

Statewide Geography and Spatial Computing in SQL Server

Kevin Karns, DBA



Database
Maintenance And
Performance Tuning

Statewide
Geography and
Spatial Computing
in SQL Server.

Kevin Karns

Level: Beginner

About me

Quick Bio:

Currently Database Administrator – State of NM

SQL Server DBA since 1992 version 3.8

ESRI user since 1985 version 1

Contact Info:

blog: www.kkarnsdba.com

twitter: @kevinkarns



Payson, AZ is where I call home – photo: Spirit Rock trail.

Agenda

Part 1

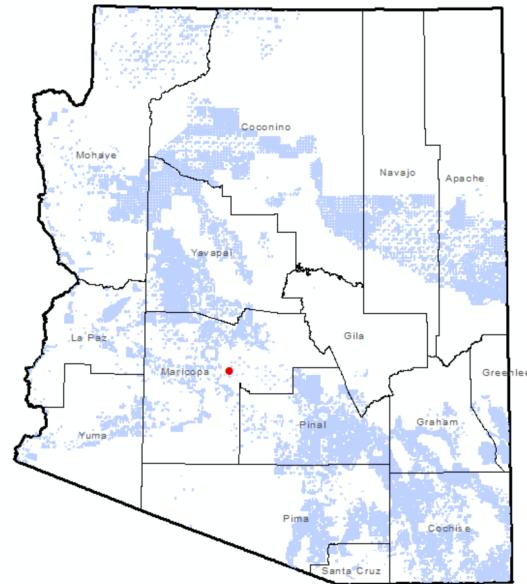
- Background about Arizona land and the PLSS.
- Applications that consume spatial data in SQL Server. (brief)
- ETL challenges with the PLSS and SQL Server spatial data. (very brief)
- GIS security challenges. (brief)
- SQL Server spatial features overview: Spatial tables, Views, Indexes, OGC Method calls

Part 2

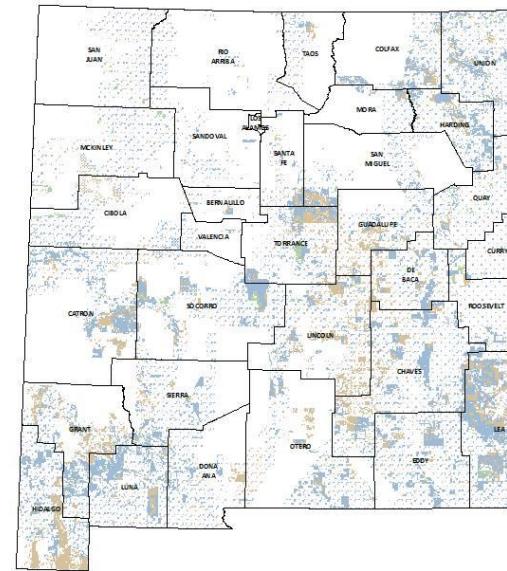
Drilling into a spatial database performance lesson from a DBA's viewpoint:
Look at index maintenance plans, Python, traditional indexes, spatial indexes, ending
with tools available for spatial index diagnostics.

Disclaimer

Arizona State Trust Lands Overview



New Mexico Performance & Tuning Lessons

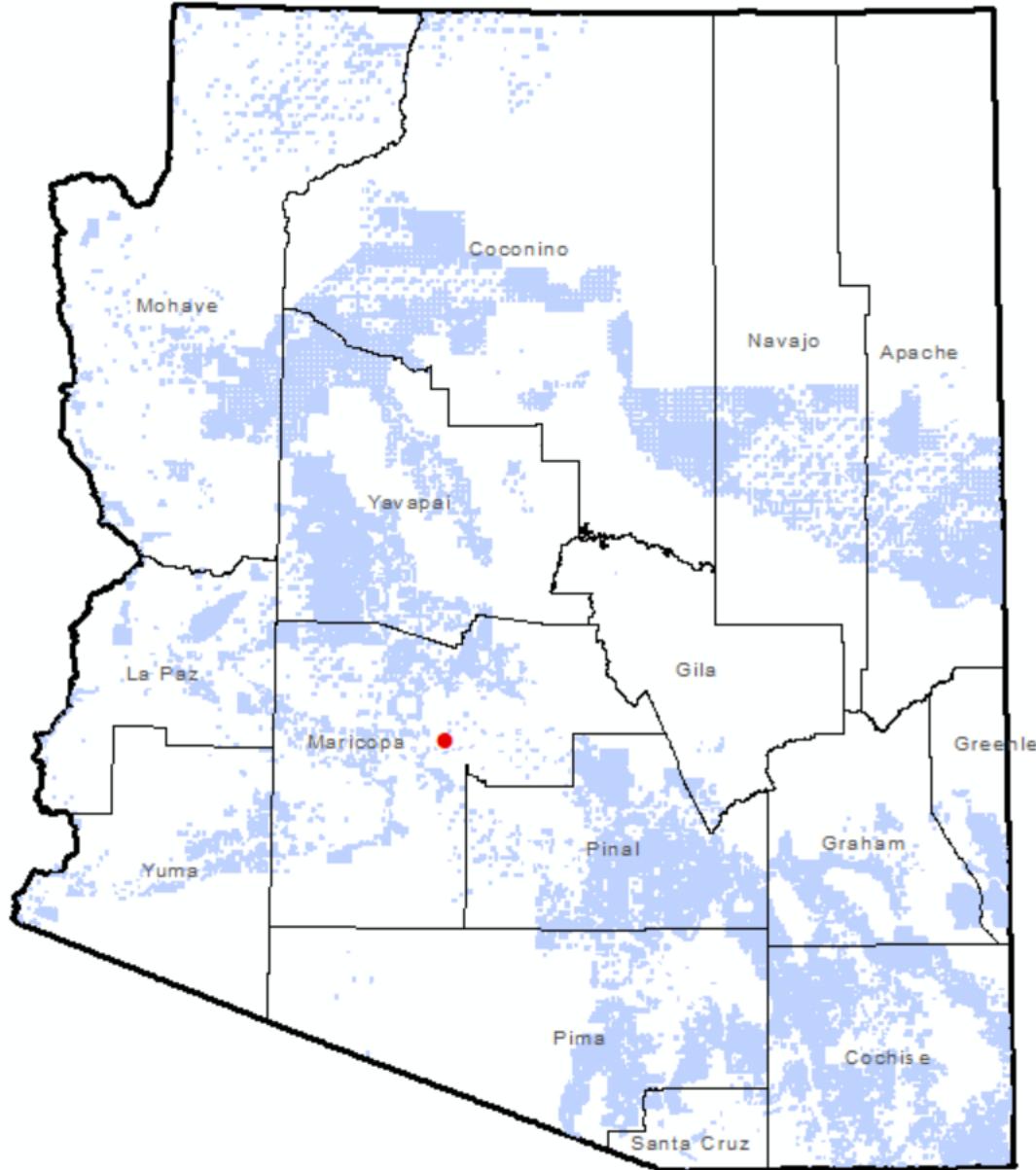


Goal is to teach about tuning the squirrelly Geography/Geometry data type in SQL Server, not speaking on behalf of either Commission.
All examples today are from data downloaded from public clearinghouses and manipulated with an inexpensive ArcGIS license for personal use.

Background on the statewide data sets.

Arizona State Trust Lands

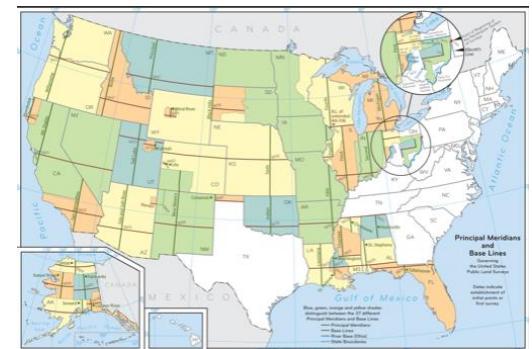
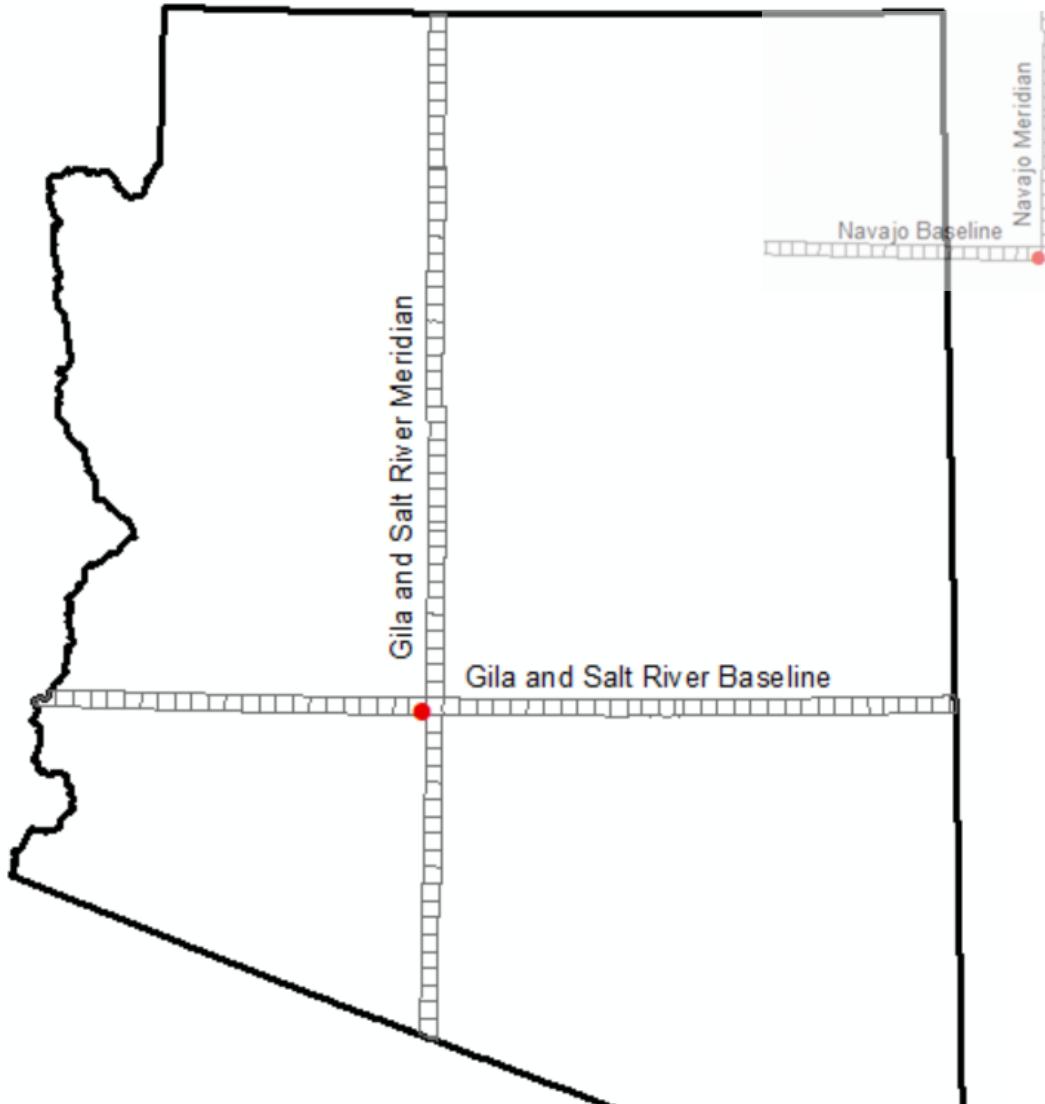
- Approx. 9.3 million acres of surface lands.
- Approx. 9.0 million acres of separated rights.
- Approx. 370,000 state parcels out of 1.3 million discrete PLSS land parcels statewide.
- "Trust" Granted by the Feds to the States to be held in trust to fund the State's public institutions.



Federal Public Land Survey System (PLSS)

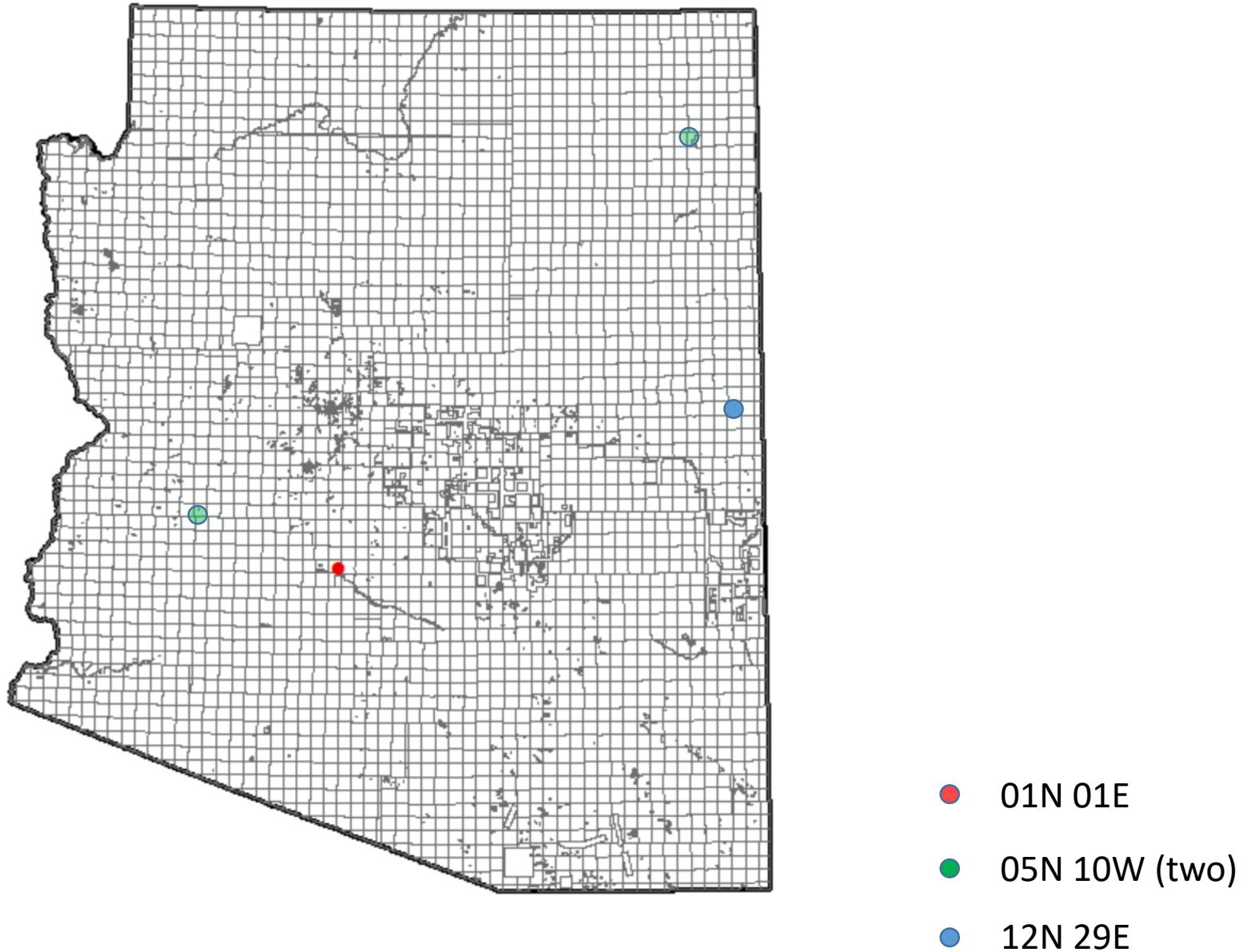
Top level Meridian

- Most land systems of record use the PLSS to uniquely identify lands, instead of latitude longitude or UTM coordinates
- Meridian largest unit of measure in the database.
- Two Meridians/Grids in AZ.



Next Level PLSS Townships

- Over 3500 townships,
approx. 6 miles wide.
- Except in grant lands.



Next Level PLSS Sections ...

36 one-mile sections
in a township.

TL;DR History:

- General Land Ordinance (1785)
 - fund public education with land.
- Organic Act (1850). Sections 16 and 36. Beneficiary: Common schools.
- Ferguson Act (1898). Additional lands and additional beneficiaries.
- Enabling Act (1910). Sections 2 and 32.
- In lieu selections .

