QGIS Application - Bug report #15752
Degradation of rendering performances in MSSQL provider

2016-10-25 05:17 AM - Andre Jesus

Status:	Closed		
Priority:	High		
Assignee:	Nyall Dawson		
Category:	Data Provider/MSSQL		
Affected QGIS version:	2.18.16	**Regression?:**	Yes
Operating System:	Windows	**Easy fix?:**	No
Pull Request or Patch supplied:	No	**Resolution:**	
Crashes QGIS or corrupts data:	No	**Copied to github as #:**	23674

Description

All version since 2.14.x raised the render time compared to 2.8.x:

https://s9.postimg.org/ip5dt924f/Screenshot_from_2016_10_25_09_07_44.png

I used the same virtual machine (install, test and used snapshot prior install) and the same data source.
I doesn't matter the source of the data, MSSQL, Shapefile, Postgres and the bigger the geometry quantity, bigger the render difference.
I looked for something in the settings to revert It without compromising the image quality but I couldn't find anything.

Is this a problem or It is just the way QGIS will work for now on?

Related issues:

Related to QGIS Application - Bug report # 16239: master: (much) slower rende...	**Closed**	**2017-02-27**
Related to QGIS Application - Bug report # 16577: Extremely slower time to op...	**Closed**	**2017-05-18**

Associated revisions

Revision 5798a82c - 2016-10-26 09:20 AM - Nyall Dawson

Speed up point layer rendering - don't calculate unused label obstacles

Cuts render time by ~60%. Fix #15752.

Revision 49432a84 - 2016-10-26 09:22 AM - Nyall Dawson

Optimise QgsAbstractGeometry

Make nCoordinates virtual, and provide shortcuts for some
geometry types. The base method which calls coordinateSequence()
is quite slow in certain circumstances.

Speeds up rendering point layers by ~25%, also likely to
speed up lots of geometry heavy operations throughout QGIS

Refs #15752

Revision 85897885 - 2016-10-26 11:22 AM - Nyall Dawson

Use QgsExpressionContextScope::addVariable instead of setVariable

...where appropriate (ie, read-only, non user set variables).
It's much faster as it doesn't need to check whether the
variable already exists.

Results in ~10% improvement in rendering speed. Refs #15752.

Revision 65a379be - 2016-10-27 11:58 PM - Nyall Dawson

Speed up point layer rendering - don't calculate unused label obstacles

Cuts render time by ~60%. Fix #15752.

(cherry-picked from 5798a82c8011ea7f44a1ed1d55ef0719786e8056)

Revision 0af73314 - 2016-10-28 12:25 AM - Nyall Dawson

Optimise QgsAbstractGeometry

Make nCoordinates virtual, and provide shortcuts for some
geometry types. The base method which calls coordinateSequence()
is quite slow in certain circumstances.

Speeds up rendering point layers by ~25%, also likely to
speed up lots of geometry heavy operations throughout QGIS

Refs #15752

(cherry-picked from 49432a84683e99bdc2592e7ae808e81fa3bc40cd)

Revision ee4acb0b - 2016-10-28 02:16 AM - Nyall Dawson

Use QgsExpressionContextScope::addVariable instead of setVariable

...where appropriate (ie, read-only, non user set variables).
It's much faster as it doesn't need to check whether the
variable already exists.

Results in ~10% improvement in rendering speed. Refs #15752.

(cherry-picked from 85897885445c7383938ac89318d12a7d37024bb4)

Revision 68cbf070 - 2016-10-28 05:26 AM - Nyall Dawson

Speed up point layer rendering - don't calculate unused label obstacles

Cuts render time by ~60%. Fix #15752.

(cherry-picked from 5798a82c8011ea7f44a1ed1d55ef0719786e8056)

Revision 5f01a872 - 2016-10-28 05:27 AM - Nyall Dawson

Optimise QgsAbstractGeometry

Make nCoordinates virtual, and provide shortcuts for some
geometry types. The base method which calls coordinateSequence()
is quite slow in certain circumstances.

