

Status:	Open	
Priority:	High	
Assignee:		
Category:	Data Provider/MSSQL	
Affected QGIS version:	3.2.2	Regression?: Yes
Operating System:	Microsoft Windows [Version 10.0.17134.228]	Easy fix?: No
Pull Request or Patch supplied:	No	Resolution:
Crashes QGIS or corrupts data:	No	Copied to github as #: 27618

Description

Description:=====

Opening a .QGS which has a layer from a MS SQL Server view, takes drastically more startup time than when the layer is connection directly with the underlying table/featureclass.

The difference in my testcase from ~1 sec. to ~157 sec. (see below), and occurs at everyone QGIS start up!

Testcase & side conditions:=====

I did some evaluation and testing with the SQL Server Profiler and a table/featureclass with ~ 1.7 millions rows/objects.

Here are the results of the profiling:

1. SQL Trace from starting qgis with the underlying table/featureclass its.dkm_gst => execution time ~1 sec.

```
exec sp_datatype_info 11
go
SET QUOTED_IDENTIFIER ON
go
SELECT name FROM sys.columns WHERE is_computed = 1 AND object_id = OBJECT_ID('[its].[DKM_GST]')
go
exec sp_columns @table_name = 'N'DKM_GST', @table_owner = 'its'
go
exec sp_pkeys @table_name = 'N'DKM_GST', @table_owner = 'its'
go
SELECT min(bounding_box_xmin), min(bounding_box_ymin), max(bounding_box_xmax), max(bounding_box_ymax) FROM
sys.spatial_index_tessellations WHERE object_id = OBJECT_ID('[its].[DKM_GST]')
go
SET QUOTED_IDENTIFIER ON
go
SELECT [OBJECTID],[SHAPE]FROM [its].[DKM_GST] where [SHAPE].STIsValid() = 1 AND
[SHAPE].STIntersects([geometry]::STGeomFromText('POLYGON',31255)) = 1
go
SET QUOTED_IDENTIFIER ON
go
SELECT [OBJECTID],[SHAPE]FROM [its].[DKM_GST] where [SHAPE].STIsValid() = 1 AND
[SHAPE].STIntersects([geometry]::STGeomFromText('POLYGON',31255)) = 1
go
-----
```

2. SQL trace from starting qgis with the corresponding flat view its.v_dkm_gst (=> select * from [its].[DKM_GST]) => execution time ~157 sec.

```
-----
exec sp_datatype_info 11
go
SET QUOTED_IDENTIFIER ON
go
SELECT name FROM sys.columns WHERE is_computed = 1 AND object_id = OBJECT_ID('[its].[v_DKM_GST]')
go
exec sp_columns @table_name = N'v_DKM_GST', @table_owner = 'its'
go
exec sp_pkeys @table_name = N'v_DKM_GST', @table_owner = 'its'
go
select count(distinct [OBJECTID]), count([OBJECTID]) from [its].[v_DKM_GST]
go
SELECT min(bounding_box_xmin), min(bounding_box_ymin), max(bounding_box_xmax), max(bounding_box_ymax) FROM
sys.spatial_index_tessellations WHERE object_id = OBJECT_ID('[its].[v_DKM_GST]')
go
select min([SHAPE].MakeValid().STPointN(1).STX), min([SHAPE].MakeValid().STPointN(1).STY),
max([SHAPE].MakeValid().STPointN(1).STX), max([SHAPE].MakeValid().STPointN(1).STY) from [its].[v_DKM_GST]
go
SET QUOTED_IDENTIFIER ON
go
SELECT [OBJECTID],[SHAPE]FROM [its].[v_DKM_GST] where [SHAPE].STIsValid() = 1 AND
[SHAPE].STIntersects([geometry]::STGeomFromText('POLYGON',31255)) = 1
go
SET QUOTED_IDENTIFIER ON
go
SELECT [OBJECTID],[SHAPE]FROM [its].[v_DKM_GST] where [SHAPE].STIsValid() = 1 AND
[SHAPE].STIntersects([geometry]::STGeomFromText('POLYGON',31255)) = 1
go
-----
```