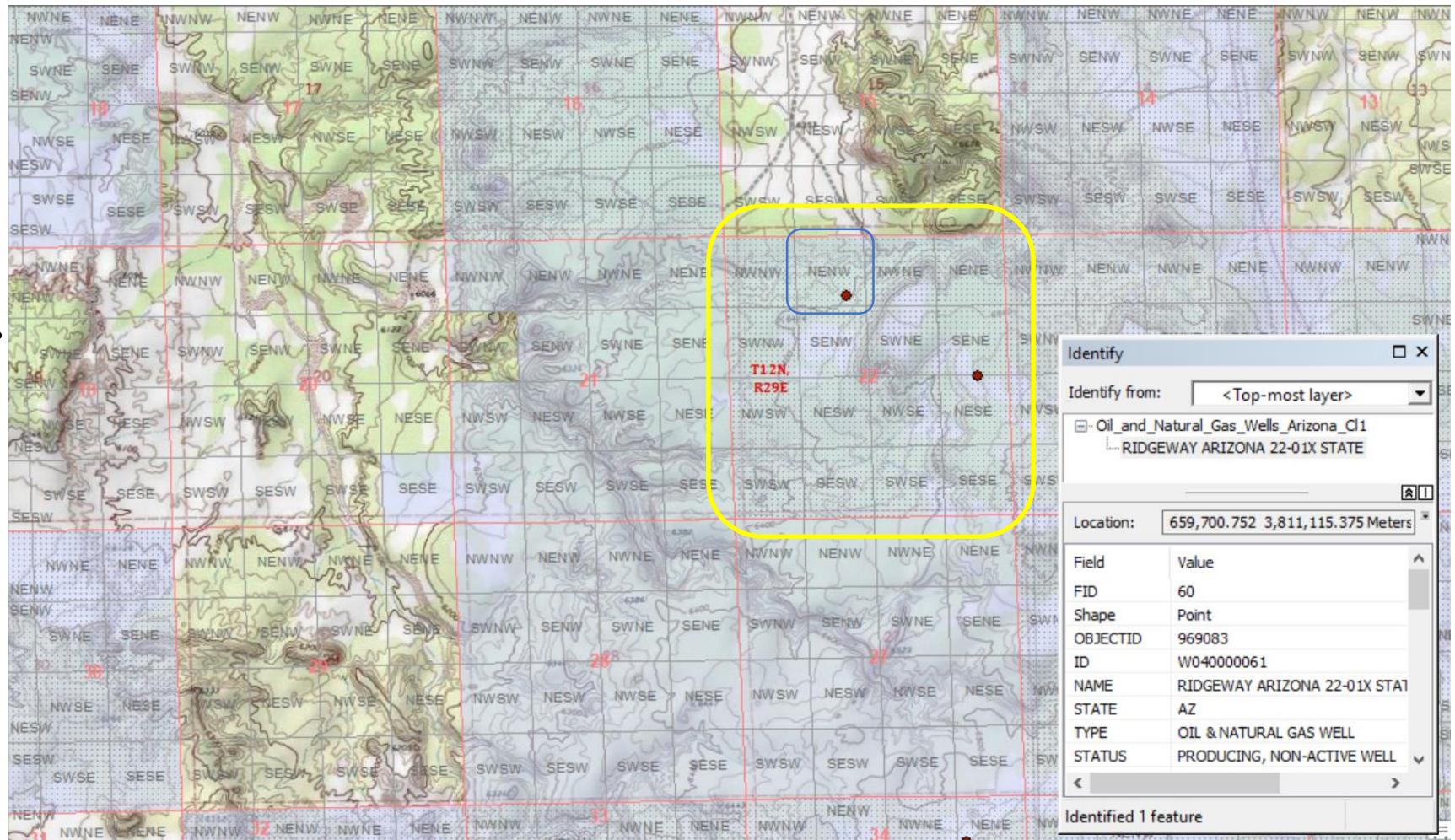


Lowest level – PLSS SubSections

(aka quarter-quarters)

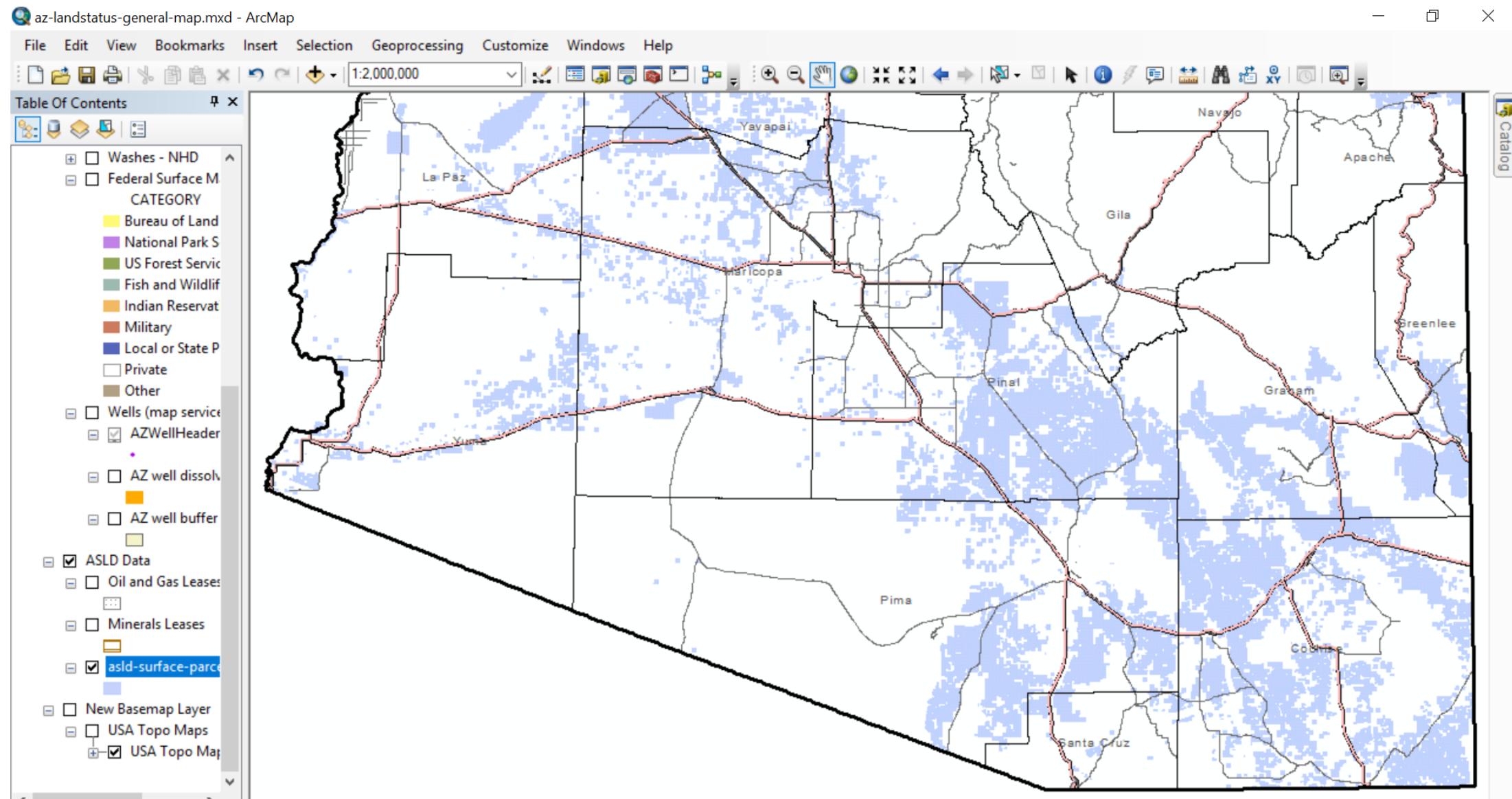
Sixteen 40 acre
subsections in a section.

- Subsection NENW
(in blue)
- 1/16th of Section 22
(in yellow)
- 1.3 million of these in AZ
- Later slide
Spatial Query
on Helium well
in NENW.



Front End Applications on SQL Server Spatial data

(#1) ESRI ArcGIS ArcMap v10.2.2 for Desktop

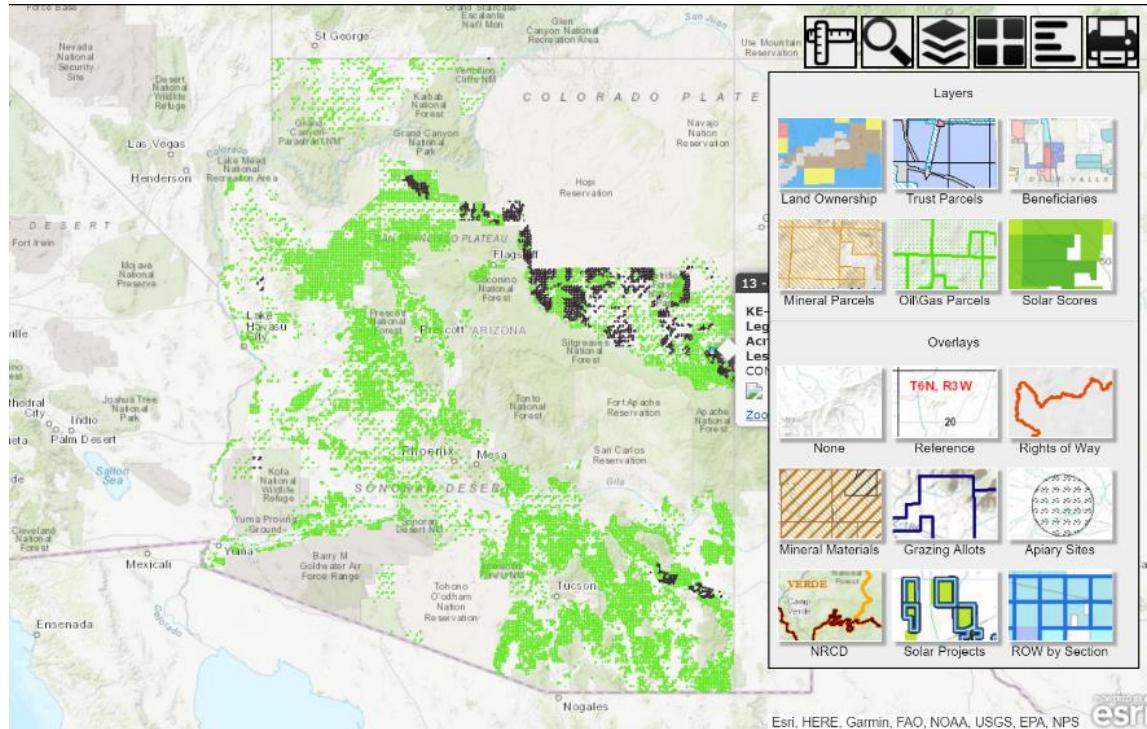


Demo

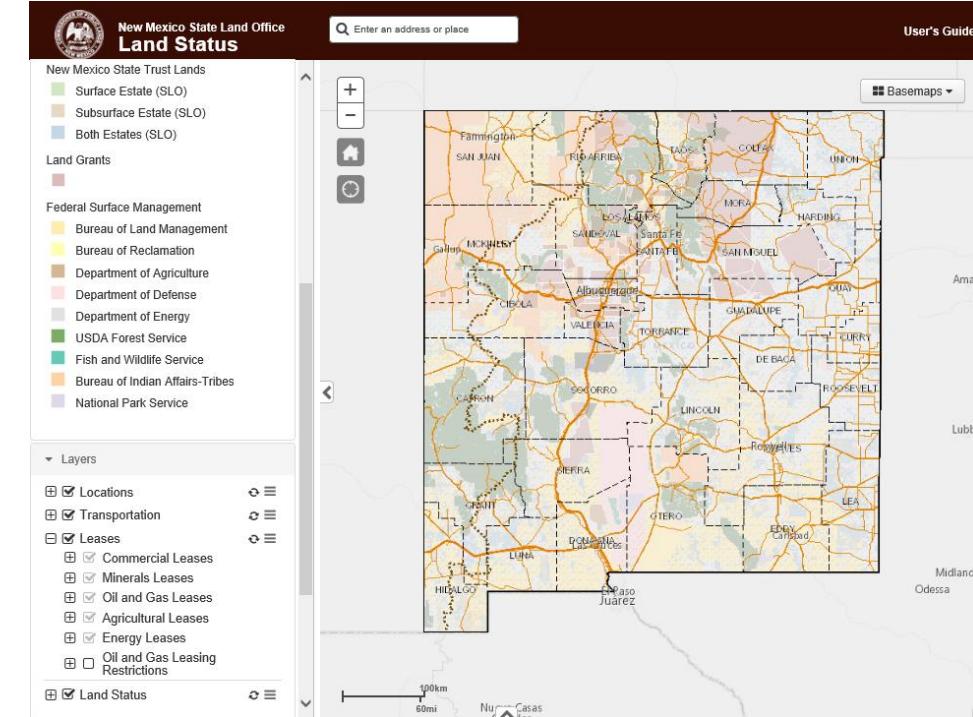
Front End Applications on SQL Server Spatial data

(#2) Interactive webmaps

<http://gis.azland.gov/webapps/parcel/>

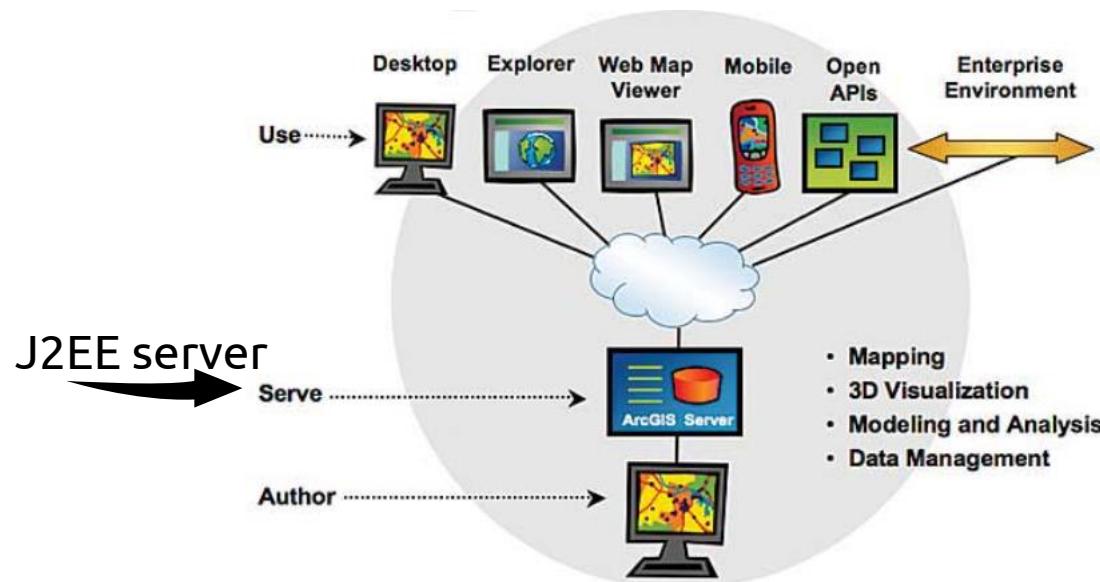


<http://www.nmstatelands.org/interactive-maps.aspx>



(#3) Other apps: things that consume Web Services in ESRI ArcGIS Server v10.2.2

ArcGIS Server Manager



- Web services for internal .Net apps
- Google Earth .KML exports
- Mobile – Getac devices & custom application
- Rights of Ways – ESRI Workflow Manager
- ArcGIS Portal

ETL Challenge: converting from PLSS (Township Range Section Subsection) to Geometry data type

(1970s
Survey point
Numbering
System

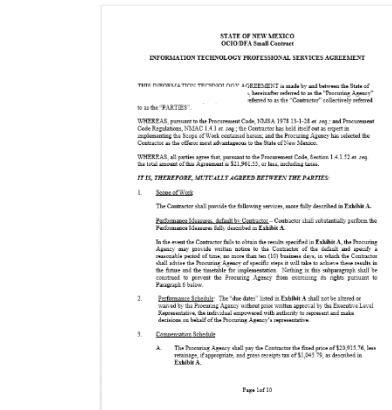
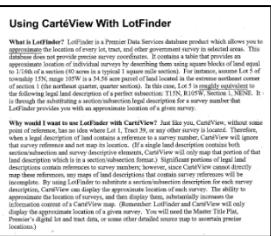
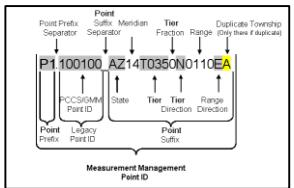
80s

