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

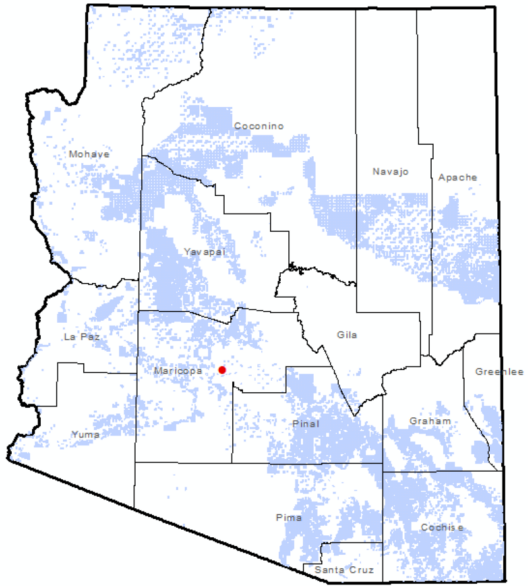
- Background about Arizona land and the PLSS.
- Applications that consume spatial data in SQL Server. (brief)
- ETL challenges with the PLSS and SQL Server spatial data. (very brief)
- GIS security challenges. (brief)
- SQL Server spatial features overview: Spatial tables, Views, Indexes, OGC Method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

Part 2

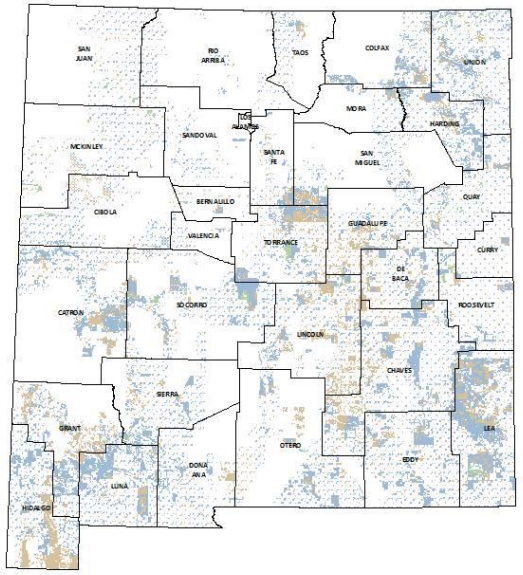
Drilling into a spatial database performance lesson from a DBA's viewpoint:
Look at index maintenance plans, Python, traditional indexes, spatial indexes, ending with tools available for spatial index diagnostics.

Disclaimer

Arizona
State Trust Lands Overview



New Mexico
Performance & Tuning Lessons

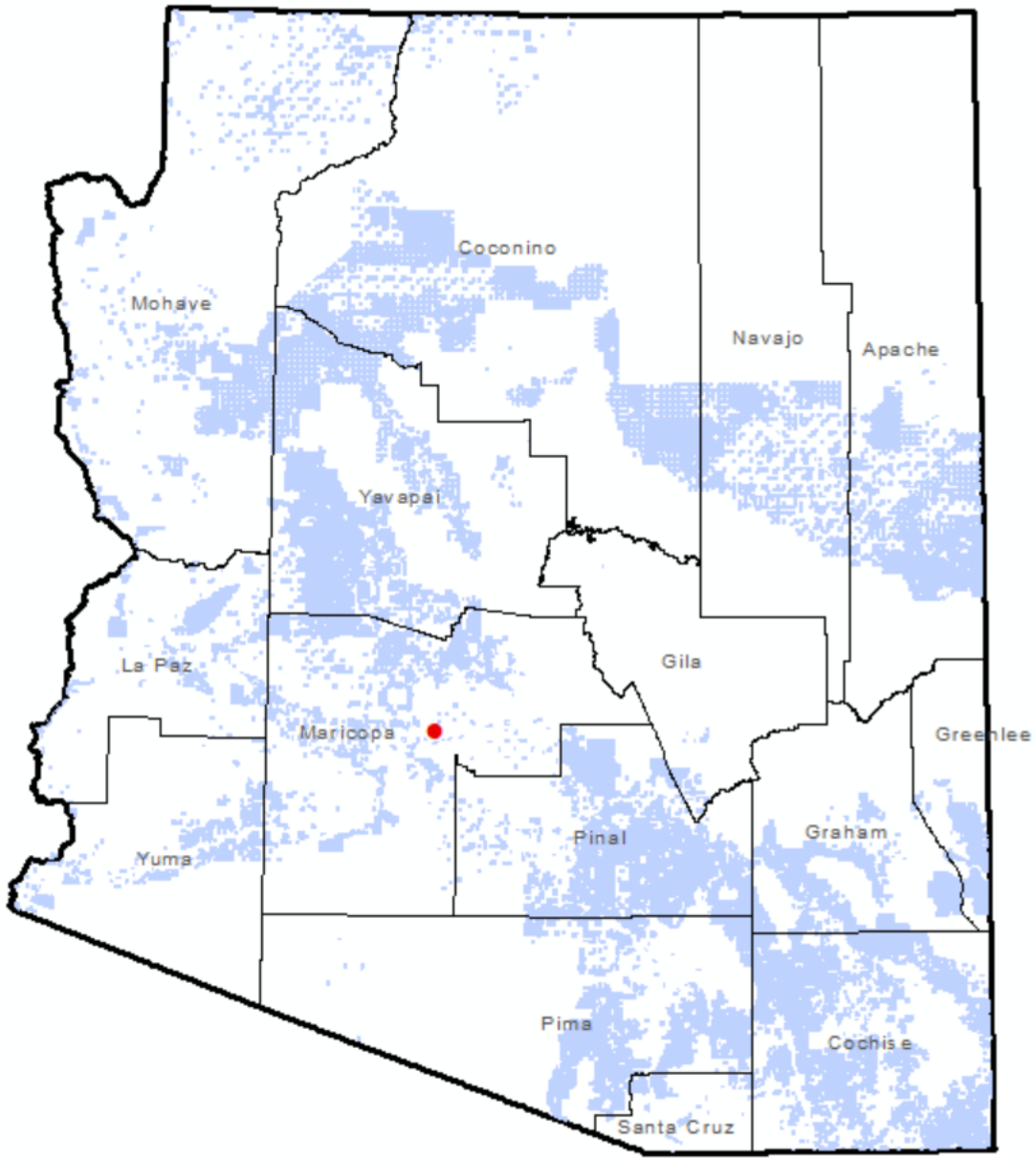


Goal is to teach about tuning the squirrelly Geography/Geometry data type in SQL Server, not speaking on behalf of either Commission.
All examples today are from data downloaded from public clearinghouses and manipulated with an inexpensive ArcGIS license for personal use.

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.



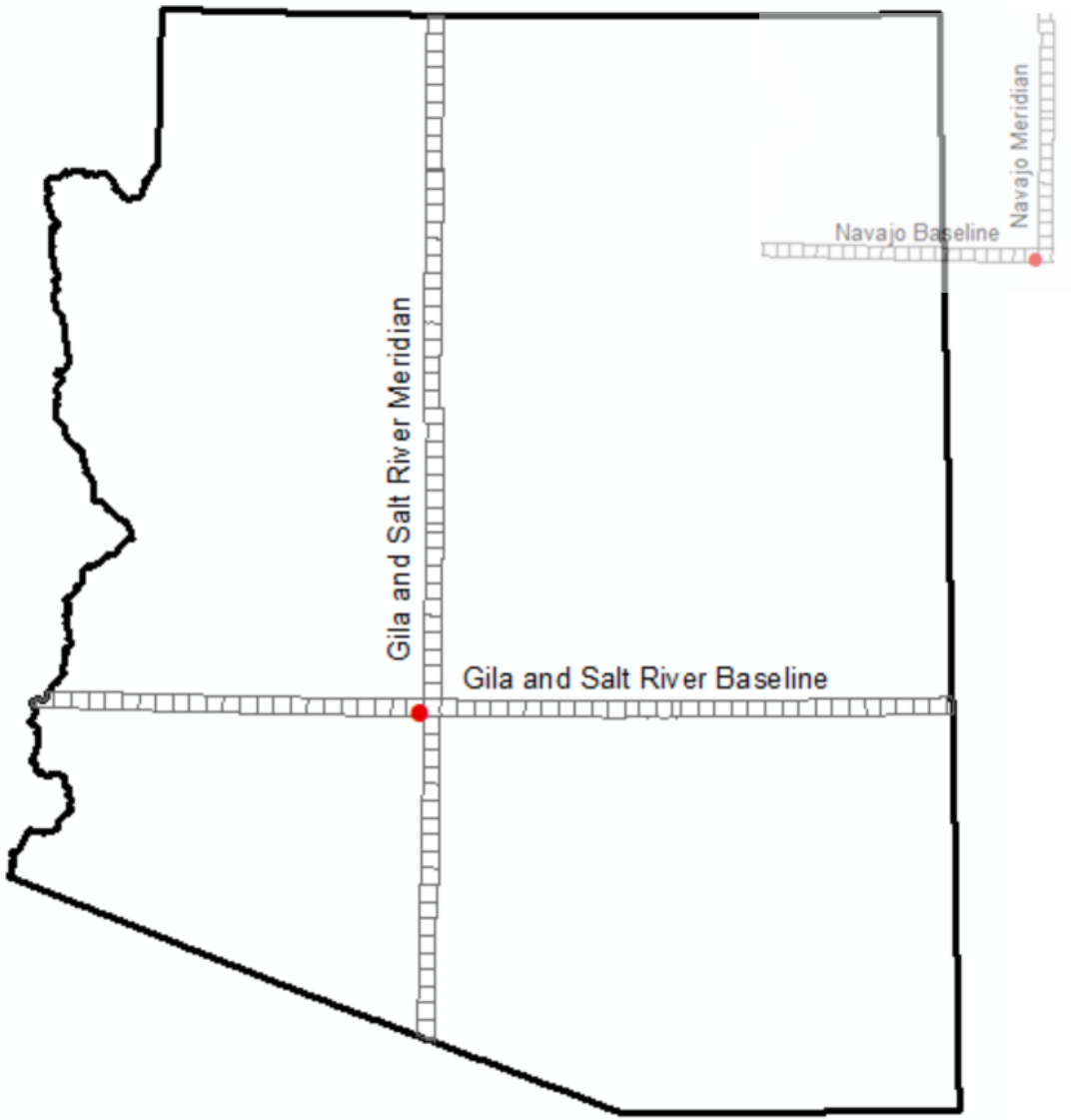
Federal Public Land Survey System (PLSS)

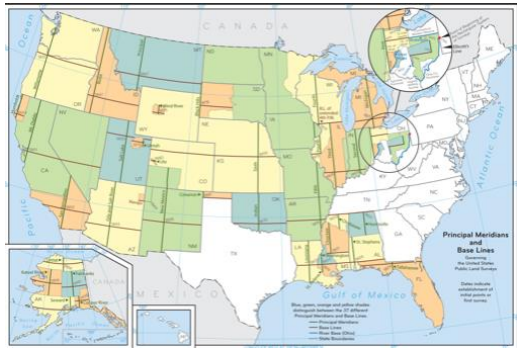
Top level Meridian

- 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.

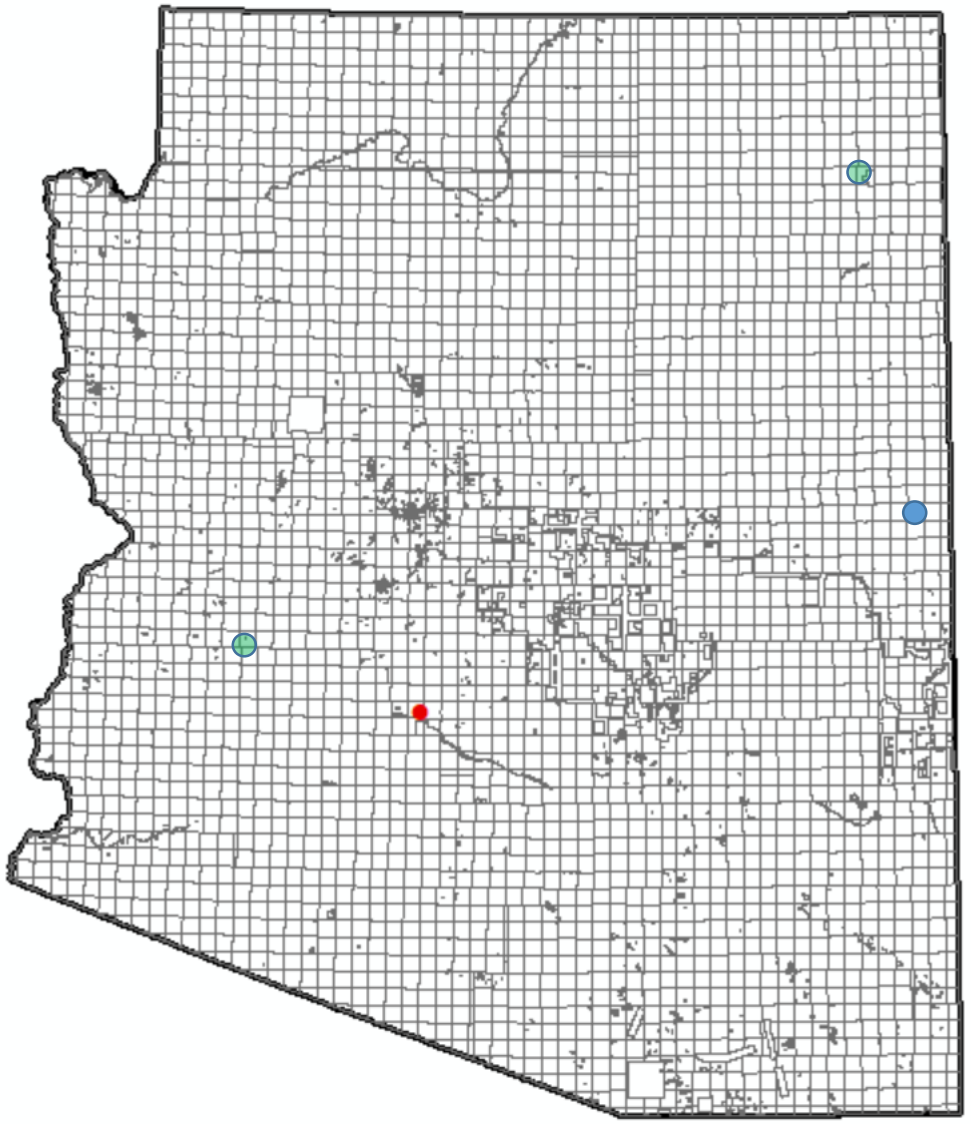




Next Level
PLSS Townships

- Over 3500 townships, approx. 6 miles wide.

