ArcSDE (Enterprise) Database Administration: Configuration, Storage, Tuning, Monitoring, Management, and Maintenance

Greg Tudor

Abstract

The Washington Department of Natural Resources (DNR) implemented SDE 3 (Spatial Database Engine) in 1997. Since that time, many changes and improvements have been made to the ArcSDE enterprise database and the Geodatabase (GDB) extensions. When did we last review our SDE configuration? As part of the DNR's GIS migration project, Database Administrators (DBAs) re-evaluated the ArcSDE/GDB environment focusing on configuration, storage, tuning, monitoring, management, and maintenance. DBAs evaluated and modified the operating system, Oracle database, and SDE configuration parameters. DBAs improved storage portability and revised the sdedbtune keyword definitions. Spatial reference definitions, spatial indexes, and spatial views were tuned. SQL views of common orphan SDE/GDB objects and monitoring scripts were developed to assist with identifying problems and tracking usage trends. Several developing ArcSDE management and productivity tools were reviewed. The ArcSDE evaluation findings, recommendations, and results of implementation revisions are presented.

Introduction

From the perspective of a Database Administrator, administration of the Spatial Database Engine and the Geodatabase extension for editing and more is problematic. The SDE is a template for storing spatial features in a relational database management system (RDBMS) that makes use of a mix of some but not all of the features of the database whether Oracle, SQL-Server, DB2, or Informix. The Geodatabase further complicates the picture by modifying the editing environment to support versioned editing. The Database Administrator is responsible for the integrity of the database and must understand the interaction between the SDE, the Geodatabase, and the database on which it is based. The DBA must provide his customers with the database consistency that they have come to expect from an enterprise RDBMS.

Background

Before relational database management systems, databases were free-form linkages between tables with database software providing some tools for using the data more efficiently than home-grown programs. Developers were still free to relate tables through any means using rules of their own. ESRI developed the coverages format to store spatial data in this environment with the INFO database. Coverages are integrated data sets that have geometric topology designed into the data structure. The integrated node-arc-poly-point structure makes operating on the data very complicated and ESRI moved away from this and began using a more object-oriented approach. Administering a coverage

database was fairly easy – make file system backups, and recover tables when users make errors or a storage disk failed. The primary difficulty of administering coverages is keeping track of the database design with data models.

Shapefiles are a first attempt at object oriented spatial features. They are very simple floating-point geometry features. The feature attributes are in the database file (dbf), the shapes are in the shape file (shp), and the features are indexed in the shape index (shx). There is no topology. ESRI did not build many tools off the shelf to edit or maintain geometric integrity, so users relied on ArcView extensions built by other users and distributed by ESRI for these purposes. Administration of shapefiles is similar to coverages except with fewer structural integrity requirements – just make sure that all files for a shapefile are present.

The Spatial Database Engine is the implementation of shapefiles in a more robust and powerful database management system. The file structure is almost the same; attributes are in the layer business table, the shapes are in the F table, and the spatial index in the S table. (The F and S tables are hidden by ArcCatalog and ArcMap.) ESRI improved query and display speeds with integer coordinates, feature envelopes, and spatial indexes. However, ESRI released SDE without editing or integrity tools, although some users did develop some of these tools as ArcView extensions. Straight SDE has been primarily used for its speed in data warehouse applications such as Internet Map Service. Administration of SDE requires database administration of the base relational database, such as Oracle, and administration of the SDE dictionary files to keep track of user SDE tables for layers and rasters.

The Geodatabase is an extension of SDE that is designed and optimized to work with the ArcMap software which includes functionality for editing and data integrity. The GDB is based on the same SDE shapefiles. The Geodatabase makes use of version control in its editing functionality by tracking changes from a base version of a feature. The database stores a base (B) version of a feature, along with any additions (A) or deletions (D) tracked by a state identifier. The current state of a feature is a combination of the base and any adds or deletes – an addition operation adds a record to the A table, a delete operation adds a record to the D table, and update operations add a record to both the D and A tables to delete current state and add the changes. The F, S, A, and D tables are hidden by ArcCatalog and ArcMap. The personal GDB is a little less functional than the enterprise GDB (ArcSDE), but feature datasets and feature classes can be copied back and forth without data loss or change of format. Versioned editing requires the database administrator to compress the geodatabase edits on a regular basis to simplify the states being tracked by the GDB, otherwise Geodatabase performance suffers.

Topology rules are a Geodatabase extension at ArcGIS 8.3. Unlike the structural topology built in coverages, Geodatabase topology is a set of business rules for business layers. Because the rules are potentially too complex to enforce with every update and users may not want results until complex transactions are complete, users validate topology on demand, and they may limit the area to be validated. ESRI software tracks areas that have been edited and only validates those areas needing validation; if an area

has already been validated then there is no need to validate it again. Validation identifies errors and offers options for fixing those errors. Users are further allowed to make exceptions to topology rules, accepting the error as correct. Topology is made possible by integer coordinate storage; feature locations either match exactly or they do not.

ArcSDE Geodatabase administration includes all of the SDE administration requirements plus administration of additional GDB dictionary files to keep track of user edit tables, topology, metadata, domains, feature datasets.

Configuration

The first task we undertook was to understand the configuration of our Oracle database and the ArcSDE Geodatabase. The SDE/GDB Configuration Guidelines describe the different levels at which ArcSDE can be tuned for better performance and organized for easier administration. These levels include operating system environment variables, Oracle database configuration parameters, ArcSDE connection parameters, and ArcSDE configuration parameters. We focused on the parameters that vary from defaults. Refer to the multi-page spreadsheet SDEGDBConfig.xls for more details; there is a separate sheet for each section below.

