



Raster Data in ArcSDE[®] 8.3

An ESRI[®] White Paper • September 2003

Copyright © 2003 ESRI
All rights reserved.
Printed in the United States of America.

The information contained in this document is the exclusive property of ESRI. This work is protected under United States copyright law and other international copyright treaties and conventions. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as expressly permitted in writing by ESRI. All requests should be sent to Attention: Contracts Manager, ESRI, 380 New York Street, Redlands, CA 92373-8100, USA.

The information contained in this document is subject to change without notice.

U.S. GOVERNMENT RESTRICTED/LIMITED RIGHTS

Any software, documentation, and/or data delivered hereunder is subject to the terms of the License Agreement. In no event shall the U.S. Government acquire greater than RESTRICTED/LIMITED RIGHTS. At a minimum, use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in FAR §52.227-14 Alternates I, II, and III (JUN 1987); FAR §52.227-19 (JUN 1987) and/or FAR §12.211/12.212 (Commercial Technical Data/Computer Software); and DFARS §252.227-7015 (NOV 1995) (Technical Data) and/or DFARS §227.7202 (Computer Software), as applicable. Contractor/Manufacturer is ESRI, 380 New York Street, Redlands, CA 92373-8100, USA.

ESRI, the ESRI globe logo, ArcCatalog, ArcGIS, ArcIMS, ArcInfo, ArcMap, ArcObjects, ArcSDE, ArcToolbox, ArcView, SDE, Spatial Database Engine, www.esri.com, and @esri.com are trademarks, registered trademarks, or service marks of ESRI in the United States, the European Community, or certain other jurisdictions. Other companies and products mentioned herein are trademarks or registered trademarks of their respective trademark owners.

Raster Data in ArcSDE 8.3

An ESRI White Paper

Contents	Page
Introduction.....	1
Other Suggested Resources	1
Why Put Images in a Database?.....	1
Multiuser Access.....	1
Data Management	1
Data Security.....	1
Data Query	1
Uses of ArcSDE Raster Data	2
Basemaps—Enterprise GIS (utility or local government).....	2
Data Management/Distribution (data providers)	2
Feature Attributes (utility, real estate, or local government)....	2
Basic Raster Concepts	2
How Raster Data Is Stored in the Database.....	3
Supported Input Formats	4
Storage Parameters	4
Pyramids	5
Tile Size	6
Compression	6
Configuration Keyword	6
Append.....	6
Update.....	6
Statistics	6
Custom Applications.....	6
Data Loading.....	7
Preprocessing of Continuous Rasters	7
Loading Parameters	7
Data Loading in ArcGIS	9
To a New SDE Raster	9
To an Existing SDE Raster	11
Batch Loading.....	12

Contents	Page
Data Loading With SDERASTER.....	12
Creating a Mosaic.....	12
Requirements for Appending Rasters to an Existing Raster in a Database.....	13
What Is Exact Pixel Registration?	13
Creating a Raster Catalog.....	13
Referenced Raster Catalog.....	14
Embedded Raster Catalog.....	14
 ArcObjects Sample	 15
ArcObjects for Loading	15
Loading Sample	16
 Data Viewing	 18
Viewing From ArcGIS.....	19
ArcCatalog	19
ArcMap	19
Viewing in ArcIMS	19
 Tips and Tricks	 20
Loading Data.....	20
Storage	20
Compression With ArcObjects	20
Georeferencing.....	20
Mosaic.....	20
Mosaic Rasters With Color Map	20
No Data	21
Loading One-Bit Data.....	21
Viewing Data	21
ArcGIS	21
Mosaic Versus Raster Catalog.....	22
Limitations of Raster Catalogs by Reference in a Database.....	22
Performance Example	22
Attribute Table.....	22
Copy/Paste Raster Data Between or Within Database	22
 Appendix A—Bibliography.....	 23
Raster Concepts	23
Raster Data Storage	23
Performance and Tuning.....	23

Contents	Page
ArcSDE Management	23
Loading Data	23
Appendix B—Raster Table Schema	24
RASTER_COLUMNS Table	24
Business Table	25
Raster Table	
(SDE_RAS_< rastercolumn_id >)	25
Raster Band Table	
(SDE_BND_< rastercolumn_id >)	26
Raster Blocks Table	
(SDE_BLK_< rastercolumn_id >)	27
Raster Band Auxiliary Table	
(SDE_AUX_< rastercolumn_id >)	28
Appendix C—SDERASTER Command	29
Usage Syntax	29
Operations	31
Options	31
Notes	32

Raster Data in ArcSDE 8.3

Introduction This document highlights the basic concepts, user experience, and product overview for the support of raster data in ESRI® ArcSDE®. The focus of the document is to explain how ArcSDE 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® 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?