Speeds up rendering point layers by ~25%, also likely to
speed up lots of geometry heavy operations throughout QGIS

Refs #15752

(cherry-picked from 49432a84683e99bdc2592e7ae808e81fa3bc40cd)

Revision c93d56a9 - 2016-10-28 05:31 AM - Nyall Dawson

Use QgsExpressionContextScope::addVariable instead of setVariable

...where appropriate (ie, read-only, non user set variables).
It's much faster as it doesn't need to check whether the
variable already exists.

Results in ~10% improvement in rendering speed. Refs #15752.

(cherry-picked from 85897885445c7383938ac89318d12a7d37024bb4)

Revision 62af54ec - 2017-06-02 04:33 AM - Nyall Dawson

[mssql] Use Filter instead of STIntersects to improve query performance

...and refine validity test from 57dc3c7

Using Filter is more performant since it does a bounding box only
check when an appropriate spatial index is available. This matches
the behavior with other providers, where the provider filter only
does a bounding box check and callers must perform the actual
intersection check if required.

While Filter also avoids SQL server closing the iterator upon
encountering invalid geometries, we can't rely on this because SQL
server will transparently fall back to STIntersects if it
decides there's no suitable spatial indexes available, and then
throw an exception on invalid geometries. So we still require
the STISValid check when using Filter.

The extent calculation from 57dc3c7 has been refined to
avoid the very expensive STMakeValid call. Instead we use a (ugly!)
workaround to skip invalid geometries using STIsValid only
inside the min/max x/y aggregates. Note that we can't just
dump a WHERE STIsValid clause in to the statement because SQL

server still throws the exception. Gotta love it!

Fix #15752, #10947

Revision dafeaf43 - 2018-10-08 08:15 AM - Nyall Dawson

[mssql][needs-docs] Add connection setting to ignore invalid geometry handling

Sets whether the connection should skip all handling of records with
invalid geometry, which are slow and costly.

This speeds up the provider, however, if any invalid geometries
are present in a table then the result is unpredictable and may
include missing records. Only check this option if you are certain
that all geometries present in the database are valid, and any newly
added geometries or tables will also be valid!

Why would we want this? Well, SQL Server invalid geometry handling
is AWEFUL. A seriously lame, data mangling and corrupting
piece of s***. Use Postgres instead. But if you can't, then you
can at least choose to use your layers at full speed, if you
can take the responsibility that a SINGLE invalid geometry
hiding somewhere in the table will result in a whole bunch
of missing (valid) features.

SQL server is at fault here, not us. There's nothing we (or
GDAL, or MapServer, or GeoServer, or anyone else) can do
to fix this.

Suffice to say, this option is off by default, as we're better
to have a slow provider which actually shows all features.

Fixes #15752

Rant over

Revision 5503ba48 - 2018-10-08 08:15 AM - Nyall Dawson

[mssql] Forward port more of 62af54e

But avoid the inexact Filter test when we are doing an
exact intersection request

Refs #15752

History

#1 - 2016-10-25 11:58 AM - Nyall Dawson

What renderer/symbol style are you using?

#2 - 2016-10-25 11:58 AM - Nyall Dawson

- Status changed from Open to Feedback

#3 - 2016-10-25 12:26 PM - Andre Jesus

What renderer/symbol style are you using?
I'm using default QGIS style.

The same occurs using WFS:

ly1 WFS/linestring

2.8.x = 16 ms
2.14.x+ = 517 ms

ly2 WFS/polygon

2.8.x = 17 ms
2.14.x+ = 219 ms

ly3 WFS/point

2.8.x = 15 ms
2.14.x+ = 647 ms

Always using the same bookmark and data.

I have 20 other machines with the same results.

#4 - 2016-10-25 12:29 PM - Andre Jesus

Sorry, I don't know if my answer was clear. I'm using default Single Symbol style without any change.