- Except in grant lands.



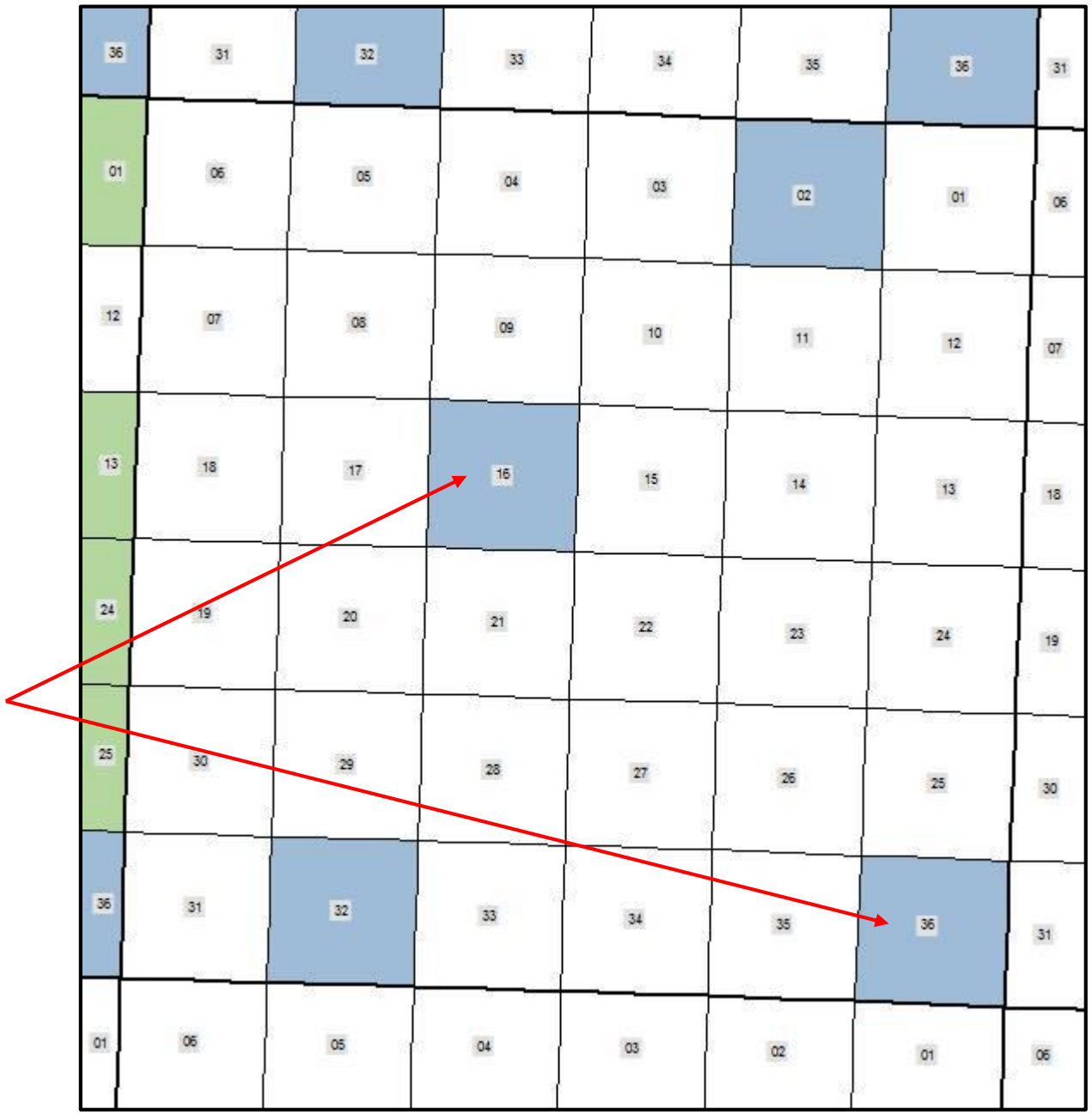
- 01N 01E
- 05N 10W (two)
- 12N 29E

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 .

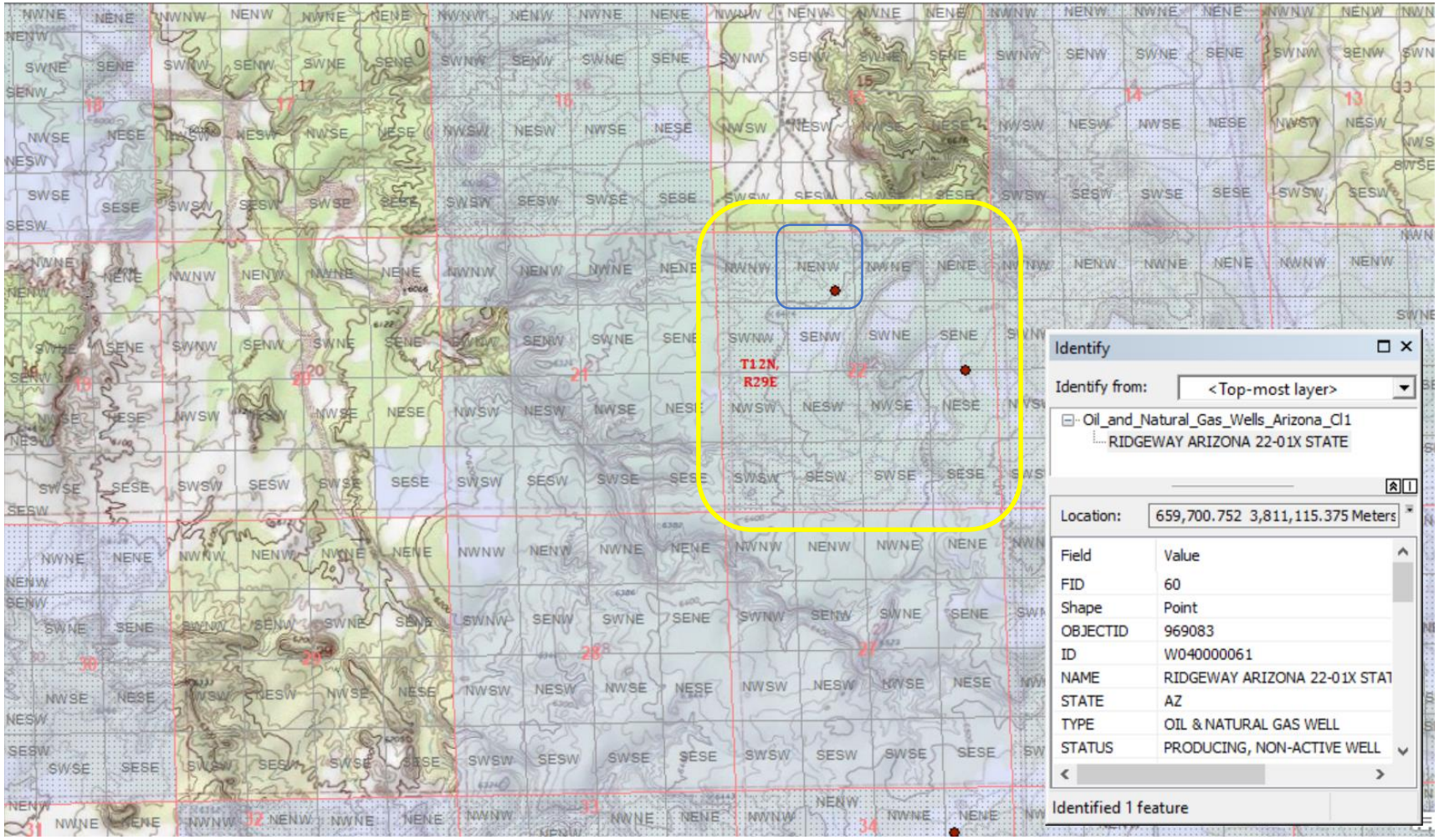


Lowest level – PLSS SubSections

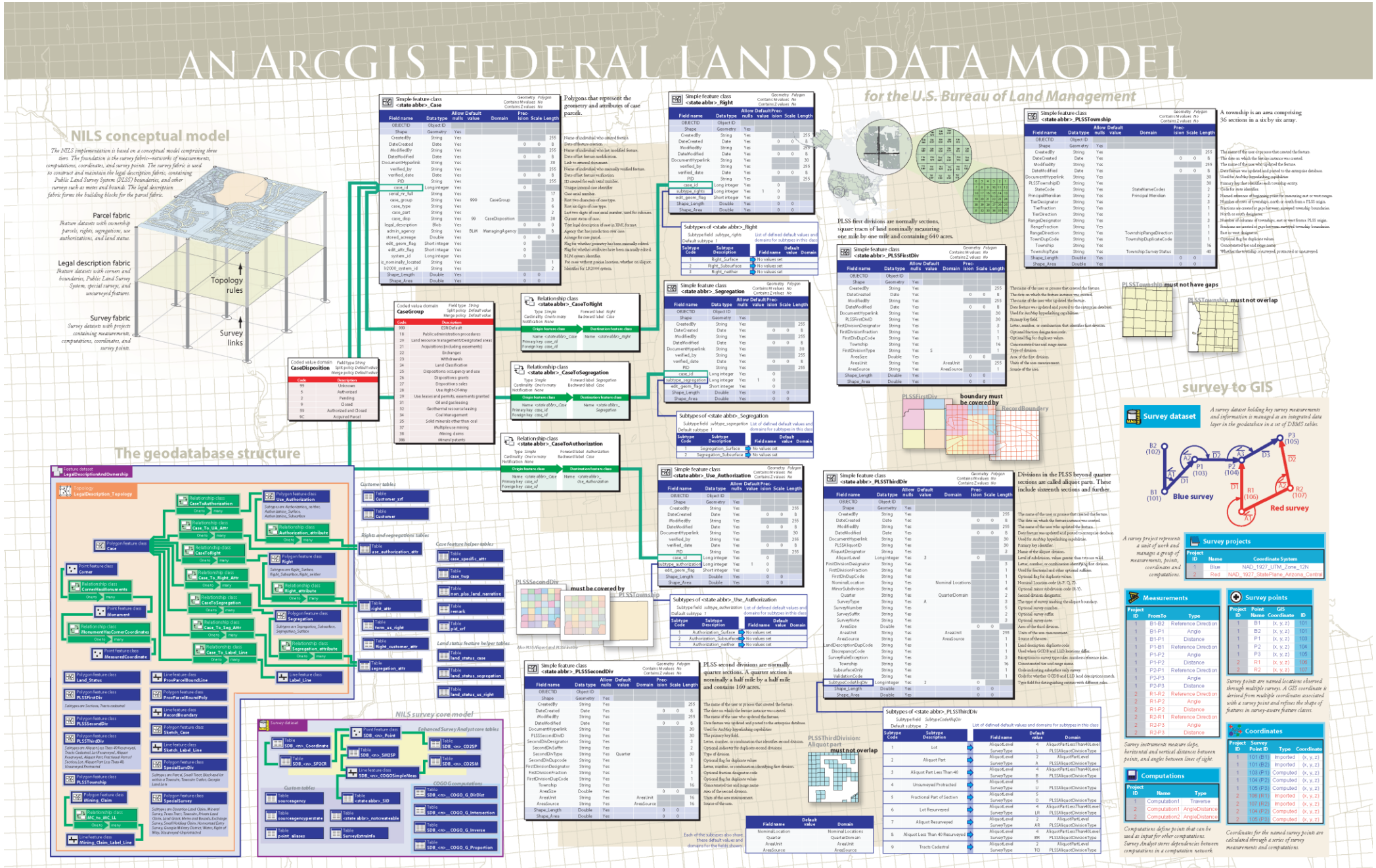
(aka quarter-quarters)

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.