UNIX OS Environmental Variables
The environmental variables allow the ArcSDE software to run without full path invocations. These are set in /etc/profile for the esri_sde user on each server. The PATH and LD_LIBRARY_PATH allow appending new execution and library paths onto the current path as shown; more may be appended in the profile than shown in the next example. These variables do not need to be exported since a new OS shell is invoked which inherits the set values.

```
ESRIHOME=/solaris/esri
SDEHOME=${ESRIHOME}/sdeexe90_2
PATH=${PATH}:${SDEHOME}/bin
LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${SDEHOME}/lib
```

Some environmental variables are set for specific ArcSDE instances used for the DNR development, demonstration, and production environments. These are set in a Korn shell script /am/dnr/macros/bin/dnrenv. These must be exported to take effect in the current OS shell.

```
export SDESERVER=rodan
export SDEINSTANCE=esri_sde
```

After these are set, running dnrenv to set the environment will allow SDE admin commands to work without entering the -s server and -i instance options.

The Oracle environmental variables must also be set for ArcSDE to work. Some are set to allow the Oracle software to be run without full path invocations. These are similar to

other home, execution path, and library path variables, but TNS_ADMIN sets the location of the tnsnames.ora file which defines the server and path for each Oracle instance. PATH and LD_LIBRARY_PATH examples show only the appends which apply to Oracle. The values may vary from server to server

```
ORACLE_HOME=/solaris/oracle/product/9.2
PATH=${PATH}:${ORACLE_HOME}/bin
LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${ORACLE_HOME}/lib
TNS_ADMIN=${ORACLE_HOME}/network/admin
```

Some Oracle environmental variables can be reset to specific Oracle instances using script macros as well.

```
. oraenv
```

Note: the dnrenv and oraenv scripts must be dotted to take effect in the current shell.

The SDEUSER is not set since there is no standard user for most situations, and SDEPASSWORD is not set because of the security risk.

INIT.ORA

The init.ora file contains parameters for setting up the Oracle database instance on which the ArcSDE instance is founded. These parameters can strongly influence the performance of ArcSDE, however changing the Oracle parameters influences all of the other applications using that Oracle instance. The Oracle parameters should be set to balance the needs of all applications.

ESRI recommends setting some of the Oracle parameters differently.

- cursor_space_for_time allocates more storage for cursors to save time.
- db_block_size sets the block size used for tablespaces (currently 8K for transactional, 32K for read only).
- open_cursors sets a limit on the number of cursors that can be open.
- optimizer_index_caching and optimizer_index_cost_adj work together to set the weighting on using indexes and discounting the penalty for using indexes.
- optimizer_mode.
- prepage_sga to true – no explanation from ESRI.
- session_cached_cursors sets a limit on the number of cursors that can be cached per session.
- sort_area_size for manual extent management, but this is not required for auto extend.

The recommended changes to be made in the init.ora file include setting open_cursors to 2000, and session_cached_cursors to 50. (In esri_sde, open_cursors is set to 20,000 for the Faster system.)

services.sde

The services.sde file contains the port numbers for individual ArcSDE instances from within each SDE directory. The system /etc/services file must also include a matching port designation for each esri_sde instance. This must be reset if the port designation changes.

dbinit.sde
The dbinit.sde file contains the parameters for the Oracle database instance upon which the SDE instance is based. The esri_sde SDE instance is based on the Oracle SDE instance, esri_read_only based on Oracle READ_ONLY, etc. ORACLE_HOME and ORACLE_SID are set, and TWO_TASK is unset.

SDE.SERVER_CONFIG or giomgr.def
The SDE.SERVER_CONFIG table and the giomgr.def file contain the parameters used for initializing the ArcSDE instance. Nearly all of the parameters have defaults that are used when starting the instance. The giomgr.def sets parameters only at startup. The SDE.SERVER_CONFIG parameters can be changed interactively and using sdeconfig –o import referencing the giomgr.def file.

> sdeconfig –o import –f /sde/esri_sde/etc/giomgr.def -i esri_sde –s panhead –u sde –p password

The SDE default parameters are generally used, but some have been changed. MAXTIMEDIFF is unset to -1 to reduce timeout. STREAMPOOLSIZE is boosted to 6 at ESRI's recommendation, a new default not included in upgrades. TCPKEEPALIVE is set to 1 (TRUE) to reduce extra 3 tier connections. TLMINTERVAL may be increased for static data model. DISABLEAUTOREG is unset to 0 since DNR is not using Oracle Spatial.

Over the course of ArcSDE upgrades (8.0.2, 8.1, and 9.0) many parameters are now obsolete. (See the SDEGDBConfig.xls, SERVER_CONFIG sheet). These changes reduced limitations on memory management, locks, and number of layers/rasters.

There is one parameter PAGESIZE used only for MS-SQLServer.

SDE Logfiles
SDE logfiles have been a significant issue for users selecting data in ArcMap sessions. Many problems with SDE logfiles were recognized by ESRI and new alternatives were offered in ArcGIS 9. This reviews the discussion of the alternatives presented at the ArcGIS Enterprise Systems Performance and Scalability training and from other sources.

Shared Logfile Tables
All selections for each user in all of their sessions are written to that user's SDE_LOGFILE_DATA tables. As selected features are unselected, they are deleted from the logfile table. Crashed sessions are never deleted, and the logfile grows without manual cleanup. The user must have CREATE_TABLE and CREATE_INDEX privileges and quota on the USERS and USERS_I tablespaces.

SERVER_CONFIG settings are:

MAXSTANDALONELOGS	0
ALLOWSESSIONLOGFILES	FALSE
LOGFILEPOOLSIZE	0

Session Logfile Tables

An SDE_SESSION table is created for each session. All selections for each user session are written to the session logfile. When all selections are dropped then the logfile is truncated instead of using the delete from process. When the session is ended then the logfile is dropped. Partial selection drops or reselection use delete from processing. Crashed sessions are not deleted, truncated, or dropped, and must be manually cleaned up. Users must have CREATE_TABLE and CREATE_INDEX privileges and quota on the USERS and USERS_I tablespaces.

SERVER_CONFIG settings are:

MAXSTANDALONELOGS	0
ALLOWSESSIONLOGFILES	TRUE
LOGFILEPOOLSIZE	0

Pooled Session Logfile Tables

Pooled session logfiles are similar to session logfiles except that the logfile tables are owned by SDE. Since users may still make inappropriate selections and reduce system performance, this does not seem like a good idea. Only a limited number of pooled logfiles are allowed to reduce the risk to SDE performance. There may be more administration overhead keeping track of the logfile pool size shared by the SDE user. This approach uses the truncate process to unselect features, but does not require users to have system privileges to create tables or indexes. When the session ends, logfiles are not dropped, but simply truncated and returned to the SDE pool for reuse.