#5 - 2016-10-26 12:21 AM - Nyall Dawson

- Status changed from Feedback to Closed

Fixed in changeset commit:"5798a82c8011ea7f44a1ed1d55ef0719786e8056".

#6 - 2016-11-25 05:52 AM - Steve Lowman

- Target version changed from Version 2.18 to Version 2.14
- Status changed from Closed to Reopened

Has this been fixed in 2.18 but not 2.14? I just changed the Target version from 2.18 to 2.14 - hope that's ok.

We use a large feature set of UK Ordnance Survey MasterMap (OSMM) Topographic Layer in shapefiles in the LTR (currently 2.14.8), about 100K ha area. The data are stored on a local server in the same building, so bandwidth is not a big part of this issue. Topographic Line is geometrically the most complex OSMM feature set, and this has a noticeable lag in rendering with this large feature set. The lag is > 2 secs at 1:20K scale, slower at smaller scales (larger visible extent), quicker at larger scales (smaller visible extent).

Our workaround is to break up the feature set into five smaller chunks, and then the lag is a lot less at any given scale. This is a pain, because we have to divide it up every time we get an updated OSMM feature set.

When this first started happening was the first release of 2.14, and I assumed it was probably caused by the more complex geometry engine that allows curved lines.

#7 - 2016-11-25 11:58 AM - Nyall Dawson

- Status changed from Reopened to Closed

These fixes are included in 2.14.9 and 2.18.1. Try with those versions. Otherwise this is likely a different issue and a new report needs to be opened.

Fyi, new geoemtry was introduced in 2.10

#8 - 2016-11-30 07:50 AM - Andre Jesus

- Target version changed from Version 2.14 to Version 2.18
- Status changed from Closed to Reopened

I did the compare again using QGIS 2.18.1

https://s17.postimg.org/p5b8324nj/DATA.png

I can see some improvements rendering polygons, but not on points and linestrings.

ps.: Render times are rounded.

#9 - 2016-11-30 02:35 PM - Nyall Dawson

Hmm... that's quite odd. You should definitely see an improvement in 2.18.1.

Are you using labels in these tests?

#10 - 2016-12-01 07:51 AM - Andre Jesus

The last time I tested It I did a dirt install.
To cross It out I set up a new VM with Windows 7 32 bits, made a snapshot before installing any software.
I didn't tweaked any setting in QGIS but enable Debug: Map Canvas Refresh

Here the new results:

https://s18.postimg.org/kxwxdp4kp/data2.png

Yes, I was using simple style without labels or any change to default QGIS style.

https://s17.postimg.org/lnh8gvuxb/data3.png

#11 - 2017-01-04 11:20 PM - Giovanni Manghi

- Category set to Map Canvas

#12 - 2017-03-06 09:44 AM - Andre Jesus

Latest update: Using 2.18.4

Render time seems to keep rising every update.

Every relatively big layer I have now became a slug with version 2.18.4.

To me, It looks like QGIS doesn't pass the bounding box when querying data from the database, bringing all data all the time. If I query 1000 or all 170.000 features, the rendering time is about the same. While, if I do the same using PostgreSQL I jump from 42 ms (1.000 feature count) to 2400 ms (173.000 features)

173.000 polygon layer, average render time (3 runs):

1:125.000 (173.000 features)
Postgres: 2383 ms
Mssql: 20151 ms

1:4.000 (1.300 features)
Postgres: 140 ms
Mssql: 13756 ms

As shown above, PostgreSQl had a 94% render time reduction and MSSQL only 31%, that considering a 99% features displayed reduction.

#13 - 2017-03-06 09:52 AM - Giovanni Manghi
- Affected QGIS version changed from 2.18.0 to 2.18.4

#14 - 2017-03-07 10:05 AM - Andre Jesus

I did some more testing...

Using the same map, layers and zoom levels.