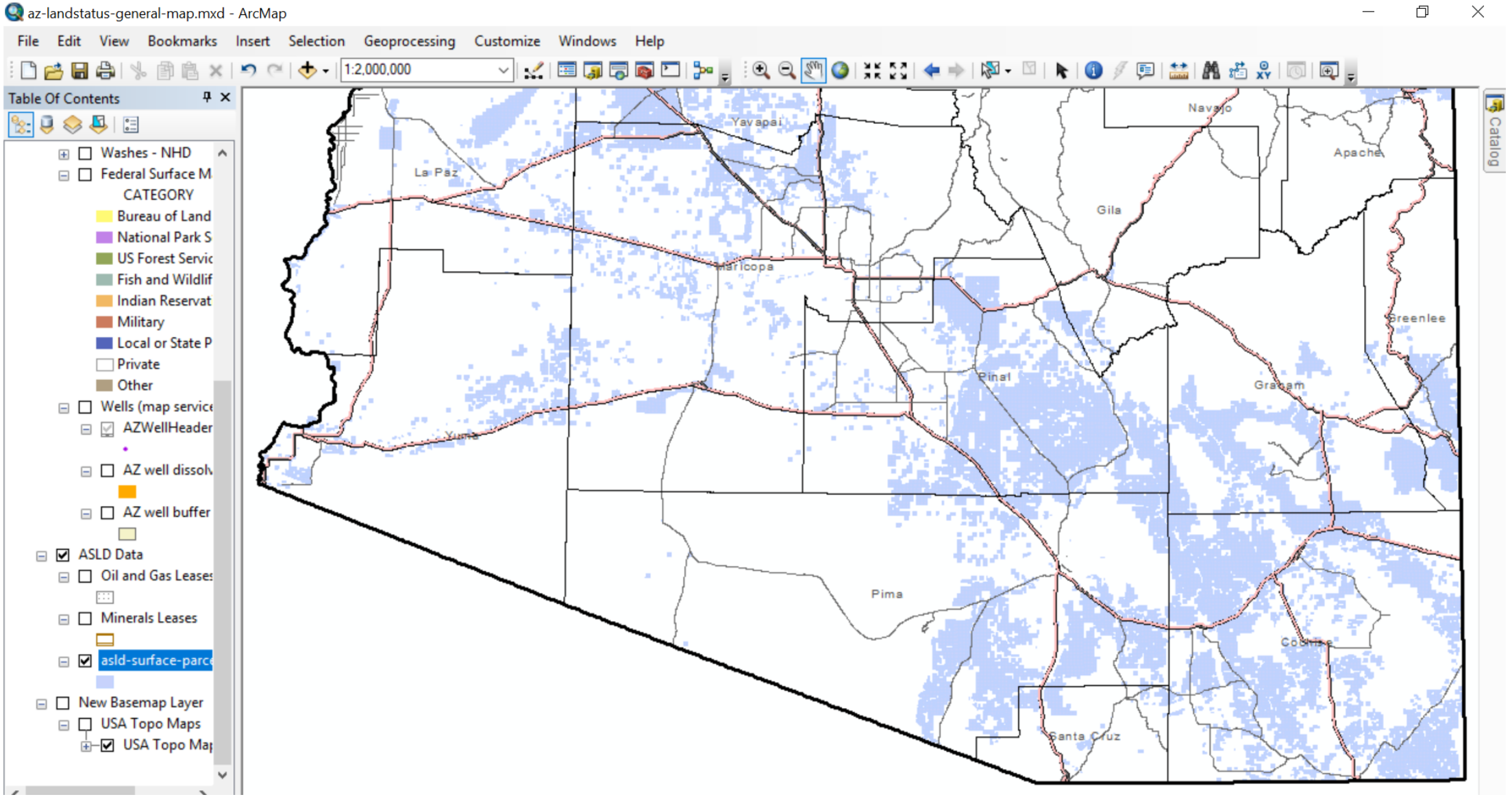


A BLM/ESRI Standard Land Data Model – (NILS) from late 90's… now called Geocommunicator or Navigator.



Front End Applications on SQL Server Spatial data

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

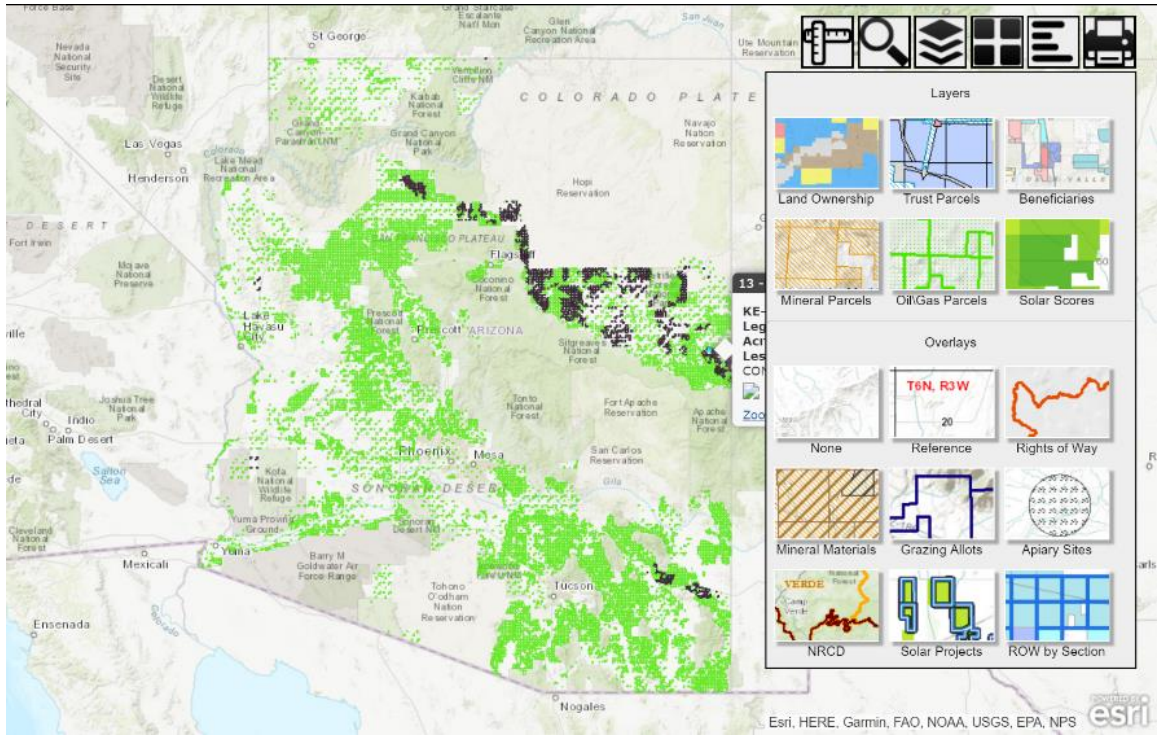


Demo

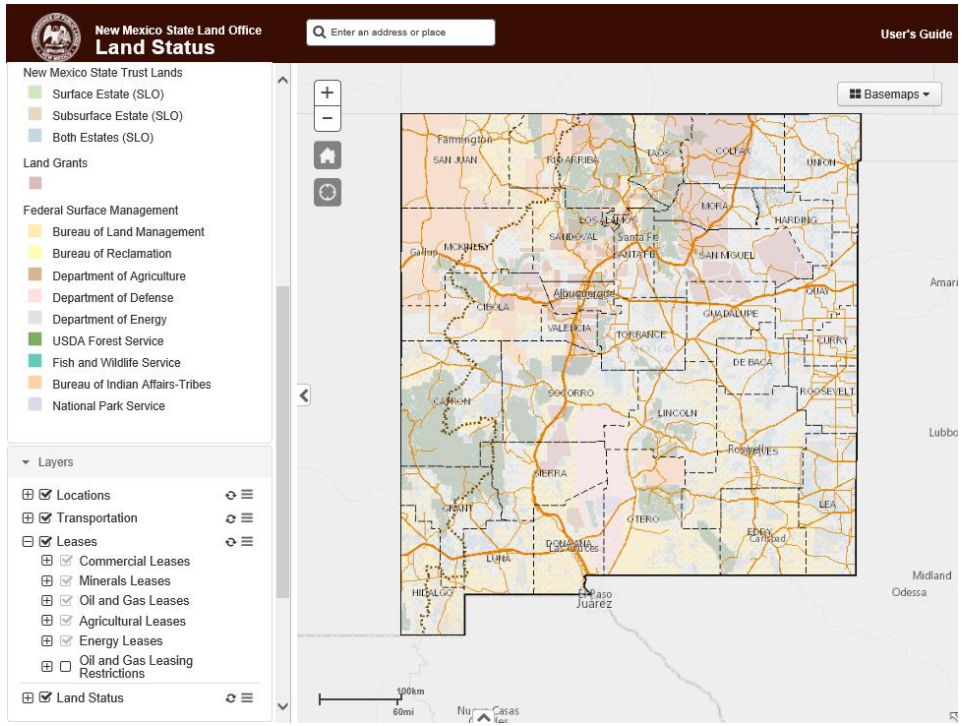
Front End Applications on SQL Server Spatial data

(#2) Interactive webmaps

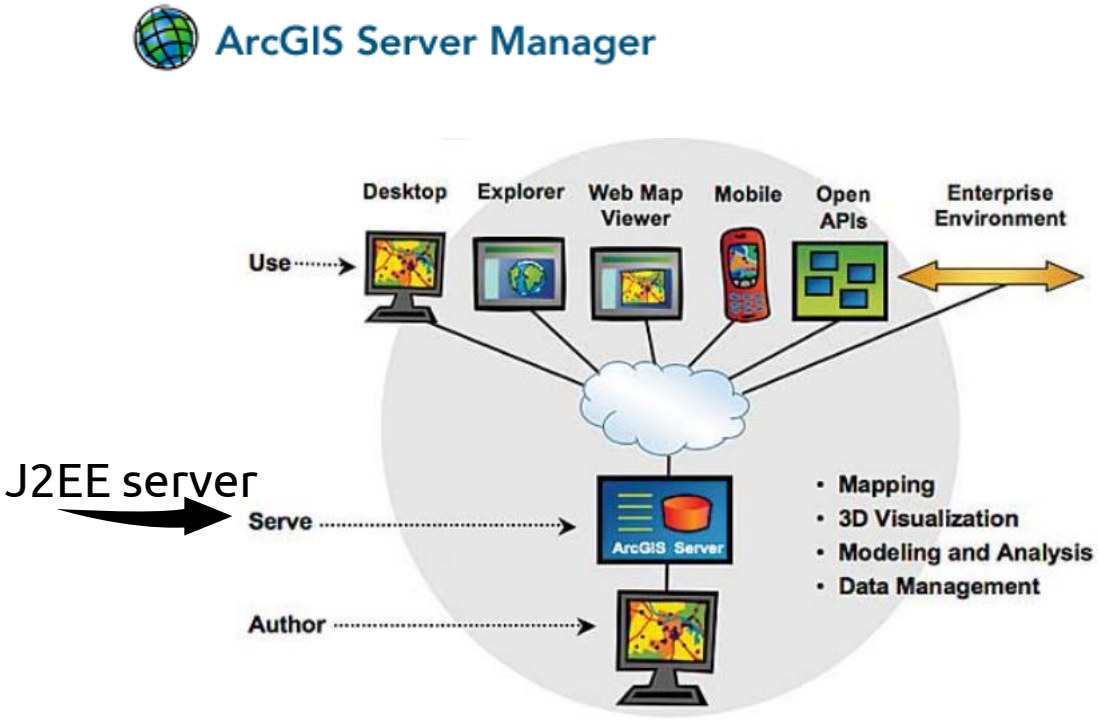
http://gis.azland.gov/webapps/parcel/



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



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



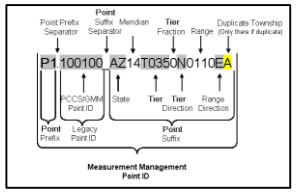
- Web services for internal .Net apps
- Google Earth .KML exports
- Mobile – Getac devices & custom application
- Rights of Ways – ESRI Workflow Manager
- ArcGIS Portal

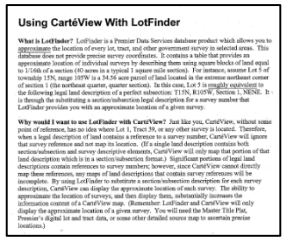
ETL Challenge:
converting from PLSS (Township Range Section Subsection) to Geometry data type

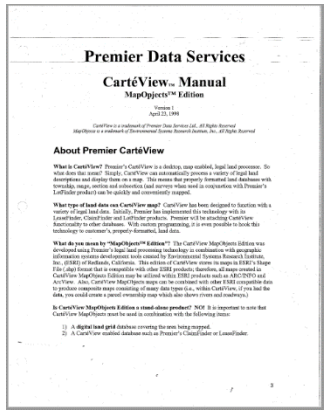
(1970s 80s)
BLM - PLSS → Lotfinder® → CartéView® → LUMAS 1 → LUMAS 2 → (2013) LIMS → Python based LUMAS

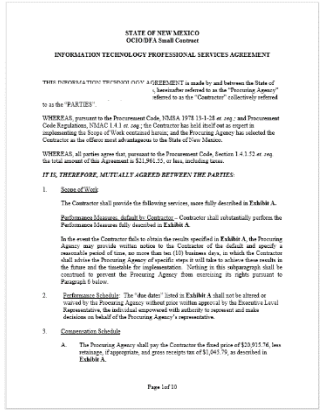
Survey point Numbering System

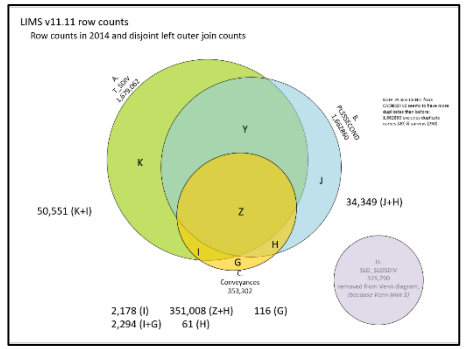
Cadnsdi+PLSS+DB2 +Conveyances LUMASGridPrimary