SERVER_CONFIG settings are:

MAXSTANDALONELOGS	0
ALLOWSESSIONLOGFILES	TRUE
LOGFILEPOOLSIZE	<N>

Standalone Logfile Tables

An SDE_LOGDATA table is created for each layer in the user's session. All selections for each layer are written to the session logfile. When selections on a layer are dropped then the logfile is truncated instead of using the delete from process, so users could select from multiple layers simultaneously without significant performance losses. When the session is ended then the logfiles are dropped. Crashed sessions are not deleted, truncated, or dropped, and must be manually cleaned up. Users must have CREATE_TABLE and CREATE_INDEX privileges and quota on the USERS and USERS_I tablespaces.

SERVER_CONFIG settings are:

MAXSTANDALONELOGS	<N>
ALLOWSESSIONLOGFILES	FALSE
LOGFILEPOOLSIZE	0

The Shared Logfile Tables alternative seems best suited to the SDE editing environment. Editors are not likely to make large selections, and are likely to reuse SQL in repeated operations. Editors should be aware that they would experience some contention if they log into multiple sessions that share the same logfile. They should also be aware that large selections should not be made in this environment since the cleanup from selections is very slow using the delete from process instead of truncation.

The Session Logfile Tables alternative seems best suited to the esri_read_only query and display environments. There are many IMS applications using single user login ids and each should have its own session-based logfile. Users are likely to make large selections to create new data sets or develop complex queries. The truncation process to unselect features is much faster than the delete from process and is well suited to these environments. Users should be aware that if they have multiple selection sets (selections on multiple feature classes) that the delete from process will apply and may be quite slow. By unselecting all features they can truncate the logfile and formulate the query as a whole with better performance. If there are many users then there is some overhead to the create table and drop table operations each time sessions begin and end.

Logfiles from crashed sessions need to be cleaned up periodically. Therefore, the shared logfiles of users who are not logged in should be truncated nightly, and the session logfiles of users who are not logged in should be dropped (SDE_SESSION_1234), delete the index row from the SDE_LOGFILES, and rebuild the index.

Storage

The next item that we focused on in our system review was storage tablespace allocation. SDE/GDB Storage Guidelines are intended to document best practices for DBAs assigning storage for ArcSDE data.

The dbtune.sde file is the core of the ESRI storage definition. The file contains the definition of the configuration keywords used by ArcCatalog to create data in specific tablespaces. Users creating data must use the dbtune.sde keywords to have data stored in their assigned tablespaces. The configuration keyword parameteres also set the column format definitions for binary data. ESRI provides a template of the dbtune.sde file with default parameter settings. The ESRI default dbtune.sde file is located in the $SDEHOME/etc directory. Each instance must have a dbtune.sde file under the instance directory, e.g. /sde/esri_sde/etc. Examples of the dbtune.sde configuration keywords and storage parameters are in the SDEGDBStorage.xls spreadsheet. On starting a new instance a blank dbtune.sde file is created which is smaller than the ESRI default. The ESRI default dbtune.sde file gives essential parameters for each keyword. From its DEFAULTS keyword, new keywords can be created to store static vector, versioned vector, and raster tablespaces for specific users or special project raster data.

The storage parameters can be very specific when defining the tablespace, but setting up the tablespace in Oracle is a better practice. So the tablespace name is all that is needed to associate layers and tables with tablespaces for storage needed by the schema owner. The tablespace definition is not needed if the tablespace has been set up in advance. For example:

B_INDEX_SHAPE	"PCTFREE 0 INITRANS 4 TABLESPACE USER_I"

rather than

B_INDEX_SHAPE	"PCTFREE 0 INITRANS 4 STORAGE (FREELISTS 4 INITIAL 6M NEXT 6M MINEXTENTS 1 MAXEXTENTS 200 PCTINCREASE 0) NOLOGGING TABLESPACE USER_I"

Vector Storage

For transportable tablespaces, binary fields should be set up to use binary large object (BLOB) storage format. Set the binary parameters below:

ATTRIBUTE_BINARY	"BLOB"
GEOMETRY_STORAGE	"SDELOB"
RASTER_BINARY_TYPE	"BLOB"
XML_DOC_TEXT_TYPE	"BLOB"

ATTRIBUTE_BINARY should be set for all keywords having attribute tables, especially B_ tables. GEOMETRY_STORAGE should be set for all keywords having geometry tables, e.g. F_. RASTER_BINARY_TYPE should be set for all keywords having RAS_ tablespace parameters, and XML_DOC_TEXT_TYPE should be set for any keywords having XML_DOC_ tablespace parameters.

Versioned Storage

Any GDB tables that are versioned for editing need storage parameters for A and D tables. The A and D table storage parameters are not required for loaded, static, or unversioned datasets such as the READ_ONLY data warehouse or raster data/images.

Raster Storage

The raster tablespaces must be specifically designated, or the AUX, BLK, BND, and RAS tables will end up stored in the default tablespaces for USER. The B, F, and S storage parameters are necessary for storing the raster footprints visible when zoomed out. These should be put in the same raster tablespaces to keep the data together for portability.

Raster datasets are each very large (10-650 GB) and static; they do not need to be backed up very often. Raster data should be owned by a separate schema owner from the transactional vector data. Having a different owner allows the export backups to be segregated by owner rather than by individual table. For example, the READ_ONLY schema owner needs data to be backed up weekly, the RASTER schema owner only needs a backup when the raster data stored under the schema is updated, and then only a few tables may need to be updated. Separating transactional and raster data into different schemas reduces the amount of overall time spent backing up data in general and raster in particular.

Another alternative is to store rasters in a difference database instance to allow even more flexibility for backups without interfering with existing backup systems.

For transportable tablespaces, set RASTER_BINARY_TYPE to BLOB.