Feature list and count:
http://i.imgur.com/qHBfJee.png

Zoom: 1:400.000
http://i.imgur.com/opB6UHE.png

http://i.imgur.com/LyWia7l.png

Zoom: 1:7.000
http://i.imgur.com/JJc3wvR.png

http://i.imgur.com/vcf3SaK.png

Wider zoom levels:
2.8.9 > 2.14.12 = + 41% render time
2.14.12 > 2.18.4 = **+ 290% render time**

Smaller zoom levels:
2.8.9 > 2.14.12 = + 7% render time
2.14.12 > 2.18.4 = **+ 4604% render time**

I know MSSQL data provider is not the focus in development but it's impossible to work with It in 2.18.4.

#15 - 2017-03-07 10:09 AM - Giovanni Manghi

> *I know MSSQL data provider is not the focus in development but it's impossible to work with It in 2.18.4.*

please raise this matter on the users and/or developers mailing lists.

#16 - 2017-03-07 10:35 AM - Andre Jesus

Giovanni Manghi wrote:

> *I know MSSQL data provider is not the focus in development but it's impossible to work with It in 2.18.4.*
>
> *please raise this matter on the users and/or developers mailing lists.*

Ok.

Each one would be more suitable: User or Developers mailing list? I've never joined one before.
I'm no developer but I'd like to contribute as I can.

#17 - 2017-03-07 10:49 AM - Giovanni Manghi

Andre Jesus wrote:

> *Giovanni Manghi wrote:*
>
> > *I know MSSQL data provider is not the focus in development but it's impossible to work with It in 2.18.4.*
> >
> > *please raise this matter on the users and/or developers mailing lists.*
>
> *Ok.*
>
> *Each one would be more suitable: User or Developers mailing list? I've never joined one before.*
> *I'm no developer but I'd like to contribute as I can.*

the developers one (which is not followed just by developers) is the right place for me: there you find the people that may help understand because of this regression.

#18 - 2017-03-08 04:17 AM - Giovanni Manghi

Andre Jesus wrote:

> *Giovanni Manghi wrote:*
>
> > *I know MSSQL data provider is not the focus in development but it's impossible to work with It in 2.18.4.*
> >
> > *please raise this matter on the users and/or developers mailing lists.*
>
> *Ok.*
>
> *Each one would be more suitable: User or Developers mailing list? I've never joined one before.*
> *I'm no developer but I'd like to contribute as I can.*

One question: have you any idea if this also affects other providers, like PostGIS?

#19 - 2017-03-08 06:44 AM - Andre Jesus

Giovanni Manghi wrote:

> *One question: have you any idea if this also affects other providers, like PostGIS?*

It doesn't.

Running the same bench test using Shapefiles or PostGIS both gives me ~1500 ms

#20 - 2017-03-08 01:28 PM - Andre Jesus

Just checking how performance went with other data providers over the 3 versions:

Using the same map but with a different zoom level (1:177.000)
http://i.imgur.com/7xm8tHf.png

full printscreens

While doing the benchmark, QGIS force closed 5 times using 2.18.4 and 2 using 2.14.12 when refreshing MSSQL layers. None with the other providers.

2.8.9 is by far the fastest, but It's understandable by the amount of features added to QGIS It would slow It down.

#21 - 2017-03-08 02:46 PM - Giovanni Manghi

> *2.8.9 is by far the fastest, but It's understandable by the amount of features added to QGIS It would slow It down.*

well... I don't think that we should assume that is all ok also with other providers... it seems to me that is pretty bad also for postgis and shapefiles.

I noticed a very noticeable degradation of performances in QGIS3 master compared to 2.18.4 and reported it, hopefully this will be sorted and fixed for all providers.

