Raster Data in ArcSDE™ 8.1.2

An ESRI® White Paper • February 2002

ESRI 380 New York St., Redlands, CA 92373-8100, USA • TEL 909-793-2853 • FAX 909-793-5953 • E-MAIL info@esri.com • WEB www.esri.com

J-8843

Raster Data in ArcSDE 8.1.2

An ESRI White Paper

Contents **Page**

Contents **Page**

J-8843

Contents **Page**

J-8843

Raster Data in ArcSDE 8.1.2

Introduction

This document highlights the basic concepts, user experience, and product overview for the support of raster data in ESRI® ArcSDE™ 8.1.2. The focus of the document is to explain how ArcSDE 8.1.2 provides efficient storage and retrieval of raster data in a client/server environment by supporting raster data types. Where possible, best practices for the loading, storage, and retrieval of raster layers are given. Raster data can be accessed by ArcGIS™ 8.1 or applications customized with the ArcSDE C API or ArcObjects™ COM API.

Other Suggested Resources

Modeling Our World—The ESRI Guide to Geodatabase Design
Understanding ArcSDE
Managing ArcSDE Services
ArcSDE Configuration and Tuning Guide (Database Specific)
ArcSDE Developer Help
ArcObjects Developer Help
Performance Tips and Tricks for ArcGIS Desktop 8.1

A list of specific resources is included in Appendix A, Bibliography.

Why Put Images in a Database?

Multiuser Access

When many users are accessing the same raster files simultaneously, better performance is possible from a properly tuned, centralized database than from a file-based system.

Data Management

A database allows common data management and retrieval for all geospatial data including raster, vector, and tabular data. A database also provides access to extremely large images (many gigabytes to terabytes) of continuous spatial data (e.g., 30-meter digital elevation model composite of North America).

Data Query

A database allows for a common query environment. Queries can be made to show all data related to an area during a particular time period or for a particular subject matter.

Uses of ArcSDE Raster Data

Basemaps—Enterprise GIS (utility or local government)

ArcSDE is used to manage large data holdings and to have a central repository, a management system, and query capabilities across the entire database. Large organizations need everyone to use the same data and be able to access it at the same time. Their primary need for raster support in ArcSDE is a background image. They have scanned maps or aerial photographs of an extensive area, and they want to create a seamless mosaic as a background to all their vector data.

Data Management (data providers)

Satellite or remote sensing data providers have large amounts of data. These organizations could use ArcSDE to manage their data. All of these organizations are in the business of data dissemination—some for financial profit, others to make it easier and cheaper for more people to have access to data.

Feature Attributes (utility, real estate, or local government)

An organization may have pictures of locations that it needs to attach to a spatial feature. This could be a picture of a house linked to a parcel boundary or a picture of a pump or valve linked to a hydraulic network.

Basic Raster Concepts

Vector data, such as coverages and shapefiles, represents geographic features with lines, points, and polygons. Rasters, such as images and grids, represent geographic features by dividing the world into discrete squares called cells. Cells are laid out in a grid, where each cell has a location relative to an origin and a value describing the feature being observed; for instance, the cell values in an aerial photograph represent the amount of light reflecting off the earth's surface[1].

Some rasters have a single band (a measure of some characteristic) of data while others have multiple bands. When you create a layer from a raster, you can choose to display a single band of data or form a color composite from multiple bands. A satellite image commonly has multiple bands representing different wavelengths of energy from the ultraviolet through the visible and infrared portions of the electromagnetic spectrum.

Rasters also have a measure of the number of colors able to be stored in a cell. This is the bit or color depth. A bilevel or one-bit image will be able to display two colors—black or white. An eight-bit image will be able to display 256 colors (2^8). The higher the bit depth, the more colors available for display but the larger the storage requirements.

To learn more about rasters, see *Understanding ArcSDE* (pp. 17–19) or *Modeling Our World* (Chapter 9).

How Raster Data Is Stored in the Database

Raster layers in a database are seen by client applications as one of two distinct types.

- Raster—a single picture of an object or a seamless image covering a spatially continuous area.
- Raster catalog—a collection of georeferenced rasters that are spatially continuous and can be displayed as a single layer. Initially these must be all in the same coordinate system. This is a table in the database that points to each of the rasters.

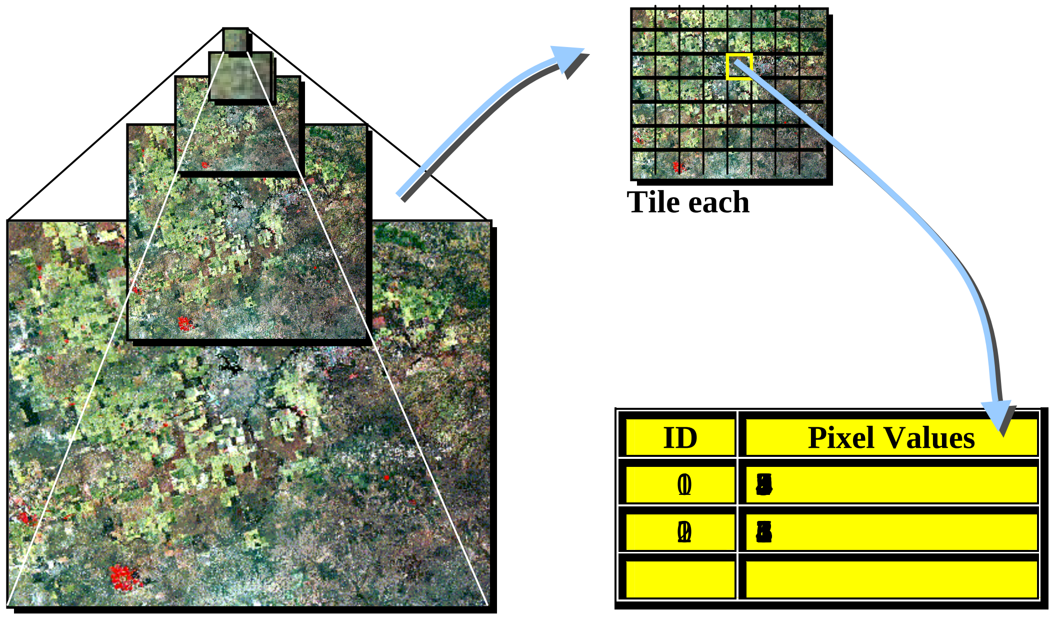
Note: All rasters in a catalog must be in the same database connection to avoid possible problems with permissions.

Rasters can be stored individually, appended together to form mosaics or, if using ArcGIS, collectively referenced as a raster catalog. However, ArcIMS® cannot see an ArcSDE raster catalog. Rasters that will be used by ArcIMS can be stored as a multi-row

[1] *ArcGIS Desktop Help*, "Rasters, Described," ESRI, 2001.

raster. A multi-row raster has many rasters stored in one table, each with a unique ID, and is displayed by ArcIMS as a seamless layer.

When raster data is loaded into an ArcSDE database, it is converted into the Spatial Database Engine™ (SDE®) raster format. The raster is stored as many small Binary Large Objects (BLOBs) or tiles in a set of ArcSDE system tables and user tables. Each time the raster is queried, only the necessary tiles are returned instead of the whole data set. Display performance is optimized by reducing the amount of data transferred between the client and the server. To do this, ArcSDE stores multiple resolutions of the raster called pyramids, cuts these into tiles that become BLOBs, and spatially indexes them. This makes it possible to store seamless raster data sets (tens to hundreds of gigabytes) and serve them quickly to a client for display.



Build reduced resolution layers (pyramids).