DBTUNE Cleanup
DBAs created templates with a complete set of the minimum parameters in the /sde/esri_read_only/etc directory on panhead. dbtune_dnr_template.sde is generic for most environments. dbtune_dnr_read_only_template.sde is generic for the esri_read_only environments. From these templates, the existing dbtune.sde files were expanded to include complete parameter sets for each configuration keyword and specific binary storage settings. Configuration keywords were simplified to follow the schema owner name, rather than schema.default or schema.alt; no differences in the tablespace definitions were found for these keyword differences. Orphan views of keywords, and layer keywords, table keywords, raster keywords, and xml_keywords helped find unused keywords and keyword references that had changed. Keyword references in LAYERS and TABLE_REGISTRY were changed to the current keyword names. All dbtune files revised for this cleanup are dated dbtune.sde.20051020.

Tuning

Spatial Reference Tuning
The spatial reference includes the parameters for storing coordinates as integers in the ArcSDE database. Since there is a limited range of values in the integer coordinate space, the range of real world coordinates must be offset and scaled to fit. For a given location vicinity and location accuracy there may be much leeway for a fitting the integer coordinate space.

There are two types of tuning to improve performance in the spatial reference: the X/Y, Z, and M location domains, and the X/Y, Z, and M precision. Of these the precision tuning has the greatest effect. When comparing a Z-enabled point feature class that has a precision of 1000 (0.001 ft) with the same feature class loaded with a precision of 1 (1 ft), the average row length is reduced from 110 Bytes to 70 Bytes. This tuning has the greatest effect on full extent display, reducing the display time by 1/3. The larger the scale the less this tuning has a noticeable effect. So, geoprocessing or displaying data for large areas takes significantly less time with less precision by default.

Tuning the Z and M precisions does not appear to affect the row length unless the Z and M domains are enabled. Two Z-disabled point layers with Z precision of 1000 versus a Z precision of 1 have essentially equal row lengths. A Z-enabled point layer with an X/Y and Z precision of 1000 (0.001 ft) has a row length of 110 Bytes; a point layer with Z disabled and an X/Y precision of 1000 has a row length of 107 Bytes.

The location domains are tuned by fitting the coordinate range of the features as closely as possible to the lower end of the integer machine coordinates. This limits the creation of spatial index grid cells having no features which are searched before finding grid cells with features. This tuning primarily improves the performance of creating and recreating the spatial index grid (see spatial index tuning below).

The spatial reference template was created to help users create ArcSDE layers tuned to an appropriate spatial reference. Users have the choice of using the spatial reference template which maximizes performance for features in Washington State, or using the Cadastral Spatial Reference which maximizes performance for 0.001 foot precision. (See SDESpatialReferenceTemplate.doc for more information about the template's spatial reference parameters and using the template.)

Spatial reference tuning may become obsolete with high precision data storage in ArcGIS 9.2.

Spatial Index Tuning

The spatial index is an index of the features that are within a regular grid of each feature class. All feature classes must have a spatial index grid defined. The map tips tool will not work without a spatial index grid defined; this creates problems for spatial views which may or may not have a spatial grid defined. The optimal size of the spatial grid is determined from iteratively defining the grid size and then checking the spatial index statistics. The spatial index grid should be large enough to fit hundreds of features, but small enough to fit the largest features in 2 or 3 cells. The best tuning for grid size will have a grids/feature ratio of about 1.12. A good rule of thumb for a first iteration is to at least double the spatial index calculated by ArcGIS when creating the layer; round to the nearest 10,000 to help distinguish tuned spatial indexes from ArcGIS calculated indexes. The desc_sde_layers.ksh script will describe and calculate the spatial index statistics for every layer in every database instance. After reviewing these, use the sdelayer –o si_stats command to help tune each layer individually. Additional iterations of setting the grid size will bring the grids/feature ratio to 1.12. Points always have a grids/feature ratio of 1.00; so for points adjust the grid size to get about 200 features per grid for best performance.

Spatial View Tuning

Spatial views are used for joining a single feature class with attribute tables, and redefining column names. The more attribute tables and the more attribute columns that are included in the spatial view the worse the spatial view's performance gets. No more than one or two attribute tables are recommended, and no more than 40-60 attributes

columns are recommended. Joining with a single materialized view joining several tables is preferable to joining directly with the feature class. In all cases, index the columns used for relating attribute tables; only ObjectID and Shape are already indexed by ArcSDE. Make sure that the first table referenced in the SQL view's "FROM" statement is the spatial feature class; do not reference an attribute table before the feature class. Although more than one feature class can be referenced by a spatial view, only one will function spatially, joining additional layers can only be used for joining attributes and cannot take the place of overlay analysis.

Spatial Feature Tuning for Static Layers

The access to spatial features can be tuned by exporting the features in feature order, (truncating the data), and then importing the features back into ArcSDE. The effect is to group features together by geographic location so that they are accessed more quickly by location. This sorting by location should not have any effect on tabular queries. The greatest effect is on layers with many features. However, the export and import operations effectively triple the amount of time needed to transport the layer from one environment to another, so it is not recommended for layers which are regularly updated. It is useful primarily for large datasets that are static. Export layer data in spatial order using the sdeexport –O option.

> sdeexport –o create –l layer,shape –O –a all –f exportfile –i esri_read_only –s panhead –u sde –p password

Raster Tuning

Load raster data into owner schemas designated for rasters, e.g. READ_ONLY_raster, ger_raster, to limit regular exports of static data. Load large raster datasets into their own dedicated tablespaces, e.g. elev_lidar, ger_mine_reg. Set the cell size to no less than the resolution of the imagery to save storage space.

Raster Loading Tuning

Load the raster from uncompressed sources; this eliminates uncompressing as part of the load process. Determine whether the raster pixel values must be preserved as source data (lossless LZ77 compression) or may just be saved as images (lossy JPEG2000 compression). Test different compression qualities for lossy compression to assure reasonable picture quality with the most compression. Use nearest neighbor for pyramids of lossless rasters; bilinear for pyramids of lossy rasters. Estimate the total size requirements based on test results with the pyramid.