#22 - 2017-03-09 09:04 AM - Andre Jesus

It's not related how QGIS request data from the database as all version uses the same intersects method with the same limit.
And that kills my theory he may be request all geometries all the time.

```
SELECT [OBJECTID],[GEOMETRY]FROM [dbo].[LIGACOES] where [GEOMETRY].STIsValid() = 1 AND
[GEOMETRY].STIntersects([geometry]::STGeomFromText('POLYGON((589164.07187500002328306 8262041.58740026596933603,
614628.80312499997671694 8262041.58740026596933603, 614628.80312499997671694 8286332.41259973403066397,
589164.07187500002328306 8286332.41259973403066397, 589164.07187500002328306 8262041.58740026596933603))',29191)) = 1
```

But checking SQL Server Profiler I noticed different values in CPU cycles and duration, even they all using the exactly same query, the values ares persistent.

Again a 3 times avg:

http://i.imgur.com/gInM45M.png

It could be a good lead to find out what went wrong, but my knowledge ends here.

#23 - 2017-03-31 05:15 AM - Andre Jesus

Update versions: 2.14.13 and 2.18.5

Loading a 5 layers project (feature count: 201.528 points, 36.521 linestrings, 24.957 linestrings, 9.231 polygons, 9.875 polygons), 5x avg canvas refresh time, roughly rounded values:

2.14.13
3500 ms mssql
1500 ms postgis

2.18.5
19443 ms mssql
1400 ms postgis

2.14.13 seems to have stabilized the performance using MSSQL. On the other hand 2.18.5 is getting worse, jumping from ~15.000 ms to ~20.000 ms (25% increase)

Edit: 2.14.13 feels faster now

#24 - 2017-04-12 04:54 PM - Nyall Dawson

- Status changed from Reopened to Feedback

To summarise - I believe that when disregarding the comparisons between release and debug builds the issue is reduced to a regression in the mssql provider alone. Is this correct?

#25 - 2017-04-13 04:09 AM - Andre Jesus

None of the tests I did here were using debug versions (by that time I didn't even know I was able to install and use them).

I think the performance degradation does impact all providers, just not as much as It impacts MSSQL (see note #15752-20).

#26 - 2017-04-29 06:23 AM - Giovanni Manghi

- Status changed from Feedback to Open

I think that note 20 says a lot. Many thanks to the reported for this very detailed analysis!

#27 - 2017-04-30 05:06 PM - Giovanni Manghi

- Regression? set to Yes

#28 - 2017-04-30 05:08 PM - Giovanni Manghi

- Priority changed from Severe/Regression to High

#29 - 2017-05-01 01:10 AM - Giovanni Manghi

- Easy fix? set to No

#30 - 2017-05-02 06:09 PM - Andre Jesus

Update: 2.14.14 LTS / 2.18.7

http://i.imgur.com/urWF6Dq.png

Using the same layers used to benchmark on note 20.
Testing the first run and 3 consecutive canvas refreshes using 1:255.000 and 1:7.000 scales. Using default OSGeo installation, no extra plugin installed.
I had an OS problem and I lost 2.8.9 installer, that's why there is no results for it.

MSSQL performance is still terrible, but PostGIS had a nice improvement.

#31 - 2017-05-18 11:25 PM - Giovanni Manghi

- Description updated

see also #16577

#32 - 2017-05-19 08:56 AM - Jürgen Fischer

- Related to Bug report #16577: Extremely slower time to open attribute table in 2.18.7 compared to 2.14.14 added

#33 - 2017-05-20 09:37 PM - Giovanni Manghi

see also: https://lists.osgeo.org/pipermail/qgis-psc/2017-May/005258.html

#34 - 2017-05-23 10:47 AM - Nyall Dawson

- Status changed from Open to Closed
- % Done changed from 0 to 100

Applied in changeset commit:qgis|5798a82c8011ea7f44a1ed1d55ef0719786e8056.

#35 - 2017-05-23 12:52 PM - Giovanni Manghi

Nyall Dawson wrote:

> *Applied in changeset commit:qgis|5798a82c8011ea7f44a1ed1d55ef0719786e8056.*

Dear Andre Jesus would you mind see if the above commit improves things as expected? thanks!