Multuser 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, metadata, and tabular data. A database also provides access to extremely large images (many gigabytes to many terabytes) of continuous spatial data (e.g., 30-meter digital elevation model composite of North America).

Data Security

A database has tools for multiple security levels to be established and enforced. Users can be given access to the imagery that is relevant to the job they are being asked to work on.

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)*

A water supply company had a lot of legacy data stored on paper and Mylar® maps. It needed to be able to supply these to its users as background images to be combined with vector data into a seamless hybrid map. Eventually, these images will be completely replaced by vector data. The images were scanned as one-bit TIFF images. The company has stored about 3,000 (40 GB) images centrally using ArcSDE and supplies them as background images to users running ArcMap™. Its users are inserting new vector data as well as updating existing vector data with new information.

*Data Management/
Distribution (data
providers)*

An association of governments needed a central repository of imagery that could be easily accessed by its members as well as citizens who requested the imagery. It needed to provide the images in downloadable formats as well as set up and maintain a Web site that allowed interested parties to view the images online. The data was in eight-bit, three-band, one-meter digital orthophoto quarter quadrangles (DOQQ). It has stored 1 TB of imagery using ArcSDE with more images expected as the areas are reflown by satellites and aircraft.

*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.

Some rasters have a single band (a measure of some characteristic) of data while others have multiple bands (more than a single measure). 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 grayscale aerial photo has a single band representing different levels of the land's surface reflectance. 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 (2⁸). 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).

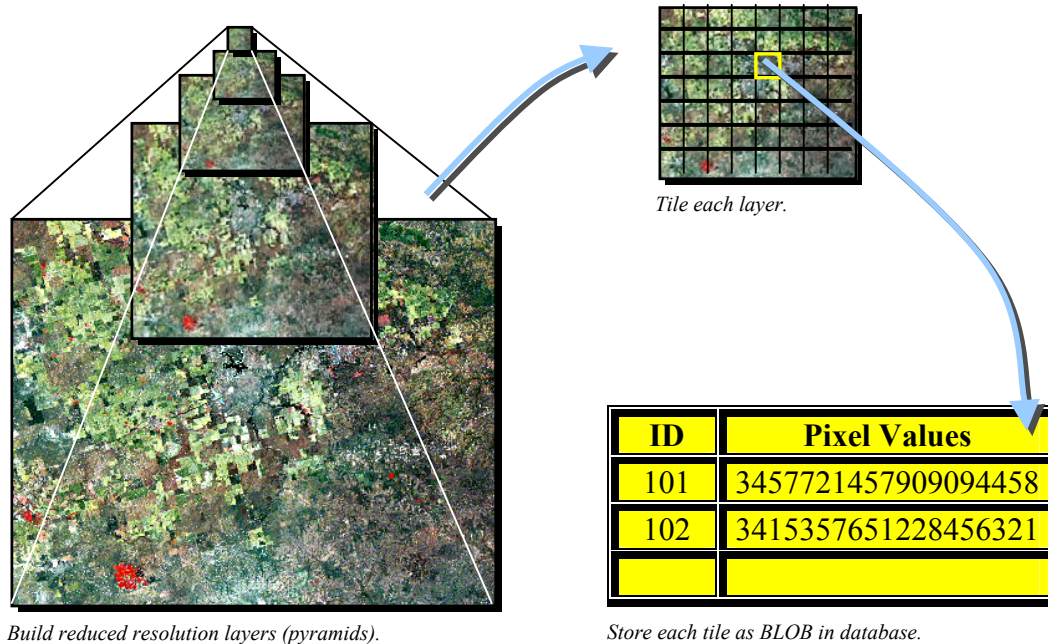
Raster layers are seen by client applications as one of two distinct types.

- **Raster Data Set**—A single picture of an object or a seamless image covering a spatially continuous area. This may be a single original image or the result of many images appended together.

- **Raster Catalog**—A collection of rasters displayed as a single layer. These must all be in the same coordinate system and should have the same data type.

How Raster Data Is Stored in the Database

When raster data is loaded into a database, it is converted into the Spatial Database Engine™ (SDE®) raster format. The raster is tiled using the user-specified tile size, and spatial indexes are built. The data is then resampled using the method specified to create pyramids. The tiles are stored as many small binary large objects (BLOBs) in a set of ArcSDE system and user tables. By doing this, when the raster is queried, only the necessary tiles are returned instead of the whole data set, thus improving end user performance. Client display performance is optimized by reducing the amount of data transferred to the client application. This makes it possible to store large seamless raster data sets and serve them quickly to a client for display.