Monitoring

The SDE/GDB environment must be monitored just as the Oracle database and UNIX and Network file systems are monitored. Monitoring helps identify trends in data usage and database problems before they become serious. The SDE/GDB environment has some different situations that need to be watched and logged. For example, the SDE environment can become corrupted by non-ESRI applications, such as modifying, dropping, or editing SDE registered tables in Oracle. A table dropped in Oracle will still

be listed in ArcCatalog, but the table cannot be previewed, used, or deleted. Then, the orphan SDE data dictionary reference(s) must be cleaned up by a DBA.

Usage/Trend Monitoring

ESRI's ArcSDE Monitor
ArcSDE Monitor is a GIU SDE administration tool that can be downloaded from ESRI. The tool is still in an alpha development stage. Some of the tool buttons are just place holders for future functionality. The main drawback of the tool is that each instance must be reconnected every time the tool is opened. Otherwise, the graphical aspect of the interface could be somewhat useful.

SDE Connections
ArcIMS applications are constantly accessing SDE data. Determine the trend in application connections to each SDE instance to ensure that the correct production data is being accessed.

Layer Description, Statistics and Spatial Index Statistics
Layer descriptions provide a week-to-week baseline of the current layers in each SDE instance and their spatial index tuning. The scripts desc_sde_layers.ksh and desc sde_rasters.ksh find each feature class and raster dataset in the SDE dictionary.

sdelayer -o describe_long
sdelayer -o stats
sdelayer -o si_stats

sderaster -o describe
sderaster -o list -verbose -storage

Miscellaneous SDE Views
A set of SDE views was created to help make SDE administration easier and more consistent. Maintaining domains is problematic. We use a view of domains and domain usages to determine the domain owners and the owners, tables, and columns that use each domain so that the dependencies can be temporarily disabled and enabled. ESRI has suggested that the sde user be the owner of domains to simplify this further. However, we feel that the Data Stewards should maintain their own domains to the extent possible, and are reluctant to make the SDE owner responsible for making content changes. A list of tables registered with SDE but hidden from view by ArcCatalog or ArcMap is useful for understanding what is in the SDE dictionary but not easily administered. A list of tables versioned in the GDB for editing helps the DBAs understand the current editing environment. A view of feature class locks helps users maintain their own tables by identifying other users who have locked those tables. Other examples are below:

SDE.DOMAINS_V
SDE.DOMAIN_USAGES_V
SDE.FEATUREDATASET_OBJECTS_V

SDE.GISUSER_FC_LOCKS_V
SDE.HIDDEN_TABLES_V
SDE.LAYER_NULL_ROWID_V
SDE.METADATA_V
SDE.MISALIGNED_COLUMNS_V
SDE.MISSING_METADATA_V
SDE.MULTIVERSION_VIEWS_V
SDE.NETWORK_LAYERS_V
SDE.SPATIAL_VIEWS_V
SDE.SPAT_REF_TMPLT_FC_V
SDE.SPAT_REF_TMPLT_FD_V
SDE.SPAT_REF_TMPLT_OBJECTS_V
SDE.TABLES_NO_ROWID_V
SDE.TOPOLOGY_LAYERS_V
SDE.UNCOMPRESSED_TABLES_V
SDE.UNREGISTERED_TABLES_V
SDE.VERSIONED_TABLES_V

Problem Detection

ESRI's ArcSDE Check Schema
The Windows ArcSDE package comes with a diagnostics tool called checkschema.exe that will verify that the SDE schema is intact in any environment, including Oracle/SDE. The tool comes with instructions, but it should be noted that the Oracle service should be entered into the "Server:" input box for the ODBC connection.

The ArcSDE application must be installed on Windows to get the check schema tool installed. The tool and its documentation install under the C:\ArcGIS\ArcSDE\ora9iexe\tools\generic directory.

The tool is simple to run, just double click on the executable. The tool is a screen with four tabs in the order of the process to check the SDE schema.

On the Connect to Server tab click Build ODBC Connection String. Select a Driver for accessing Oracle. Either of the Oracle ODBC (open database connectivity) drivers must be installed completely for the tool to work – Microsoft ODBC for Oracle, or Oracle in ORA9x_CLIENT. Enter the Oracle service (SID from the TNS file) for the Server. The User Name will be entered as SDE. Enter the Password for the SDE user and click OK. Click Connect.

On the Select Server Type tab, select the DBMS Type as Oracle. Then click Query DBMS to get the current ArcSDE Version.

On the Run Tests tab click Run. If there are errors in the SDE schema then they will show up on the screen, but are printed so small that the report should be created to view them.

On the Create Report tab, click Write Test Results to Text File. Save the results to a file under a file named after the server and instance such as CheckSchema_rodan_sde.txt.

There are 16 tests that Check Schema performs against the SDE schema instance.

1. Is database running (Environment)
2. LAYERS table existence (Missing system tables)
3. TABLE_REGISTRY.CONFIG_KEYWORD in DBTUNE (Referential Integrity)
4. TABLE_REGISTRY entries exist as tables or views (Referential Integrity)
5. LAYERS in TABLE_REGISTRY (Referential Integrity)
6. LAYERS in GEOMETRY_COLUMNS (Referential Integrity)
7. LAYERS.SRID in SPATIAL_REFERENCES (Referential Integrity)
8. LAYERS.LAYER_CONFIG in DBTUNE (Referential Integrity)
9. LAYERS entries exist as tables or views (Referential Integrity)
10. GEOMETRY_COLUMNS in LAYERS (Referential Integrity)
11. RASTER_COLUMNS in TABLE_REGISTRY (Referential Integrity)
12. RASTER_COLUMNS.SRID in SPATIAL_REFERENCES (Referential Integrity)
13. RASTER_COLUMNS.CONFIG_KEYWORD in DBTUNE (Referential Integrity)
14. RASTER_COLUMNS entries exist as tables or views (Referential Integrity)
15. GEOMETRY_COLUMNS entries exist as tables or views (Referential Integrity)
16. ArcSDE for Oracle Compressed Binary format complete (Index Analysis)

Most are referential integrity checks to be sure that there are matches between related tables in the SDE schema. There are also checks to be sure that all of the SDE tables exist, that the environment is complete, and that indexes are complete.