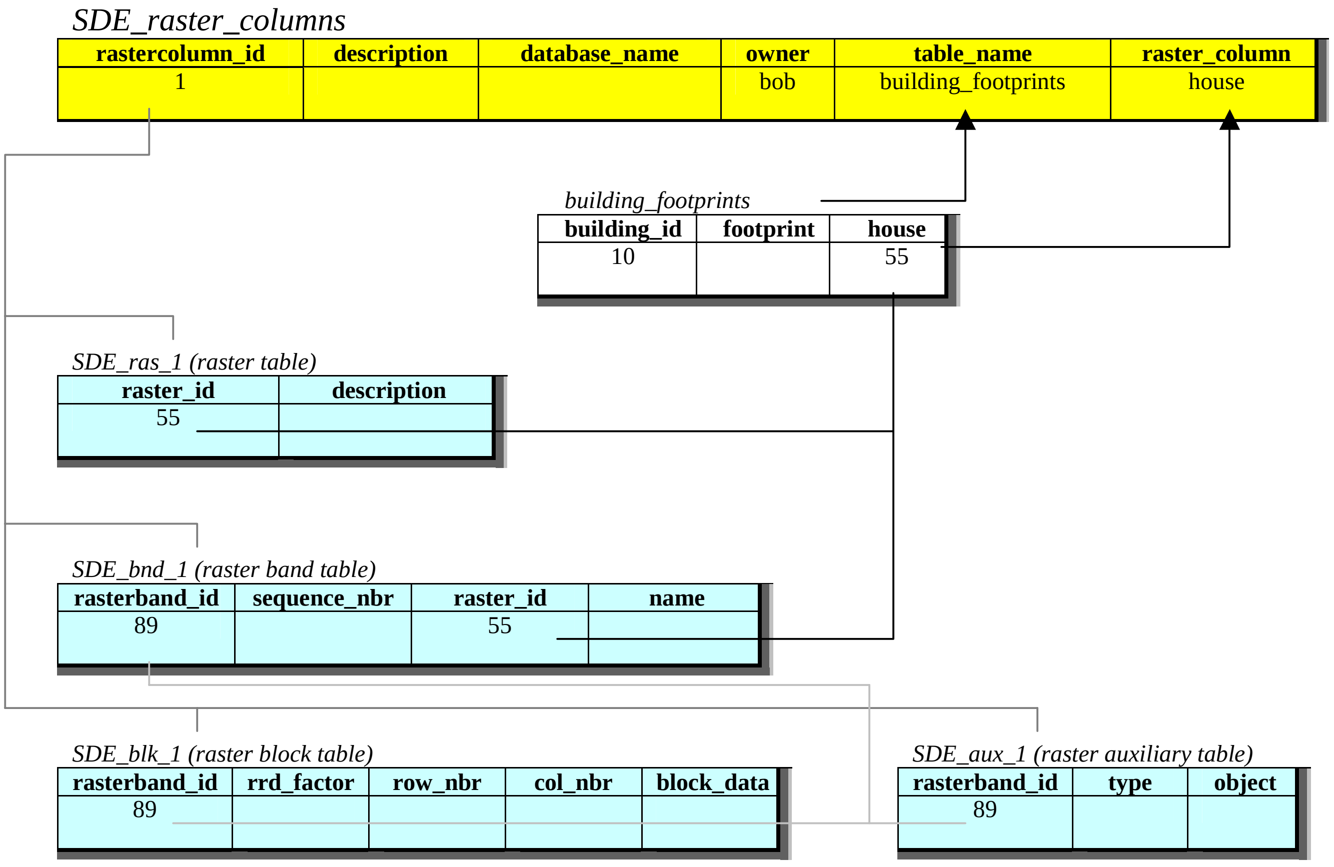
Store each tile as BLOB in database.

When a raster is imported into an ArcSDE database, ArcSDE adds a raster column to the business table of your choice. You may name the raster column whatever you like, as long as it conforms to the underlying database column naming convention. ArcSDE has a restriction of one raster column per business table.

The raster column is a foreign key reference to the raster_id column of the raster table (SDE_ras_*n*) created during the addition of the raster column. Also joined to the raster table's raster_id primary key, the raster bands table (SDE_bnd_*n*) stores the bands of the image. The raster auxiliary table (SDE_aux_*n*) joins one-to-one to the raster bands table by rasterband_id, which stores the metadata of each raster band. The rasterband_id also joins the raster band's table to the raster blocks table (SDE_blk_*n*) in a many-to-one relationship. The raster blocks table rows store blocks of pixels determined by the dimensions of the block.

When ArcSDE adds a raster column to a table, it records that column in the sde user's SDE_raster_columns table. The rastercolumn_id is used in creating the names of the raster, raster bands, raster auxiliary, and raster blocks tables.

See Appendix A for a detailed description.


SDE_raster_columns
rastercolumn_id | description | database_name | owner | table_name | raster_column
1 | | | bob | building_footprints | house
building_footprints
building_id | footprint | house
10 | | 55
SDE_ras_1 (raster table)
raster_id | description
55
SDE_bnd_1 (raster band table)
rasterband_id | sequence_nbr | raster_id | name
89 | | 55
SDE_blk_1 (raster block table)
rasterband_id | rrd_factor | row_nbr | col_nbr | block_data
89
SDE_aux_1 (raster auxiliary table)
rasterband_id | type | object
89

ArcSDE Raster Tables Version 8.1.2

🟦	System Tables stored in user schema	🟨	System Tables stored in SDE schema
⬜	Business Table stored in user schema		

Supported Input Formats

The ArcGIS data loading application is built on top of Raster Data Objects (RDO) and ArcObjects and can access rasters in a large number of standard formats. The same formats supported by ArcMap™ and ArcCatalog™ can be loaded into ArcSDE.

SDERASTER is an ArcSDE tool that supports TIFF or BSQ formats. However, it requires the open source libtiff library (http://www.libtiff.org) to load TIFF files.

See "Supported Raster Formats" in *ArcGIS Desktop Help* for a full listing.

Storage Parameters

The user can use default storage parameters to store the raster data or set the storage parameters to suit particular data and server setup. They are specified when loading raster data to the database.

Pyramids

Pyramids are reduced resolution representations of your data set that are used to improve performance. Pyramids can speed up display of raster data by fetching only the data at a specified resolution that is required for the display. Pyramids are created by resampling the original data. The resample methods instruct the server how to resample the data to build the pyramids. Three resampling methods are supported.

- Nearest neighbor should be used for nominal data or raster data sets with color maps such as land use data, scanned maps, and pseudocolor images.
- Bilinear or cubic should be used for continuous data such as satellite imagery or aerial photography.
- The default option is to build pyramids using the bilinear resample method.

Pyramid building is performed on the ArcSDE server side. If the data is compressed, the server will first decompress the data, then build the pyramid and compress the data again to insert into the block table.

Tile Size

The tile size controls the number of pixels you want to store in each BLOB and, therefore, the size of the BLOB. This is specified as a number of pixels in *x* and *y*. The default value is 128 x 128, which should be satisfactory for most applications. The best tile size setting depends on many factors such as data type (bit depth), database settings, and network settings. A smaller tile size (100 x 100) may result in more records in the raster block table, which will slow down the queries; a larger tile size (300 x 300) will require more memory to process though it may create fewer records in the block table. Experiment with your data to choose a tile size if you do not want to use the default.

Compression

Data compression compresses the blocks of data before storing them in the geodatabase. The compression methods used are LZ77 or JPEG. The LZ77 algorithm is a lossless compression, meaning the values of cells in your raster data set will not be changed. This is the same compression used by the PNG image format and in ZIP compression. The amount of compression will depend upon the data. The fewer unique cell values, the higher the compression ratio. JPEG compression is lossy, meaning the values of cells in the raster data set will be changed. JPEG compression only applies on eight-bit data without color map. The primary benefit of compressing your data is that it requires less storage space. The user can specify quality for JPEG compression using values from 5 to 95, where 95 is the best quality and 75 is the default.

The ArcSDE client performs compression and decompression. The ArcSDE client sends compressed data to the server at loading, and the server always returns compressed data to its client at retrieval. Where retention of pixel values is important, use LZ77 compression; if individual pixel values are not important, use JPEG compression.

Configuration Keyword

The configuration keyword specifies the predefined set of options for storing the data. It is defined in the DBTUNE table. To improve performance, it is best to store the indexes and data in different locations on different physical devices.

Append

Appending, or mosaicking, to an existing raster in the database creates one seamless raster. The overlapping area is resolved by replacing the existing data with the data from the new raster data set.

Update

Updating will delete the existing raster in the database and load input raster to the database with the name of the existing raster.

Statistics

The statistics of the raster data set in a database can be stored along with its pixel data; a histogram is also stored in the database. Statistics are normally required for displaying rasters with different stretch methods. Having current statistics built on a raster layer will always improve layer drawing performance.

Custom Applications

ArcObjects is a wrapper of the ArcSDE C API and is implemented in RDO. ArcGIS loading application is a Windows application written using ArcObjects and runs on both ArcCatalog and ArcToolbox™. The C API has the core functionalities for managing raster data in a database. The SDERASTER command line application is based on the C API.

When writing customized loading applications, ArcObjects or the C API can be used to suit different situations. ArcObjects is only available on the Windows platform, but the ArcSDE C API is available on all the supported platforms.

The C API for raster data implements Raster Streams, Raster Column, Raster Value, Raster Band, Raster Cell, Raster Block, Raster Pyramids, and Raster Band Statistics. Logically, the SDE Raster API is similar to the SDE Geometry API with Streams, Layers, Geometry Columns, and Shapes.

Raster Columns are added to Business Tables and registered with the system in a shared Raster Columns table. The value in a Raster Column is a foreign key reference to a raster. A raster is a collection of one or more Raster Bands that, taken together, form the raster. Raster Bands store the actual pixel data and associated statistics. Applications access the pixel data of a raster or Raster Band through a Raster Stream.