#36 - 2017-05-23 03:08 PM - Giovanni Manghi

Giovanni Manghi wrote:

> *Nyall Dawson wrote:*
>
> > *Applied in changeset commit:qgis|5798a82c8011ea7f44a1ed1d55ef0719786e8056.*
>
> *Dear Andre Jesus would you mind see if the above commit improves things as expected? thanks!*

see also https://drive.google.com/file/d/0B8VD1V_leaeWNmpaOFNxZGxicVU/view?usp=sharing

#37 - 2017-05-26 04:23 PM - Giovanni Manghi

- Subject changed from Slow render time to Degradation of rendering performances in MSSQL provider
- Status changed from Closed to Open

#38 - 2017-05-26 04:32 PM - Giovanni Manghi

reopened as requested here

https://lists.osgeo.org/pipermail/qgis-psc/2017-May/005319.html

#39 - 2017-06-03 08:03 AM - Nyall Dawson

- Status changed from Open to Closed

Applied in changeset commit:qgis|62af54ecb2666d0bde0f7af102d663ded5cda97d.

#40 - 2017-06-06 02:39 PM - Andre Jesus

Giovanni Manghi wrote:

> *Dear Andre Jesus would you mind see if the above commit improves things as expected? thanks!*

Sorry I was on vacation.

Are these changes applied to the final Windows build?
Comparing the numbers alone I don't see improvements, but in the daily usage I find 2.14.15 LTR to be good enough.
On the other hand, 2.18.x is a totally skip.

I compared It using #note-14 because I could use the same layers/scales.

Using QGIS 2.18.9 64 bits, Windows Standalone installer. Projected layers.
http://i.imgur.com/GG5aYsH.png

Using QGIS 2.14.15 LTR 64 bits, Windows Standalone installer. Projected layers.
http://i.imgur.com/rUUy2hU.png

#41 - 2017-06-06 02:42 PM - Giovanni Manghi

you probably have to try qgis master or 2.18.9 nightly or wait the next 2.18 build (but I'm not sure the fix has been backported)

Andre Jesus wrote:

> *Giovanni Manghi wrote:*
>
> > *Dear Andre Jesus would you mind see if the above commit improves things as expected? thanks!*
>
> *Sorry I was on vacation.*
>
> *Are these changes applied to the final Windows build?*
> *Comparing the numbers alone I don't see improvements, but in the daily usage I find 2.14.15 LTR to be good enough.*
> *On the other hand, 2.18.x is a totally skip.*
>
> *I compared It using #note-14 because I could use the same layers/scales.*
>
> *Using QGIS 2.18.9 64 bits, Windows Standalone installer. Projected layers.*
> *http://i.imgur.com/GG5aYsH.png*
>
> *Using QGIS 2.14.15 LTR 64 bits, Windows Standalone installer. Projected layers.*
> *http://i.imgur.com/rUUy2hU.png*

#42 - 2017-06-06 09:02 PM - Nyall Dawson

These changes aren't in master yet, or any of the released 2.18 versions. Can you please install the qgis-rel-dev version using osgeo4w add test using that?

#43 - 2017-06-06 11:28 PM - Andre Jesus

Nyall Dawson wrote:

> *These changes aren't in master yet, or any of the released 2.18 versions. Can you please install the qgis-rel-dev version using osgeo4w add test using that?*

Aren't those versions debug-enabled? I remember from another issue the comparison between final and dev build irrelevant as the performance is degraded because of the debug flags.

I installed QGIS-OSGeo4W-2.99.0-23-Setup-x86_64.exe (05-Jun-2017 06:29 463M) from the weekly builds (http://qgis.org/downloads/weekly/).

The results are close or worse than the QGIS 2.18.9 final build times.

#44 - 2017-06-07 12:19 AM - Nyall Dawson

You could compare it against qgis-ltr-dev. That's a debug enabled 2.14 release.

What's the commit number from the about screen in the weekly you are running? I suspect its probably from before the recent changes.

#45 - 2017-06-07 12:20 AM - Nyall Dawson

Actually I just realised you mentioned 2.99. The fixes aren't in master builds yet. You'll need to test qgis-rel-dev from osgeo4w, that's the only pre-built version with these changes available.