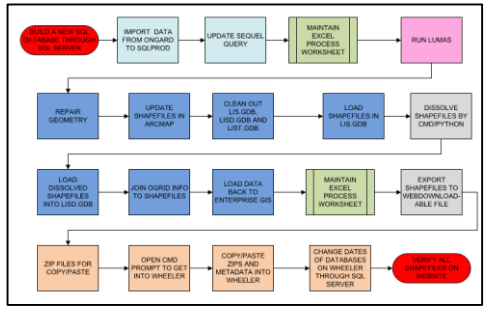








```
import arcpy
import urllib
import json
import ptl_MainTool


# Determine the database to write GIS WIP Status to
def findDBownership(notes):
    if                  in notes and "dev" not in notes and "sde" not in notes:
        return          Production.sde"
    elif "              " in notes:
        return          _Development.sde"
    elif "              " in notes:
        return          est.sde"
    else:
        arcpy.AddError("Environment could not be determined. LUMAS could not be run.")

        returnCode = 1
        arcpy.SetParameterAsText(3,returnCode)
        import urllib
        import json
        assignURL = '{0}/jobs/assign?user={1}'\
            '&jobs={2}'\
            '&assignedtype={3}'\
            '&assignedto={4}&f=pjson'.format(wmxPath,
                                             wmxUser,
                                             2,
                                             groupName)

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

def findSISdb(notes):
    arcpy.AddMessage(notes)
```



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

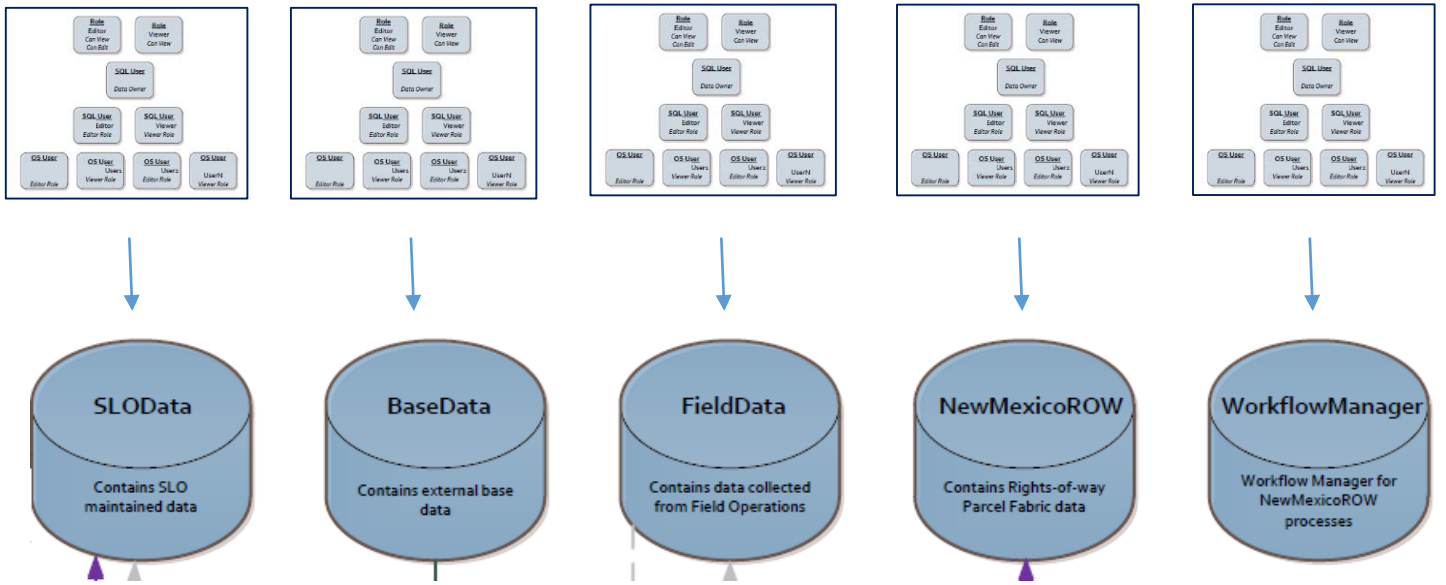
contractor #1
security model #1

internal staff
security model #2

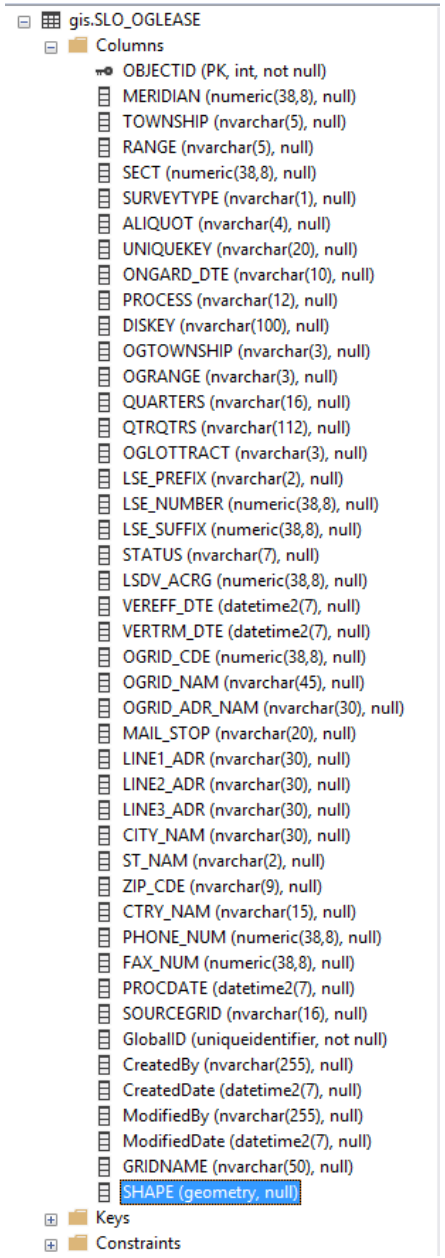
contractor #2
security model #3

contractor #3
security model #4

ESRI third party S/W
security model #5



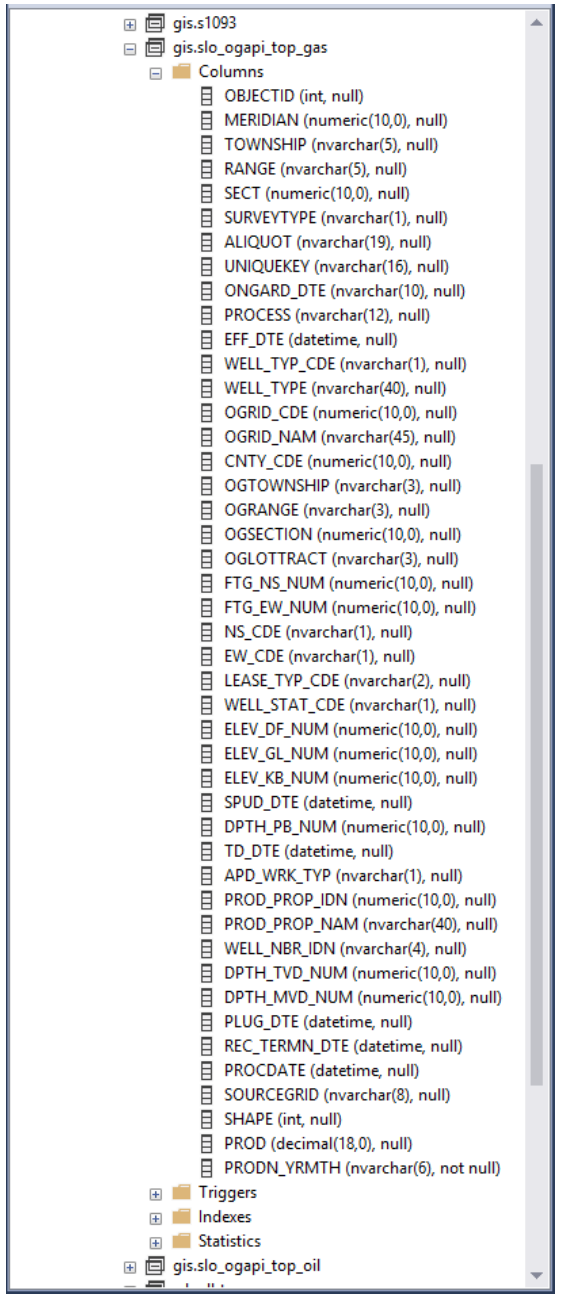
Spatial table




```
-- 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           0xA20C000000000000A9E1EDF209B6D3C8E106A562ED8B309B6
```

Spatial View

"ESRI specific"



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

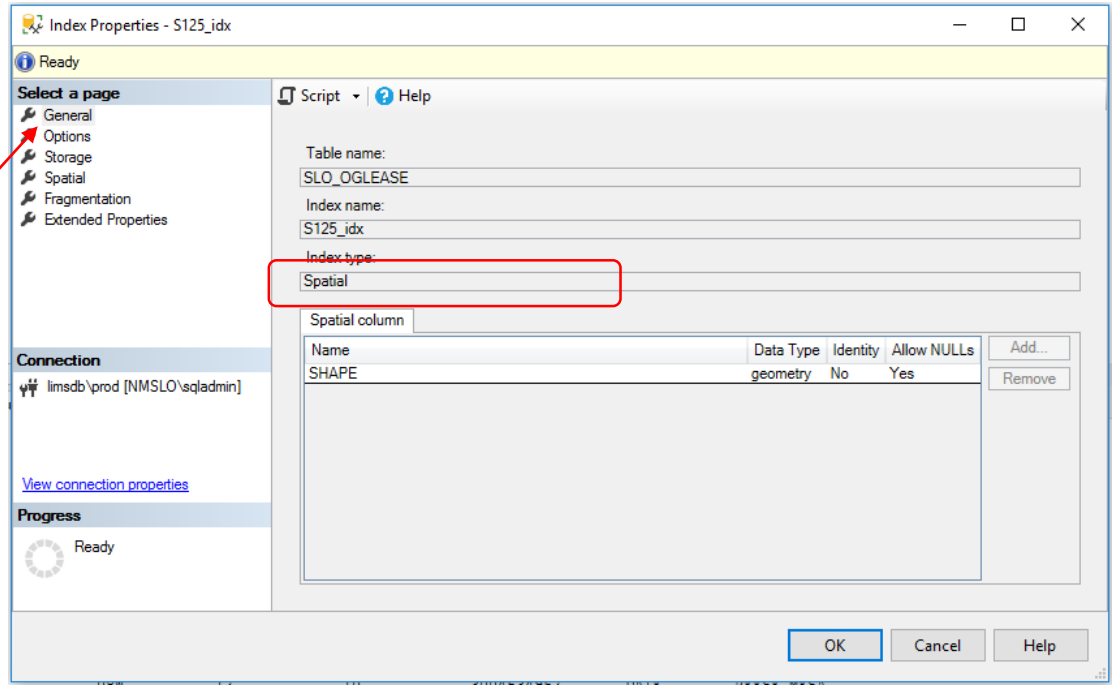
```
sdetable -o create_view -s servername -D dbname -u username -p passwd -i esri_sde
    -T SLO_OGAPI_TOP_GAS -t "SDE.GIS.SLO_OGAPI" -c "*"
```

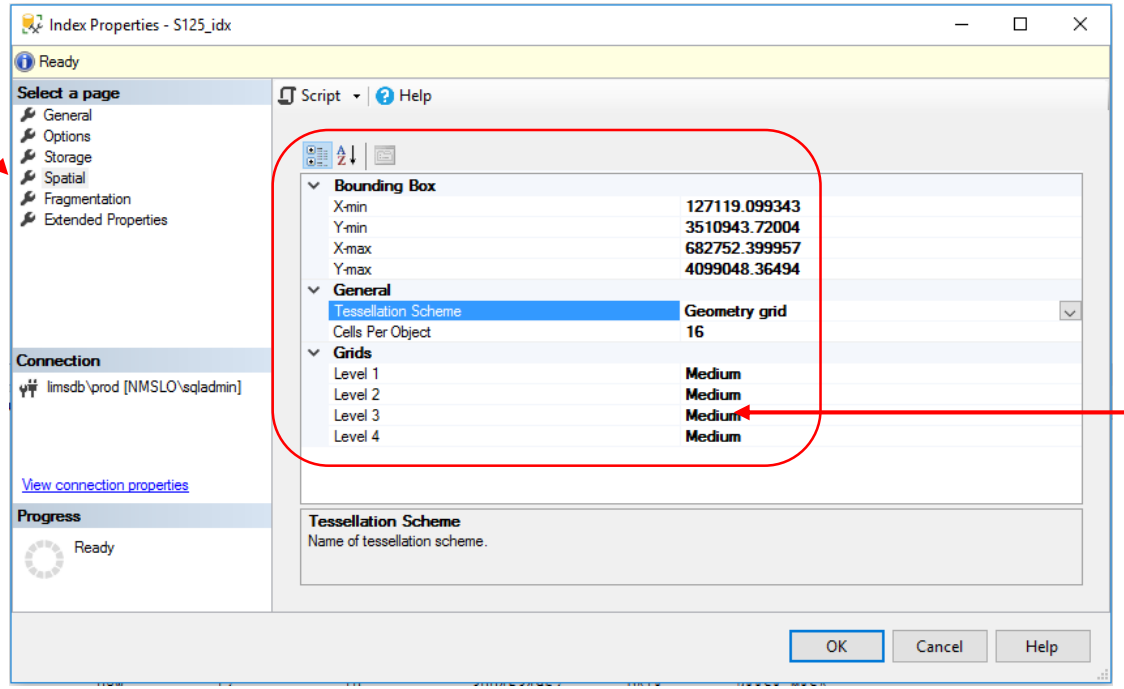
- Using ESRI ArcPy Python commands

```
""" 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
- STIsSimple
- STIsValid
- STLength
- STNumCurves (geometry Data Type)
- STNumGeometries
- STNumInteriorRing
- STNumPoints
- STOverlaps
- STPointN
- STPointOnSurface
- STRelate
- STSrid
- STStartPoint
- STSymDifference
- STTouches
- STUnion
- STWithin
- STX
- STY

https://docs.microsoft.com/en-us/sql/t-sql/spatial-geometry/ogc-methods-on-geometry-instances?view=sql-server-2017

ArcGIS Desktop | Documentation | Support | SEARCH | Sign In | English | esri

ArcGIS Help

Home | Get Started | Map | Analyze | Manage Data | Tools | More...

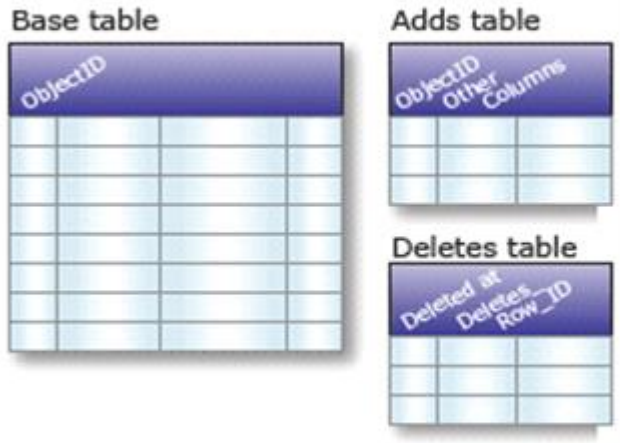
Manage Data > Geodatabases > Archiving data

- 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, 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_OGLEASE'
```