The Check Schema test battery is equivalent to several individual checks already commonly used at the DNR to validate the SDE schema.

1. sdemon –o info –I config
2. desc SDE.LAYERS
3. SDE.ORPHAN_TABLE_CONFIG_KEYWORD_V;
4. SDE.ORPHAN_TABLE_REGISTRY_V;
5. SDE.ORPHAN_LAYER_TABLE_REGISTRY_V;
6. SDE.ORPHAN_LAYER_GEOMETRY_COLS_V;
7. SDE.ORPHAN_LAYER_SPATIAL_REF_V;
8. SDE.ORPHAN_LAYER_CONFIG_KEYWORD_V;
9. SDE.ORPHAN_LAYERS_V;
10. SDE.ORPHAN_GEOMETRY_COL_LAYERS_V;
11. SDE.ORPHAN_RASTER_TABLE_REGISTRY_V
12. SDE.ORPHAN_RASTER_SPATIAL_REF_V;
13. SDE.ORPHAN_RASTER_CONFIG_KEYWORD_V;
14. SDE.ORPHAN_RASTERS_V
15. SDE.ORPHAN_GEOMETRY_COLUMNS_V

16. no check

However, there are additional checks used at the DNR to validate the SDE schema.

SDE.ORPHAN_ADTABLES_V
SDE.ORPHAN_COLUMN_REGISTRY_V
SDE.ORPHAN_CONFIG_KEYWORD_V
SDE.ORPHAN_FSTABLES_V
SDE.ORPHAN_GDB_DEFAULTVALUES_V
SDE.ORPHAN_GDB_FD_SPATIAL_REF_V
SDE.ORPHAN_GDB_FEATURECLASSES_V
SDE.ORPHAN_GDB_FIELDINFO_V
SDE.ORPHAN_GDB_OBJECTCLASSES_V
SDE.ORPHAN_GDB_SUBTYPES_V
SDE.ORPHAN_GDB_USERMETADATA_V
SDE.ORPHAN_GEOM_COLS_SPATIAL_REF_V
SDE.ORPHAN_RASTER_TABLES_V
SDE.ORPHAN_SPATIAL_REFERENCES_V
SDE.ORPHAN_SPATIAL_VIEWS_V
SDE.ORPHAN_XML_CONFIG_KEYWORD_V

Just a couple of problems were found by running the schema check against all DNR SDE instances. A couple of tables were still referenced in the SDE.GEOMETRY_COLUMNS table that had been dropped without using the ArcGIS interface. These rows were deleted from the SDE.GEOMETRY_COLUMNS table.

The other problem was a table was still referenced in the SDE.TABLE_REGISTRY table. The SDE.ORPHAN_TABLE_REGISTRY_V also detected this error. Again the referenced table was dropped without using the ArcGIS interface. This problem was found in esri_read_only, and rows were deleted from the SDE.TABLE_REGISTRY to fix the problem.

So, the overall health of the SDE schemas at DNR is very good. The SDE.ORPHAN views have served well to find problems that would otherwise cause ArcGIS usage problems or prevent ArcSDE software upgrades. Users have also gotten better at using the ArcGIS tools to add and delete layers cleanly after receiving notice about the problems caused by manipulating layers outside of the ArcGIS tools.

One of the desired results of testing did not materialize. The access problem with the USER environment related to SELECT_ANY_TABLE privileges was not detected. The USER environment was clean according to the check schema test. This problem turned out to be caused by users naming layers with periods as separators, e.g. SE.hydro.table.

Orphan Views
To ensure that the SDE data dictionary has not become corrupted through manipulation by external applications, orphan views were developed to identify errors in the SDE/GDB

data dictionary which would cause application errors in the ArcGIS application. Typical errors include having no Oracle table or view for an SDE layer or table, still having columns, domains, for tables that no longer exist, and having configuration keywords or spatial references that are not used by any layers or tables. The orphan views are named after the SDE or GDB data dictionary tables that they monitor. (See the SDEGDBMonitoring.xls spreadsheet for a complete list of the orphan views.)

A and D Table with Rows After Compress
Identify tables that may need a full compression. Determine whether users require special support for compressing versioned tables.

SRIDs with Duplicate Parameters
Find spatial references that are duplicated within SDE that can be cleaned up. This makes API software easier to program and run more efficiently.

Management

When ArcGIS users seek help on a number of issues related to usage of the ArcSDE Geodatabase then a proactive approach to manage database usage is required. User training is the primary means of getting users on board with the ArcGIS environment. The switch from Workstation ARC/INFO to ArcGIS is daunting. Even simple operations are no longer intuitive to long-time Workstation users. The Database Administration staff developed several presentations for annual GIS user meetings on new developments, better ways to access data, data inventories, spatial reference template, loading their own data, and precision and cluster tolerance effects on editing.

Maintenance

Compress Geodatabases (weekly or daily). Performance of the edit environment can be significantly reduced when the GDB is not compressed regularly. If users are using topology tools to make global edits across the database then the GDB may need to be compressed on an hourly basis.

Clean up orphan SDE/GDB references (monthly). Track down users dropping and modifying tables outside ArcGIS applications if necessary.

Clean up orphan locks and edit states (as needed). This was more of a problem before ArcSDE 9.0 SP3.

Install ArcSDE upgrades and service packs (as needed).

Clean up SRIDs with duplicate parameters (annually).

Implementing Recommendations

The job of a Database Administrator is a balancing act, trying to get the biggest improvements in performance without trading off something else. First, do no harm. Our Oracle databases are not dedicated entirely to GIS applications; we have many financial and other business applications. After reviewing our ArcSDE/GDB environments, we documented what we found, and what we thought we should change. Next, we prioritized those items which we thought would give GIS users better performance while not sacrificing performance in other applications. Then, we tested the changes in development environments before implementing those changes in production. Some users got considerable improvements in their performance based on our suggestions. In general, probably only moderate improvements in database performance resulted; the database was generally tuned well to start with and there were any significant high level changes necessary.