See the *ArcSDE Developers Guide* for more information.

Data Loading

The data loading process consists of preprocessing the raster images (if required), defining a set of appropriate parameters to hold the raster, and actually populating the Raster Columns table and a set of ArcSDE system tables.

There are several ways to load raster data into a database. This document gives an outline of

- ArcGIS—Graphical interface
- SDERASTER—Command line loader

- ArcObjects COM API—Customized application

Preprocessing of Continuous Rasters

The preprocessing of images can be done in software products like ArcInfo™, ERDAS IMAGINE®, and, to a limited extent, ArcView®. Preprocessing of data is most important for people wanting to create a seamless raster layer, edgematch, spectral match, or georeference before loading a collection of images.

Loading Parameters

In ArcGIS, rasters can be loaded into the database as individual layers or a seamless layer by appending it to an existing raster. For the case of mosaicking rasters into a spatial continuous raster data set in the database, all the rasters must have the same spatial definition and data type.

Default storage parameters include

- *Spatial reference:* the spatial reference of the raster data set
- *Update mode:* append to existing raster
- *Statistics:* compute
- *Compression type:* LZ77
- *Tile size:* 128 x 128
- *Pyramid option:* build pyramid
- *Pyramid resample method:* bilinear
- *Configuration keyword:* default

Users can specify whether they want to generate reduced resolution layers (pyramid layers) and how the pyramid layers will be built. The reduced resolution layers cut down on access time in passing data up to the client by only grabbing the coarsest possible resolution of imagery to paint the screen. It increases database size but decreases access time.

Color maps are automatically loaded to the database when raster data is loaded, if available. The color map is used to map pixel values to RGB colors so that the raster data is displayed the same way by default.

Statistics storage is calculated upon request. When loading raster data to the database, the user can specify if the statistics will be calculated and stored in the database. Since calculation of statistics on large data sets can be time-consuming, it is suggested that they be calculated once the data is loaded if the data is to be permanent in the database. If mosaicking, loading performance will improve if statistics are calculated only once at the end.

Both ArcGIS and the SDERASTER command line application have advantages and disadvantages.

ArcGIS Loading Application

Advantages

- Graphical user interface (GUI)-based Windows® application

- Supports all raster formats
- Less restrictions on mosaicking (must be spatial-referenced, have no color map, and have the same bit depth)

Disadvantages

- Does not work on UNIX®.
- Batch loading can be tedious if not using default storage parameters because each individual raster loading parameter has to be changed, but this can be resolved by developing a customized application using ArcObjects.

SDERASTER Command Line Application

Advantages

- Works on both Windows NT® and UNIX
- Has several options such as add/delete color map, specify value as nodata, and eliminate background
- Can copy between databases
- Can export data of a specified area
- Faster than the ArcGIS loading application

Disadvantages

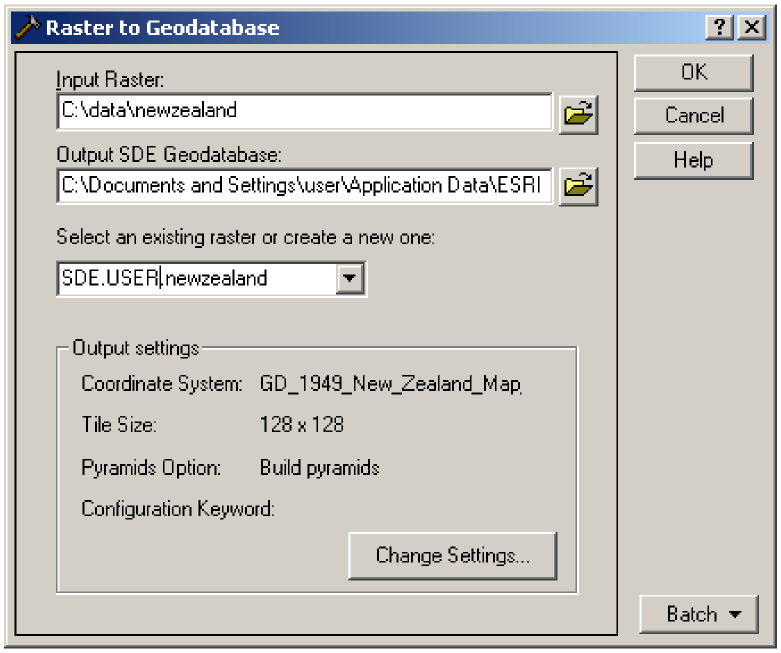
- Only loads BSQ and TIFF formats
- Requires open source libtiff library (http://www.libtiff.org) to load TIFF
- More restrictions on mosaicking

Data Loading in ArcGIS

The source raster can be either file based or in a database, and it can be imported to either an existing or new SDE raster. The Raster to Geodatabase tool or Raster to Geodatabase wizard is in ArcCatalog and ArcToolbox. Rasters also can be loaded into a geodatabase using Export -> Raster to Different Format in ArcCatalog; however, this loading process only uses default storage parameters. Depending on the selected context menu (from raster or geodatabase), the input raster or output geodatabase may need to be defined.

To New SDE Raster

With the input raster and geodatabase connection defined, the new raster name has to be entered. By default, the output raster will inherit the spatial reference from the input raster; however, it can be modified or changed.



Input Raster: C:\data\newzealand

Output SDE Geodatabase: C:\Documents and Settings\user\Application Data\ESRI

Select an existing raster or create a new one: SDE.USER.newzealand

Output settings

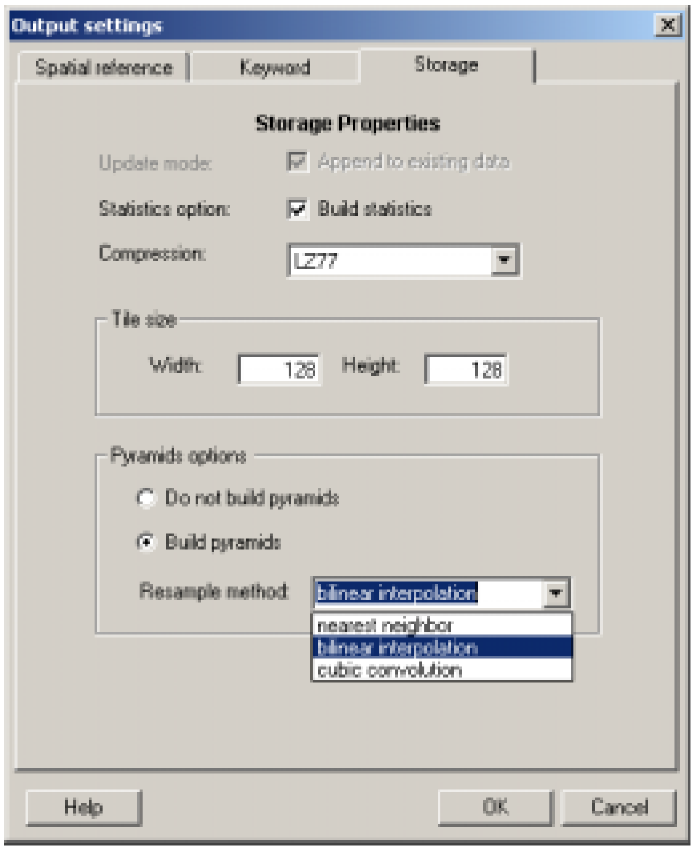
Coordinate System: GD_1949_New_Zealand_Map

Tile Size: 128 x 128

Pyramids Option: Build pyramids

Configuration Keyword:

Other raster properties, including pyramids options, tile (block) size, configuration keyword, and compression type, can be defined at the time of importing data. If the raster has a color map file, it will be loaded at the same time. Then additional attributes for the new raster can be added later, which will be saved in the business table. Statistics calculation is optional at data loading.



Storage Properties

Update mode: ✓ Append to existing data

Statistics option: ✓ Build statistics

Compression: LZ77

Tile size

Width: 128 Height: 128

Pyramids options

○ Do not build pyramids

● Build pyramids

Resample method: bilinear interpolation

nearest neighbor

bilinear interpolation

cubic convolution

To an Existing SDE Raster

If the ArcSDE raster has preexisting data, update options (e.g., mosaic or replace, rebuild pyramids or delete old pyramids, rebuild statistics or not) have to be set.

Data update acts at the tile level and has two modes.

- Mosaic mode appends the loading data to the original data, so only the blocks that the new data cover are updated; the overlapping area is resolved by replacing the old data with the new data. In this mode, tile size and spatial reference cannot be modified. If the original data in the database has stored statistics, the statistics will be removed. The user could elect to recalculate the statistics for the whole data after mosaicking is performed.

- Replace mode deletes the original data completely and inserts the new data. All properties except spatial reference information can be modified.