)

(1990s)

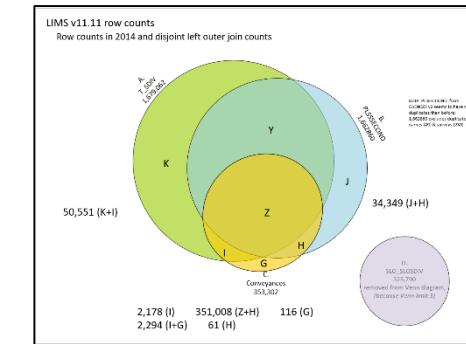
(2000s)

BLM - PLSS → Lotfinder® → CartéView® → LUMAS 1 → LUMAS 2 →



(2013)
LIMS
Cadnsdi+PLSS+DB2
+Conveyances
LUMASGridPrimary

→ Python based
LUMAS



```

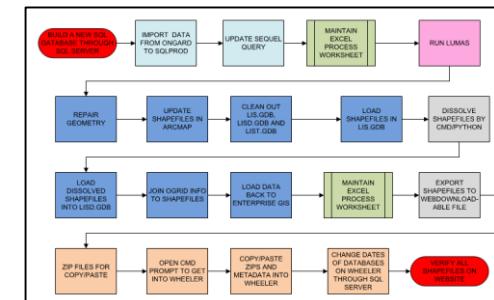
import arcpy
import urllib
import json
import psl_MainTool

# Determine the database to write GIS WIP Status to
def findDBOwnership(notes):
    if 'gis' in notes and 'gis' not in notes and "gis" not in notes:
        return Production.gdb
    elif 'gis' in notes:
        return Development.gdb
    else:
        return est.gdb
    else:
        arcpy.AddWarning("Environment could not be determined. LUMAS could not be run.")

returnCode = 1
arcpy.SetParameterAsText(0,returnCode)
import urllib
import json
assignURL = '{0}/jobs/assignUser={1}\' + \
            'jobs/{2}\' + \
            'assignedType={3}\' + \
            'assignedGeo={4}\'if pslMain.gis == ' + \
            '2, \
            groupname)'

arcpy.AddWarning(assignURL)
assignResult = urllib.urlopen(assignURL).read()
assignJson = json.loads(assignResult)
arcpy.AddMessage(assignJson)
import sys
sys.exit()

```



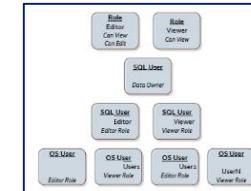
Spatial data security models

Unfortunately most interactions between GIS and DBA staff members.

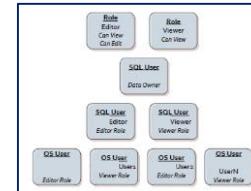
Starting to make some progress
at the time of server migrations
using 300 line cleanup template

```
1 -- new-template-for-geodatabase-migration.sql
2
3 -- identify backup to use
4
5 -- move backup file (map drives if needed)
6
7 -- db#1 SLOData
8 -- use RESTORE FILELISTONLY to verify that files are in the right place and are accessible for next steps
9
10 -- restore backup
11
12 -- db#2 Basedata
13 -- use RESTORE FILELISTONLY to verify that files are in the right place and are accessible for next steps
14
15 -- restore backup
16
17 -- enable query_store
18
19 -- put baseline to vac
20 -- point DLL baseline saved
21
22 -- resolve orphaned accounts
23
24 -- First review ads groups
25 -- refer to runas notes
26
27 net group /domain "newmexicogeouser"
28 net group /domain "gis Admin Group"
29 net group /domain "gis User Group"
30
31
32 -- review security audit log from source server
33
34 -- enumerate existing permissions on the database objects (on prior databases and on new databases) with one of two grants-script UNKNOWN.sql
35 -- First group by roles, sort it by role name, then user
36 -- Second group is grants, database grants are at the top - sort those by grant/applySQL, object grants are next sort those by granted object
37
38 -- since this is production, review enumerated permissions (on new databases) and watch out for potentially elevated dev permissions that came over from (source server)
39
40 -- remove incorrectly applied grants to individual users, and duplicate redundant grants
41
42 -- check for orphaned users (in each database) ... and follow the 9 different use cases to accept or resolve -- this will be in the section below from users.sql
43
44
45 -- orphaned-users.sql
46 select
47   case when [user_name] like '%_orphaned%' then 'Orphaned'
48     else 'Normal' end as [user_type]
49   ,isnull([user_name], 'Orphaned') as [Login_name]
50   ,isnull([user_id], 0) as [User_ID]
51   ,isnull([object_id], 0) as [Object_ID]
52
53 from
54 sys.database_principals dp
55 left join sys.server_principals sp on (dp.sid = sp.sid)
56 where
57   dp.type in ('S','U','G')
58   and dp.principal_id > 4
59   order by [user_name]
60
61
62 -- annotate each of the results with one of these below and fix in a separate users.sql script
63
64 -- SQL_USER Orphaned| NULL
65 -- SQL_USER Orphaned| NULL | ) reset SID
66 -- SQL_USER Orphaned| SQL_LOGIN NULL | ) change SID should not be reset, reset the SID anyway
67 -- XENON_USER Orphaned| NULL | ) remove it
68 -- WINDMILL_USER Orphaned| NULL | ) add to logins
69
70
```

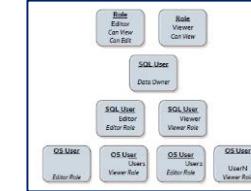
contractor #1
security model #1



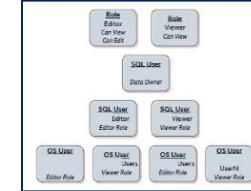
internal staff
security model #2



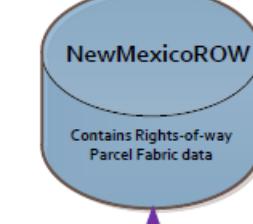
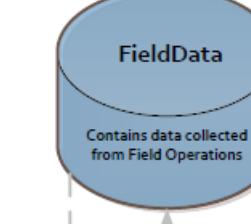
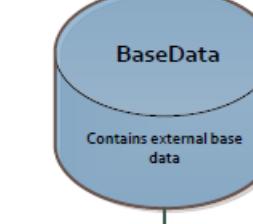
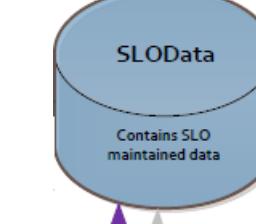
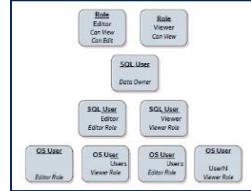
contractor #2
security model #3



contractor #3
security model #4



ESRI third party S/W
security model #5



Spatial table

gis.SLO_OGLEASE	
	Columns
	OBJECTID (PK, int, not null)
	MERIDIAN (numeric(38,8), null)
	TOWNSHIP (nvarchar(5), null)
	RANGE (nvarchar(5), null)
	SECT (numeric(38,8), null)
	SURVEYTYPE (nvarchar(1), null)
	ALIQUOT (nvarchar(4), null)
	UNIQUEKEY (nvarchar(20), null)
	ONGARD_DTE (nvarchar(10), null)
	PROCESS (nvarchar(12), null)
	DISKEY (nvarchar(100), null)
	OGTOWNSHIP (nvarchar(3), null)
	OGRANGE (nvarchar(3), null)
	QUARTERS (nvarchar(16), null)
	QTRQTRS (nvarchar(12), null)
	OGLOTRACT (nvarchar(3), null)
	LSE_PREFIX (nvarchar(2), null)
	LSE_NUMBER (numeric(38,8), null)
	LSE_SUFFIX (numeric(38,8), null)
	STATUS (nvarchar(7), null)
	LSDV_ACRG (numeric(38,8), null)
	VEREFF_DTE (datetime2(7), null)
	VERTRM_DTE (datetime2(7), null)
	OGRID_CDE (numeric(38,8), null)
	OGRID_NAM (nvarchar(45), null)
	OGRID_ADR_NAM (nvarchar(30), null)
	MAIL_STOP (nvarchar(20), null)
	LINE1_ADR (nvarchar(30), null)
	LINE2_ADR (nvarchar(30), null)
	LINE3_ADR (nvarchar(30), null)
	CITY_NAM (nvarchar(30), null)
	ST_NAM (nvarchar(2), null)
	ZIP_CDE (nvarchar(9), null)
	CTRY_NAM (nvarchar(15), null)
	PHONE_NUM (numeric(38,8), null)
	FAX_NUM (numeric(38,8), null)
	PROCDATE (datetime2(7), null)
	SOURCEGRID (nvarchar(16), null)
	GlobalID (uniqueidentifier, not null)
	CreatedBy (nvarchar(255), null)
	CreatedDate (datetime2(7), null)
	ModifiedBy (nvarchar(255), null)
	ModifiedDate (datetime2(7), null)
	GRIDNAME (nvarchar(50), null)
	SHAPE (geometry, null)
	Keys
	Constraints

-- MS SQL Server Spatial

```
SELECT  
    SHAPE,  
    SHAPE.ToString() AS WKT
```

SHAPE	WKT
0x21690000010C1E1DAF4714EB2341DC1E3046FF714B41	POINT (652682.1400078868 3597310.5483435225)

-- Oracle Spatial

```
CREATE TABLE OracleSpatialTable (  
    OFFSET_FROM    NUMERIC(12,3),  
    OFFSET_TO      NUMERIC(12,3),  
    GEOM          PUBLIC.SDO_Geometry  
)
```

OFFSET_FROM OFFSET_TO GEOM

54.822	86.977	{4002,null,null,{1,2,1},{555828.9771,3610888.9205,0,54.822,555842.5771,3610916.543,0,
--------	--------	---

-- Oracle Spatial can transform projections on-the-fly in SQL

```
sdo_cs.transform(SDO_Geometry(4001, 26913, SDO_Point_Type(t.x, t.y, t.z),null,null),8307) AS GEOM_WGS84
```

-- old ESRI Spatial Binary (pre SQL Server 2008)

```
SELECT  
    SDE.GIS.SLO_OGPUNSTUNITD.UNIQUEKEY,  
    SDE.GIS.SLO_OGPUNSTUNITD.SHAPE,  
    SHAPE.points F_points  
FROM  
    SDE.GIS.SLO_OGPUNSTUNITD  
LEFT JOIN  
    SDE.GIS.f557 SHAPE  
ON  
    SHAPE.fid = SDE.GIS.SLO_OGPUNSTUNITD.SHAPE
```

UNIQUEKEY	SHAPE	F_points
1001156	2	0xA20C0000000000A9E1EDF209B6D3C8E106A562ED8B309B6

Spatial View

"ESRI specific"

gis.s1093

gis.slo_ogapi_top_gas

Columns

- OBJECTID (int, null)
- MERIDIAN (numeric(10,0), null)
- TOWNSHIP (nvarchar(5), null)
- RANGE (nvarchar(5), null)
- SECT (numeric(10,0), null)
- SURVEYTYP (nvarchar(1), null)
- ALIQUOT (nvarchar(19), null)
- UNIQUEKEY (nvarchar(16), null)
- ONGARD_DTE (nvarchar(10), null)
- PROCESS (nvarchar(12), null)
- EFF_DTE (datetime, null)
- WELL_TYP_CDE (nvarchar(1), null)
- WELL_TYPE (nvarchar(40), null)
- OGRID_CDE (numeric(10,0), null)
- OGRID_NAM (nvarchar(45), null)
- CNTY_CDE (numeric(10,0), null)
- OGTOWNSHIP (nvarchar(3), null)
- OGRANGE (nvarchar(3), null)
- OGSECTION (numeric(10,0), null)
- OGLOTTRACT (nvarchar(3), null)
- FTG_NS_NUM (numeric(10,0), null)
- FTG_EW_NUM (numeric(10,0), null)
- NS_CDE (nvarchar(1), null)
- EW_CDE (nvarchar(1), null)
- LEASE_TYP_CDE (nvarchar(2), null)
- WELL_STAT_CDE (nvarchar(1), null)
- ELEV_DF_NUM (numeric(10,0), null)
- ELEV_GL_NUM (numeric(10,0), null)
- ELEV_KB_NUM (numeric(10,0), null)
- SPUD_DTE (datetime, null)
- DPTH_PB_NUM (numeric(10,0), null)
- TD_DTE (datetime, null)
- APD_WRK_TYP (nvarchar(1), null)
- PROD_PROP_IDN (numeric(10,0), null)
- PROD_PROP_NAM (nvarchar(40), null)
- WELL_NBR_IDN (nvarchar(4), null)
- DPTH_TVD_NUM (numeric(10,0), null)
- DPTH_MVD_NUM (numeric(10,0), null)
- PLUG_DTE (datetime, null)
- REC_TERMN_DTE (datetime, null)
- PROCDATE (datetime, null)
- SOURCEGRID (nvarchar(8), null)
- SHAPE (int, null)
- PROD (decimal(18,0), null)
- PRODN_YRMTH (nvarchar(6), not null)

Triggers

Indexes

Statistics

gis.slo_ogapi_top_gas

Spatial processing with ESRI toolbox often uses inefficient cursors ...
but in SQL we can create view on a table with a geometry/geography column joined to additional data ... much faster.

Then "register" the view in ESRI with one of the two approaches:

- Using ESRI ArcSDE administration commands

```
sdetable -o create_view -s servername -D dbname -u username -p passwd -i esri_sde  
-T SLO_OGAPI_TOP_GAS -t "SDE.GIS.SLO_OGAPI" -c "*"
```

- Using ESRI ArcPy Python commands

```
""" Create view containing SQL geometry type column """

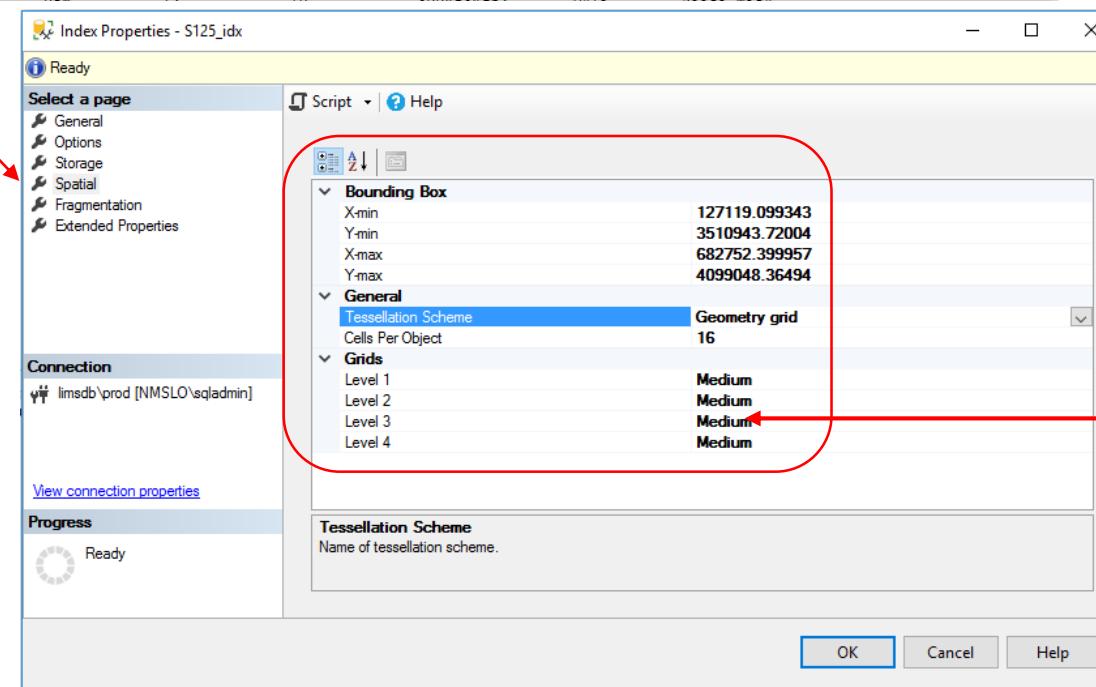
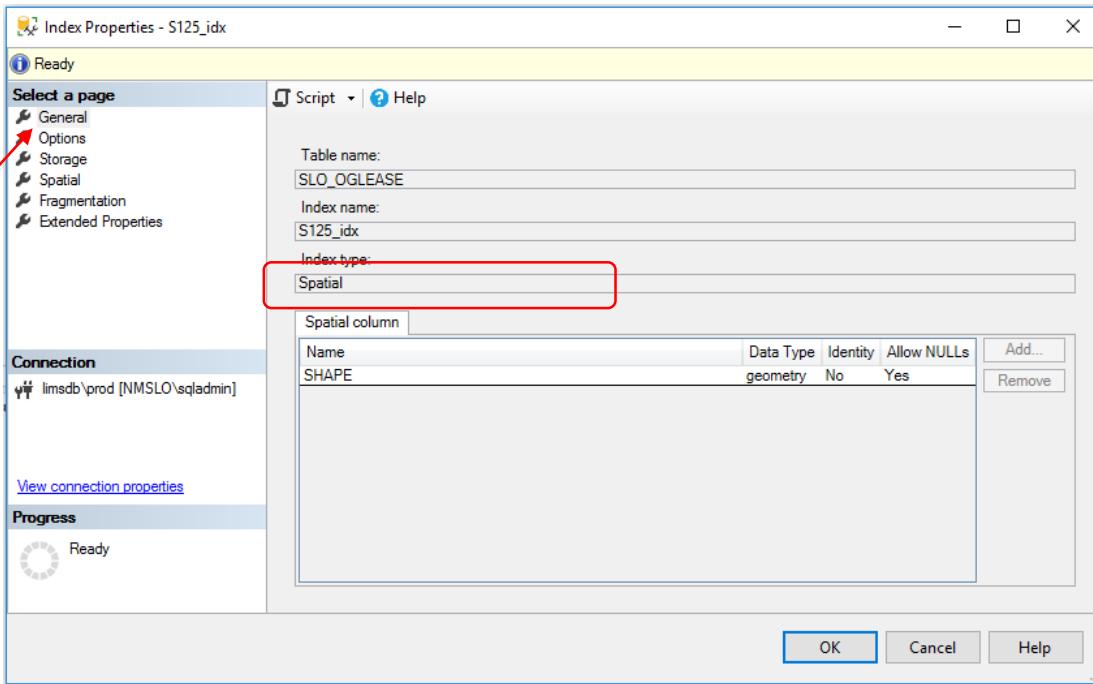
import arcpy
import sys


def CreateSpatialView(input_database, view_name, view_definition):
    """ Create view containing SQL geometry type column """
    try:
        arcpy.CreateDatabaseView_management(input_database, view_name, view_definition)
    except arcpy.ExecuteError:
        print(arcpy.GetMessages(2))

if __name__ == "__main__":
    arguments = sys.argv[1:]
    CreateSpatialView(*arguments)
```