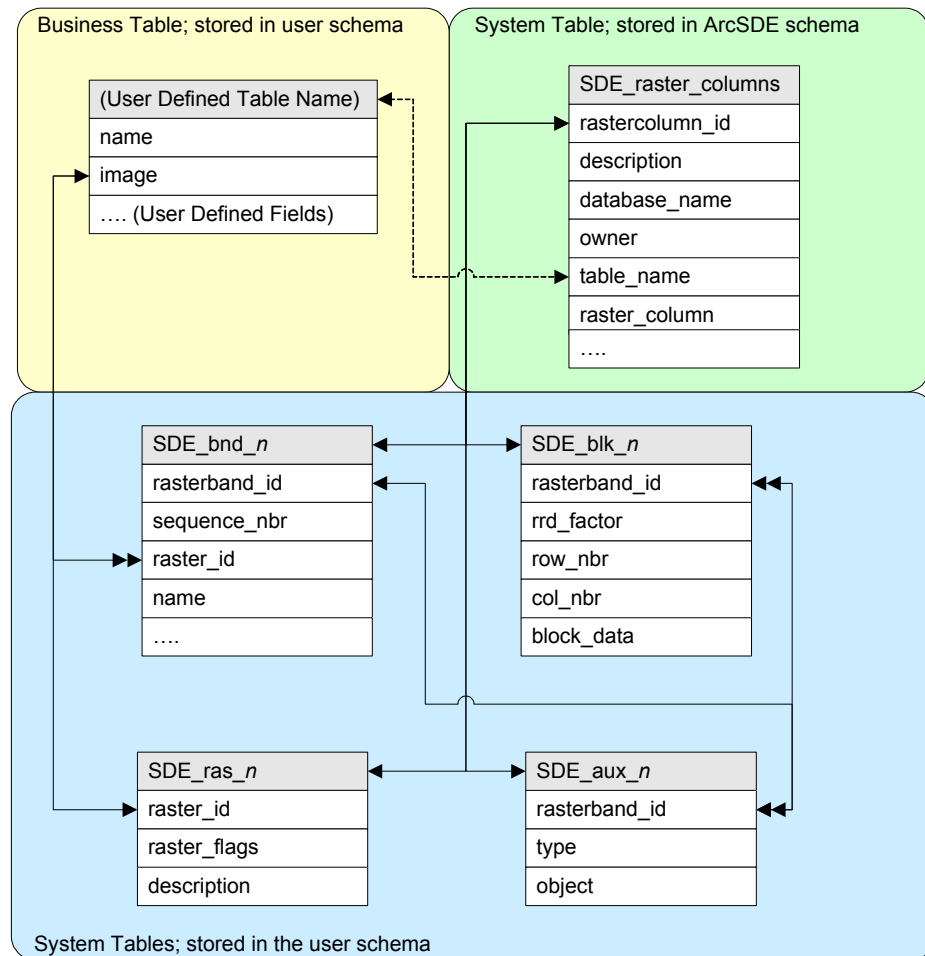


When a raster is created in an ArcSDE database, ArcSDE adds a raster column to the business table of the user's choice. This table may be an existing table or created for the user. Users may name the raster column whatever they like, as long as the name conforms to the underlying database column naming 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 for each 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 block table (SDE_blk_n) in a many-to-one relationship. The raster block table's rows store the tiles 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.



Supported Input Formats

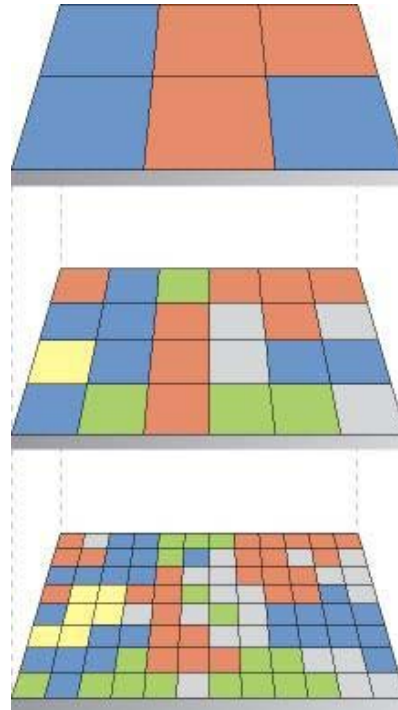
The ArcGIS data loading application is built on top of Raster Data Objects (RDO) and ArcObjects and can access rasters in a large number of standard formats. The same formats supported by ArcMap and ArcCatalog™ can be loaded into ArcSDE. See ArcCatalog > Working with rasters > Supported Raster Formats in *ArcGIS Desktop Help* for a full listing.