Statistics and pyramids of the existing data in ArcSDE will be erased when appending. The user then has the option to rebuild statistics/pyramids or not to rebuild them.

Batch Loading

The Raster to Geodatabase tool in ArcGIS also supports batch loading of multiple rasters into the geodatabase. The rasters can be loaded as a seamless raster into the geodatabase or as individual rasters. The user can also load one raster into a different geodatabase using batch loading.

Data Loading with SDERASTER

To import an existing image into ArcSDE using SDERASTER, import the image as follows:

```
sderaster -o import -l topo,image -c lz77 -g -f topo_562.tif
Connecting to server ultra, port 7000, as user gis1
Creating user table: topo Creating raster layer: topo.image
Image Dimension..............: 500, 1108, 1 Pixel
Type...................: uchar Raster ID : 1
Total Time: 00:00:02
Complete...
```

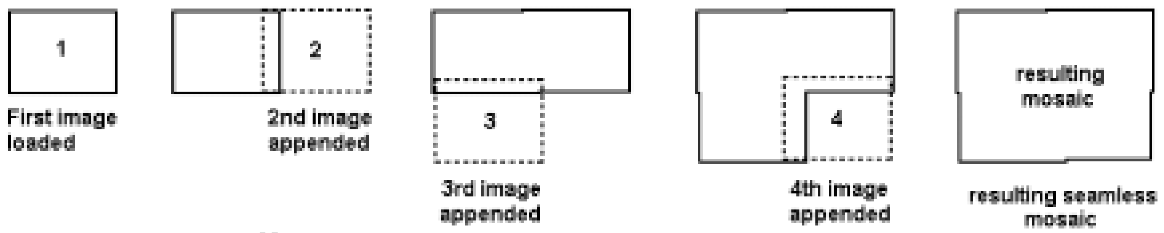
To add the raster to an existing business table (multirow raster), add the raster as follows:

```
sderaster -o insert -l bsq2,image –c lz77 -f mlc2.bsq -N -n bsq2

Opening file mlc2.bsq...
Connecting to server piopio, instance esri_sde, as user gis1

Image Dimension..............: 891, 1001, 1
Pixel Type...................: uchar
Raster ID : 2
       Total Time: 00:00:01
Complete...
```

For more command options, see Appendix B in *ArcSDE 8.1.2 Developer Help* or online at http://arconline.esri.com.

Creating a Mosaic

A mosaic is a raster composed of multiple input images. Mosaicking can be thought of as merging or appending. First a single image is loaded into ArcSDE. Then subsequent images are appended onto the original image, resulting in a new larger image.



N
ote that the images being appended may overlap or even have gaps inbetween them. For overlapping images, the new image being appended will overwrite the existing image's pixels where they overlap.

Requirements for Appending Rasters to an Existing Raster in Database

If using ArcObjects or ArcGIS loading tools, the following conditions must be satisfied in order to have a successful mosaic operation:

- All the rasters must have the same data bit depth.
- All the rasters must have the same number of bands.
- None of the rasters can have color maps.
- The existing raster in the database must have a spatial reference.

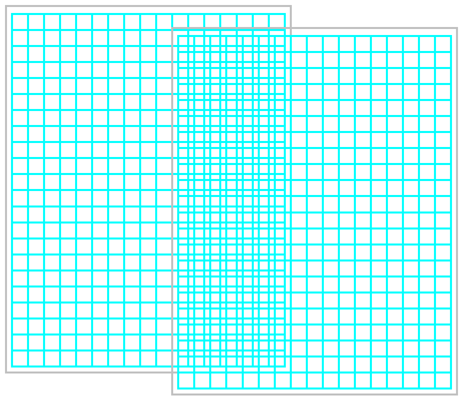
For the SDERASTER command line loader, there are two additional requirements.

- All the rasters must have exact pixel registration.
- All the rasters must have the same cell size.

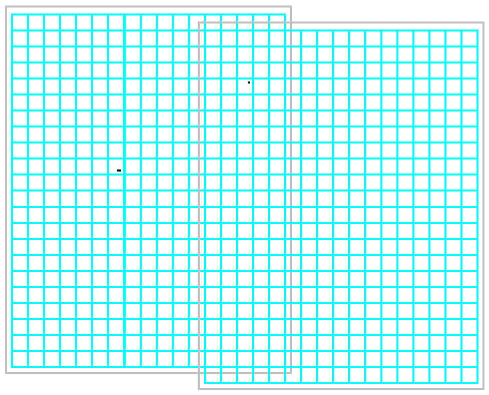
What Is Exact Pixel Registration?

Exact pixel registration means that pixels from multiple images line up exactly. This should not be confused with overlaps or gaps, which are permitted. But the cells have to fall on an even multiple of the cell width and height from one another, and adjacent images cannot have cells starting halfway into the cells of the original image. It is required for mosaicking using SDERASTER or C API.

Imagine two pieces of ordinary 8.5- by 11-inch graph paper as representing two images.



The cells of the second sheet of graph paper (representing pixels) do not line up with the first. The edges of the second sheet's cells fall in the middle of the first sheet's cells. These do not have exact pixel registration.



Here, the second sheet's cells line up with the first. The edges of the second sheet's cells fall precisely on the edges of the first sheet's cells. This is exact pixel registration.

Creating a Raster Catalog

A raster catalog is a method of displaying individual rasters as one layer in ArcGIS without appending them together. The table structure for a raster catalog in the database is the same as it is in a file system. This table must be created manually. There are five fields

Column Name	Data Type
Image	String
Xmin	Float
Ymin	Float

Xmax	Float
Ymax	Float

This example creates a raster catalog named "mycatalog."

```
CREATE TABLE sde.mycatalog (Image varchar(50) NOT NULL, Xmin float(53)
NOT NULL, Ymin float(53) NOT NULL, Xmax float(53) NOT NULL,
Ymax float(53) NOT NULL)
```

The value for Image field is [<database>].<owner>.<raster>, the same as it appears in ArcCatalog treeview.

Image	Xmin	Ymin	Xmax	Ymax
[<Database>].<Owner>.<Raster1>	0	0	100.50	200.50
[<Database>].<Owner>.<Raster2>	100.5	0	150	200.50

ArcObjects Sample

ArcObjects for Loading

ArcObjects functions are implemented in RDO based on ArcSDE C APIs. These functions are primarily for loading rasters into a database. There is one CoClass that implements four interfaces. The user can develop customized applications for loading raster data to suit specific requirements using ArcObjects, which may provide additional functionalities that ArcGIS does not support directly on the user interface; for example, using IRaster as input allows loading part of a raster to the database. It also make it possible to preprocess the data before loading.

IRasterSDEConnection defines the connection information that includes output workspace name, input raster name, output raster name, and bit mask file.

IRasterSDEConnection2 implements IRasterSDEConnection and adds one member, Raster, so not only raster data set name string, but also raster object, can be input for loading to the database.

IRasterSDEStorage holds the parameters for storage, spatial reference, tile size, compression type, configuration keyword, and pyramid option.

IRasterServerOperation defines the operation for the SDE session such as Create, Delete, Mosaic, Update, BuildPyramids, and ComputeStatistics.

Loading Sample

Note: More samples can be found in the *ArcObjects Developer Help* (http://arconline.esri.com/arcobjectsonline/).

Example 1
Loading a Raster Data Set to Database Using Default Storage Parameter Values

```
Sub LoadRasterToSDE(sInputraster As String, sServer As String, sInstance As
String, sDatabase As String, sUser As String, sPasswd As String, sSDERaster
As String)
   Dim pRasterSDELoader As IRasterSdeConnection
   Dim pRasterStorage As IRasterSdeStorage
   Dim pRasterOp As IRasterSdeServerOperation
   Dim pSR As ISpatialReference
   ' ----------- Set up connection ------------
   Set pRasterSDELoader = New RasterSdeLoader
   pRasterSDELoader.ServerName = sServer
   pRasterSDELoader.instance = sInstance
   pRasterSDELoader.Database = sDatabase
   pRasterSDELoader.UserName = sUser
   pRasterSDELoader.Password = sPasswd
   pRasterSDELoader.InputRasterName = sInputraster
   pRasterSDELoader.SdeRasterName = sSDERaster
   Set pRasterStorage = pRasterSDELoader

   ' pSR Can be the spatial reference of the
   ' input raster or user specified
   ' ----------- Set spatial reference -------------
   Set pRasterStorage.SpatialReference = pSR
   ' ---------- Load data -------------
   Set pRasterOp = pRasterSDELoader
   pRasterOp.Create

      ' --------------- Clean up -------------
      Set pRasterSDELoader = nothing
      Set pRasterStorage = nothing
      Set pRasterOp = nothing
End sub
```