Spatial Index

- gis.SLO_OGLEASE
 - Columns
 - Keys
 - Constraints
 - Triggers
 - Indexes
 - I1479UNIQUEKEY (Non-Unique, Non-Clustered)
 - R166_pk (Clustered)
 - S125_idx (Spatial)**
 - UUID_166 (Non-Unique, Non-Clustered)
 - UUID_OID_166 (Unique, Non-Clustered)
 - Statistics
 - _WA_Sys_00000007_6522C3C0
 - I1479UNIQUEKEY
 - R166_pk
 - UUID_166
 - UUID_OID_166



Histogram function later on

SQL Server has Spatial Methods - Follows ISO 19125 Guidelines

OGC Methods on Geometry Instances

03/13/2017 • 2 minutes to read • Contributors 

APPLIES TO:  SQL Server (starting with 2012)  Azure SQL Database  Azure SQL Data Warehouse  Parallel Data Warehouse

SQL Server supports the Open Geospatial Consortium (OGC) methods on geometry instances.

For more information on OGC specifications, see the following:

- [OGC Specifications, Simple Feature Access Part 1 - Common Architecture](#)
- [OGC Specifications, Simple Feature Access Part 2 – SQL Options](#)

In This Section

- [STArea](#)
- [STAsBinary](#)
- [STAsText](#)
- [STBoundary](#)
- [STBuffer](#)
- [STCentroid](#)
- [STContains](#)
- [STConvexHull](#)
- [STCrosses](#)
- [STCurveN \(geometry Data Type\)](#)
- [STCurveToLine \(geometry Data Type\)](#)
- [STDifference](#)
- [STDimension](#)
- [STDisjoint](#)
- [STDistance](#)
- [STEndpoint](#)
- [STEnvelope](#)
- [STEquals](#)
- [STExteriorRing](#)
- [STGeometryN](#)
- [STGeometryType](#)
- [STInteriorRingN](#)
- [STIntersection](#)
- [STIntersects](#)
- [STIsClosed](#)
- [STIsEmpty](#)
- [STIsRing](#)
- [STIsValid](#)
- [STLength](#)
- [STNumCurves \(geometry Data Type\)](#)
- [STNumGeometries](#)
- [STNumInteriorRing](#)
- [STNumPoints](#)
- [STOverlaps](#)
- [STPointN](#)
- [STPointOnSurface](#)
- [STRelate](#)
- [STSrid](#)
- [STStartPoint](#)
- [STSymDifference](#)

ArcGIS Help

Home

Get Started

Map

Analyze

Manage Data

Tools

[More...](#)

Manage Data > Geodatabases > Archiving data

- An overview of the geodatabase
 - Designing a geodatabase

The archive process

ArcMap 10.3 | Other versions

Queries on transactional versions are still on the base and delta tables:

Base table

Adds table

ObjectID
Other
Columns

Deletes table

Deleted at Deletes_ID Row_ID		

Base, adds, and deletes tables

How to get the registration_id

```
SELECT
    registration_id,
    database_name,
    table_name
FROM [SLOData].[dbo].[SDE_table_registry]
WHERE
    table_name = 'SLO_OGLELEASE'
```

```
registration_id database_name table_name
-----
50           SLODATA      SLO_OGLEASE

(1 row affected)
```

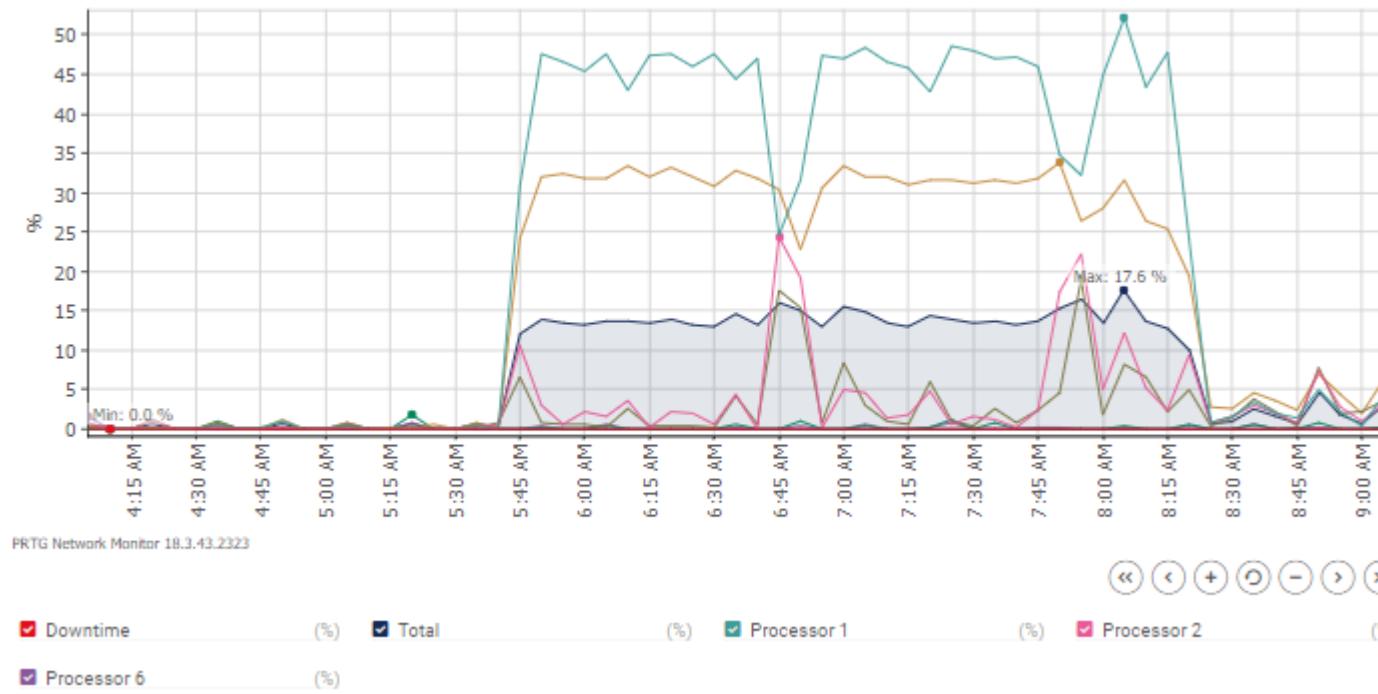
Q&A from Part 1

.... any more questions on etl from system of record into a feature class in a spatial database? ...

Part 2

Drilling into a DBA's production performance problem

OMG! What is all that CPU?



Almost completed migration to Query Store ... Still using Old 2012 performance tools - Qure

Analysis Information

Total Number of Events: 56297
Location: C:\files\qure\2018-01-26-a.qwan

Workload Traces

Trace	Time Range	Event Count
X:\files\apps\itb-lims\2018-01-26-e	1/26/2018 6:15 AM - 1/26/2018 6:20 AM	56297

Resource Consumption Statistics

Overall Resource Consumption

Resource	Total	Average	Min	Max
Duration	3.12 min	3.33 ms	22 µs	48.01 sec
CPU	42.56 sec	1 ms	0 µs	24.23 sec
Reads	21.7M	385.4	0	20.38M
Writes	97.98K	1.74	0	30.52K
Row Count	59.18K	1.05	0	18.24K

Top Consuming Batches

Resource	Top Consumer	Event Count	Total	Average	More Information
Duration	EXEC sp_execute (3)	54322	2.69 min	2.97 ms	Show Batches by Duration
CPU	select OBJECTID, MERIDIAN	1	24.23 sec	24.23 sec	Show Batches by CPU
Reads	select OBJECTID, MERIDIAN	1	20.38M	20.38M	Show Batches by Reads
Writes	EXEC sp_execute (3)	54322	97.45K	1.79	Show Batches by Writes
Row Count	EXEC sp_execute (3)	54322	54.32K	1	Show Batches by Row Count

Top Consuming Databases

Resource	Top Consumer	Event Count	Total	Average	More Information
Duration	SLOData	55938	3.12 min	3.34 ms	Show Databases by Duration
CPU	SLOData	55938	42.12 sec	1 ms	Show Databases by CPU
Reads	SLOData	55938	21.49M	384.22	Show Databases by Reads
Writes	SLOData	55938	97.98K	1.75	Show Databases by Writes
Row Count	SLOData	55938	58.1K	1.04	Show Databases by Row Count

Top Consuming Hosts

Resource	Top Consumer	Event Count	Total	Average	More Information
Duration	SLOGISDATA	55938	3.12 min	3.34 ms	Show Hosts by Duration
CPU	SLOGISDATA	55938	42.12 sec	1 ms	Show Hosts by CPU
Reads	SLOGISDATA	55938	21.49M	384.22	Show Hosts by Reads
Writes	SLOGISDATA	55938	97.98K	1.75	Show Hosts by Writes
Row Count	SLOGISDATA	55938	58.1K	1.04	Show Hosts by Row Count

Workload View

Some kind of cursor foo on a spatial table

```

SELECT *
,SHAPE
,SHAPE.STArea()
,SHAPE.STLength()
FROM (
  SELECT /* ArcSDE_NORMAL_FILTER */
    b./*
  FROM SLODATA.GIS.slo_oglease b
  LEFT JOIN (
    SELECT SDE_DELETES_ROW_ID
    ,SDE_STATE_ID
  FROM SLODATA.GIS.d166
  WHERE SDE_STATE_ID = 0
    AND DELETED_AT IN (
      SELECT 1.lineage_id
      FROM SLODATA.dbo.SDE_state_lineages l
      WHERE l.lineage_name = @P1
        AND l.lineage_id <= @P2
    )
  ) d ON b.OBJECTID = d.SDE_DELETES_ROW_ID
  WHERE d.SDE_STATE_ID IS NULL
UNION ALL

SELECT a./*
FROM SLODATA.GIS.a166 a
LEFT JOIN (
  SELECT SDE_DELETES_ROW_ID
  ,SDE_STATE_ID
  FROM SLODATA.GIS.d166
  WHERE SDE_STATE_ID > 0
    AND DELETED_AT IN (
      SELECT 1.lineage_id
      FROM SLODATA.dbo.SDE_state_lineages l
      WHERE l.lineage_name = @P3
        AND l.lineage_id <= @P4
    )
  ) d ON (a.OBJECTID = d.SDE_DELETES_ROW_ID)
  AND (a.SDE_STATE_ID = d.SDE_STATE_ID)
WHERE a.SDE_STATE_ID IN (
  SELECT 1.lineage_id
  FROM SLODATA.dbo.SDE_state_lineages l
  WHERE l.lineage_name = @P5
    AND l.lineage_id <= @P6
  )
  AND d.SDE_STATE_ID IS NULL
) V_166
  
```

Found thousands of these
... entity framework cursor?
No, joining into add&delete
tables repeatedly.

Colleague found the task and the *.bat file and the Python script call

The screenshot displays three windows illustrating a scheduled task and its associated files:

- Task Scheduler Window:** Shows a task named "Oil-Gas Lease Generation" set to run "Ready At 5:35 AM every day". The "Actions" tab is selected, showing an "Action" row where "Start a program" is set to "E:\OGLease_Generate\OGLease.bat".
- File Explorer Window:** Shows a folder structure. The "OGLease" folder, which contains a Python file ("OGLease.py") and a batch file ("OGLease.bat"), is highlighted with a red circle.
- Notepad Window:** Displays the contents of the "OGLease.py" file. The code performs database operations, joins tables, and appends lease records. It includes a note about updating the Python executable path and a command-line call to "pythonw.exe" pointing to the "OGLease_Generate" directory.

```
## Script copies the LUMAS_OGLEASE_ALL table from the LIMS database and the subsurface ownership GIS data
## from GEOSTAGING into the temporary workspace OGLease_Workspace.gdb, calculates an ID value in the LIMS
## table, joins the attribute table to the subsurface GIS data, and exports only those records successfully
## joined. It then appends the problem leases located in a separate layer in the workspace. It then deletes
## the records in the oil/gas lease feature class within each geodatabase and appends with the new lease
## layer created within the workspace geodatabase. All workspace data are deleted prior to the next day's
## execution, except for the problem leases.
##
## NOTE: The pythonw.exe pathname will need to be updated as new versions of
##       ArcGIS are installed.

C:\Python27\ArcGIS10.2\pythonw.exe E:\OGLease_Generate\OGLease.py
```