SDERASTER is a powerful ArcSDE command line tool that supports importing TIFF or band sequential (BSQ) formats.

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. Parameters are specified when loading raster data to the database.

Pyramids Pyramids are reduced resolution representations of the data set that are used to improve display performance. The base layer of the pyramid is always the highest resolution. Pyramids can speed up display of raster data by fetching only the data at a specified resolution that is required for the display.



Pyramid Creation. The spatial extent stays the same, but cell sizes change.

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.

1. Nearest neighbor assignment should be used for nominal or ordinal data, where each value represents a class, member, or classification (categorical data such as land use, soil, or forest type), or for color mapped data such as scanned topographic sheets.
2. Bilinear interpolation should be used for continuous data.
3. Cubic convolution should also be used for continuous data.

Continuous data types include elevation, slope, intensity of noise from an airport, salinity of the groundwater near an estuary, and satellite or aerial imagery.

If continuous data is used, then benchmarking to compare image quality and performance with either of the last two interpolation methods should be used. Pyramid building is performed on the ArcSDE server side. If the original data is compressed, the server will

first decompress the data, then build the pyramids and compress the data again to insert into the block table.

Tile Size The tile size controls the number of pixels users want to store in each BLOB field. 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) will 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 will create fewer records in the block table. If users do not want to use the default setting, they can experiment with data to choose a tile size.

Compression Data compression (optional, but recommended) compresses the tiles of data before storing them in the geodatabase. The compression methods available are none, LZ77, or JPEG. The LZ77 algorithm is a "lossless" compression, meaning the unique values of cells in the raster data set can be recovered. This is the same compression used by the PNG image format and in ZIP compression. JPEG has a high compression ratio but is "lossy," meaning the values of cells in the raster data set may be changed slightly. JPEG compression only applies to unsigned eight-bit data without a color map. The user can specify quality for JPEG compression using values from five to 95, where 95 is the best quality and 75 is the default.

The primary benefit of compressing data is that compressed data requires less storage space and is smaller to transfer to the client application. This results in better display performance at the client application. The amount of compression will depend on the data. The fewer unique cell values, the higher the compression ratio. The ArcSDE client performs compression and decompression. Where retention of pixel values is important, such as in categorical data or data used for analysis, use LZ77 compression. If individual pixel values are not important, such as in simple background images, use JPEG compression.

Configuration Keyword The configuration keyword specifies 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. Note that the largest table will be the SDE_blk_n table as it holds the actual image data. See the *ArcSDE Configuration and Tuning Guide* for the DBMS for more information.

Append Appending, or mosaicking, to an existing raster in the database creates one seamless raster. Any overlapping areas are 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 the input raster to the database with the name of the existing raster.

Statistics The statistics of each band 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. The 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 supported ArcSDE 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. 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 user interface (GUI)
- 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. The ArcObjects Developer Help has several sample scripts that can help with preprocessing of images.

Loading Parameters

In ArcGIS, rasters can be loaded into the database as individual layers or as seamless layers by appending them 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 grabbing only 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 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.

■ Platform Support

- SDERASTER is available for all the platforms on which ArcSDE is supported (HP® Tru64 UNIX™ and HP-UX®, IBM® AIX®, Red Hat® Linux®, SGI™ IRIX™, Sun™ Solaris™, and Windows NT® and 2000).
- ArcGIS tools are supported on Windows NT, Windows 2000, and Windows XP.

■ User Interface

- SDERASTER is a command line tool with no GUI.
- The ArcGIS tools are Windows applications that have a GUI.

■ Input Formats

- SDERASTER supports importing TIFF or BSQ formats.
- The ArcGIS tools can load a wide range of formats (including GeoTIFF).

■ Automation

- SDERASTER can be automated using operating system shell scripts.
- ArcGIS tools can be customized using ArcObjects; sample scripts are available in the ArcObjects Developer Help. The ArcGIS tools offer a batch mode that is able to convert multiple rasters at one time.