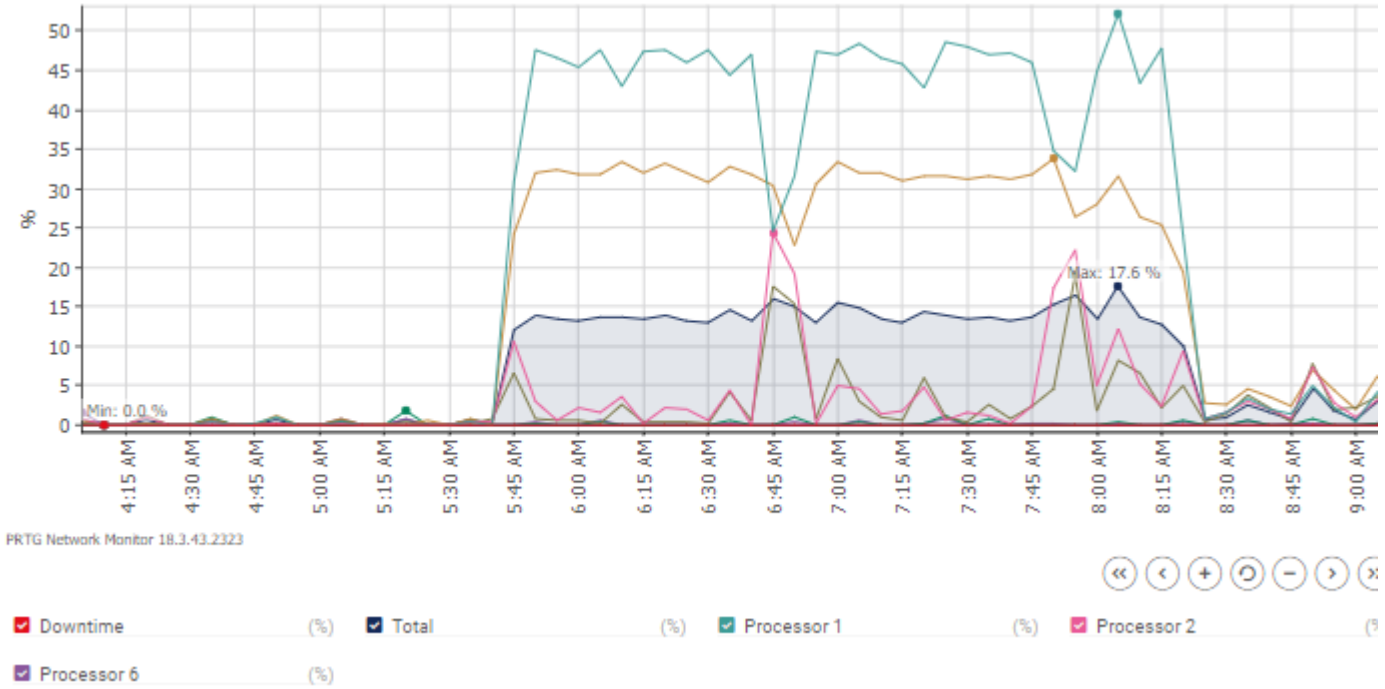
```
registration_id database_name table_name
--------------- ------------- ------------
50              SLODATA       SLO_OGLEASE

(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

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

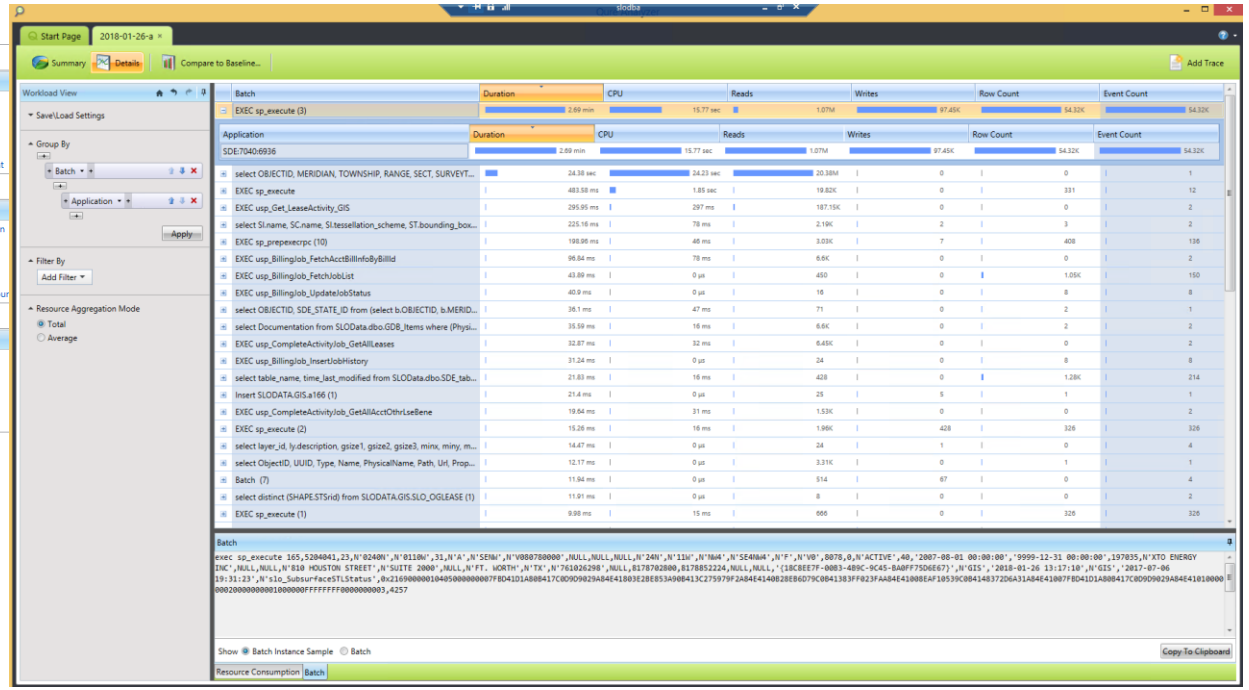
Resource	Top Consumer	Event Count	Total	Average	More Information
Duration	EXEC sp_execute (3)	54322	2.69 min	2.97 ms	Show Batches by Duration
CPU	select OBJECTID, MERIDIA	1	24.23 sec	24.23 sec	Show Batches by CPU
Reads	select OBJECTID, MERIDIA	1	20.38M	20.38M	Show Batches by Reads
Writes	EXEC sp_execute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

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

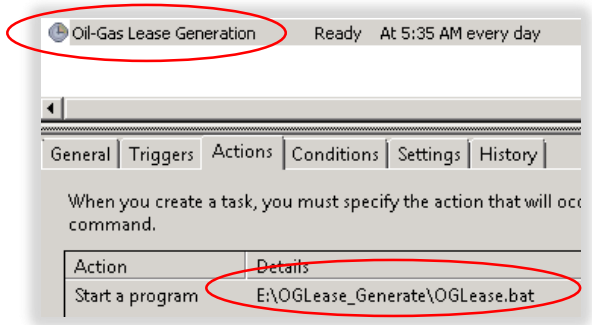


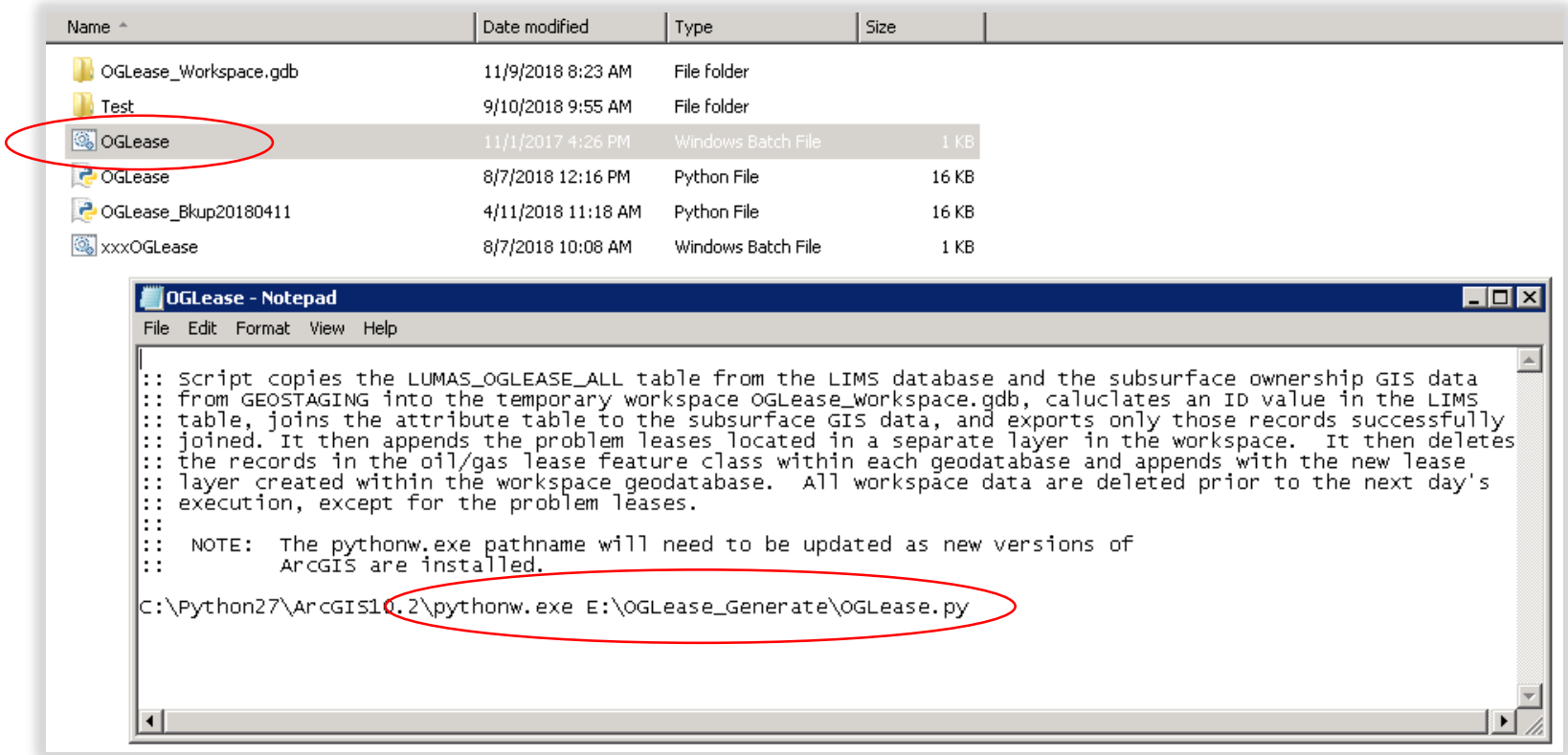
Some kind of cursor foo on a spatial table

```
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 l.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 l.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 l.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.