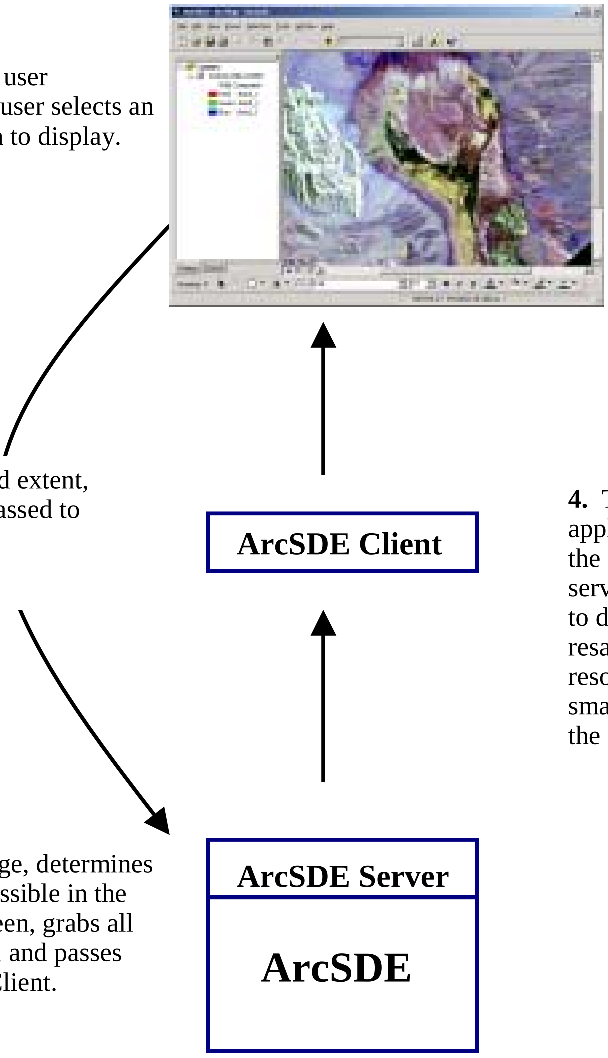
Example 2
Batch Loading and Mosaicking Rasters from a File Directory to a Seamless Raster in the Database

```
Sub MosaicDirToSDE(sDir As String, pWKName as IWorkspaceName, sSDERaster _
   As String)

      Dim pSDEConnection As IRasterSDEConnection
      Dim pSDEStorage As IRasterSDEStorage
      Dim pSDEServerOp As IRasterSDEServerOperation
      Dim pWsFact As IWorkspaceFactory
      Dim pWs As IWorkspace
      '---- all the datasetnames in the dir ---
      Dim pEnumDNs As IEnumDatasetName
      Dim pDsName As IDatasetName
      Dim iCount As Long
      Dim pName As IName
      Dim pGeoDs As IGeoDataset

      ' -------- Open the workspace of specified dir ---------
      Set pWsFact = New RasterWorkspaceFactory
      Set pWs = pWsFact.OpenFromFile(sDir,0)
```

```
        ' ------- Get all the datasetnames -------------
        Set pEnumDNs = pWs.DatasetNames(esriDTRasterDataset)

        ' ------- Set SDERasteLoader ---------------
        Set pSDEConnection = New RasterSDELoader
        pSDEConnection.SDEWorkspaceName = pWKName
        pSDEConnection.SDERasterName = sSDERaster

        '-------- Loop through all the datasetnames and mosaic them ---------
        iCount = 0
        Set pDsName = pEnumDNs.Next
    Do While Not pDsName Is Nothing
        pSDEConnection.InputRasterName = sDir + "\" + pDsName.Name '--full
path name --
        Set pSDEStorage = pSDEConnection

        ' ---------- Do not build pyramid until mosaic is finished -----
        pSDEStorage.PyramidOption = esriRasterSdePyramidDonotBuild
        Set pSDEServerOp = pSDEConnection

        ' ---------- Create the first one and mosaic the rest ----------
        if iCount = 0 then
            ' ----------- Get spatial reference ------------
            Set pName = pDsName
            Set pGeoDs = pName.Open
            Set pSDEStorage.SpatialReference = pGeoDs.Spatialreference
            ' ---------- Create ---------------
            pSDEServerOp.Create
        Else
            ' ---------- Mosaic ----------
            pSDEServerOp.Mosaic
        End If

        ' ---------- Get next datasetname -----------
        Set pDsName = pEnumDNs.Next
        iCount = 1
    Loop

    ' ------------- Build pyramid and calculate stats ---------------
    pSDEServerOp. ComputeStatistics
    pSDEStorage.PyramidOption = esriRasterSdePyramidBuildWithFirstLevel
    pSDEServerOp.BuildPyramids
End Sub
```

Data Viewing

Rasters in a database behave the same as rasters in a file system, just as features in a database are treated similar to any other feature source. The difference is that you have to make a connection to the database in order to access the raster data in that database.



1. From an end user application, the user selects an image they wish to display.

2. The image name, desired extent, and screen resolution are passed to ArcSDE.

3. ArcSDE finds the image, determines the coarsest resolution possible in the pyramids to paint the screen, grabs all the tiles to cover the area, and passes them up to the ArcSDE Client.

4. The ArcSDE Client application will uncompress the data if compressed on the server, mosaic the tiles, clip to display extent, and resample to display canvas resolution. Pass up the smallest possible image to the end user application.

Viewing from ArcGIS 8.1.2

ArcCatalog

Rasters from SDE can be browsed in ArcCatalog under a database connection using table view or geography view. A raster table is a normal user table with a raster column in it. Each time the user loads rasters to the database, a record is added to the specified raster table. The user can modify the table by adding new fields to store some attributes for the raster data set such as date of the imagery, type, and cloud coverage. But the raster column should not be modified because it records the connection to the actual data tables.

ArcMap

You can also display in ArcMap using the same tools available for all other raster formats. The property page is a little different; on the Source tab, the data source box appears as follows:

```
Data Type: SDE Raster
    Server:<servername>
    User:<username>
    Instance:<instance>
    Raster: <rastername>
    Status:
    Coordinate System:
```

Querying cells with the identify tool will return values from the base pyramid of the raster. This means that no matter what level of the pyramid you are currently viewing, the values will be those stored by the data.

Viewing in ArcIMS

ArcSDE raster data can be served over the Internet by the means of ArcIMS as file-based raster data. The following is an example of <MAP> tag in an AXL file.

```
<MAP>
   <PROPERTIES>
<ENVELOPE minx="-125" miny="20" maxx="-80" maxy="60"
name="Initial_Extent" />
   <MAPUNITS units="decimal_degrees" />
   </PROPERTIES>
   <WORKSPACES>
<SDEWORKSPACE name="sde_ws-0" server="spserver"
instance="port:5151" database="raster" user="raster"
password="go" geoindexdir="C:\Temp\" />
   </WORKSPACES>
   <LAYER type="image" name="US NED" visible="true" id="0">
<DATASET name="RASTER.RASTER.US_NED.IMAGE"
workspace="sde_ws-0"
/>
   </LAYER>
</MAP>
```

Tips and Tricks

Loading Data

To improve performance for loading rasters into a geodatabase, it is recommended that pyramids are built and statistics are calculated after loading. The loading speed is about 80 MB per minute without building pyramids. Loading with the SDERASTER command application is about 20 percent faster than ArcObjects on tested data.

Storage

As each raster tile is stored as a BLOB in the raster block table, in order to minimize storage requirements when choosing a tile size, consider the current DBMS page size to make sure that a good proportion (75 percent) of the tiles will be stored in line with the rest of the row data.

Georeferencing

Rasters can hold georeferencing information in their file headers (e.g., GeoTIFF) or in World files. SDERASTER expects World files to exist and hold the referencing information required. It loads the coordinate information provided in relation to the spatial reference (projection) specified. ArcGIS will read the georeferencing information in the file header.

Mosaic

The best way to mosaic multiple raster data sets to a seamless raster data set is to do it in batch mode or write a customized application using ArcObjects. As intermediate pyramids and statistics results will be erased when the mosaicking operation starts, it is strongly recommended to always set "do not build pyramids" and "do not calculate statistics" for each of the data sets except the last one.

Mosaic Rasters with Color Map

As a requirement, if any of the rasters intended for mosaicking operation have a color map, the operation will fail in ArcGIS. However, if the rasters are in TIFF or band sequential (BSQ) format, the user can use the SDERASTER command to load and mosaic the data using –N option to ignore the color maps; and after the mosaicking operation completes, use –o color map to apply the color map back to the mosaicked raster.