ESRI Model Builder

"it's like Microsoft SSIS for GIS professionals"
Result is a Python Script

```
# -- coding: utf-8 --
# -----
# OGLease.py
# Created on: 2017-09-22 09:43:15.00000
# (generated by ArcGIS/ModelBuilder)
#
# Description: Creates an Oil/Gas Lease Layer from the LIMS table view. The program brings over the table view and the GIS Subsurf
# into a temporary work space (file geodatabase), creates a field called LLDID, joins the subsurface data with the tab
# keeping only the matching records, copies the new layer to a feature class in the work space, and dissolves the new
# the Lease ID. Once the subdivision-level and dissolved layers are created, the layers in GEOSTAGING, LIMSDB, and AT
# through Truncate/Append tools. The data in the work space is then deleted in preparation for the next day's run.
#
# This creates new layers each day after the Oil/Gas lease table view in LIMS is created from ONGARD.
#
# NOTE: Remove Code beginning at line 70 when land description for leases have been corrected.
# Leases are located in 31N 6W (Tract 40) and 26S 23E (TX border).
#
# -----
#
# Import arcpy module
import arcpy

# Local variables:
# Connect to SDE Geodatabases and Public Web Server File Geodatabase
GEOSTAGING = "Database Connections\\"
LIMSDB = "Database Connectio"
ATALAYA = ""

# Location of Subsurface Ownership Data and Oil/Gas Table in LIMS (from ONGARD)
slo_SubsurfaceSTLStatus = GEOSTAGING+"SLOData.GIS.slo_SubsurfaceSTLStatus"
LIMS_PROD_dbo_LUMAS_OGLEASE_ALL = "Database Connecti
OGLEASE_ALL"

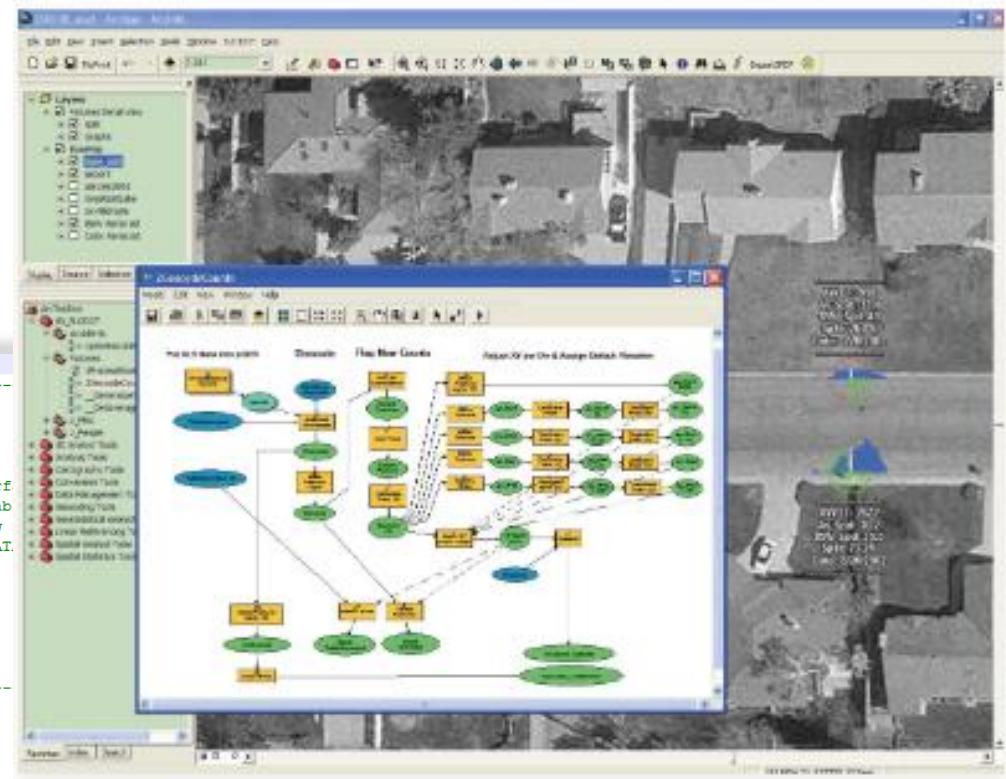
# Workspace and Temporary Data Sets (on GEOGROD)
OGLEase_Workspace =
..gdb"
SubSurface_Ownership = OGLEase_Workspace+"\\"SubSurface_Ownership"
Subsurface_Ownership_Layer = "Subsurface_Ownership_Layer"

# Copy Subsurface Ownership GIS Data to File Geodatabase With Only the LLDID Attribute
arcpy.FeatureClassToFeatureClass_conversion(slo_SubsurfaceSTLStatus, OGLEase_Workspace, "SubSurface_Ownership", "", "LLDID \"LLDID\" true false false 64 Text 0 0 ,First,#,"+slo_SubsurfaceSTLStatus+",LLDID,-1,-1;I

# Copy LIMS Table View into File Geodatabase (work space)
arcpy.TableToTable_conversion(LIMS_PROD_dbo_LUMAS_OGLEASE_ALL, OGLEase_Workspace, "LUMAS_OGLEASE_ALL", "", "MERIDIAN \"MERIDIAN\" true true false 2 Text 0 0 ,First,#,Database Connect
IMS_P

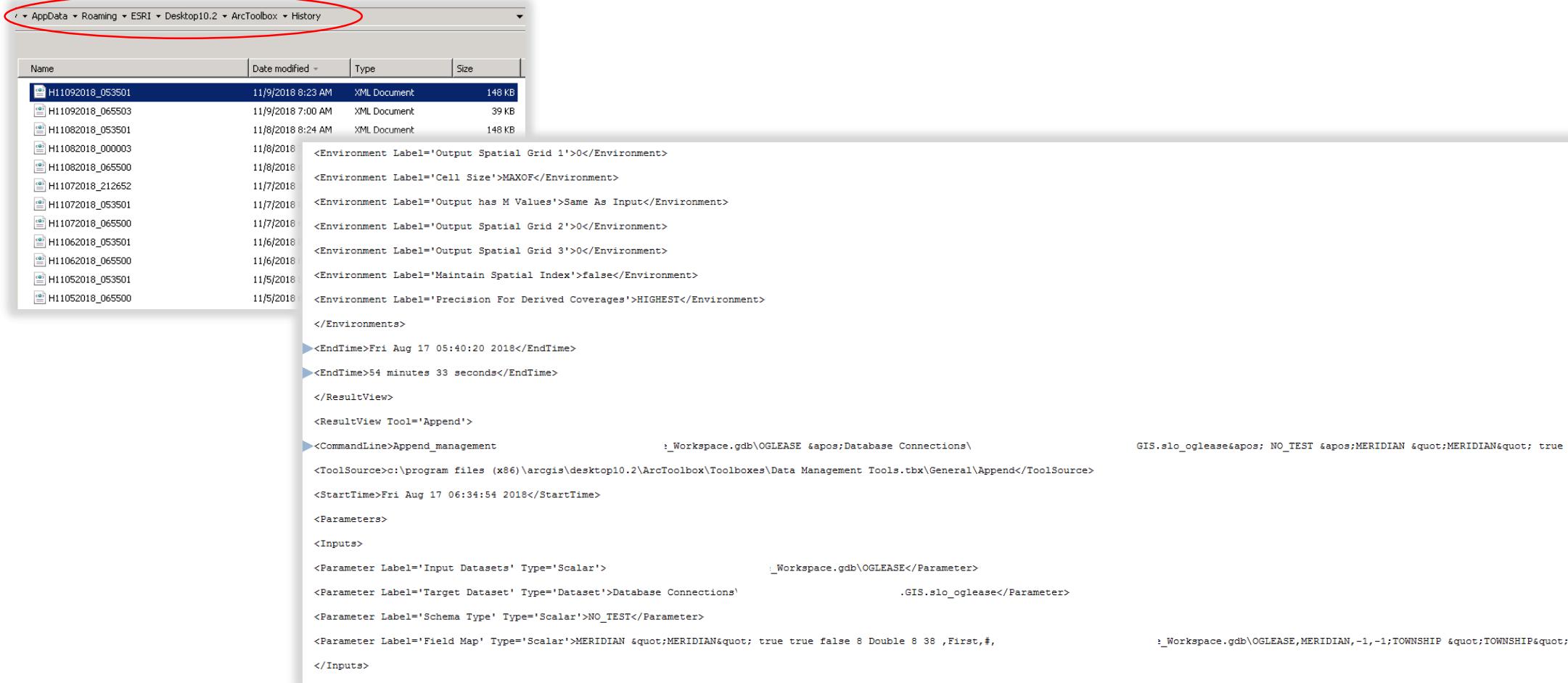
# Add LLDID Field for Joining
arcpy.AddField_management(OGLEase_Workspace+"\\"LUMAS_OGLEASE_ALL", "LLDID", "TEXT", "", "", "64", "", "NULLABLE", "REQUIRED", "")

# Calculate LLDID Field By Concatinating TRSQQ. If Section Number is 9 or less, Add Another "0" Before the Section
arcpy.CalculateField_management(OGLEase_Workspace+"\\"LUMAS_OGLEASE_ALL", "LLDID", "!MERIDIAN!+ !TOWNSHIP!+ !RANGE!+LLIDZero( !SECTION! )+ !SECTION!+ !SURVEYTYPE!+ !ALIQUOT!", "PYTHON_9.3", "def LLIDZero(section):
    # Add GRIDNAME Field and Populate with "Subsurface" STL Status
```



Locating the slow lines of code in the script.

ESRI Model Builder XML Log files - Timestamps



The screenshot shows a Windows File Explorer window with the following path highlighted:

```
/ AppData Roaming ESRI Desktop10.2 ArcToolbox History
```

The window displays a list of XML log files with their names, dates modified, types, and sizes. The files are listed in chronological order from top to bottom. The first few files are:

Name	Date modified	Type	Size
H11092018_053501	11/9/2018 8:23 AM	XML Document	148 KB
H11092018_065503	11/9/2018 7:00 AM	XML Document	39 KB
H11082018_053501	11/8/2018 8:24 AM	XML Document	148 KB

Below the table, the XML content of the first file is partially visible:

```
<Environment Label='Output Spatial Grid 1'>0</Environment>
<Environment Label='Cell Size'>MAXOF</Environment>
<Environment Label='Output has M Values'>Same As Input</Environment>
<Environment Label='Output Spatial Grid 2'>0</Environment>
<Environment Label='Output Spatial Grid 3'>0</Environment>
<Environment Label='Maintain Spatial Index'>false</Environment>
<Environment Label='Precision For Derived Coverages'>HIGHEST</Environment>
</Environments>
▶<EndTime>Fri Aug 17 05:40:20 2018</EndTime>
▶<EndTime>54 minutes 33 seconds</EndTime>
</ResultView>
<ResultView Tool='Append'>
▶<CommandLine>Append_management _Workspace.gdb\OGLEASE &apos;Database Connections' GIS.slo_oglease&apos; NO_TEST &apos;MERIDIAN &quot;MERIDIAN&quot; true t
<ToolSource>c:\program files (x86)\arcgis\desktop10.2\ArcToolbox\Toolboxes\Data Management Tools.tbx\General\Append</ToolSource>
<StartTime>Fri Aug 17 06:34:54 2018</StartTime>
<Parameters>
<Inputs>
<Parameter Label='Input Datasets' Type='Scalar'> _Workspace.gdb\OGLEASE</Parameter>
<Parameter Label='Target Dataset' Type='Dataset'>Database Connections .GIS.slo_oglease</Parameter>
<Parameter Label='Schema Type' Type='Scalar'>NO_TEST</Parameter>
<Parameter Label='Field Map' Type='Scalar'>MERIDIAN &quot;MERIDIAN&quot; true true false 8 Double 8 38 ,First,#, _Workspace.gdb\OGLEASE,MERIDIAN,-1,-1,TOWNSHIP &quot;TOWNSHIP&quot;
</Inputs>
```

Lines 89 & 90 take over an hour each line

```
68 # Copy Features into Feature Class OGLEASE
69 arcpy.CopyFeatures_management("OGLEASE_Layer", OGLease_Workspace+"\\OGLEASE", "", "0", "0", "0")
70
71
72 # ***** REMOVE WHEN LEASES ARE CORRECTED - Bad Land Descriptions *****
73 # Append missing lease geometry to oil/gas leases. These are produced separately as a static data set.
74 arcpy.Append_management(OGLease_Workspace+'\\MissingLeases', OGLease_Workspace+"\\OGLEASE", "NO_TEST","","")
75
76
77 # Dissolve on LEASE ID and Other Lessee Information
78 arcpy.Dissolve_management(OGLease_Workspace+"\\OGLEASE", OGLease_Workspace+"\\OGLEASE_D", "UNIQUEKEY;LSE_PREFIX;LSE_NUMBER;LSE_SUFFIX;STATUS;VEREFF_DTE;VERTRM_DTE")
79
80 # Alter the Name of the Summed Acreage Field
81 arcpy.AlterField_management(OGLease_Workspace+"\\OGLEASE_D", "SUM_LSDV_ACRG", "LSE_ACRG", "Lease Acreage")
82
83
84 #Import Dissolved Lease Layer           SLOData.gdb) (truncate/append) from the OGLEASE_Workspace
85 arcpy.DeleteFeatures_management(      +"slo_ogleased")
86 arcpy.Append_management(OGLease_Workspace+"\\OGLEASE_I"           _ogleased", "NO_TEST", "", "")
87
88 #Import Subdivision-level and Dissolved Lease Layers          ncate/append) from the OGLEASE_Workspace
89 arcpy.DeleteFeatures_management(      "SLOData.GIS.slo_oglease")
90 arcpy.Append_management(OGLease_Workspace+"\\C"           "SLOData.GIS.slo_oglease", "NO_TEST", "", "")
91 arcpy.DeleteFeatures_management(      "SLOData.GIS.slo_ogleased")
92 arcpy.Append_management(OGLease_Workspace+"\\OGL"           "SLOData.GIS.slo_ogleased", "NO_TEST", "", "")
93
94 #Import Subdivision-level and Dissolved Lease Layers          (truncate/append) from the OGLEASE_Workspace
95 arcpy.DeleteFeatures_management(      :NG+"SLOData.GIS.slo_oglease")
96 arcpy.Append_management(OGLease_Workspace+"\\OGLE"           +"SLOData.GIS.slo_oglease", "NO_TEST", "", "")
97 arcpy.DeleteFeatures_management(      :NG+"SLOData.GIS.slo_ogleased")
98 arcpy.Append_management(OGLease_Workspace+"\\OGLEASE_2"       :"SLOData.GIS.slo_ogleased", "NO_TEST", "", "")
99
100
101 # Delete Temporary Data Out of OGLease_Workspace.gdb in Preparation for the Next Day's Run
102 arcpy.Delete_management(OGLease_Workspace+"\\LUMAS_OGLEASE_ALL", "Table")
103 arcpy.Delete_management(SubSurface_Ownership, "FeatureClass")
104 arcpy.Delete_management(OGLease_Workspace+"\\OGLEASE", "FeatureClass")
105 arcpy.Delete_management(OGLease_Workspace+"\\OGLEASE_D", "FeatureClass")
106
```

OMG! Gotta be the indexes. Double check index maintenance plans on the spatial table.