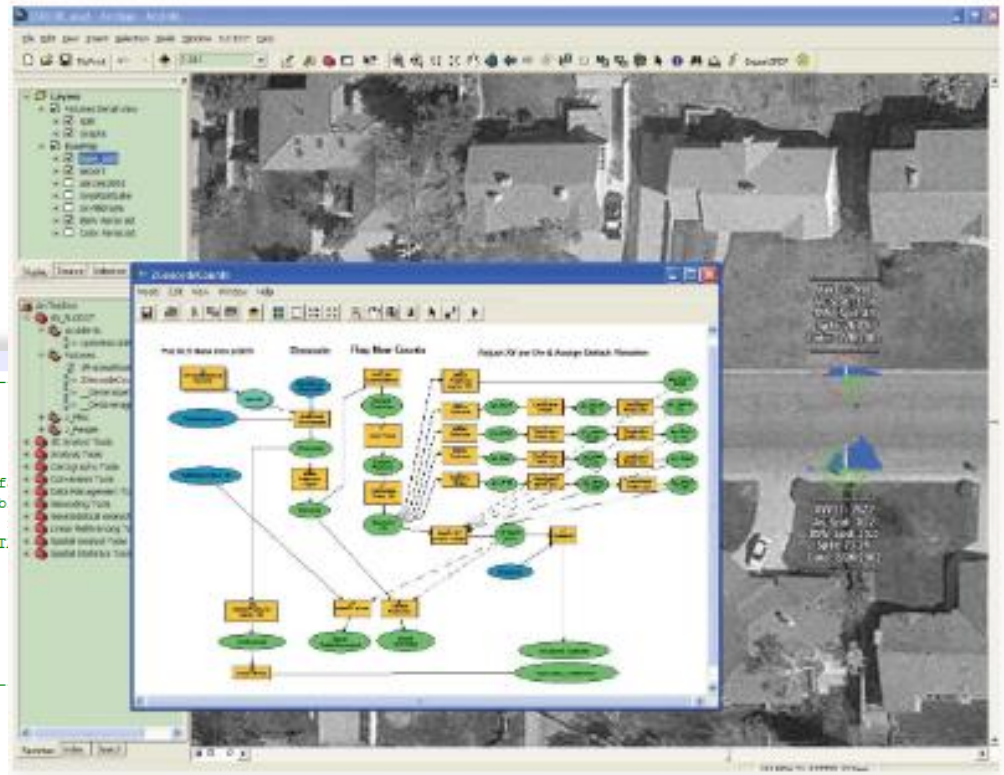
Colleague found the task and the *.bat file and the Python script call





ESRI Model Builder

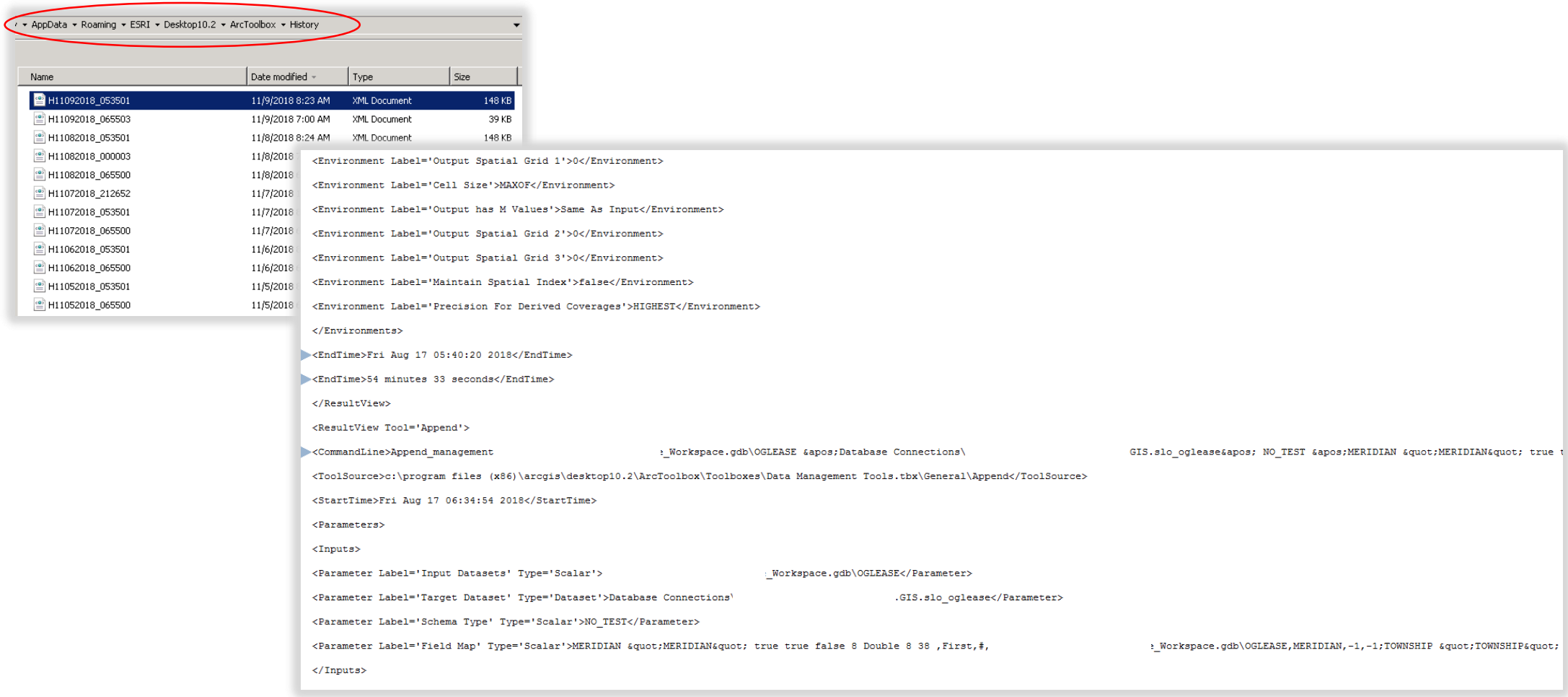
"it's like Microsoft SSIS for GIS professionals"
Result is a Python Script



```
# -*- 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;

# 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_PR

# 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



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_ , _____."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
```

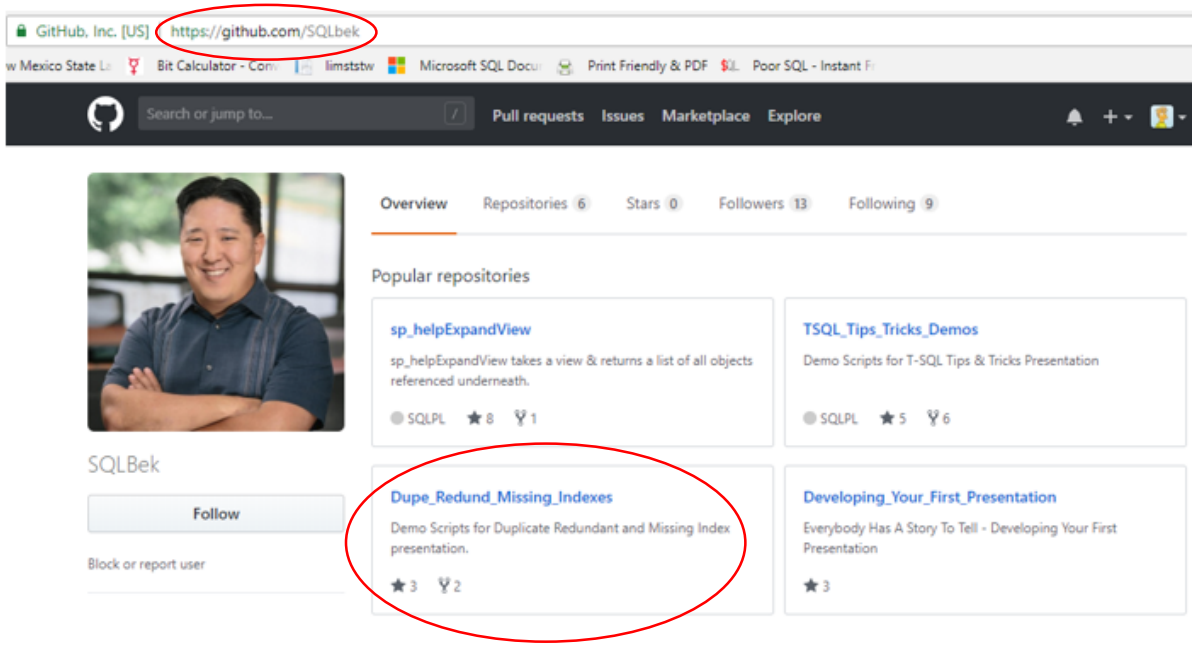
OMG! Gotta be the indexes. Double check index maintenance plans on the spatial table.

```
  FROM [DBAUtility].[dbo].[CommandLog]
  where ObjectName IN ('slo_oglease', 'a166', 'd166')
  order by StartTime DESC
------------------------*/
```

ID	DatabaseName	SchemaName	ObjectName	ObjectType	IndexName	IndexType	StatisticsName	PartitionNumber	ExtendedInfo	Command	CommandType	StartTime	EndTime	Erro
2694841	SLOData	gis	D166	U	NULL	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 20:34:54.250	0
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	0
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	0
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	0
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	0
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	0
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	0
2686284	SLOData	gis	D166	U	NULL	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	0
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	0
2686278	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	0
2686277	SLOData	gis	a166	U	UUID_OID_166_a	2	NULL	1	<ExtendedInfo><PageCount>95106</PageCount><Fragmentation>5.47284</Fragmentation></ExtendedInfo>	ALTER INDEX [UUID_OID_166_a] ON [SLOData].[gis].[a166] REORGANIZE WITH (LOB_COMPACTION = ON)	ALTER_INDEX	2018-10-10 20:30:07.460	2018-10-10 20:30:57.133	0
2686244	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	0
2686243	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 20:29:54.977	0
2686242	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	0
2686241	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	0
2682083	SLOData	gis	D166	U	NULL	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 20:29:49.540	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 20:29:36.597	2018-10-09 20:29:43.480	0
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 20:29:27.540	2018-10-09 20:29:34.237	0
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 20:29:09.957	2018-10-09 20:29:18.320	0
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	0
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	0
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 20:28:09.313	2018-10-09 20:28:45.580	0
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 20:28:07.410	0
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 20:26:58.643	2018-10-09 20:28:00.817	0
2677941	SLOData	gis	D166	U	NULL	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	0
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	0
2677938	SLOData	gis	D166	U	d166_pk	1	NULL	1	<ExtendedInfo><PageCount>77739</PageCount><Fragmentation>5.28949</Fragmentation></ExtendedInfo>	ALTER INDEX [d166_pk] ON [SLOData].[gis].[D166] REORGANIZE WITH (LOB_COMPACTION = ON)	ALTER_INDEX	2018-10-08 20:26:03.890	2018-10-08 20:26:25.470	0
2677934	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 20:25:47.127	2018-10-08 20:25:55.393	0
2677933	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 20:25:46.127	0
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 20:25:25.970	2018-10-08 20:25:35.403	0
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	0

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

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



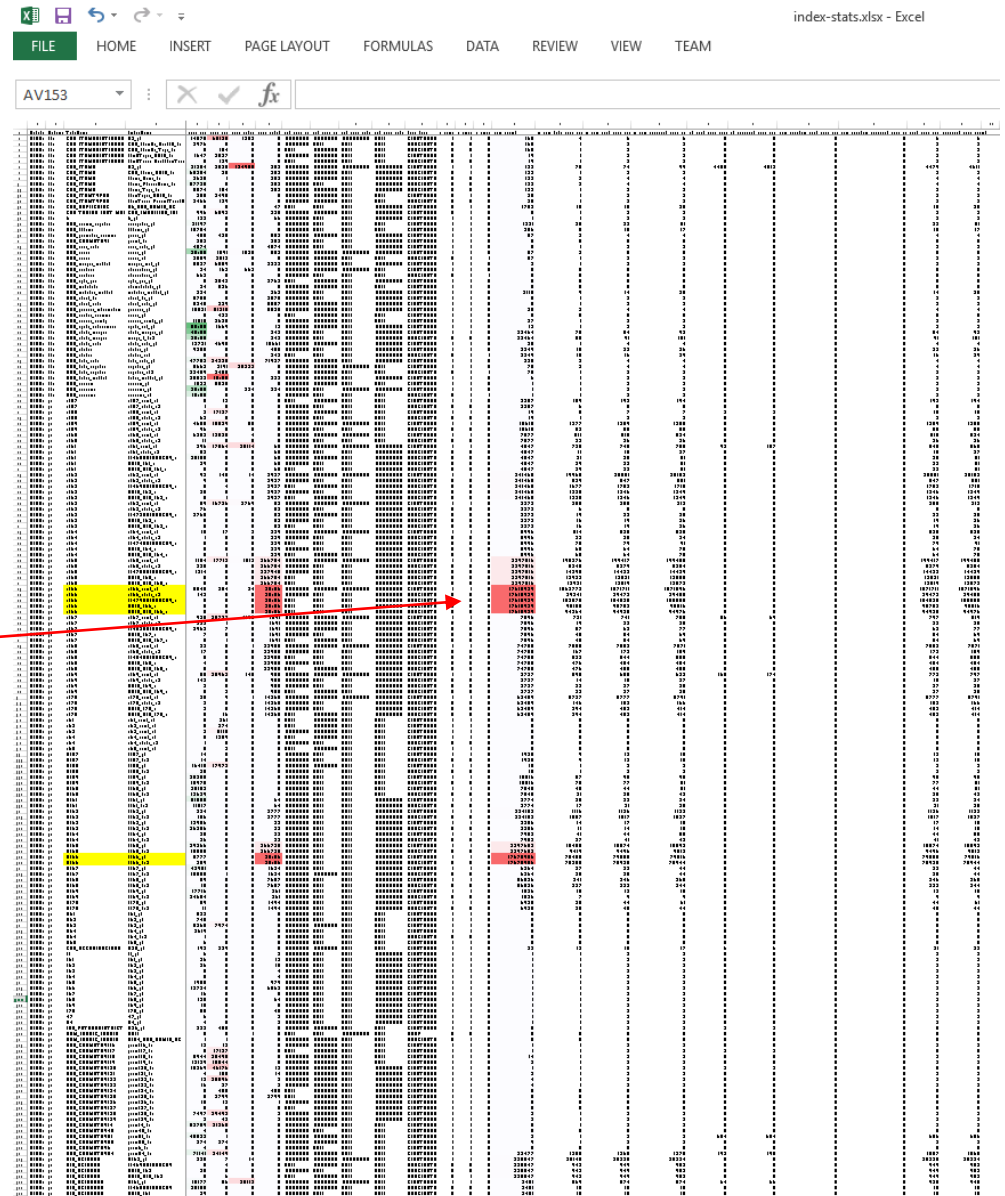
Yun's UsageStatsQuery – his "go to" query … result: it was index updates but only showed traditional indexes.