MrSID

MrSID™ rasters are compressed using lossy compression algorithm. When loading a MrSID raster to database, the loader application uncompresses the data and then loads the raw pixel data to the database. This process delays loading tremendously and does not benefit data quality, but can take a lot of space. Consequently it is not recommended to load MrSID rasters into a database. If it is necessary to load a MrSID raster, there are a couple of approaches—either load the original data from which the MrSID is created, or convert the MrSID raster to other format using "Export to Other Format" menu in ArcCatalog and then load the data.

No Data

If the input raster data set has no data, a bit mask will be generated on the fly at the time of loading to the database if using ArcObjects or ArcGIS loading tools. The bit mask is stored along with pixel data on each band. When the ArcSDE Client accesses the data, the no data area will be returned as no data too. Also –a option in the SDERASTER command line loader will eliminate the specified value as no data.

Loading One-Bit Data

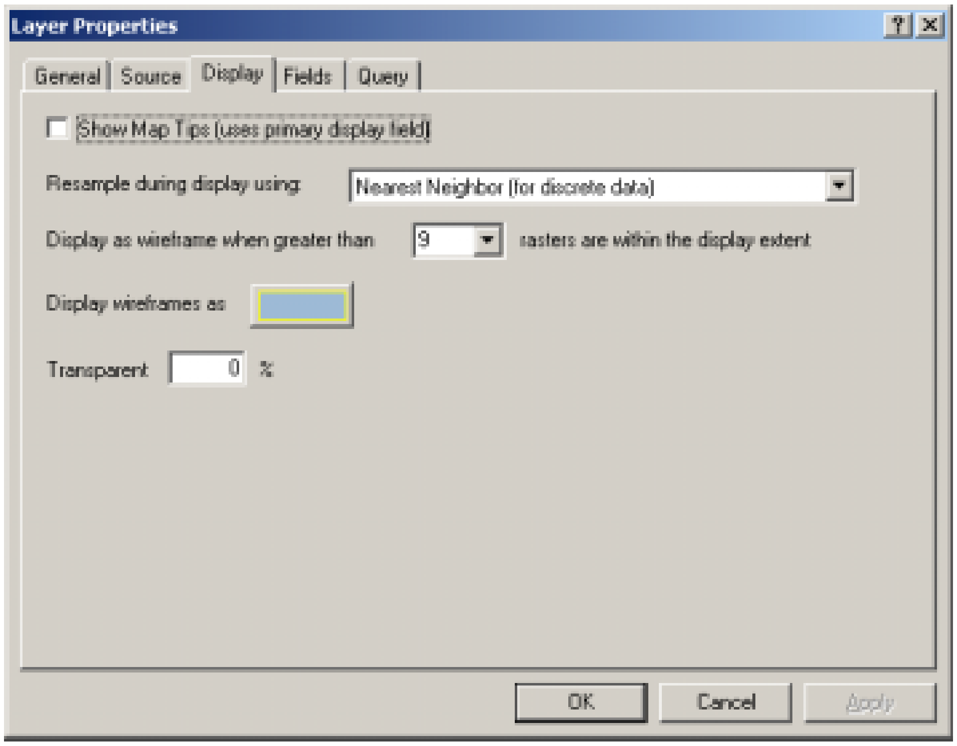
When loading one-bit raster data to the database, the user can specify the background value as no data value to be excluded. This will significantly reduce the time spent on loading and building pyramid layers and also improve drawing performance. This can be achieved by using ArcObjects or the SDERASTER command application. It is noticeably slower loading one-bit data using ArcObjects than using the SDERASTER command.

Viewing Data

To improve viewing performance of rasters, ensure statistics have been calculated on the layers.

ArcGIS

To improve the viewing performance of raster catalogs, set the layer properties in ArcMap to show a wire frame instead of the data when zoomed out or when many rasters will be visible within the display extent.



Show Map Tips (uses primary display field)

Resample during display using: Nearest Neighbor (for discrete data)

Display as wireframe when greater than 9 rasters are within the display extent

Display wireframes as

Transparent 0 %

The default resampling method for viewing raster layers in ArcGIS is nearest neighbor, which does not affect the method specified when loading the data. This is the fastest method of resampling when viewing the data.

Mosaic Versus Raster Catalog

Mosaicked rasters perform better than a raster catalog with the same set of rasters. Because a mosaicked raster is one entity, it eliminates the possibility of color mismatching that could happen in a raster catalog. However, it is easier for a single raster in a raster catalog to be accessed or updated than updating the same area of a mosaicked one, because updating a mosaicked raster means that pyramids and statistics will be removed.

Performance Example

Environment setting:

Server: Oracle 8*i*™ with ArcSDE 8.1.2, Windows NT, 512 MB RAM (dual CPU)
Client: ArcGIS 8.1.2, Windows 2000, 1GB RAM
Source data: 2000, 640 x 480, 8-bit, 1 band raster data (total extent of 32168 x 19680)

Test scenario:

The data was loaded as a seamless mosaic and raster catalog that included individual rasters in LZ77 compression.

Table 1
Performance Comparison of Raster Catalog and Seamless Mosaic in ArcGIS 8.1.2

Default setting (display catalog if <= 9 rasters)	Raster Catalog (single raster per layer)		Seamless Mosaic
Open connection		2 minutes	3 seconds
Full extent	(Wire frame)	40 seconds	3 seconds
16000 x 10000	(Wire frame)	16 seconds	1 second
8000 x 5000	(Wire frame)	14 seconds	1 second
4000 x 2500	(Wire frame)	14 seconds	1 second
2000 x 1250	(Wire frame)	13 seconds	1 second
1000 x 630	(Display data)	2 minutes	1 second

Attribute Table

The current server implementation of raster in ArcSDE does not support the concept of attribute tables, and ArcGIS cannot extract them from the ArcSDE server. However, the user can always add attributes to the raster business table and extract the attribute data by writing a customized application.

Copy/Paste Raster Data Between or Within Database

This function is not currently available from within ArcCatalog; however, the user can write a customized application to implement this using ArcObjects. This function is available using the SDERASTER command with –f "–l…" argument.

Limitations of Raster Catalogs in a Database

Each individual raster in a raster catalog is an entity in the database with its own business table. If a raster catalog contains many rasters, that means the database has at least that many raster feature classes. This will cause performance issues when trying to open the database connection due to the sheer number of layers in the database.

Appendix A—Bibliography

Related topics can also be found in the documentation that is installed with ArcSDE and hard-copy documentation with ArcSDE and ArcGIS.

Raster Concepts

- *Modeling Our World,* Michael Zeiler, ESRI, 1999. "Cell-based modeling with rasters," Ch. 9.
- *Understanding ArcSDE,* Robert West, ESRI, 2001. "Raster data in a geodatabase," pages 17–19.

Raster Data Storage

- *ArcSDE Configuration and Tuning Guide for DB2,* ESRI, 2001. "Appendix A, Storing raster data," pages 41–52 (pdf pages 45–56), config_tuning_guide_db2.pdf
- *ArcSDE Configuration and Tuning Guide for Informix,* ESRI, 2001. "Appendix B, Storing raster data," pages 81–92 (pdf pages 85–96), config_tuning_guide_informix.pdf
- *ArcSDE Configuration and Tuning Guide for Oracle,* ESRI, 2001. "Appendix B, Storing raster data," pages 119–129 (pdf pages 124–134), config_tuning_guide_oracle.pdf
- *ArcSDE Configuration and Tuning Guide for SQL Server,* ESRI, 2001. "Appendix C, Storing raster data," pages 179–189 (pdf pages 183–193), config_tuning_guide_sqlserver.pdf
- *Managing ArcSDE Services,* Mark Harris, ESRI, 2000. "Raster tables," pages 110–111 (pdf pages 116–117), Managing_ArcSDE_Services.pdf
- *Understanding ArcSDE,* Robert West, ESRI, 2001. "Raster Data Storage," page 35.

Performance and Tuning

- *ArcSDE Configuration and Tuning Guide for Oracle,* ESRI, 2001. "Appendix A, Estimating the size of your tables and indexes: The raster data tables," pages 114–117 (pdf pages 119–122), config_tuning_guide_oracle.pdf

ArcSDE Management

- *ArcSDE 8.1.2 Developer Help,* ESRI. "Administrator command references," "Concepts."
- *Managing ArcSDE Services,* "Raster parameters," page 35 (pdf page 41), Managing_ArcSDE_Services.pdf

Loading Data

- *Building a Geodatabase,* A. MacDonald, ESRI, 1999. "Migrating existing data to Geodatabase," pages 79–107.

Appendix B—Raster Table Schema

The section that follows describes the schema of the tables associated with the storage of raster data. It can also be found in the online ArcSDE 8.1.2 Developer Help, "Concepts" chapter.

RASTER_COLUMNS Table

When you add a raster column to a business table, ArcSDE adds a record to the RASTER_COLUMNS system table maintained in the sde user's schema. ArcSDE also creates four tables to store the raster images and metadata associated with each one.