ID	DatabaseName	SchemaName	ObjectName	ObjectType	IndexName	IndexType	StatisticsName	PartitionNumber	ExtendedInfo	Command	CommandType	StartTime	EndTime	Error
2694841	SLOData	gis	D166	U	NULL	WA_Sys_00000002_7FD6B9FC	NULL	NULL		UPDATE STATISTICS [SLOData].[gis].[D166] [WA_Sys_00000002_7FD6B9FC] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:34:54.253	2018-10-12 20:35:00.067	0
2694840	SLOData	gis	D166	U	d166_idx2	2	d166_idx2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_idx2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:34:47.467	2018-10-12 20:34:50.250	
2694839	SLOData	gis	D166	U	d166_pk	1	d166_pk	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_pk] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:34:38.407	2018-10-12 20:34:45.057	
2694791	SLOData	gis	a166	U	UUID_OID_166_a	2	UUID_OID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_OID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:33:52.613	2018-10-12 20:34:00.570	
2694790	SLOData	gis	a166	U	UUID_166_a	2	UUID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:33:42.097	2018-10-12 20:33:52.600	
2694789	SLOData	gis	a166	U	I1479UNIQUEKEY_a	2	I1479UNIQUEKEY_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [I1479UNIQUEKEY_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:33:31.940	2018-10-12 20:33:41.050	
2694788	SLOData	gis	a166	U	a166_state_ix2	2	a166_state_ix2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [a166_state_ix2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:33:25.140	2018-10-12 20:33:30.080	
2694787	SLOData	gis	a166	U	a166_rowid_ix1	1	a166_rowid_ix1	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [a166_rowid_ix1] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-12 20:32:23.040	2018-10-12 20:33:24.410	
2686284	SLOData	gis	D166	U	NULL	WA_Sys_00000002_7FD6B9FC	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [WA_Sys_00000002_7FD6B9FC] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:31:34.697	2018-10-10 20:31:40.473	0	
2686283	SLOData	gis	D166	U	d166_idx2	2	d166_idx2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_idx2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:31:27.560	2018-10-10 20:31:34.693	
2686282	SLOData	gis	D166	U	d166_pk	1	d166_pk	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_pk] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:31:14.553	2018-10-10 20:31:20.977	
2686270	SLOData	gis	a166	U	UUID_OID_166_a	2	UUID_OID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_OID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:30:57.133	2018-10-10 20:31:05.440	
2686277	SLOData	gis	a166	U	UUID_OID_166_a	2	NULL	1	<ExtendedInfo><PageCount>95106</PageCount><Fragmentation>5.47284</Fragmentation><ExtendedInfo>	ALTER INDEX [OID_OID_166_a] ON [SLOData].[gis].[a166] REORGANIZE WITH (LOB_COMPACTION = ON)	ALTER_INDEX	2018-10-10 20:30:07.460	2018-10-10 20:30:57.130	
2686276	SLOData	gis	a166	U	UUID_166_a	2	UUID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:29:56.100	2018-10-10 20:30:06.397	
2686244	SLOData	gis	a166	U	I1479UNIQUEKEY_a	2	I1479UNIQUEKEY_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [I1479UNIQUEKEY_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:29:46.100	2018-10-10 20:29:45.977	
2686243	SLOData	gis	a166	U	a166_state_ix2	2	a166_state_ix2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [a166_state_ix2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:29:39.193	2018-10-10 20:29:44.140	
2686242	SLOData	gis	a166	U	a166_rowid_ix1	1	a166_rowid_ix1	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [a166_rowid_ix1] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-10 20:28:37.130	2018-10-10 20:29:38.500	
2682083	SLOData	gis	D166	U	NULL	WA_Sys_00000002_7FD6B9FC	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [WA_Sys_00000002_7FD6B9FC] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:29:43.480	2018-10-09 20:29:48.540	0	
2682082	SLOData	gis	D166	U	d166_idx2	2	d166_idx2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_idx2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:29:36.592	2018-10-09 20:29:43.480	
2682081	SLOData	gis	D166	U	d166_pk	1	d166_pk	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_pk] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:29:27.560	2018-10-09 20:29:34.237	
2682077	SLOData	gis	a166	U	UUID_OID_166_a	2	UUID_OID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_OID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:29:05.857	2018-10-09 20:29:18.320	
2682076	SLOData	gis	a166	U	UUID_166_a	2	UUID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:28:54.903	2018-10-09 20:29:05.317	
2682075	SLOData	gis	a166	U	I1479UNIQUEKEY_a	2	I1479UNIQUEKEY_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [I1479UNIQUEKEY_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:28:45.580	2018-10-09 20:28:53.987	
2682074	SLOData	gis	a166	U	I1479UNIQUEKEY_a	2	NULL	1	<ExtendedInfo><PageCount>103670</PageCount><Fragmentation>5.17797</Fragmentation><ExtendedInfo>	ALTER INDEX [I1479UNIQUEKEY_a] ON [SLOData].[gis].[a166] REORGANIZE WITH (LOB_COMPACTION = ON)	ALTER_INDEX	2018-10-09 20:28:05.133	2018-10-09 20:28:45.580	
2682073	SLOData	gis	a166	U	a166_state_ix2	2	a166_state_ix2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [a166_state_ix2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:28:02.240	2018-10-09 20:28:07.412	
2682072	SLOData	gis	a166	U	a166_rowid_ix1	1	a166_rowid_ix1	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [a166_rowid_ix1] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-09 20:26:59.643	2018-10-09 20:28:01.817	
2677941	SLOData	gis	D166	U	NULL	WA_Sys_00000002_7FD6B9FC	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [WA_Sys_00000002_7FD6B9FC] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-08 20:26:40.463	2018-10-08 20:26:46.267	0	
2677940	SLOData	gis	D166	U	d166_idx2	2	d166_idx2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_idx2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-08 20:26:33.767	2018-10-08 20:26:40.460	
2677939	SLOData	gis	D166	U	d166_pk	1	d166_pk	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[D166] [d166_pk] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-08 20:26:25.470	2018-10-08 20:26:31.300	
2677938	SLOData	gis	D166	U	UUID_OID_166_a	2	UUID_OID_166_a	NULL	NULL	ALTER INDEX [d166_pk] ON [SLOData].[gis].[D166] REORGANIZE WITH (LOB_COMPACTION = ON)	ALTER_INDEX	2018-10-08 20:26:03.390	2018-10-08 20:26:25.470	
2677937	SLOData	gis	a166	U	UUID_OID_166_a	2	UUID_OID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_OID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-08 20:25:46.593	2018-10-08 20:26:05.393	
2677936	SLOData	gis	a166	U	UUID_166_a	2	UUID_166_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [UUID_166_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-08 20:25:36.500	2018-10-08 20:26:06.127	
2677932	SLOData	gis	a166	U	I1479UNIQUEKEY_a	2	I1479UNIQUEKEY_a	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [I1479UNIQUEKEY_a] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-08 20:25:25.370	2018-10-08 20:25:35.403	
2677931	SLOData	gis	a166	U	a166_state_ix2	2	a166_state_ix2	NULL	NULL	UPDATE STATISTICS [SLOData].[gis].[a166] [a166_state_ix2] WITH FULLSCAN	UPDATE_STATISTICS	2018-10-08 20:25:18.637	2018-10-08 20:25:24.093	

Base table never reorg or rebuild
Spatial indexes never reorg or rebuild

Run Andy Yun's index analysis scripts (on Github).



Uncovering Duplicate, Redundant, & Missing Indexes - A Sneak Peek

Andy Yun, Senior Solutions Engineer, SentryOne
Moderated By: Mark Broadbent



Andy Yun
Senior Solutions Engineer, SentryOne

- [Blog](https://blogs.sentryone.com/andyyun/)
- [GitHub](https://github.com/SQLBek)
- [@SQLBek](https://twitter.com/SQLBek)
- [Email](mailto:ayun@sentryone.com)
- SQLBek@gmail.com

SQL Server DBA & DB Developer

Working with SQL Server since 2001

Chicago Suburban User Group
Chapter Leader

Speaking since Early 2014



The screenshot shows a GitHub user profile for 'SQLBek'. The URL in the address bar is highlighted with a red oval. The profile page includes a photo of a smiling man, Andy Yun, and navigation links for Overview, Repositories (6), Stars (0), Followers (13), and Following (9). Below the photo, there's a section for 'Popular repositories' featuring 'sp_helpExpandView' and 'TSQL_Tips_Tricks_Demos'. A large red oval highlights the repository 'Dupe_Redund_Missing_Indexes', which is described as 'Demo Scripts for Duplicate Redundant and Missing Index presentation.' It has 3 stars and 2 forks.

Yun's UsageStatsQuery – his “go to” query ... result: it was index updates but only showed traditional indexes.

Database Schema Table Name Index Name user_seeks user_scans user_lookups user_updates last_user_seek last_user_scan last_user_lookup last_user_update type_desc is_primary is_unique is_unique_row_count in_row_data_page_count in_row_used_page_count in_row_reserved_page_count lob_used_page_count lob_reserved_page_count row_overflow_used_page_count

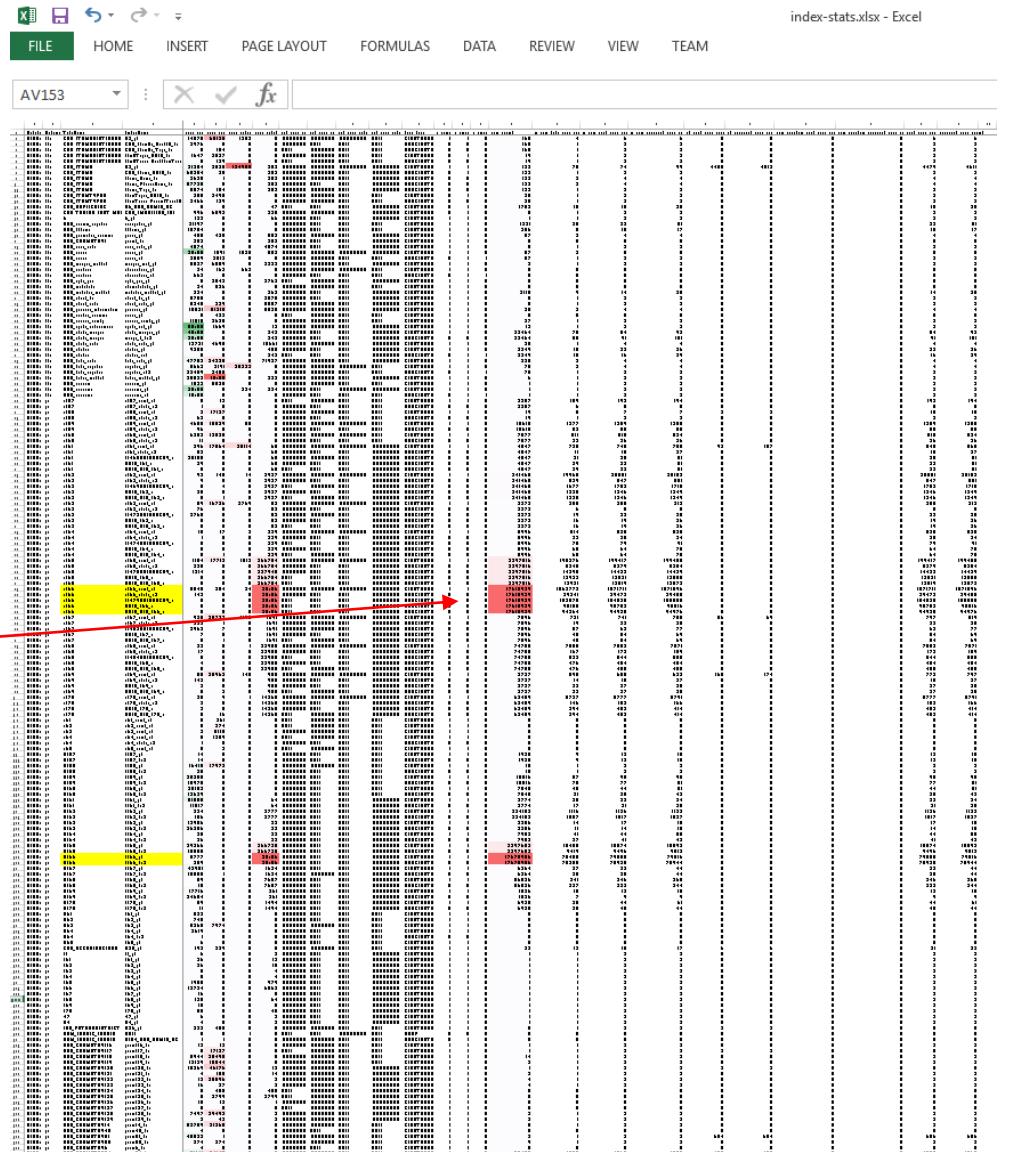
SLOData	dbo	GDB_ITEMRELATIONSHIPS	R3_pk	14570	68135	1202	0 #####	#####	10/12/18 15:00	NULL	CLUSTERED	1	1	0	160	4	6	0	0	0	0
SLOData	dbo	GDB_ITEMRELATIONSHIPS	GDB_ItemRel_DestID_idx	3976	0	0	0 #####	NULL	NULL	NULL	NONCLUSTERED	0	0	0	160	1	2	2	0	0	0

-- Andy Yun's personal sys.dm_db_index_usage_stats query

```
SELECT
    databases.Name AS DatabaseName,
    schemas.Name AS SchemaName,
    objects.Name AS TableName,
    indexes.Name AS IndexName,
    dm_db_index_usage_stats.user_seeks,
    dm_db_index_usage_stats.user_scans,
    dm_db_index_usage_stats.user_lookups,
    dm_db_index_usage_stats.user_updates,
    dm_db_index_usage_stats.last_user_seek,
    dm_db_index_usage_stats.last_user_scan,
    dm_db_index_usage_stats.last_user_lookup,
    dm_db_index_usage_stats.last_user_update,
    indexes.type_desc,
    indexes.is_primary_key,
    indexes.is_unique,
    indexes.is_unique_constraint,
    dm_db_partition_stats.row_count,
    dm_db_partition_stats.in_row_data_page_count,
    dm_db_partition_stats.in_row_used_page_count,
    dm_db_partition_stats.in_row_reserved_page_count,
    dm_db_partition_stats.lob_used_page_count,
    dm_db_partition_stats.lob_reserved_page_count,
    dm_db_partition_stats.row_overflow_used_page_count,
    dm_db_partition_stats.row_overflow_reserved_page_count,
    dm_db_partition_stats.used_page_count,
    dm_db_partition_stats.reserved_page_count
FROM master.sys.dm_db_index_usage_stats
INNER JOIN master.sys.databases
    ON dm_db_index_usage_stats.database_id = databases.database_id
INNER JOIN sys.objects
    ON dm_db_index_usage_stats.object_id = objects.object_id
INNER JOIN sys.schemas
    ON schemas.schema_id = objects.schema_id
INNER JOIN sys.indexes
    ON dm_db_index_usage_stats.index_id = indexes.index_id
    AND dm_db_index_usage_stats.object_id = indexes.object_id
INNER JOIN sys.dm_db_partition_stats
    ON dm_db_index_usage_stats.index_id = dm_db_partition_stats.index_id
    AND dm_db_index_usage_stats.object_id = dm_db_partition_stats.object_id
WHERE objects.type = 'U'
    AND databases.name = db_name()
ORDER BY databases.name, schemas.name, objects.name, indexes.type_desc, indexes.is_primary_key DESC, indexes.name
```

Use Excel “conditional formatting” on results to find needle in haystack of hundreds of indexes

2.7 M user updates
On 17.6 M row_count



Complete, but scary picture of what was going on.

50 MB data
7 MB index
====
57 MB

56,709 rows

SLO_OGLEASE	
OBJECTID (PK)	int
OWNER_NAM	numeric(38, 8)
TOWNSHIP	char(5)
RANGE	char(5)
SECT	numeric(38, 8)
SURVEYTYPE	char(1)
ALIQUOT	char(4)
UNIQUEKEY	char(20)
ONWARD_DTE	date
PROCESS	char(12)
DISNEY	char(100)
OGTOWNSHIP	char(3)
OGRANGE	char(3)
QUARTERS	char(16)
OTRTRACT	char(3)
OGLTRACT	char(3)
LSE_PREFIX	char(2)
LSE_NUMBER	numeric(38, 8)
LSE_SUFFIX	numeric(38, 8)
STATUS_DTE	char(7)
USL_ACRO	char(8)
VEREFF_DTE	datetime2(7)
VERTRM_DTE	datetime2(7)
OGRID_CODE	numeric(38, 8)
OGRID_NAM	char(45)
OGRID_ADR_NAM	char(30)
MAIL_STOP	char(20)
LINE1_ADR	char(30)
LINE2_ADR	char(30)
LINE3_ADR	char(30)
CITY_NAM	char(30)
ST_NAM	char(2)
ZIP_CODE	char(9)
CTRY_NAM	char(15)
PHONE_NUM	numeric(38, 8)
FAX_NUM	numeric(38, 8)
PROCDATE	datetime2(7)
SOURCEGRID	char(16)
GlobalID	uniqueidentifier
CreatedBy	char(255)
CreatedDate	datetime2(7)
ModifiedBy	char(255)
ModifiedDate	datetime2(7)
GridName	char(50)
SHAPE	geometry
SDE_STATE_ID (PK)	bit

12.2 GB data
2.7 GB index
====
14.9 GB

17,618,929 rows

a166	
OBJECTID (PK)	int
MERIDIAN	numeric(38, 8)
TOWNSHIP	char(5)
RANGE	char(5)
SECT	numeric(38, 8)
SURVEYTYPE	char(1)
ALIQUOT	char(20)
UNIQUEKEY	char(20)
ONWARD_DTE	date
PROCESS	char(12)
DISNEY	char(100)
OGTOWNSHIP	char(3)
OGRANGE	char(3)
QUARTERS	char(16)
OTRTRACT	char(3)
OGLTRACT	char(3)
LSE_PREFIX	char(2)
LSE_NUMBER	numeric(38, 8)
LSE_SUFFIX	numeric(38, 8)
STATUS_DTE	char(7)
USL_ACRO	char(8)
VEREFF_DTE	datetime2(7)
VERTRM_DTE	datetime2(7)
OGRID_CODE	numeric(38, 8)
OGRID_NAM	char(45)
OGRID_ADR_NAM	char(30)
MAIL_STOP	char(20)
LINE1_ADR	char(30)
LINE2_ADR	char(30)
LINE3_ADR	char(30)
CITY_NAM	char(30)
ST_NAM	char(2)
ZIP_CODE	char(9)
CTRY_NAM	char(15)
PHONE_NUM	numeric(38, 8)
FAX_NUM	numeric(38, 8)
PROCDATE	datetime2(7)
SOURCEGRID	char(16)
GlobalID	uniqueidentifier
CreatedBy	char(255)
CreatedDate	datetime2(7)
ModifiedBy	char(255)
ModifiedDate	datetime2(7)
GridName	char(50)
SHAPE	geometry
SDE_STATE_ID (PK)	bit

613 MB data
559 MB index
====
1174 MB

17,675,906 rows

D166	
SDE_STATE_ID (PK)	bit
SDE_DELETED_ROW_ID (PK)	int
DELETED_AT (PK)	bit

[R166_PK]