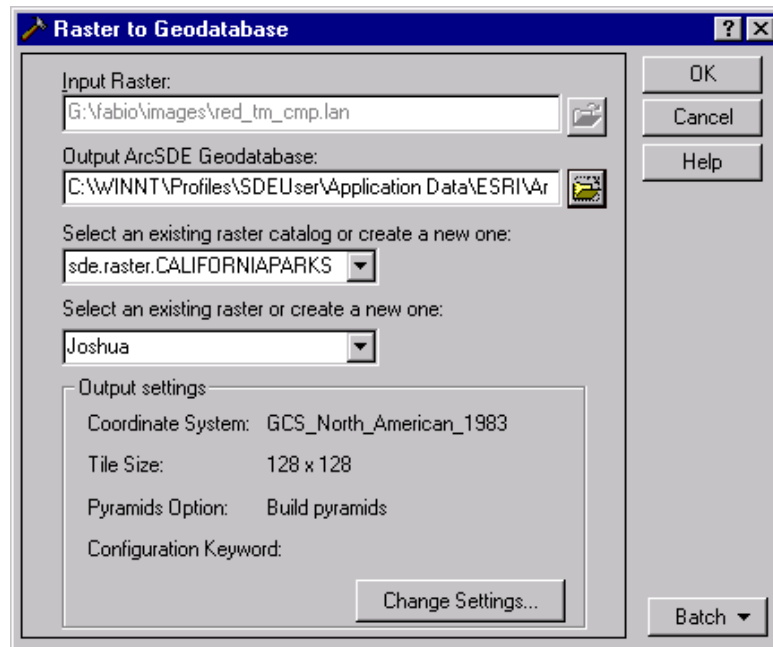
- Data Loading Options
 - SDERASTER allows the user to specify custom pyramid levels, remove color maps on loading, specify no data values, and specify whether or not to build pyramids or statistics at load time.
 - ArcGIS tools allow you to specify whether or not to build pyramids or statistics at load time.
- Data Exporting Options
 - SDERASTER allows the clipping and extraction of a raster layer.
 - ArcGIS can clip and extract portions of a layer with the use of ArcObjects.
- Mosaicking Requirements
 - SDERASTER requires the rasters to be spatially referenced and have the same bit depth, the same number of bands, exact pixel registration, and the same cell size.
 - ArcGIS specifies that raster must be spatially referenced, have no color map, and have the same bit depth.

Data Loading in
ArcGIS

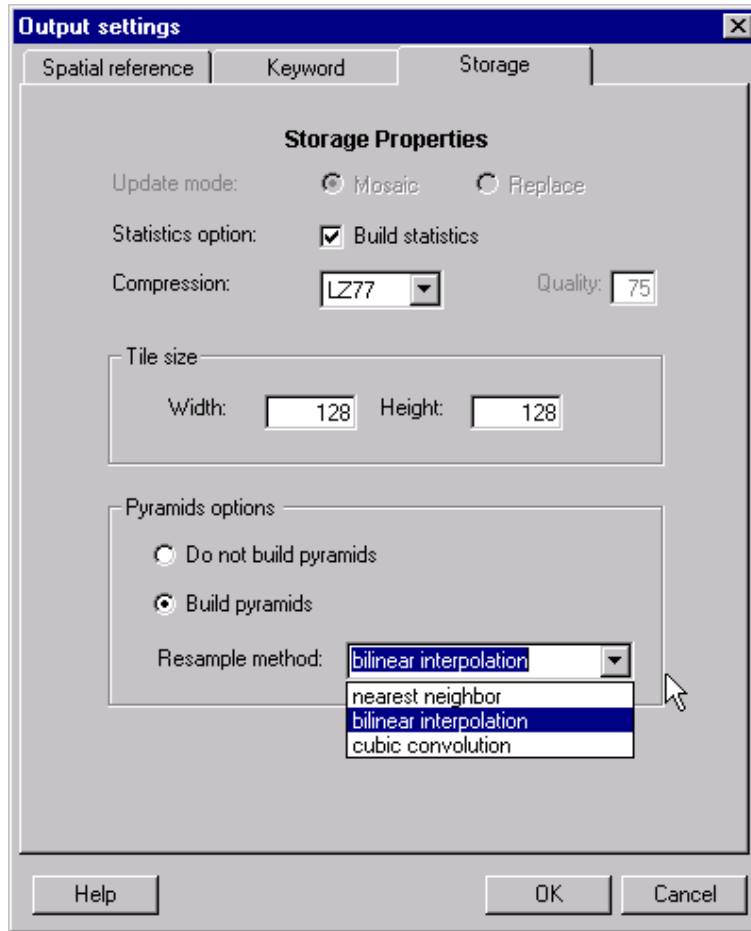
The source raster can either be file-based or in a database, and it can be imported to either an existing or new SDE raster. The Raster to Geodatabase tool or wizard is in ArcCatalog and ArcToolbox. Rasters also can be loaded into a geodatabase using Export -> Raster to Different Format in ArcCatalog; 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 a 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 (this does not reproject to the new spatial reference). If you would like to create a raster catalog, enter the name for a catalog into the Raster Catalog field.