Implementation Results

First of all, we understand the ArcSDE/GDB environment much better. That understanding has given us peace of mind for answering user and manager questions. We have gained a lot of expertise on tracing and improving performance of the GIS environment for users. Simultaneously with our SDE/GDB review, we converted many Workstation ARC/INFO coverages and grids to SDE. We have tuned the spatial indexes for those layers, and we have provided users with datasets in the ArcGIS native environment for fast and efficient ArcMap display and editing. As users became more familiar with using ArcMap, they began to show interest in using the editing tools. So, we provided information on how to design their datasets to take advantage of domains and subtypes, use relationship classes in the edit environment versus spatial views in the read-only environment, version their data, and check geometric relationships with topology. Further, since users have become used to maintaining their own Workstation workspaces on the file system, we have provided an environment in which users can create their own data and administer their own datasets. This has had an added benefit of getting users familiar with some of the difficulties of administering the ArcSDE Geodatabase and had improved their appreciation of what we do as DBAs.

A few case studies of implementation are the domain maintenance strategy and spatial reference template.

Domains from Code Tables

Coded domains are used by ESRI ArcGIS software as values for validation of coded attributes. Domains are stored in the geodatabase instance as binary objects to speed the loading process after connecting to the GDB. Unfortunately, this means that the codes are not directly accessible to other applications such as Oracle. Multiple applications, both ESRI and external, need to make use of coded values for editing and validation processes. These coded values need to be consistent, and having multiple sources of these coded values makes synchronization an issue. This complaint was passed on directly to Jack Dangermond via the Cadastral Project, and in response ESRI provided a work-around with domain-to-table and table-to-domain conversion tools as a substitute for direct table/domain compatibility. Also, as users began to use domains more in the

design of their applications, they found that different interfaces in ArcMap provided access to either the code or the description, but not both.

The domain-to-table conversion tool creates a new table based on an existing coded value domain. The table cannot exist previously, so must be dropped and recreated each time the domain is unloaded to table form. The table created has 3 columns: objectid, code, and description, for which the code and description fields can be named but not formatted by the user. The code and description field formats are created as character fields with a length of 255. This output table could be reloaded into a user defined and formatted code table. The best process to unload domains to tables would be to run the domain to table tool, load a formatted user code table from the tool's output table, and then drop the tool's output table.

The table-to-domain conversion tool creates a new domain or appends to an existing domain. The user selects which table columns map to the domain's code and description fields. The domain code can be either integer or character. The description can be a character field up to 1024 in length; longer fields are truncated to 1024. However, since the coded domain description is used for map labeling, a description this wide is much too big. The DNR standard might best be to map the table code to the domain code, and the name field (length of 50) to the description field. The label name field (length of 12) seems too short to be useful for the domain description field, although a field length of 20 would probably suffice if the label name format were changed. An individual analysis of DNR's existing code table values might be necessary to determine the best match of code and description fields. The best process to load tables to domains would be to run the table to domain tool using a specific column mapping for each code table. Another alternative is to create views of the code tables that have a standard format for the code and description fields to be automatically loaded with the table to domain tool.

Synchronizing code tables with domains is critical to aligning ArcGIS with independent enterprise applications. Given the limitations of the domain to table conversion tool with respect to format type and description length, the best approach to synchronizing code tables and domains is probably to load code tables into domains. The ESRI domains will accept a wider array of formats than the ESRI domain export process provides. Loading domains from tables would preserve the format type of the code without any intermediate conversion or post-processing.

Code table to domain scripts were developed to load code tables into domains. Views of code tables in SDE were created to redefine INFO code, label, name, and description items and map them to code and code_description columns to be used by load scripts. The views also concatenated the code and description in the code_description column so that users are able to see both, since ArcMap interfaces may show codes or descriptions, but have no option to choose one over the other. Users were very happy with the implementation of the code tables as domains, but still feel some frustration with administering domains themselves.

Spatial Reference Template

The template for spatial reference (SR) has been set up as a feature dataset to simplify and standardize cartographic work within Washington State. The template serves several purposes. The template provides a convenient source from which to import the standard coordinate system used for Washington DNR GIS. The template avoids getting stuck with spatial domains having ESRI defaults (e.g. XYDomain of -10,000, -10,000) or ESRI calculations (e.g. seemingly random XYDomain 1846.198123, 4791.987234) that are too small for successful data loading later (spatial domains cannot be changed after a feature class has been created). Feature classes created using the template will be able to load any features located within 40 to 60 miles of the state. The template improves the speed of feature display by setting the precision appropriate to GPS collected coordinates (1 foot accuracy) without excessive precision (wasted storage and processing), especially when geoprocessing statewide feature classes. Having fewer spatial references also allows ArcObjects code to run with fewer SR mismatches. The template reduces the number of extraneous spatial references administered through ArcSDE. Altogether, the template optimizes the ArcSDE/GDB tuning for Washington and still provides the necessary precision and coordinate space to allow for feature growth.

Next Steps and SDE's Future

There are many new Geodatabase features anticipated with ArcGIS 9.2. What's New in ArcGIS 9.2 lists significant new products and enhancements that will need to be investigated, such as File Geodatabase, ArcSDE Personal Edition and Workgroup Edition using SQL Server Express, geodatabase archiving, non-versioned editing, geodatabase replication, high precision storage (XYDomain), database servers, ESRI Spatial Type for Oracle with spatial SQL query access, Oracle project instances, Oracle Spatial GeoRaster, and operating system authentication. GIS Database Administrators must continue to investigate, understand, and manage these new products.

Conclusion

Systematically reviewing the ArcSDE/GDB environment provided many immediate payoffs for better usage at the DNR. We proceeded through the ArcSDE manuals and verified the basics, next we reviewed ESRI Conference technical workshops and looked for general recommendations. We documented our findings and now have a large store of information to help us quickly understand and fix problems. Throughout the review we documented problems that we and users were experiencing and searched the knowledge base and user forum for specific suggestions. We tested changes, implemented fixes, developed administration tools, and streamlined our administration procedures. There are still many more things that we would like to accomplish to ease database administration, but we now have more time to design new user datasets, and implement new products for users. As an example, we used our experience in investigating SDE administration to investigate and document a raster implementation strategy, and we are now implementing a raster database with completely different backup and recovery strategy – large static datasets with annual or biannual updates.