6364 pages,
56709 leaf
Root node
Index_id = 1 | root_page

OBJECTID INT

[i1479UNIQUEKEY]

6364 pages,
56709 leaf
Root node
Index_id = 1 | root_page

UNIQUEKEY NVARCHAR(20)

[UUID_166]

243 pages,
56709 leaf
Root node
Index_id = 1 | root_page

GLOBALID GUID

[UUID_OID_166]

243 pages,
56709 leaf
Root node
Index_id = 1 | root_page

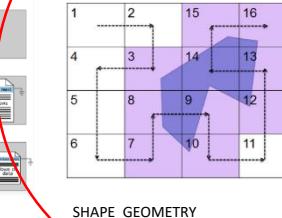
GLOBALID GUID
OBJECTID INT

[S125_IDX] (Spatial)

2533 pages,
853,517 leaf
Root node
Index_id = 1 | root_page

SHAPE GEOMETRY

[S125_IDX] (Spatial)
1,047,727 pages,
265,245,200 leaf



(111) Index reorgs > 5%frag
(9) Index rebuilds >30% frag

(110) Index reorgs > 5%frag
(9) Index rebuilds >30% frag

[d166_pk]

78,408 pages,
17,777,000 leaf
Root node
Index_id = 1 | root_page

SDESATE_ID BIGINT
SDEDELETEDROWID INT
DELETED_AT BIGINT

[d166_idx2]

70,388 pages,
17,591,600 leaf
Root node
Index_id = 1 | root_page

DELETED_AT BIGINT

[d166_idx2]

70,388 pages,
17,591,600 leaf
Root node
Index_id = 1 | root_page

DELETED_AT BIGINT

(66) Index reorgs
(2) Index rebuilds

(66) Index reorgs
(1) Index rebuilds

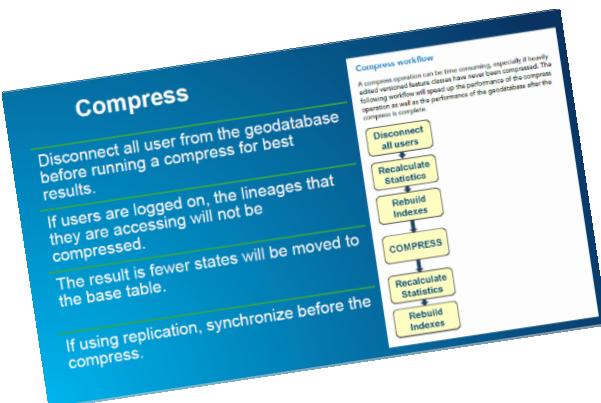
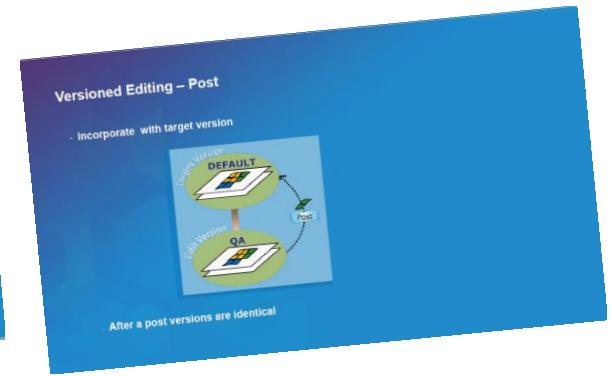
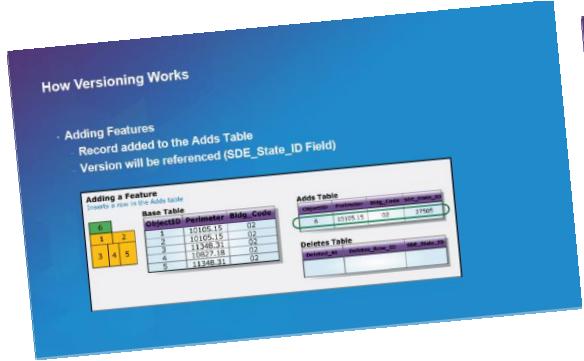
```

SELECT
    OBJECT_NAME(PARENT_OBJECT_ID) AS table_object,
    name,
    create_date,
    *
FROM sys.internal_tables it
WHERE
    INTERNAL_TYPE = 207
    AND it.name = 'extended_index_29295214_384000'

Create_date = 2018-10-12 06:50:30.743

```

Source of problem – the stuff of ESRI convention on Versioned Feature Classes
Performance Issues were a by-product of ESRI versioning.



Is it versioned?
Different for PostgreSQL
and Oracle:

```
SELECT NAME AS "Versioned feature class"
FROM dbo.GDB_ITEMS
WHERE Definition.exist('/*Versioned')[1]' = 1
AND Definition.value('/*Versioned')[1]', 'nvarchar(4)' ) = 'true'
```

But why? Digging deeper on the spatial index.

Three questions.

What tools did Microsoft give us for analyzing the index?

Is the index working, is it being used at all?

Does the index work quickly?

What tools did Microsoft give us for analyzing the index?

Very little info in PASS, but two useful papers from Microsoft on Spatial Index Internals, both co-authored by Michael Rys – now on Spark for .Net.

Spatial Indexing in Microsoft SQL Server 2008

Yi Fang, Marc Friedman, Giri Nair, Michael Rys, Ana-Elisa Schmidt

Microsoft Corp.

{yfang,marcfr,gnair,mrys,anasc}@microsoft.com

SIGMOD '08, June 9–12, 2008, Vancouver, BC, Canada.

Copyright 2008 ACM 978-1-60558-102-6/08/06...\$5.00.

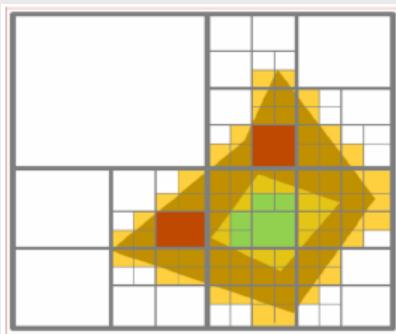


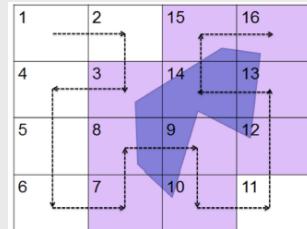
Figure-2: Grid decomposition of a polygon using 2x2 grids

Tuning Spatial Point Data Queries in SQL Server 2012



Tuning Spatial Point Data Queries in SQL Server 2012

Written by: Ed Katibah, Milan Stojic, Michael Rys, Nicholas Dritsas



Therefore, the reading pattern for an index seek over spatial index might look like:



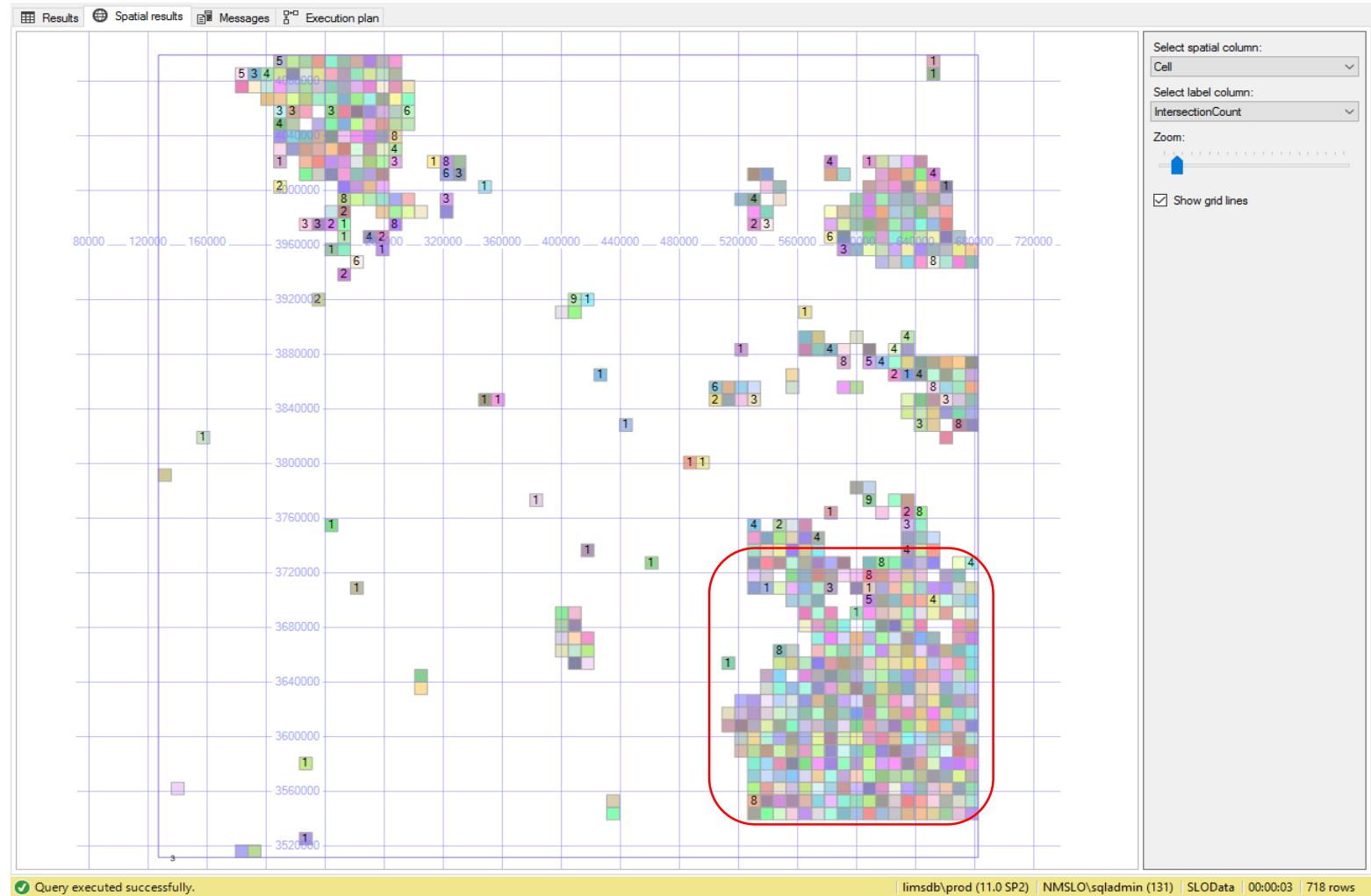
However, as the join to the base table is needed to get actual spatial objects, the IO pattern for the clustered index seek (the base table) might look like:



What tools did Microsoft give us for analyzing the index?

Diagnostic function #1: `sp_help_spatial_geometry_histogram()`

```
exec sp_help_spatial_geometry_histogram @tablename = "[ASLDData].[dbo].[ASLD_SURFACE_PARCELS]", @colname =  
'shape', @resolution = 64, @xmin = 144206, @ymin = 3466600, @xmax = 685016, @ymax = 4099054, @sample = 100;
```

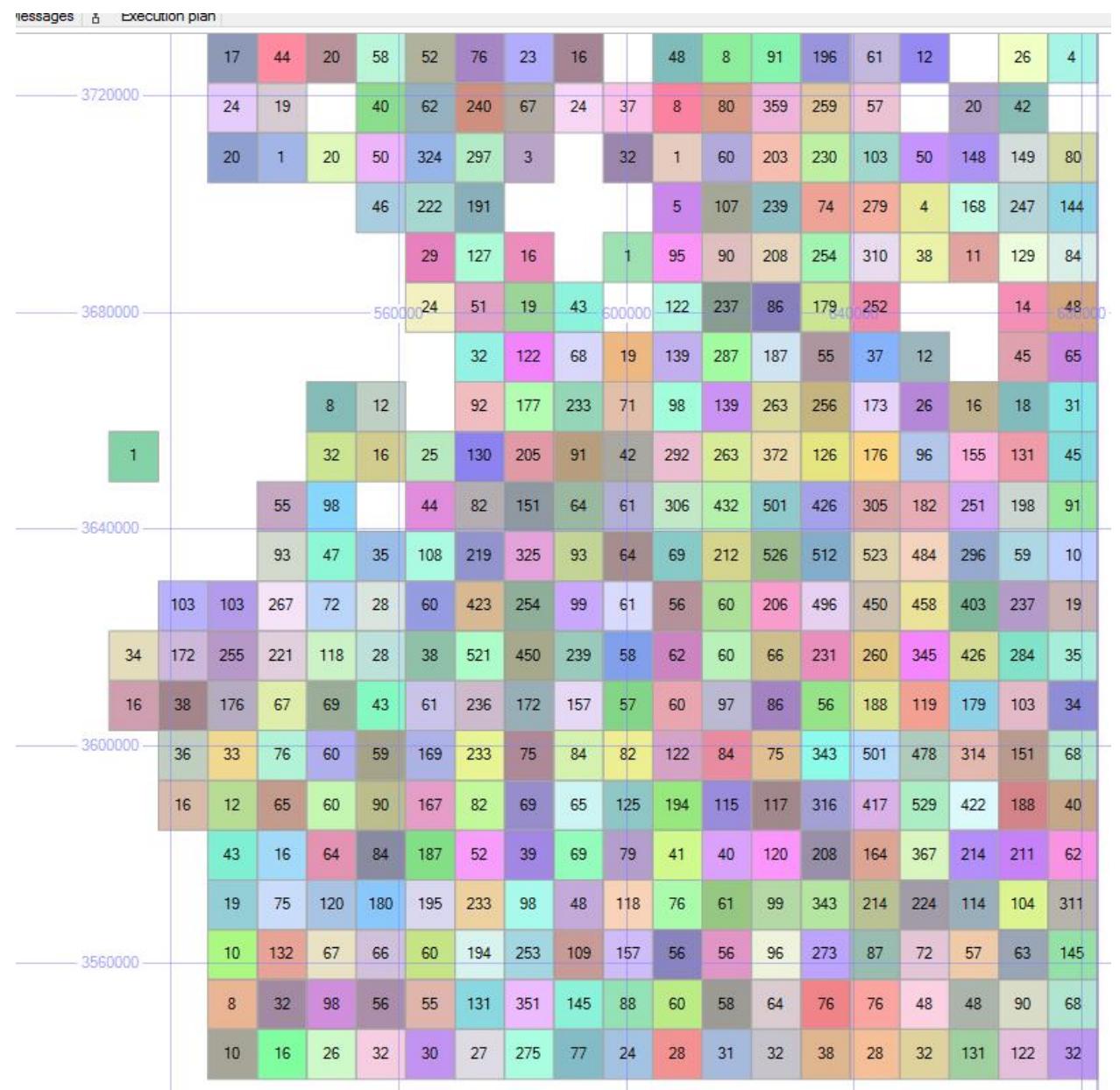


Zoom in:

Is the grid density for the spatial index evenly distributed?

`sp_help_spatial_geometry_histogram` tells us if the index is balanced or not

Values are the number of shapes in the smallest grid ... change index grid properties if unbalanced



What tools did Microsoft give us for analyzing the index?