NAME	*DATA TYPE*	*NULL?*
rastercolumn_id	SE_INTEGER_TYPE	NOT NULL
description	SE_STRING_TYPE(65)	NULL
database_name	SE_STRING_TYPE(32)	NULL
owner	SE_STRING_TYPE(32)	NOT NULL
table_name	SE_STRING_TYPE(160)	NOT NULL
raster_column	SE_STRING_TYPE(32)	NOT NULL
cdate	SE_DATE_TYPE	NOT NULL
config_keyword	SE_STRING_TYPE(32)	NULL
minimum_id	SE_INTEGER_TYPE	NULL
base_rastercolumn_id	SE_INTEGER_TYPE	NOT NULL
rastercolumn_mask	SE_INTEGER_TYPE	NOT NULL
srid	SE_INTEGER_TYPE	NULL

- rastercolumn_id—The table's primary key.
- description—The description of the raster table.
- database_name—The database that stores the table (Field is always NULL for Oracle).
- owner—The owner of a raster column's business table.
- table_name—The business table name.
- raster_column—The raster column name.
- cdate—The date the raster column was added to the business table.
- config_keyword—The DBTUNE configuration keyword whose storage parameters determine how the tables and indexes of the raster are stored in the database. For more information on DBTUNE configuration keywords and their storage parameters, review the *ArcSDE Configuration and Tuning Guide (DBMS Specific).*

- minimum_id—Defined during the creation of the raster, this establishes the value of the raster table's raster_id column.

- base_rastercolumn_id—If a view of the business table is created that includes the raster column, an entry is added to the RASTER_COLUMNS table. The raster column entry of the view will have its own rastercolumn_id. The base_rastercolumn_id will be the rastercolumn_id of the business table used to create the view. This base_rastercolumn_id maintains referential integrity to the business table. It ensures that actions performed on the business table raster column are reflected in the view. For example, if the business table's raster column is dropped, it will also be dropped from the view (essentially removing the view's raster column entry from the RASTER_COLUMNS table).

- rastercolumn_mask—Currently not used, this is maintained for future use.

- srid—The spatial reference ID (srid) is a foreign key reference to the SPATIAL_REFERENCES table. For images that can be georeferenced, the srid establishes the *x* and *y* offset translation factor and the scale factor for storage of the image coordinates into the 32-bit integer ArcSDE coordinate storage system. It also stores the coordinate reference system the image was created under.

Business Table

In the example that follows, the fictitious REDLANDS business table contains the raster column image. This is a foreign key reference to the raster table created in the user's schema. In this case the raster table contains a record for a TM image of Redlands.

NAME	***DATA TYPE***	***NULL?***
name	SE_INTEGER_TYPE	NOT NULL
image	SE_INTEGER_TYPE	NOT NULL

REDLANDS business table with house image raster column:

- name—The table's primary key

- image—A raster column and foreign key reference to a raster table containing the image

Raster Table (SDE_RAS_<raster column_id>)

The raster table, created as SDE_RAS_<raster_column_id> in the database, stores a record for each image stored in a raster column. The raster_column_id is assigned by ArcSDE whenever a raster column is created in the database. A record for each raster column in the database is stored in the ArcSDE RASTER_COLUMNS system table maintained in the sde user's schema.

NAME	***DATA TYPE***	***NULL?***
raster_id	SE_INTEGER_TYPE	NOT NULL
raster_flags	SE_INTEGER_TYPE	NULL
description	SE_STRING_TYPE(65)	NULL

Raster table schema

- raster_id—The primary key of the raster table and unique sequential identifier of each image stored in the raster table
- raster_flags—A bit map set according to the characteristics of stored image.
- description—A text description of the image (not implemented at ArcSDE 8.1.2)

Raster Band Table (SDE_BND_<raster column_id>)

Each image referenced in a raster may be subdivided into one or more raster bands. The raster band table, created as SDE_BND_<rastercolumn_id>, stores the raster bands of each image stored in the raster table. The raster_id column of the raster band table is a foreign key reference to the raster table's raster_id primary key. The rasterband_id column is the raster band table's primary key. Each raster band in the table is uniquely identified by the sequential rasterband_id.

NAME	***DATA TYPE***	***NULL?***
rasterband_id	SE_INTEGER_TYPE	NOT NULL
sequence_nbr	SE_INTEGER_TYPE	NOT NULL
raster_id	SE_INTEGER_TYPE	NOT NULL
name	SE_STRING_TYPE(65)	NULL
band_flags	SE_INTEGER_TYPE	NOT NULL
band_width	SE_INTEGER_TYPE	NOT NULL
band_height	SE_INTEGER_TYPE	NOT NULL
band_types	SE_INTEGER_TYPE	NOT NULL
block_width	SE_INTEGER_TYPE	NOT NULL
block_height	SE_INTEGER_TYPE	NOT NULL
block_origin_x	SE_FLOAT_TYPE	NOT NULL
block_origin_y	SE_FLOAT_TYPE	NOT NULL
eminx	SE_FLOAT_TYPE	NOT NULL
eminy	SE_FLOAT_TYPE	NOT NULL
emaxx	SE_FLOAT_TYPE	NOT NULL
emaxy	SE_FLOAT_TYPE	NOT NULL
cdate	SE_DATE_TYPE	NOT NULL
mdate	SE_DATE_TYPE	NOT NULL

Raster band table schema

- rasterband_id—The primary key of the raster band table that uniquely identifies each raster band.
- sequence_nbr—An optional sequential number that can be combined with the raster_id as a composite key as a second way to uniquely identify the raster band.
- raster_id—The foreign key reference to the raster tables primary key. Uniquely identifies the raster band when combined with the sequence_nbr as a composite key.
- name—The name of the raster band.
- band_flags—A bit map set according to the characteristics of the raster band.

- band_width—The pixel width of the band.
- band_height—The pixel height of the band.
- band_types—A bit map band compression data.
- block_width—The pixel width of the band's tiles.
- block_height—The pixel height of the band's tiles.
- block_origin_x—The left-most pixel.
- block_origin_y—The bottom-most pixel.
- If the image has a map extent, the optional eminx, eminy, emaxx, and emaxy will hold the coordinates of the extent.
- eminx—The band's minimum *x* coordinate.
- eminy—The band's minimum *y* coordinate.
- emaxx—The band's maximum *x* coordinate.
- emaxy—The band's maximum *y* coordinate.
- cdate—The creation date.
- mdate—The last modification date.

Raster Blocks Table (SDE_BLK_<raster column_id>)

Created as SDE_BLK_<rastercolumn_id>, the raster blocks table stores the actual pixel data of the raster images. ArcSDE evenly tiles the bands into blocks of pixels. Tiling the raster band data enables efficient storage and retrieval of the raster data. The raster blocks can be configured so that the records of the raster block table fit with an Oracle data block, avoiding the adverse effects of data block chaining.

The rasterband_id column of the raster block table is a foreign key reference to the raster band table's primary key. A composite unique key is formed by combining the rasterband_id, rrd_factor, row_nbr, and col_nbr columns.

NAME	***DATA TYPE***	***NULL?***
rasterband_id	SE_INTEGER_TYPE	NOT NULL
rrd_factor	SE_INTEGER_TYPE	NOT NULL
row_nbr	SE_INTEGER_TYPE	NOT NULL
col_nbr	SE_INTEGER_TYPE	NOT NULL
block_data	SE_BLOB_TYPE	NOT NULL

Raster blocks table schema

- rasterband_id—The foreign key reference to the raster band tables primary key.

- rrd_factor—The reduced resolution data set factor determines the position of the raster band block within the resolution pyramid. The resolution pyramid begins at 0 for the highest resolution and increases until the raster band's lowest resolution level has been reached.
- row_nbr—The block's row number.
- col_nbr—The block's column number.
- block_data—The block's tile of pixel data.

Raster Band Auxiliary Table (SDE_AUX_<raster column_id>)

The raster band auxiliary table, created as SDE_AUX_<rastercolumn_id>, stores optional raster metadata such as the image color map, image statistics, and a bit mask used for image overlay and mosaicking. The rasterband_id column is a foreign key reference to the primary key of the raster band table.

NAME	***DATA TYPE***	***NULL?***
rasterband_id	SE_INTEGER_TYPE	NOT NULL
type	SE_INTEGER_TYPE	NOT NULL
object	SE_BLOB_TYPE	NOT NULL

Raster band auxiliary table schema

- rasterband_id—The foreign key reference to the raster band table's primary key.
- type—A bit map set according to the characteristics of the data stored in the object column.
- object—May contain the image color map, image statistics, and other components.

Appendix C—8.1.2 SDERASTER Command

What's New

At ArcSDE 8.1.2, several new options have been added to SDERASTER.