DatabaseN	SchemaNa	TableName	IndexName	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	6	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

```sql
-- 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

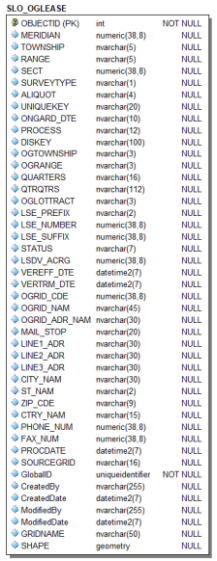
2.7 M user updates
On 17.6 M row_count



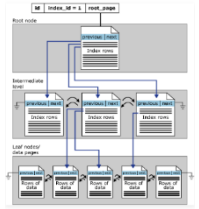
Complete, but scary picture of what was going on.

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

56,709 rows

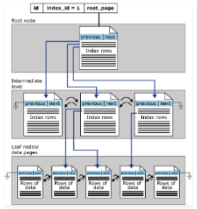


[R166_PK]
6364 pages,
56709 leaf



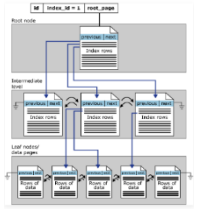
OBJECTID INT

[I1479UNIQUEKEY]
6364 pages,
56709 leaf



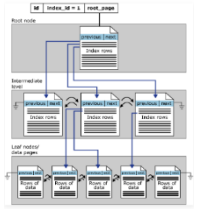
UNIQUEKEY NVARCHAR(20)

[UUID_166]
243 pages,
56709 leaf



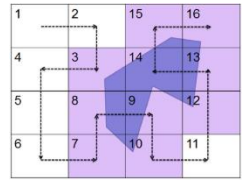
GLOBALID GUID

[UUID_OID_166]
243 pages,
56709 leaf



GLOBALID GUID
OBJECTID INT

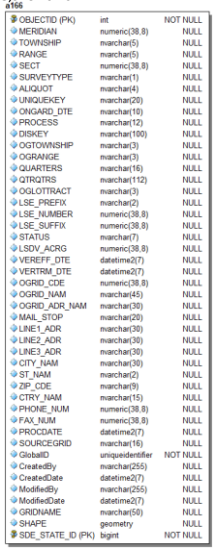
[S125_IDX] (Spatial)
2533 pages,
853,517 leaf



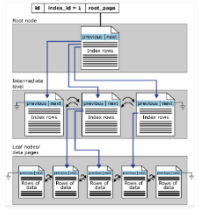
SHAPE GEOMETRY

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

17,618,929 rows

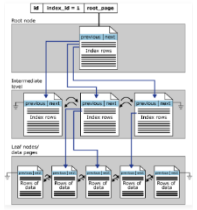


[a166_rowid_idx1]
1,562,773 pages,
17,613,700 leaf



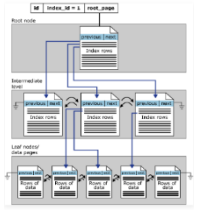
OBJECTID INT
SDESATE_ID BIGINT

[a166_state_ix2]
39,241 pages,
17,645,700 leaf



SDESATE_ID BIGINT

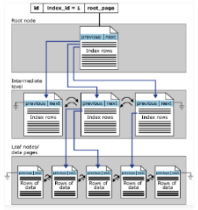
[I1479UNIQUEKEY_a]
103,875 pages,
17,609,200 leaf



UNIQUEKEY NVARCHAR(20)

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

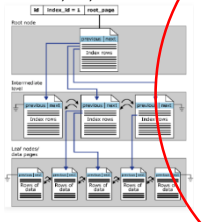
[UUID_166_a]
95,188 pages,
17,539,800 leaf



GLOBALID GUID

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

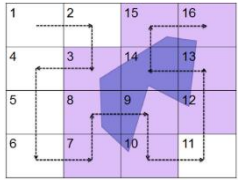
[UUID_OID_166_a]
94,364 pages,
17,712,200 leaf



GLOBALID GUID
OBJECTID INT

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

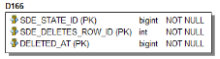
[SA125_IDX] (Spatial)
1,047,727 pages,
265,245,200 leaf



SHAPE GEOMETRY

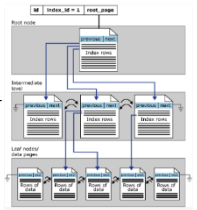
613 MB data
559 MB index
====
1174 MB

17,675,906 rows



[d166_pk]
78,408 pages,
17,777,000 leaf

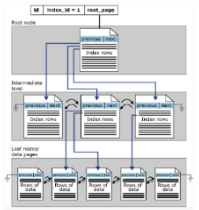
SDESATE_ID BIGINT
SDEDELETEDROWID INT
DELETED_AT BIGINT



(66) Index reorgs
(2) Index rebuilds

[d166_idx2]
70,388 pages,
17,591,600 leaf

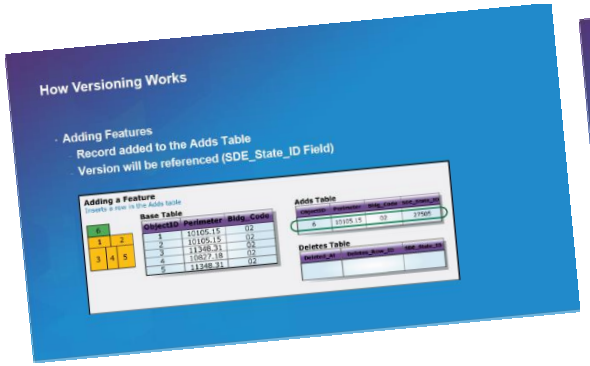
DELETED_AT BIGINT

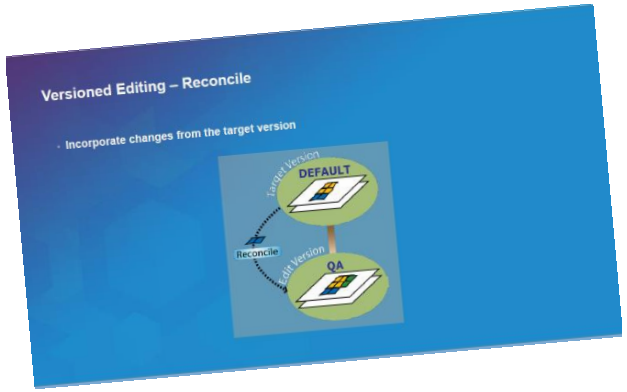


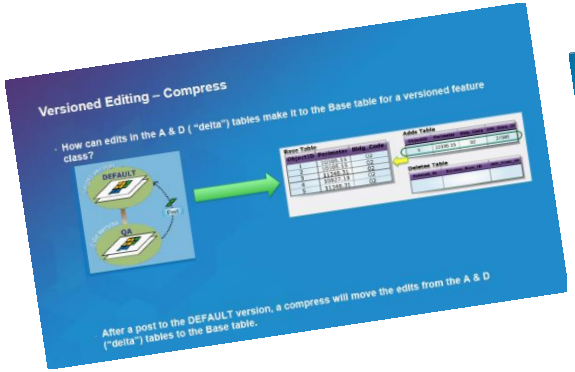
(66) Index reorgs
(1) Index rebuilds

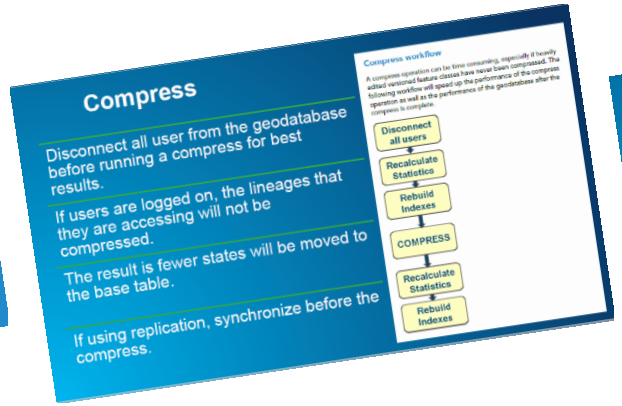
```
SELECT
   OBJECT_NAME(PARENT_OBJECT_ID) AS table_object,
   name,
   create_date,
   *
FROM sys.internal_tables it
WHERE
   INTERNAL_TYPE = 207
   AND it.name = 'extended_index_29295214_384000'

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.

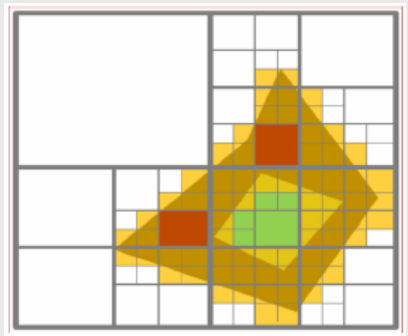


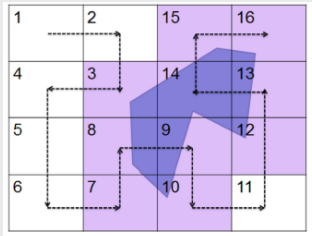
Figure-2: Grid decomposition of a polygon using 2x2 grids

Tuning Spatial Point Data Queries in SQL Server 2012

Microsoft® SQL Server®

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()

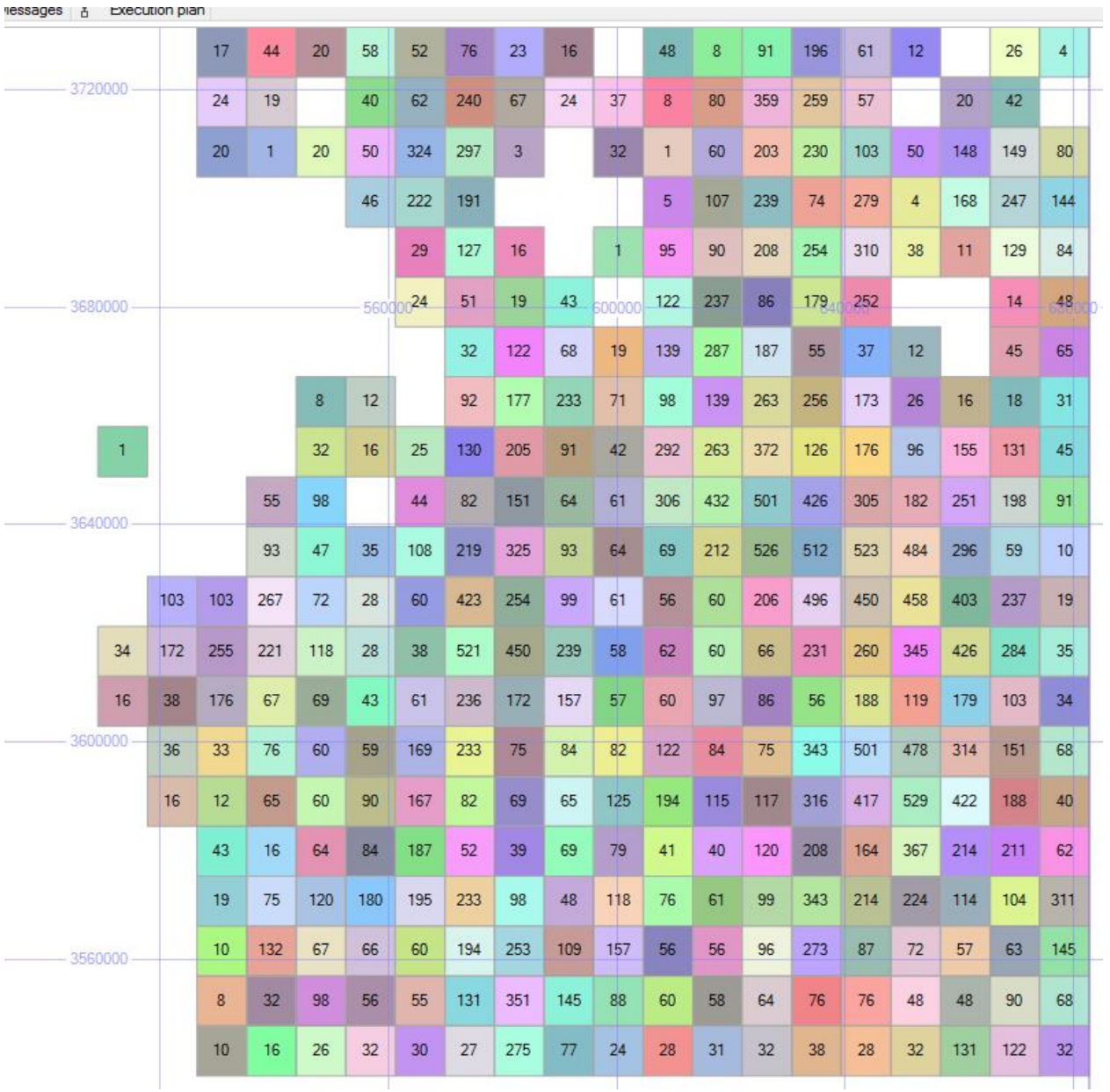
```
exec sp_help_spatial_geometry_histogram @tabname = "[ASLDData].[dbo].[ASLD_SURFACE_PARCELS]", @colname =
'shape',  @resolution = 64, @xmin = 144206, @ymin = 3466600, @xmax = 685016, @ymax = 4099054, @sample = 100;
```

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?