With the upcoming changes in ArcGIS 9.2, another systematic review of the ArcSDE Geodatabase may prove as useful.

Acknowledgements

Appreciation goes to fellow DNR Database Administrators Lowell Thacker and Kevin Kozak for discussion and review of our ArcSDE/GDB environment. Special appreciation goes to Jack Horton, and extra-special appreciation to Gudmundar Hafberg for taking the time from their work at ESRI to provide in-depth answers to our questions on the ArcSDE and Geodatabase whenever we came up blank.

References

Arctur, D., Zeiler, M. (2004). Designing Geodatabases: Case Studies in GIS Data Modeling. ESRI Press, Redlands, CA.

Brown, T. (2004). Advanced Oracle Configuration and Tuning of the Geodatabase. ESRI Conference Proceedings.

Cunningham, G., Gough, J. (2004). ArcSDE Administration for Oracle. ESRI Conference Proceedings.

Downey, M., Neild, J. (2004). Building Highly Available ArcSDE Systems. ESRI Conference Proceedings.

Environmental Systems Research Institute (2005). Managing a Versioned Geodatabase. ESRI Educational Services, Redlands, CA.

Environmental Systems Research Institute (2004). Understanding ArcSDE. ESRI, Redlands, CA. http://downloads.esri.com/support/documentation/sde_/706Understanding_ArcSDE.pdf

Environmental Systems Research Institute (2004). ArcSDE Installation Guide for Oracle. ESRI, Redlands, CA.

Environmental Systems Research Institute (2004). ArcGIS 9: Managing ArcSDE Application Servers. ESRI, Redlands, CA. http://downloads.esri.com/support/documentation/sde_/Managing_ArcSDE_Application_Servers_121704.pdf

Environmental Systems Research Institute (2004). ArcSDE Configuration and Tuning Guide for Oracle. ESRI, Redlands, CA. http://downloads.esri.com/support/documentation/sde_/706ArcSDE_Config_Gd_Oracle.pdf

Environmental Systems Research Institute (2004). ArcGIS Enterprise Systems: Performance and Scalability. ESRI Education Services, Redlands, CA.

Environmental Systems Research Institute (2004). Check Schema User Manual. ESRI, Redlands, CA.

Environmental Systems Research Institute (2004). ArcSDE Administration Command Reference.
http://edndoc.esri.com/arcsde/9.1/

Environmental Systems Research Institute (2004). SDE Monitor User Manual. ESRI, Redlands, CA.

Environmental Systems Research Institute (2003). ArcGIS: Working with Geodatabase Topology. ESRI, Redlands, CA.
http://support.esri.com/index.cfm?fa=knowledgebase.whitepapers.viewPaper&PID=43&MetaID=524.

Environmental Systems Research Institute (2002). Modeling Geodatabases Using CASE Tools. ESRI Educational Services, Redlands, CA.

Environmental Systems Research Institute (2001). ArcSDE Administration for Oracle. ESRI Educational Services, Redlands, CA.

Environmental Systems Research Institute (2001). Geodatabase Design Concepts. ESRI Educational Services, Redlands, CA.

Environmental Systems Research Institute (1999). Building a Geodatabase. ESRI, Redlands, CA.

Environmental Systems Research Institute (1997). SDE Version 3.0 for Oracle: Configuration and Tuning Guide. ESRI, Redlands, CA.

Environmental Systems Research Institute (1997). Spatial Database Engine: Getting Started with SDE Version 3.0. ESRI, Redlands, CA.

Hafberg, G., Dunn, T. (2004). Using Data Created with Non-ArcSDE Clients. ESRI Conference Proceedings.

Hafberg, G., Silvertand, G. (2004). Multiuser Geodatabases: Configuration and Performance Tuning. ESRI Conference Proceedings.

Meza, J., Hafberg, G., Sakowicz, A. (2004). ArcGIS Enterprise Systems: Performance and Scalability.

McAbee, J., Hafberg, G. (2004). ArcSDE and Organizational Databases. ESRI Conference Proceedings.

Zeiler, M. (1999). Modeling Our World: The ESRI Guide to Geodatabase Design. ESRI Press, Redlands, CA.

ESRI > For Developers > ArcSDE Online

ArcSDE Developer Help
http://arcsdeonline.esri.com/Support_files/sdehelp.htm

ESRI > Knowledge Base > Product Documentation > ESRI Software Library – ArcSDE

FAQ: ArcSDE Logfiles (8.3)
http://forums.esri.com/Attachments/9305.doc

ESRI Knowledge Base
http://support.esri.com/index.cfm?fa=knowledgebase.techarticles.articleShow&d=15862
http://support.esri.com/index.cfm?fa=knowledgebase.techarticles.articleShow&d=18336
http://support.esri.com/index.cfm?fa=knowledgebase.techarticles.articleShow&d=25040
http://support.esri.com/index.cfm?fa=knowledgebase.techarticles.articleShow&d=26481

ESRI User Forum
http://forums.esri.com/Thread.asp?c=2&f=1719&t=144787&mc=5
http://forums.esri.com/Thread.asp?c=2&f=59&t=131017#378635
http://forums.esri.com/Thread.asp?c=2&f=1719&t=127189#503996
http://forums.esri.com/Thread.asp?c=2&f=1720&t=130955#378553

Author Information

Greg Tudor, PLS
Bio:
GIS DBA, focusing on ArcGIS implementation and cadastral, transportation, and hydrography applications. MSE in Surveying and Land Information Systems, University of Washington.

Contact:
Information Technology Division
Washington State Department of Natural Resources
PO Box 47020
Olympia, WA 98504-7020

Voice: 360-902-1542
E-mail: greg.tudor@wadnr.gov
Web: http://www.dnr.wa.gov/