#46 - 2017-06-07 03:00 PM - Andre Jesus

https://i.imgsafe.org/7f7d806fd5.png

Even with the debug flags qgis-rel-dev was faster than the 2.18 final release. I guess we can expect the same performance between 2.14.x and 2.18.x for now on. Thank you, Nyall!

#47 - 2017-06-07 09:27 PM - Nyall Dawson

I'm confused - did you attach the wrong table above? It looks like it's still slower to me?

#48 - 2017-06-07 09:49 PM - Andre Jesus

Nyall Dawson wrote:

> *I'm confused - did you attach the wrong table above? It looks like it's still slower to me?*

Comparing 2.18.x vs 2.14.x both dev versions, 2.18 is slower than 2.14.
But when comparing 2.18.x dev to 2.18.x final, even with the debug flags the dev is faster than the final version.

I confess I got a little excited there and maybe the 2.18 **still** doesn't have the same performance as 2.14 has, but it's a great improvement, the render time may drop another 30-40% with debug off...

Just out of curiosity, I installed QGIS 2.8.9-2 x86 again, which is the fastest version so far, to see how far off the 2.14.x is from it. It's pretty close, 2.8.9 is still the fastest but only by ~8%.

#49 - 2017-06-08 12:25 AM - Nyall Dawson

Ok - let's do some more tests. I'm curious if it's the added IsValid check which is causing these remaining regressions (you can see discussion about why this check is required at https://github.com/qgis/QGIS/pull/4642)

Can you run these benchtests directly on your sql server to get the timing for the query on the server alone (no qgis involved):

- select everything from the layers , no where clause
- select everything from the layers, with an 'WHERE geom.STIsValid() = 1' clause
- select from the layers using a restricted bounding box check (eg something like 'WHERE geom.Filter(Polygon::STGeomFromText('POLYGON)',4326)) = 1', but with the correct crs/etc)
- same as above, but with a 'WHERE geom.STIsValid() = 1 AND geom.Filter....etc)

I'd like to see how much of an impact the validity checks are adding on your query times. It would be good to narrow down the regression to the presence of the validity check and not other factors so we can refine further investigation.

#50 - 2017-06-08 02:46 PM - Andre Jesus

results: https://drive.google.com/open?id=11GY4OxK0Xy2HULLrJ1xB4MuLAZcpF-MP7ADUKLiHBRg

tl;dr

Using IsValid has a huge impact in performance in points and polygons, linestrings also has a big slow down but not as much.
What was interesting to see was how big the difference when using both bbox filter and isValid together is. It basically sums bbox filter + isValid cpu times and doubles it!

just another 2 cents: geom.filter is slightly better than the geom.STIntersects it's currently used (commit:57dc3c7eff2337411c163222a4e5b7562fffd457)

#51 - 2017-06-27 07:42 PM - Andre Jesus

Updated results:

http://i.imgur.com/xQiIbsw.png

#52 - 2017-08-10 02:51 PM - Andre Jesus

Sorry to keep bothering about it, but I don't think it's closed as the current 2.18.x versions are still 6x slower than the 2.14.x versions.

And the 2.18.x becoming the next LTR version it really worries me.

#53 - 2017-08-10 04:41 PM - Giovanni Manghi

- Status changed from Closed to Feedback

Andre Jesus wrote:

> *Sorry to keep bothering about it, but I don't think it's closed as the current 2.18.x versions are still 6x slower then the 2.14.x versions.*
>
> *And the 2.18.x becoming the next LTR version it really worries me.*

I assume that have you tried the latest available 2.18, correct?

#54 - 2017-08-10 04:57 PM - Andre Jesus

- File 2.18.11.jpg added

Giovanni Manghi wrote:

> *I assume that have you tried the latest available 2.18, correct?*

I keep testing every version for improvements.