Reason:=====

=> Detecting the bounding box of the layer! takes ~156 sec. (1.700.000 rows!!!)

```
select min([SHAPE].MakeValid().STPointN(1).STX), min([SHAPE].MakeValid().STPointN(1).STY),
max([SHAPE].MakeValid().STPointN(1).STX), max([SHAPE].MakeValid().STPointN(1).STY) from [its].[v_DKM_GST]
go
```

In case of the layer directly connected to the table the query

```
SELECT min(bounding_box_xmin), min(bounding_box_ymin), max(bounding_box_xmax), max(bounding_box_ymax) FROM
sys.spatial_index_tessellations WHERE object_id = OBJECT_ID('[its].[DKM_GST]')
```

returns the bounding box via the SQL Server Metatable,

In case of the view

```
SELECT min(bounding_box_xmin), min(bounding_box_ymin), max(bounding_box_xmax), max(bounding_box_ymax) FROM
```

```
sys.spatial_index_tessellations WHERE object_id = OBJECT_ID('[its].[v_DKM_GST]')
```

go

returns Null (of course is not physical table)!

Solution Proposals:=====

Extending the qgis metatable "dbo.geometry_columns" with 4 bounding box fields and also a field for the primary key, than the detection of the bounding box and the primary key field goes to the responsibility of the project/db-administration!?!

Manually inserting into the "sys.spatial_index_tessellations" with a dummy record doesn't work, because it's forbidden => Ad hoc updates to system catalogs are not allowed!

After the initial startup, any further action (Zooming, Panning, etc.) works with the same excellent performance as with the underlying table!

Hopefully I could describe the issue sharply, thanks in Advance!

Alex

PS: I assigned the priority to high, because under enterprise conditions often you have
no direct access to the underlying tables and of course it occurs on each QGIS starting!
I didn't test it under QGIS Server conditions.

History

#1 - 2018-09-07 01:11 PM - Giovanni Manghi

- Priority changed from High to Normal

- Subject changed from MSSQL: Poor initial QGIS 3.2.2 starting performance when getting data from a MS SQL Server view! to MSSQL: Poor initial QGIS 3.2.2 starting performance when getting data from a MS SQL Server view

Was/is the case also on 2.18?

#2 - 2018-09-07 01:11 PM - Giovanni Manghi

- Status changed from Open to Feedback

#3 - 2018-09-07 03:34 PM - Alexander Zidek

I did a quick QGIS 2.18.9 installation and test it against the given data.

Short summary, at very first time to load the layer (view) into QGIS it takes 258 sec., but now the good news, after reopen the qgs, it needs only 1 sec, it's completely different to version 3.2.2!

This would be feasible in daily work, but if you're increasing the record count the initial time would also linearly be increased!?!

Thanks,

Alex

Below the results of the MS SQL Profiler tracings for QGIS 2.18.19!