Other raster properties, including pyramids options, tile (block) size, configuration keyword, and compression type, can be defined at the time of importing data by clicking Change Settings. 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.



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 all the 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 using the import option with the SDERASTER command. Occasionally, warnings will be returned if the TIFF file contains tags not supported by the TIFF format standards.

```
sderaster -o import -l state,image -c lz77 -f 3296011A.tif -n a3296011 -G 26714 -t 128,128 -s mapserver -D raster -i esri_sde -u mark -p pwd
```

```
3296011A.tif: Warning, unknown field with tag 33550 (0x830e) ignored.
3296011A.tif: Warning, unknown field with tag 33922 (0x8482) ignored.
3296011A.tif: Warning, unknown field with tag 34735 (0x87af) ignored.
3296011A.tif: Warning, unknown field with tag 34737 (0x87b1) ignored.
Connecting to server mapserver, instance esri_sde, as user mark
```

```
Creating user table: state
Creating raster layer: state.image
```

```
Image Dimension.....: 6614, 7663, 3
Pixel Type.....: uchar
Image Extent.....:
  minx   : 686516.500000000000000
  miny   : 3645833.000000000000000
  maxx   : 693129.500000000000000
  maxy   : 3653495.000000000000000
Raster ID : 1
```

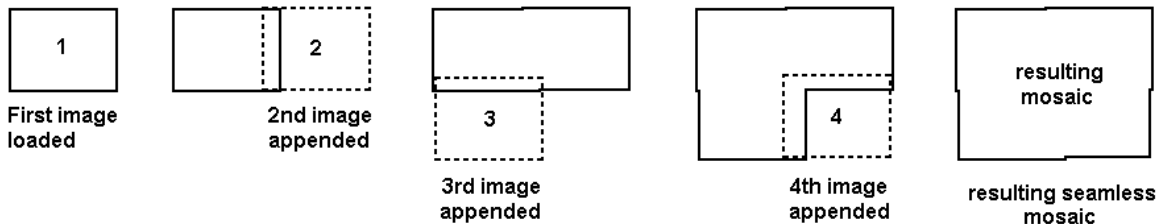
Total Time: 00:01:50

Complete...

For more command options, see Appendix B in *ArcSDE Developer Help* or visit <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.



Note that the images being appended may overlap or even have gaps between 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 a 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.

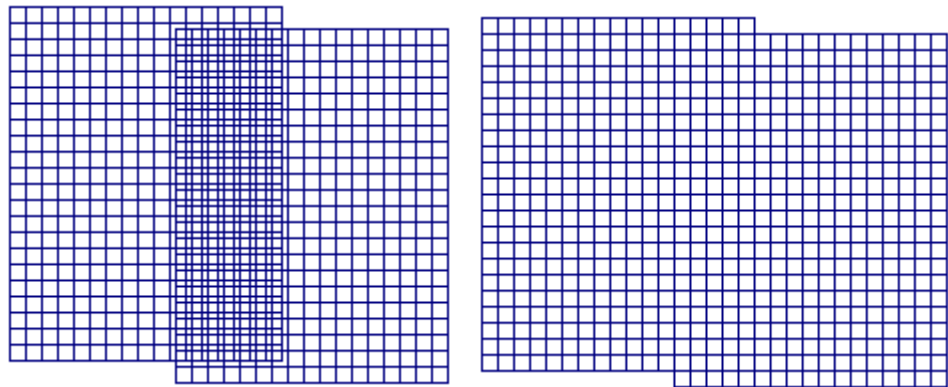
For the SDERASTER command line loader, there are two additional requirements.

- 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. Visit ArcObjects Online (<http://arcobjectsonline>) for sample Visual Basic for Applications (VBA) code to align your imagery.

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 way of displaying individual rasters as a seamless layer without having to mosaic them. They are useful for users who expect to get frequent data updates, need to maintain the data from areas where images overlap, or need to manage their holdings. ArcSDE supports two types of raster catalogs: Referenced Raster Catalogs that reference separately stored raster layers and Embedded Raster Catalogs in which the individual rasters are stored in one business table. Each raster in a catalog must use the same spatial reference and should have the same spectral properties. The decision of which to use will depend on the application.