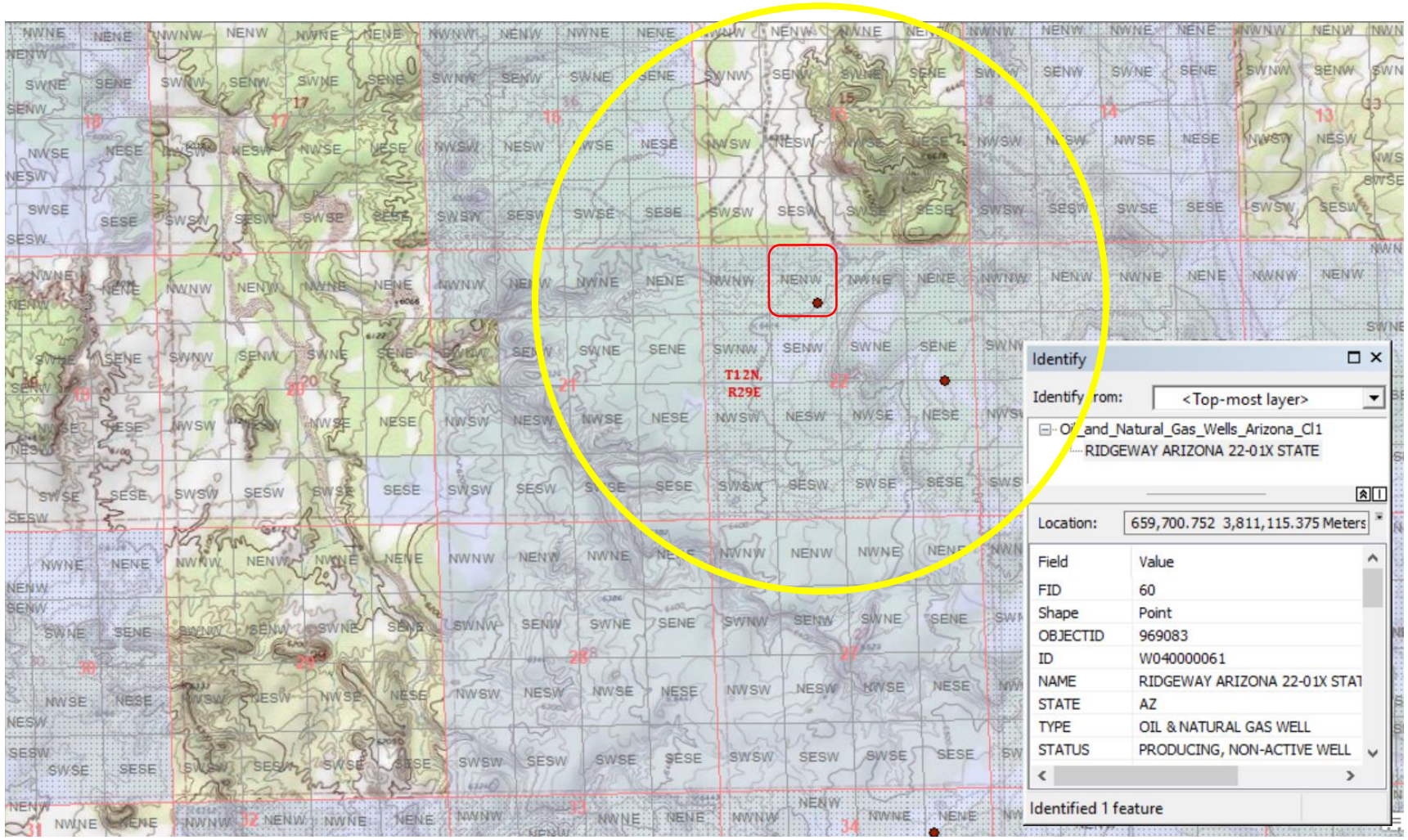
Diagnostic function #2: sp_help_spatial_geometry_index Whatif scenarios on a spatial index

```
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	NULL
Total_Number_Of_ObjectCells_In_Level0_In_Index	327	327	2	2
Total_Number_Of_ObjectCells_In_Level1_For_QuerySample	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
Total_Number_Of_Interior_ObjectCells_In_Level4_For_QuerySample	NULL		NULL	2
Total_Number_Of_Interior_ObjectCells_In_Level4_In_Index	60759124	60759124	195701	195701
Total_Number_Of_Intersecting_ObjectCells_In_Level1_For_QuerySample	28		28	485
Total_Number_Of_Intersecting_ObjectCells_In_Level3_In_Index	162574	162574	485	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	
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)	17618929	327	56709	9
Number_Of_Rows_Selected_By_Internal_Filter (by the index optimizations)	5178631	327	17635	1
Number_Of_Times_Secondary_Filter_Is_Called (the expensive operation)	12440298	0	39074	8
Number_Of_Rows_Output	0	0	0	9
Percentage_Of_Rows_NotSelected_By_Primary_Filter	0	99.9981440415589	0	99.9841295032535
Percentage_Of_Primary_Filter_Rows_Selected_By_Internal_Filter	29.3924278825347	100	31.0973566805974	11.1111111111111
Internal_Filter_Efficiency (by the index optimizations)	0.0	0.0	0.0	11.1111111111111
Primary_Filter_Efficiency	0	0	0	100

Demo

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



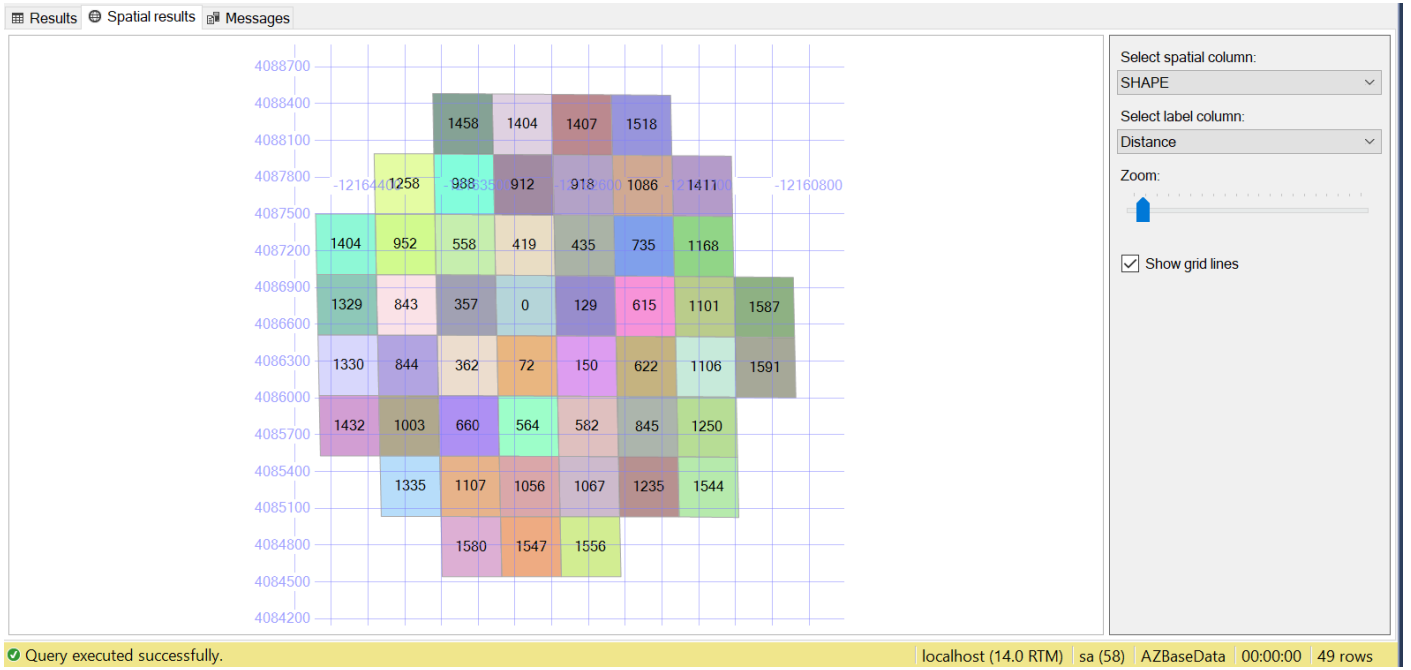
1. Select a geometry to tessellate with.

```
-- 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
```

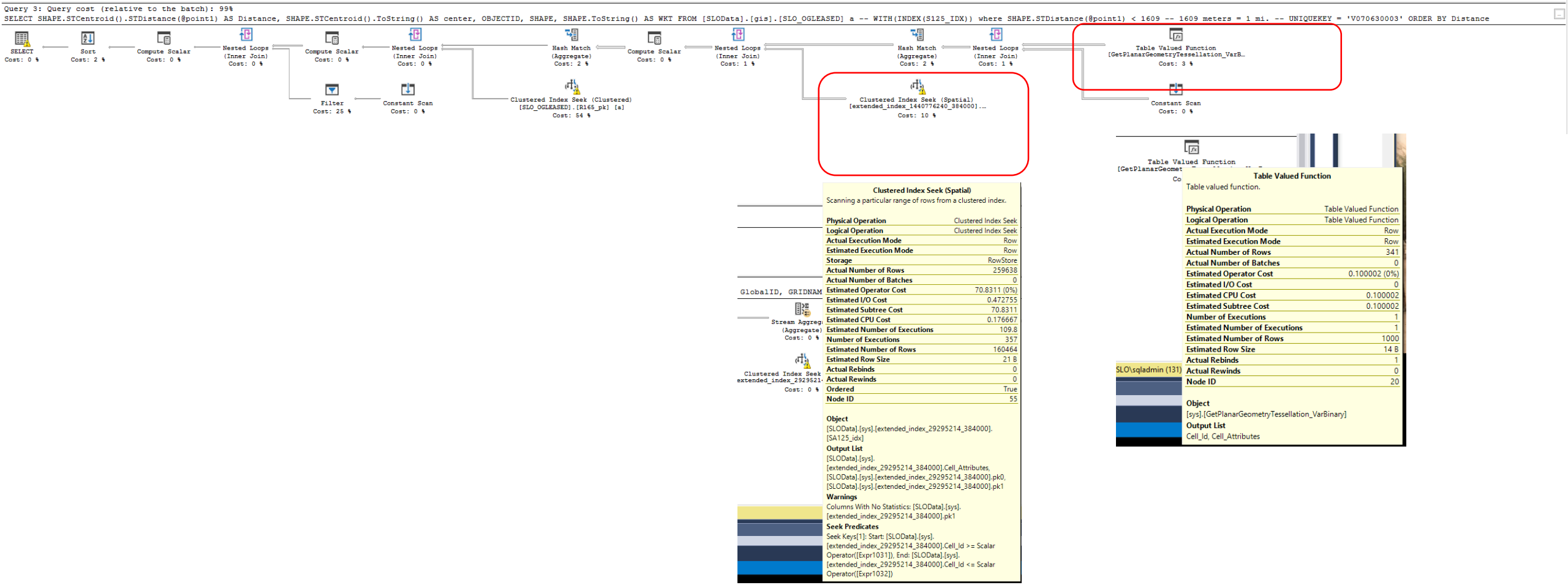
2. Use .STDistance() method in query.

```
-- find the PLSS subsections within a mile from the national blm dataset 24 million rows, 300M leaf nodes.
SELECT
    a.SHAPE,
    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

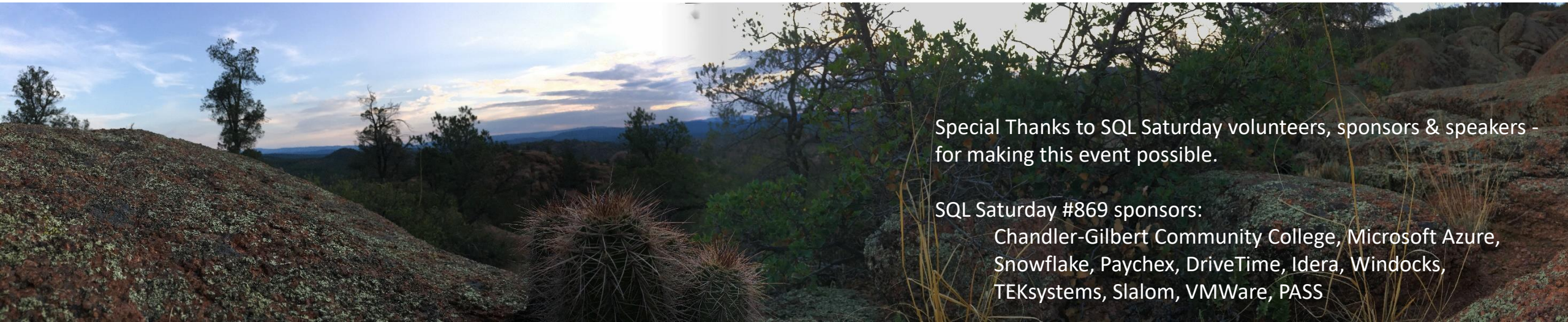
Jeff Iannucci

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