-- 2.18.19 trace at the very first time to load the layer (view) into qgis => takes 258 sec.

```
SELECT f_table_schema, f_table_name, f_geometry_column, coord_dimension, g.srid, srtext, geometry_type FROM dbo.geometry_columns g JOIN
```

```

INFORMATION_SCHEMA.COLUMNS ON f_table_schema = TABLE_SCHEMA and f_table_name = TABLE_NAME and f_geometry_column =
COLUMN_NAME left outer join dbo.spatial_ref_sys s on g.srid = s.srid
go
SELECT srtext FROM spatial_ref_sys WHERE srid = 31255
go
select count() from its.DKM_GST
go
select count() from its.DKM_GST
go
select count() from its.v_DKM_GST
go
select count() from its.EIGENTUEMER_GRUNDSTUECK
go
SELECT f_table_schema, f_table_name, f_geometry_column, coord_dimension, g.srid, srtext, geometry_type FROM dbo.geometry_columns g JOIN
INFORMATION_SCHEMA.COLUMNS ON f_table_schema = TABLE_SCHEMA and f_table_name = TABLE_NAME and f_geometry_column =
COLUMN_NAME left outer join dbo.spatial_ref_sys s on g.srid = s.srid
go
SELECT srtext FROM spatial_ref_sys WHERE srid = 31255
go
select count(*) from its.v_DKM_GST
go
select [SHAPE], [OBJECTID], [GNR], [KG], [G], [KG_GNR], [GB], [EZ], [GB_EZ], [MBL_BEZ], [VHW], [FLAECHE], [SE_ANNO_CAD_DATA],
[GDO_GID] from its.v_DKM_GST
go
SELECT f_table_schema, f_table_name, f_geometry_column, coord_dimension, g.srid, srtext, geometry_type FROM dbo.geometry_columns g JOIN
INFORMATION_SCHEMA.COLUMNS ON f_table_schema = TABLE_SCHEMA and f_table_name = TABLE_NAME and f_geometry_column =
COLUMN_NAME left outer join dbo.spatial_ref_sys s on g.srid = s.srid
go
SELECT srtext FROM spatial_ref_sys WHERE srid = 31255
go
select [SHAPE] from its.v_DKM_GST where [SHAPE].STIntersects(geometry::STGeomFromText('POLYGON',31255)) = 1
go

-- 2.18.19 trace after reopen the qgs => takes ~1 sec.
SELECT f_table_schema, f_table_name, f_geometry_column, coord_dimension, g.srid, srtext, geometry_type FROM dbo.geometry_columns g JOIN
INFORMATION_SCHEMA.COLUMNS ON f_table_schema = TABLE_SCHEMA and f_table_name = TABLE_NAME and f_geometry_column =
COLUMN_NAME left outer join dbo.spatial_ref_sys s on g.srid = s.srid
go
SELECT srtext FROM spatial_ref_sys WHERE srid = 31255
go
select count(*) from its.v_DKM_GST
go
SELECT f_table_schema, f_table_name, f_geometry_column, coord_dimension, g.srid, srtext, geometry_type FROM dbo.geometry_columns g JOIN
INFORMATION_SCHEMA.COLUMNS ON f_table_schema = TABLE_SCHEMA and f_table_name = TABLE_NAME and f_geometry_column =
COLUMN_NAME left outer join dbo.spatial_ref_sys s on g.srid = s.srid
go
SELECT srtext FROM spatial_ref_sys WHERE srid = 31255
go
select [SHAPE] from its.v_DKM_GST where [SHAPE].STIntersects(geometry::STGeomFromText('POLYGON',31255)) = 1
go

```

#4 - 2018-09-07 06:17 PM - Giovanni Manghi

- Status changed from Feedback to Open
- Regression? changed from No to Yes
- Priority changed from Normal to High

#5 - 2018-09-07 09:07 PM - Jürgen Fischer

- Status changed from Open to Feedback

did you enable use estimated metadata on the connection?

#6 - 2018-09-08 03:15 PM - Alexander Zidek

Yes, of course ... below see the datasource tag of the layer in my test.qgs:

```
<datasource>dbname='its_gis' host=its_gis estimatedmetadata=true srid=31255 type=Polygon table="its"."v_DKM_GST" (SHAPE) sql=</datasource>
```

But I think the "estimatedmetadata" tag has no effect to detecting the bounding box of the layer, when the layer data comes from a SQL Server view. The most time consumption during each start up of QGIS is triggered by following SQL query:

```
select min([SHAPE].MakeValid().STPointN(1).STX), min([SHAPE].MakeValid().STPointN(1).STY), max([SHAPE].MakeValid().STPointN(1).STX),  
max([SHAPE].MakeValid().STPointN(1).STY) from [its].[v_DKM_GST]
```

As I mentioned before this behavior is completely different against version 2.18.19!

#7 - 2018-09-12 04:16 PM - Alexander Zidek

- Status changed from Feedback to Open

Some more additional information about the environment settings:

Following database details parameter are checked:

- Only look in the geometry_columns metadata table
- Use estimated table parameters

But I think, these settings are primarily for listing and browsing within the data source, once you have the layer configured in table of contents, these settings have no impact anymore!?!

PS: I changed the status from feedback to open, if it's not desired, pls let me know!