Abnormal performance drop with length increase of "in" operator argument

New issue

Closed **rsk700** opened this issue Apr 13, 2015 · 8 comments

rsk700 commented Apr 13, 2015

Was tested with 2.2, but looks like same valid for 2.1.7

STR:

Test data:

```
unwind range(1,100000) as r
create (:page {
category: 'category-' + str(r)
})
```

Check performance:

```
match (p:page)
where p.category in ['category-1']
return count(*)
```

58 ms

```
match (p:page)
where p.category in ['category-1', 'cat-1', 'cat-2', 'cat-3', 'cat-4', 'cat-5', 'cat-6', '
return count(*)
```

1059 ms

increasing length of list leads to significant performance drop

jexp commented Apr 13, 2015 Collaborator

did you create an index on `:page(category)` ?
otherwise it will have to scan all nodes and for each node do the IN check (which is not optimized using a Set yet)

jexp commented Apr 13, 2015 Collaborator

The second query executes in less than 40ms for me, please also note you misspelt the "category-xxx" values for everything > 1

jexp commented Apr 13, 2015 Collaborator

2.2 has an issue where it doesn't automatically use the index when the collection is an expression or parameter, but you can force index usage like this:

Labels

None yet

Milestone

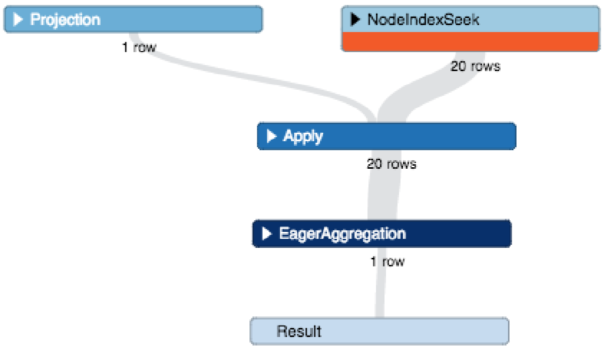
No milestone

Assignee

No one assigned

3 participants

```
profile
with [r in range(1,20) | 'category-'+r] as cats
match (p:page)
using index p:page(category)
where p.category in cats
return count(*)
```



jexp closed this Apr 13, 2015

rsk700 commented Apr 13, 2015

> did you create an index on :page(category) ?
> otherwise it will have to scan all nodes and for each node do the IN check (which is not optimized using a Set yet)

yes I know, I didn't create index on purpose, and yes both queries scan all nodes, but second query makes it much much slower

> The second query executes in less than 40ms for me, please also note you misspelt the "category-xxx" values for everything > 1

I did not misspell, it made on purpose, all queries return same value "1", "cat-x" is fake values to show issue, only one page "category-1" is counted

Can you say which neo4j version you was using? I have this performance issue on 2 servers.

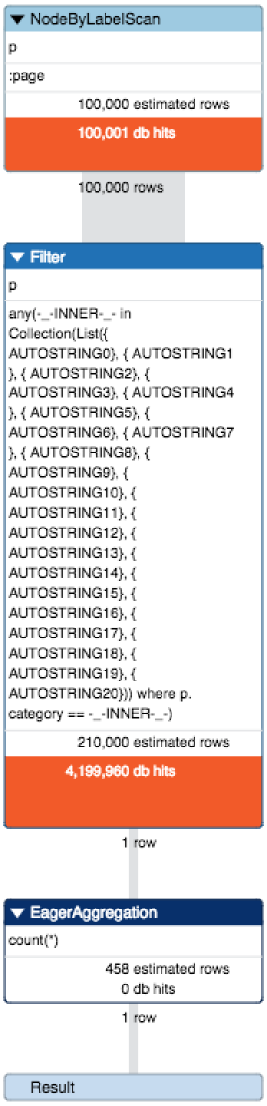
rsk700 commented Apr 13, 2015

@jexp you misunderstood issue please run queries exactly as I provided, there is definitely performance issue in comparing parameter with list

jexp commented Apr 13, 2015 Collaborator

Sure, I said so "IN comparison" across all nodes without an index is **not** an optimized operation, and if you do it for 100k nodes it takes a bit (4s on my machine).

And as you can see in the profile it creates quite a bit of db-hits:



NodeByLabelScan

p

:page

100,000 estimated rows

100,001 db hits

100,000 rows

Filter

p

any(-_-INNER-_- in Collection(List({ AUTOSTRING0}, { AUTOSTRING1 }, { AUTOSTRING2}, { AUTOSTRING3}, { AUTOSTRING4 }, { AUTOSTRING5}, { AUTOSTRING6}, { AUTOSTRING7 }, { AUTOSTRING8}, { AUTOSTRING9}, { AUTOSTRING10}, { AUTOSTRING11}, { AUTOSTRING12}, { AUTOSTRING13}, { AUTOSTRING14}, { AUTOSTRING15}, { AUTOSTRING16}, { AUTOSTRING17}, { AUTOSTRING18}, { AUTOSTRING19}, { AUTOSTRING20})) where p. category == -_-INNER-_-)

210,000 estimated rows

4,199,960 db hits

1 row

EagerAggregation

count(*)

458 estimated rows

0 db hits

1 row

Result

rsk700 commented Apr 13, 2015

Thanks for explanation, I see.

Can it be optimization request ;-) ? My point here this operation possible to make just once, from what I see neo4j runs query again and again separately for each member of list: `2 * 100000 * 21 = 4 200 000 db hits`

but it can be one operation similar to this:

```
match (p:page)
where p.category =~ 'category-1|cat-1|cat-2|cat-3|cat-4|cat-5|cat-6|cat-7|cat-8|cat-9|cat-
return count(*)
```

this gives 200 000 db hits

apivaral commented Apr 13, 2015

@jexp any plans to make "IN comparison" an optimised operation?