#55 - 2017-08-11 09:27 AM - Giovanni Manghi

- Affected QGIS version changed from 2.18.4 to 2.18.11

- Status changed from Feedback to Open

#56 - 2017-10-02 09:31 PM - Andre Jesus

New results: 2.18.13, 2.14.19, 2.8.9

https://i.imgur.com/TPOq61c.png

#57 - 2018-02-15 02:41 PM - Stijn Van der Linden

Hello,

Is this still being looked at? With the upcoming release of QGIS 3.0, the LTR version will become 2.18.x instead of 2.14.x. I just checked 2.18.16 versus 2.14.22 and the problem persists. Loading a MSSQL layer in 2.18.16 takes several minutes compared to the same layer in 2.14.22, which is loaded in a matter of seconds.

This makes 2.18.16 unusable unless I don't use SQL sources (which is notreally an option).

#58 - 2018-03-04 11:11 AM - Giovanni Manghi

- Affected QGIS version changed from 2.18.11 to 2.18.16

#59 - 2018-03-05 02:34 AM - Nyall Dawson

I've looked into this issue in depth and there's no easy answers here.

We have to choose between:

1. Full performance - yet undefined behavior whenever a layer contains invalid geometries. (Undefined behavior here will mean missing features from the mssql table in the QGIS layer. Not just those with invalid geometries, but any feature present AFTER an invalid geometry is encountered will also be discarded.)

or

2. Slower performance - yet handling invalid geometries correctly and ensuring that all features from the table are present in the QGIS layer.

There's no alternative... save from begging MS to add support for disabling the automatic geometry validity check!

So what I propose is adding a new checkbox to the MS SQL connection properties dialog - "No invalid geometry handling". If checked, you're telling QGIS that you accept all the risks and want to disable the invalid geometry handling. You'll be missing features if your tables have invalid geometries. If unchecked (default), you tell QGIS to keep running the slower code which correctly handles invalid geometries in the table.

#60 - 2018-03-05 10:29 AM - Stijn Van der Linden

Hi,

I think that would be a very good solution. The SQL geometry databases I use are very well maintained and have a quality control that insures a very low chance of invalid geometries.

Looking forward to this!

#61 - 2018-03-05 12:09 PM - Andre Jesus

Nyall Dawson wrote:

> *1. Full performance - yet undefined behavior whenever a layer contains invalid geometries. (Undefined behavior here will mean missing features from the mssql table in the QGIS layer.* ***Not just those with invalid geometries, but any feature present AFTER an invalid geometry is encountered will also be discarded****.)*

I saw this happened more than once in previous 2.x version. It would not draw any geometry if there was one invalid geometry in the bbox.
At least It would let me pin point the nearby location where that invalid geometry is if a validation were not enough to detect it.

Nyall Dawson wrote:

> *So what I propose is adding a new checkbox to the MS SQL connection properties dialog - "No invalid geometry handling". If checked, you're telling QGIS that you accept all the risks and want to disable the invalid geometry handling. You'll be missing features if your tables have invalid geometries. If unchecked (default), you tell QGIS to keep running the slower code which correctly handles invalid geometries in the table.*

That would be best approach. There is no beginners in QGIS that start using QGIS with a full relational database as data source.

Maybe the validation could occur only when the feature is added or before saving new/modified geometry, not every time a select is made.

Yet, the checkbox is the simpler and fastest solution.

by the way, I could not wait for a solution so i migrated by data to Postgresql/Postgis

#62 - 2018-10-02 10:18 PM - Nyall Dawson

- Category changed from Map Canvas to Data Provider/MSSQL

#63 - 2018-10-08 06:31 AM - Nyall Dawson

- Assignee set to Nyall Dawson
- Status changed from Open to In Progress

#64 - 2018-10-08 08:14 AM - Nyall Dawson

- Status changed from In Progress to Closed

Applied in changeset commit:qgis|dafeaf4372b9e029160438345cce852dbfcf0741.

Files

2.18.11.jpg	404 KB	2017-08-10	Andre Jesus