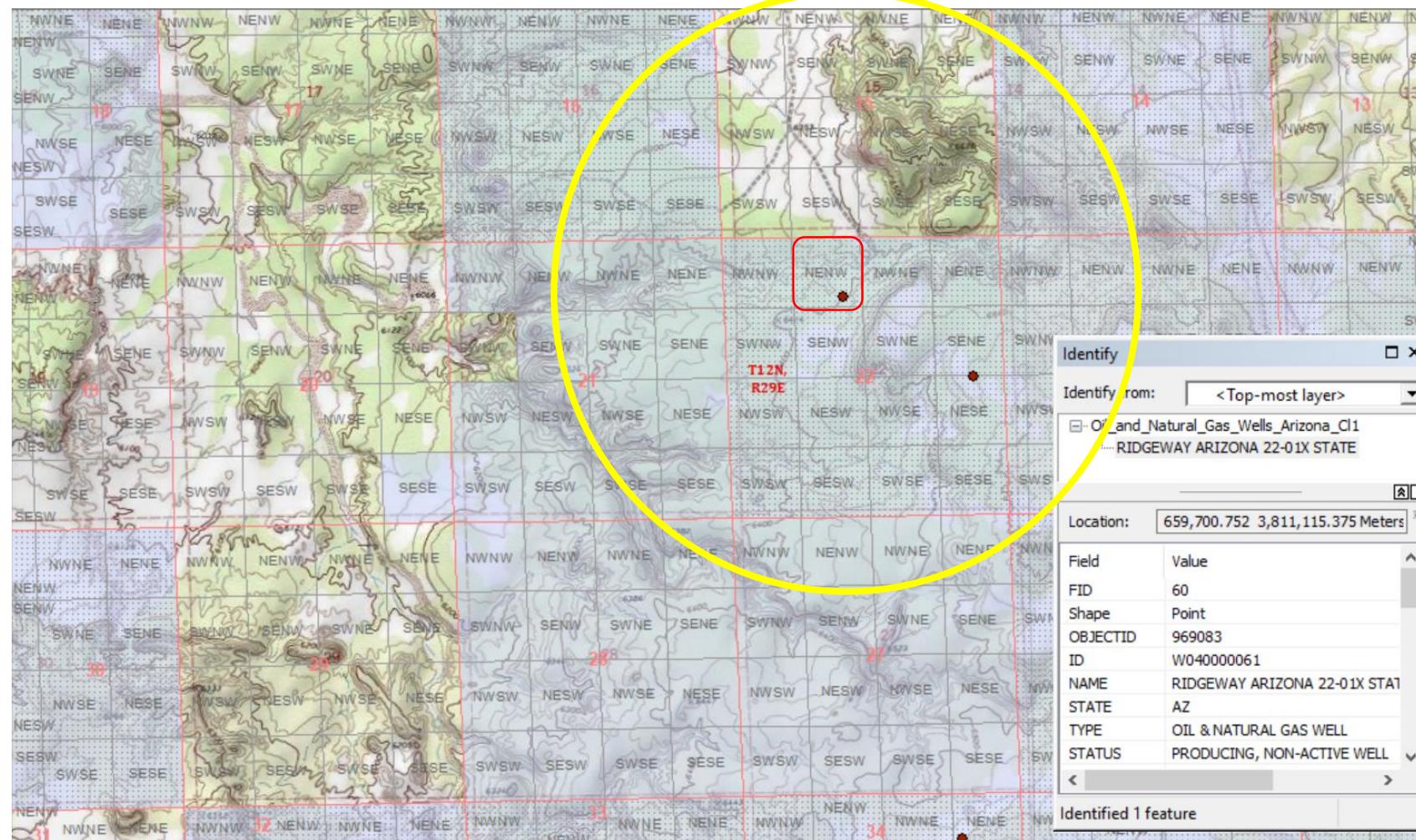
Diagnostic function #2: sp_help_spatial_geometry_index Whatif scenarios on a spatial index

```
declare @qs geometry ='POLYGON((-12162870 4086998, -12162871 4086998, -12162870 4086999, -12162870 4086998));  
exec sp_help_spatial_geometry_index @tabname = "[AZBaseData].[dbo].[PLSSSECONDDIVISION_AZ]", @indexname = 'FDO_Shape', @verboseoutput = 1,  
@query_sample = @qs;
```

propname	propval bigpoly bigsidx	propval smpoly bigsidx	propval bigpoly smallsidx	propval smpoly smsidx
Base_Table_Rows	17618929	17618929	56709	56709
Bounding_Box_xmin	127119.099343	127119.099343	127119.099343	127119.099343
Bounding_Box_ymin	3510943.72004	3510943.72004	3510943.72004	3510943.72004
Bounding_Box_xmax	682752.399957	682752.399957	682752.399957	682752.399957
Bounding_Box_ymax	4099048.36494	4099048.36494	4099048.36494	4099048.36494
Grid_Size_Level_1	64	64	64	64
Grid_Size_Level_2	64	64	64	64
Grid_Size_Level_3	64	64	64	64
Grid_Size_Level_4	64	64	64	64
Cells_Per_Object	16	16	16	16
Total_Primary_Index_Rows	265074776	265074776	853517	853517
Total_Primary_Index_Pages	1051507	1051507	2542	2542
Average_Number_Of_Index_Rows_Per_Base_Row	15	15	15	15
Total_Number_Of_ObjectCells_In_Level0_For_QuerySample	1	1	1	NULL
Total_Number_Of_ObjectCells_In_Level0_In_Index	327	327	2	2
Total_Number_Of_ObjectCells_In_Level1_For_QuerySample	64	64	64	NULL
Total_Number_Of_ObjectCells_In_Level3_In_Index	162574	162574	485	485
Total_Number_Of_ObjectCells_In_Level4_For_QuerySample	NULL	1	NULL	12
Total_Number_Of_ObjectCells_In_Level4_In_Index	264911875	264911875	853030	853030
Total_Number_Of_Interior_ObjectCells_In_Level1_For_QuerySample	36	36	NULL	NULL
Total_Number_Of_Interior_ObjectCells_In_Level4_In_Index	NULL	NULL	2	2
Total_Number_Of_Interior_ObjectCells_In_Level14_In_Index	60759124	60759124	195701	195701
Total_Number_Of_Intersecting_ObjectCells_In_Level1_For_QuerySample	28	28	485	485
Total_Number_Of_Intersecting_ObjectCells_In_Level3_In_Index	162574	162574	NULL	NULL
Total_Number_Of_Intersecting_ObjectCells_In_Level4_For_QuerySample	NULL	1	NULL	10
Total_Number_Of_Intersecting_ObjectCells_In_Level4_In_Index	204152751	204152751	657329	657329
Total_Number_Of_Border_ObjectCells_In_Level0_For_QuerySample	1	1	1	1
Total_Number_Of_Border_ObjectCells_In_Level0_In_Index	327	327	2	2
Interior_To_Total_Cells_Normalized_To_Leaf_Grid_Percentage	0.0	0.0	0.0	0.0
Intersecting_To_Total_Cells_Normalized_To_Leaf_Grid_Percentage	0.0	0.0	0.0	0.0
Border_To_Total_Cells_Normalized_To_Leaf_Grid_Percentage	0.0	0.0	0.0	0.0
Average_Cells_Per_Object_Normalized_To_Leaf_Grid	0.0	0.0	0.0	0.0
Average_Objects_PerLeaf_GridCell	0.0	0.0	0.0	0.0
Number_Of_SRIDs_Found	2	2	2	2
Width_Of_Cell_In_Level1	69454.16257675	69454.16257675	69454.16257675	69454.16257675
Width_Of_Cell_In_Level2	8681.77032209375	8681.77032209375	8681.77032209375	8681.77032209375
Width_Of_Cell_In_Level3	1085.22129026172	1085.22129026172	1085.22129026172	1085.22129026172
Width_Of_Cell_In_Level4	135.652661282715	135.652661282715	135.652661282715	135.652661282715
Height_Of_Cell_In_Level1	73513.0806125	73513.0806125	73513.0806125	73513.0806125
Height_Of_Cell_In_Level2	9189.1350765625	9189.1350765625	9189.1350765625	9189.1350765625
Height_Of_Cell_In_Level3	1148.64188457031	1148.64188457031	1148.64188457031	1148.64188457031
Height_Of_Cell_In_Level4	143.580235571289	143.580235571289	143.580235571289	143.580235571289
Area_Of_Cell_In_Level1	5105789452.37831	5105789452.37831	5105789452.37831	5105789452.37831
Area_Of_Cell_In_Level2	79777960.193411	79777960.193411	79777960.193411	79777960.193411
Area_Of_Cell_In_Level3	1246530.62802205	1246530.62802205	1246530.62802205	1246530.62802205
Area_Of_Cell_In_Level4	19477.0410628445	19477.0410628445	19477.0410628445	19477.0410628445
CellArea_To_BoundingBoxArea_Percentage_In_Level1	1.5625	1.5625	1.5625	1.5625
CellArea_To_BoundingBoxArea_Percentage_In_Level2	0.0244140625	0.0244140625	0.0244140625	0.0244140625
CellArea_To_BoundingBoxArea_Percentage_In_Level3	0.0003814697265625	0.0003814697265625	0.0003814697265625	0.0003814697265625
CellArea_To_BoundingBoxArea_Percentage_In_Level4	5.96046447753906E-06	5.96046447753906E-06	5.96046447753906E-06	5.96046447753906E-06
Number_Of_Rows_Selected_By_Primary_Filter	(may have false positives)			
Number_Of_Rows_Selected_By_Internal_Filter	(by the index optimizations)	327	56709	9
Number_Of_Times_Secondary_Filter_Is_Called	(the expensive operation)	327	17635	1
Number_Of_Rows_Output		0	39074	8
Percentage_Of_Rows_NotSelected_By_Primary_Filter		0	0	9
Percentage_Of_Primary_Filter_Rows_Selected_By_Internal_Filter		99.9981440415589	0	99.9841295032535
Internal_Filter_Efficiency	(by the index optimizations)	100	31.0973566805974	11.11111111111111
Primary_Filter_Efficiency		0	0	100

Demo

Is the spatial index working? Design a spatial query, in yellow, all SubSections within a mile of a Helium well.



1. Select a geometry to tessellate with.

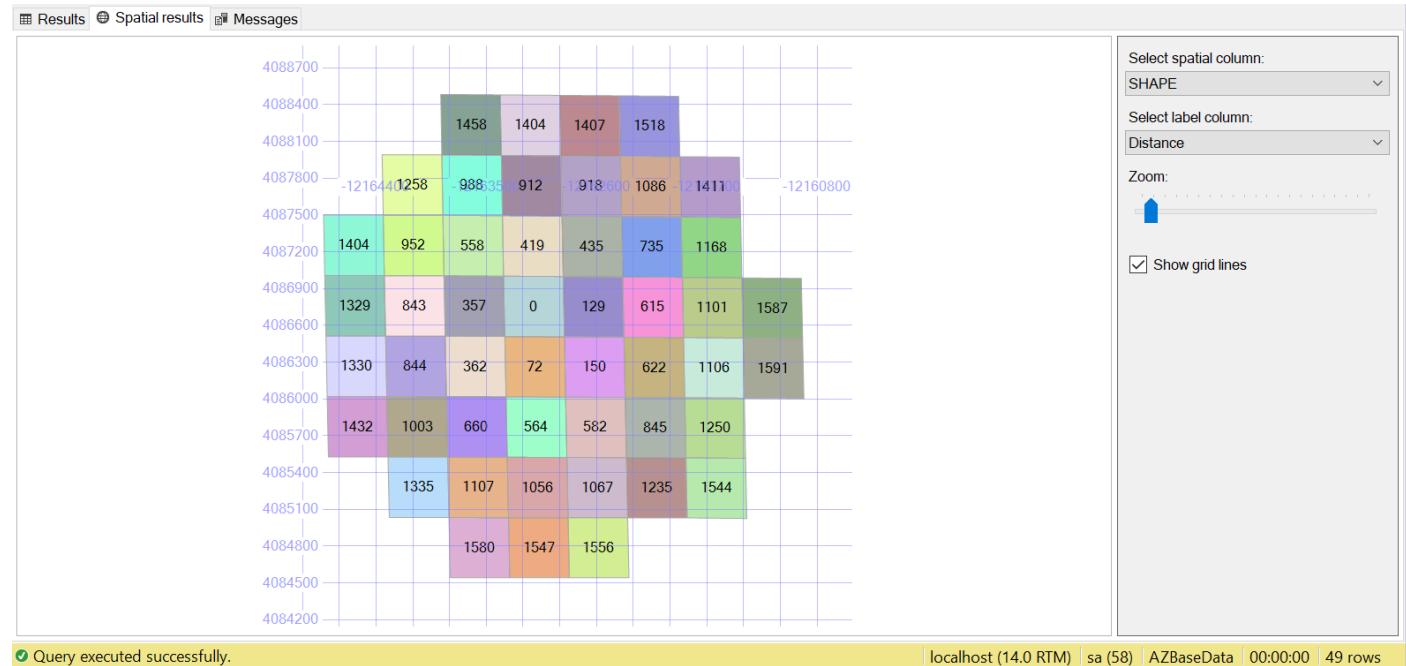
```
-- define a variable for the geometry to tessellate with, a spatial data value for a helium well
DECLARE @point1 geometry;

-- load the geometry point value for a helium well into a variable
SELECT
    @point1 = SHAPE
FROM [AZBaseData].[dbo].[AZWELLS_CLIPPED_WGS84_3857]
WHERE
    ID = 'W040000061' -- Well W040000061 - Ridgeway Arizona 22-01X STATE
```

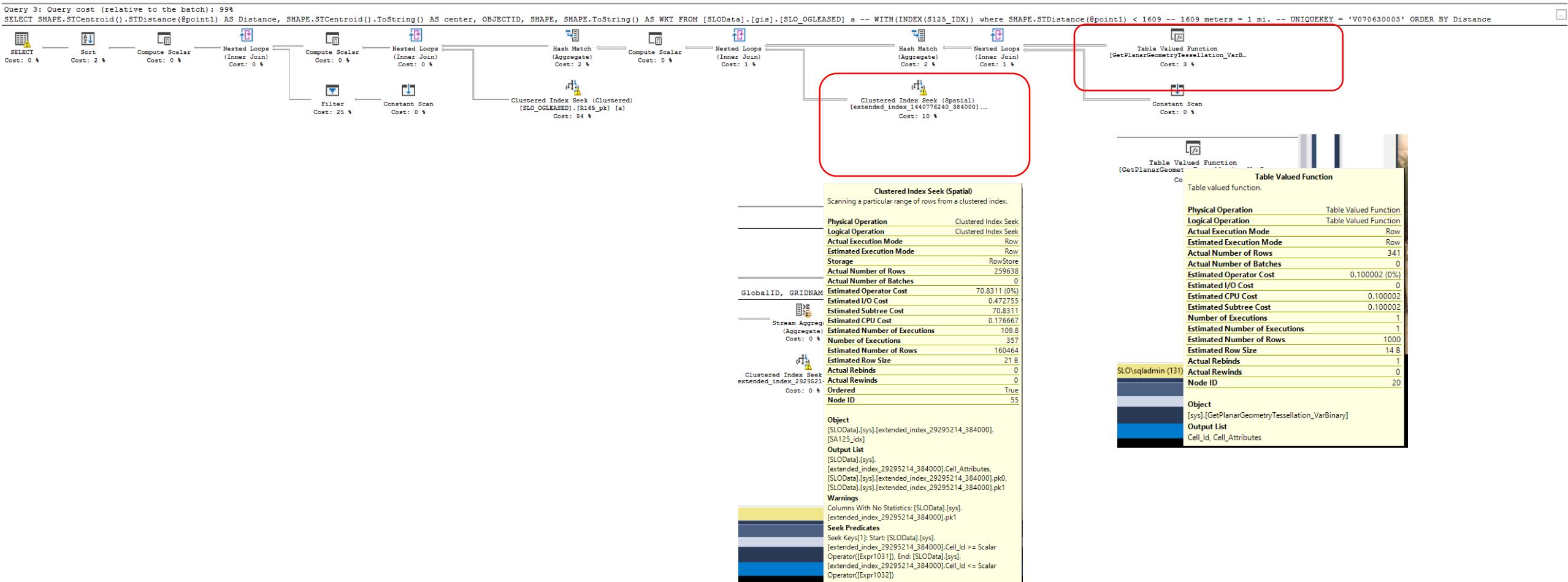
2. Use .STDistance() method in query.

```
-- find the PLSS subsections within a mile from the national blm dataset 24 million rows, 300M leaf nodes.
SELECT
    a.SHPE,
    round(a.Shape.STDistance(@point1),0) Distance,
    a.SECDIVID
FROM [AZBaseData].[dbo].[PLSSSECONDDIVISION] a -- WITH(INDEX(FDO_Shape)) -- shouldn't need index hint
WHERE
    a.Shape.STDistance(@point1) < 1609 -- 1609 meters = 1 mi.
ORDER BY Distance
```

3. Spatial Results



4. Execution plan verifies use of spatial index



Demo

Any Queries?

Next:

09:45 AM - 10:45 AM

Database
Maintenance And
Performance Tuning

Faster Transactions:
Query Tuning for
Data Manipulation

Jeff Iannucci

Level: Intermediate

Contact Info:

blog: www.kkarnsdba.com
twitter: @kevinkarns



I am humbled by the many hours of training I've received from PASS volunteers over the years, e-mail me to ask more questions or for Zoom/Webex education.

Special Thanks to SQL Saturday volunteers, sponsors & speakers - for making this event possible.

SQL Saturday #869 sponsors:

Chandler-Gilbert Community College, Microsoft Azure,
Snowflake, Paychex, DriveTime, Idera, Windocks,
TEKsystems, Slalom, VMWare, PASS