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

Referenced Raster Catalog

This type of raster catalog is a stand-alone database business table that references rasters stored elsewhere on disk or in the database. This table must be created manually in the RDBMS being used. There are six fields required.

Column Name	Data Type
IMAGE	String
XMIN	Float
YMIN	Float
XMAX	Float
YMAX	Float
RASTER_ID	Integer

This example SQL code creates a raster catalog named "mycatalog". Run this in the SQL execution environment for your RDBMS.

```
CREATE TABLE 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, RASTER_ID integer)
```

The value for the IMAGE field is the full path and file name of the file on disk or the fully qualified database name for the raster in the ArcSDE database [`<database>.<owner>.<raster>`], the same as it appears in the ArcCatalog tree view. The RASTER_ID field is used if referencing individual images stored in an embedded raster catalog. The value is the rastercolumn_id of the raster.

IMAGE	XMIN	YMIN	XMAX	YMAX	RASTER_ID
[<Database>].<Owner>.<Raster1>	0	0	100.50	200.50	1
\\computer\share\image.tif	100.5	0	150	200.50	

Embedded Raster Catalog

This type of raster catalog is a business table with a raster column that can have multiple rows corresponding to individual rasters. An embedded Raster Catalog is created by loading multiple rasters into the same business table by using the Raster to Geodatabase tool or the SDERASTER command. It has an advantage over the referenced image catalog by reducing the number of unique layers that must be stored and accessed by the clients. This will improve the performance of clients that list the layers inside the database connection.

If using the SDERASTER command, do not register the first image with the geodatabase.

```
sderaster -o import -l mapbook,image -C rgb -f p105c.tif -c jpeg -q 25 -n p105c -G file=marks.prj -L -l -I nearest -t 128,128 -i esri_sde -s mapserver -D raster -u mark -p pass
```

```
p105c.tif: Warning, unknown field with tag 33550 (0x830e) ignored.
p105c.tif: Warning, unknown field with tag 33922 (0x8482) ignored.
p105c.tif: Warning, unknown field with tag 34735 (0x87af) ignored.
p105c.tif: Warning, unknown field with tag 34737 (0x87b1) ignored.
Connecting to server mapserver, instance esri_sde, as user mark
```

```
Creating user table: mapbook
Creating raster layer: mapbook.image
```

J-8908

```

Image Dimension.....: 1369, 1643, 1
Pixel Type.....: uchar
Image Extent.....:
  minx   :          522498.0000000000000000
  miny   :          5247999.827754108200000
  maxx  :          524998.367620206900000
  maxy   :          5251001.000000000000000
Raster ID : 1

```

Total Time: 00:00:04

Complete...

Once the first image has been loaded, use `SDERASTER` to insert another image into the same business table.

```

sderaster -o insert -l mapbook,image -f p106c.tif -c jpeg -q 25 -C rgb -n
p106c -L -l -I nearest -t 128,128 -G file=marks.prj -i esri_sde -s
mapserver -D raster -u mark -p pass

```

```

p106c.tif: Warning, unknown field with tag 33550 (0x830e) ignored.
p106c.tif: Warning, unknown field with tag 33922 (0x8482) ignored.
p106c.tif: Warning, unknown field with tag 34735 (0x87af) ignored.
p106c.tif: Warning, unknown field with tag 34737 (0x87b1) ignored.
Connecting to server mapserver, instance esri_sde, as user mark

```

```

Image Dimension.....: 1369, 1642, 1
Pixel Type.....: uchar
Image Extent.....:
  minx   :          524998.0000000000000000
  miny   :          5248000.691015339500000
  maxx  :          527499.171658144680000
  maxy   :          5251001.000000000000000
Raster ID : 2

```

Total Time: 00:00:03

Complete...

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 makes 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 both raster data set name string, and 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 a 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
```

J-8908