- JPEG compression is added. The quality is variable from 5 to 95 percent.
- Export allows the definition of an extraction window, image color inversion, and specific band.
- It is possible to specify pixel values for nodata areas to be excluded when loading.

Usage Syntax

```
sderaster -h

sderaster -o add -l <table,column > [<-M minimum_id>]
          [-G {<projection_ID> | file=<proj_file>}]
          [-k <config_keyword>] [-S <description_str>]
          [-i <service> | <port#>] [-s <server_name>]
          [-D <database>] [-u <DB_user_name>]
          [-p <password>]

sderaster -o drop {-l < table,column > | -t < table > }
          [-i < service > | < port# > ] [-s < server_name > ]
          [-D < database > ] [-u < DB_user_name > ]
          [-p < password >]

sderaster -o truncate -l < table,column > [-i < service > | < port# >]
          [-s < server_name > ] [-D < database>] [-u <DB_user_name>]
          [-p <password>]

sderaster -o describe [-l <table,column>] [-V] [-i <service> | <port#>]
          [-s <server_name>] [-D <database>] [-u <DB_user_name>]
          [-p <password>]

sderaster -o list -l <table,column> [-v <raster_id>] [-V]
          [-i <service> | <port#>] [-s <server_name>]
          [-D <database>] [-u <DB_user_name>]
          [-p <password>]

sderaster -o insert -l < table,column >
          -f {< image_file > | < ArcSDE raster >} [-N]
          [ -c {lz77 | jpeg}] [-q <quality>] [-C rgb]
          [ {-R | -a <NoData>} ]  [ -n <image_name>]
          [ -L <pyramid_level>]  [ -I {nearest | bilinear | bicubic}]
          [ -t <tile_width,tile_height>]
           [ -G {<projection_ID> | file=<proj_file>}]
```

```
               [ -i <service > | < port# > ]
               [ -s <server_name> ] [ -D <database> ]
               [ -u <DB_user_name> ] [ -p <password> ]

sderaster -o delete -l <table,column> -v <raster_id>
               [-i <service> | <port#>] [-s <server_name>]
               [-D <database>] [-u <DB_user_name>]
               [-p <password>]

sderaster -o update -l <table,column> -v <raster_id>
              -f {<image_file> | <ArcSDE raster>} [-N]
              [-c {lz77 | jpeg}] [-q <quality>] [-C rgb]
              [{-R | -a <NoData>}] [-n <image_name>]
              [-L <pyramid_level>]  [-I {nearest | bilinear | bicubic}]
              [-t <tile_width,tile_height>]
              [-G {<projection_ID> | file=<proj_file>}]
              [-i <service> | <port#>]
              [-s <server_name>] [-D <database>]
              [-u <DB_user_name>] [-p <password>]

sderaster -o mosaic -l <table,column> -v <raster_id>
              -f {<image_file> | <ArcSDE raster>} [-N]
              [-C rgb] [-q <quality>] [{-R | -a <NoData>}]
              [-n <image_name>] [-L <pyramid_level>]
              [-I {nearest | bilinear | bicubic}]
              [-i <service> | <port#>]
              [-s <server_name>] [-D <database>]
              [-u <DB_user_name>] [-p <password>]

sderaster -o pyramid -l <table,column> -v <raster_id>
              [-L <pyramid_level>]
              [-I {nearest | bilinear | bicubic}]
              [-i <service> | <port#>] [-s <server_name>]
              [-D <database>] [-u <DB_user_name>]
              [-p <password>]

sderaster -o stats -l <table,column> -v <raster_id>
              [-i <service> | <port#>] [-s <server_name>]
              [-D <database>] [-u <DB_user_name>]
              [-p <password>]

sderaster -o colormap -l <table,column> -v <raster_id>
              {-d | -f <image_file>} [-i <service> | <port#>]
              [-s <server_name>] [-D <database>]
              [-u <DB_user_name>] [-p <password>]

sderaster -o import -l <table,column> [-gN] [-C rgb]
              -f {<image_file> | <ArcSDE raster>}
              [-c {lz77 | jpeg}] [-q <quality>]
              [{-R | -a <NoData>}] [-n <image_name>]
              [-M <minimum_id] [-G {<projection_ID> | file=<proj_file>}]
              [-k <config_keyword>] [-S <description_str>]
```

```
                  [-L <pyramid_level>] [-I {nearest | bilinear | bicubic}]
                  [-t <tile_width,tile_height>]
                  [-i <service> | <port#>]
                  [-s <server_name>] [-D <database>]
                  [-u <DB_user_name>] [-p <password>]

sderaster -o export -l <table,column> -v <raster_id>
                  -f <image_file> [-I]
                  [{-w | -e} <minx,miny,maxx,maxy>]
                  [-b <band_number>]
                  [-L <pyramid_level>]
                  [-i <service> | <port#>]
                  [-s <server_name>] [-D <database>]
                  [-u <DB_user_name>] [-p <password>]
```

Operations

add	Create a raster layer by adding a raster column to a business table.
drop	Drop a raster column or business table.
truncate	Truncate a raster layer.
describe	Describe one or all raster layers owned by a user.
list	List one or all rasters in a raster layer.
insert	Insert a raster into a raster layer.
delete	Delete a raster from a raster layer.
update	Update a raster.
mosaic	Perform piecewise update on a raster.
pyramid	Update a raster's image pyramid.
stats	Calculate a raster's image statistics and histogram.
color map	Update a raster's color map.
import	Import raster layer.
export	Export raster from a layer.

Operations

–a	Sets pixels with specified value as no data pixels.
–c	Compresses a raster upon entry. LZ77: uses the lossless LZ77 compression algorithm JPEG: uses the lossy JPEG compression algorithm
–C	rgb: expands the color map into a true color image
–d	Deletes color map.
–D	Database or data source name. Not supported by all database management systems (DBMS).
–e	Extraction window in world coordinates.
–f	The name of the image file or ArcSDE raster.
–g	Directs the import operation to register the raster layer with the geodatabase.
–G	Coordinate system specifier. <projection_id>: coordinate system ID (see the pedef.h file for the integer codes) file=<proj_file_name>: file containing coordinate system description string
–h	Prints usage and options.
–i	ArcSDE service name or port number

–o export only
–I Inverts bilevel images

–I The resampling technique used during the construction of the pyramid.
nearest: The nearest neighbor method selects the closest pixel.
bilinear: The bilinear method interpolates four adjacent pixels.
bicubic: The bicubic method interpolates 16 adjacent pixels.
For more information on pixel resampling, refer to *Using ArcGIS Spatial Analyst*.
–k Configuration keyword present in DBTUNE table. The storage parameters specific to the raster column will be found under the specified keyword.
–l The raster layer's business table and raster column. If you are not the owner of the table, you must qualify the table name as "owner.table".
–L The pyramid level.
Set to a number greater than 0, ArcSDE creates the levels specified unless the apex is reached first.
Set to –1, ArcSDE calculates the pyramid level.
Set to 0, the pyramid is deleted.
–M Minimum feature ID. New raster IDs are assigned the larger of the minimum ID or the maximum assigned ID plus one.
–n Image name.
–N Ignores color map in data source.
–p ArcSDE user DBMS password.
–q Compression quality for JPEG (5–95).
–R Removes pixels with background color in a rotated image.
–s ArcSDE server host name (default: localhost).
–S The raster description (quoted string).
–t The raster tile width and height measured in pixels. Each tile is stored as a separate raster block.

–o drop only
–t DBMS table name

–u ArcSDE user DBMS user name.
–v The raster ID.
–V Enable verbose mode to describe all properties.
–w Extraction window in pixel coordinates.

Notes

The codes for the coordinate system can be found in the *ArcSDE Developer Help* for pedef.h.

The supported external raster formats are ESRI BSQ and TIFF. However, to use TIFF format, you will need to install the open source libtiff library. For information on obtaining the open source libtiff library for your platform, refer to http://www.libtff.org. For the Solaris platform, the libtiff.so is included under /usr/openwin/lib. For the Solaris platform, be sure to include /usr/openwin/lib in your LD_LIBRARY_PATH.

The typical scenario for using the SDERASTER command is to create a raster layer (a business table and associated raster tables) with the import operation followed by subsequent executions of the mosaic operation to input additional image files. Finally, the pyramid operation is applied with the level set to –1, instructing SDERASTER to create a full pyramid.

Be sure to include the –g option during the import operation if you want the raster to appear in the Table of Contents of either ArcCatalog of ArcMap. Specifying –g creates a raster with an image name of ESRI_SDERASTERDATASET that is detectable by ArcCatalog and ArcMap.

The other operations of the SDERASTER command are used to make adjustments to the raster layer.

Use the add operation to add a raster column to an empty business table.