email

required, but never shown

Post Your Answer

By posting your answer, you agree to the [privacy policy](#) and [terms of service](#).

Not the answer you're looking for? Browse other questions tagged [neo4j](#) [cypher](#) or [ask your own question](#).

[question feed](#)

[tour](#) [help](#) [blog](#) [chat](#) [data](#) [legal](#) [privacy policy](#) [work here](#) [advertising info](#) [mobile](#) [contact us](#) [feedback](#)

TECHNOLOGY

[Stack Overflow](#)

[Programmers](#)

[Database](#)

LIFE / ARTS

[Photography](#)

CULTURE /
RECREATION

[English Language &](#)

SCIENCE

[Mathematics](#)

OTHER

[Stack Apps](#)

| | | | | | | |
|------------------|--------------------------------|---------------------------|---------------------------|---------------------|------------------------------|------------------------|
| Server Fault | Unix & Linux | Administrators | Science Fiction & Fantasy | Usage | Cross Validated (stats) | Meta Stack Exchange |
| Super User | Ask Different (Apple) | Drupal Answers | | Skeptics | Theoretical Computer Science | Area 51 |
| Web Applications | WordPress Development | SharePoint | Graphic Design | Mi Yodeya (Judaism) | Physics | Stack Overflow Careers |
| Ask Ubuntu | | User Experience | Movies & TV | Travel | | |
| Webmasters | Geographic Information Systems | Mathematica | Seasoned Advice (cooking) | Christianity | MathOverflow | |
| Game Development | Electrical Engineering | Salesforce | Home Improvement | Arqade (gaming) | Chemistry | |
| TeX - LaTeX | Android Enthusiasts | ExpressionEngine® Answers | Personal Finance & Money | Bicycles | Biology | |
| | Information Security | more (13) | Academia | Role-playing Games | more (5) | |
| | | | more (9) | more (21) | | |