```

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

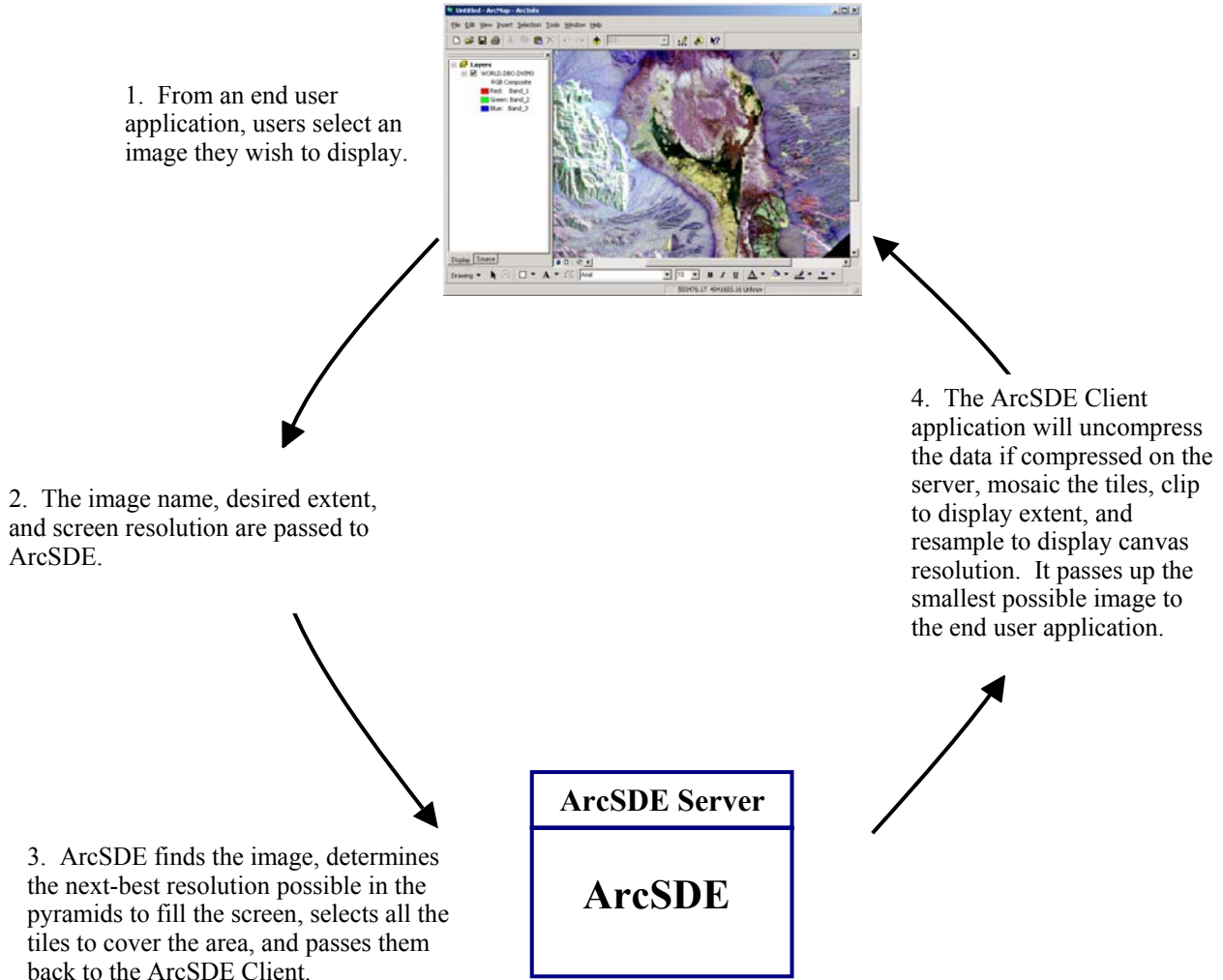
  ' ----- Get next datasetname -----
  Set pDsName = pEnumDNs.Next
  iCount = 1
Loop

' ----- 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.



Viewing From ArcGIS


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 pixel 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 using ArcIMS®. Use a text editor to create the AXL file referencing your raster layer. 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" geindexdir="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 be built and statistics calculated after loading. Loading with the SDERASTER command application is about 20 percent faster than ArcObjects on tested data. Because building pyramids is computationally intensive, always use the machine(s) with the fastest processor to build them; in some cases this may mean using direct connect from a client machine.

By using separate loading processes to load rasters into different rows (images) in the same raster catalog, it is possible to speed up the loading of imagery. As performance will vary depending on the processor, it is recommended that the load process be tested on a small amount of data. It is not possible to have multiple processes load into the same image simultaneously.

Storage Each raster tile is stored as a BLOB in the raster block table 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.

Compression With ArcObjects The RDO constants that hold compression information are referenced by IRasterSDEStorage. The constants are as follows:

Constant	Value	Description
esriRasterSdeCompressionTypeUncompressed	0	No compression.
esriRasterSdeCompressionTypeRunLength	1	LZ77 encoding.
esriRasterSdeCompressionTypeJPEG	2	JPEG encoding.

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. (Search *ArcObjects Developer Help* for "createworldfile". This is a tool to create a world file from a raster.)

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.

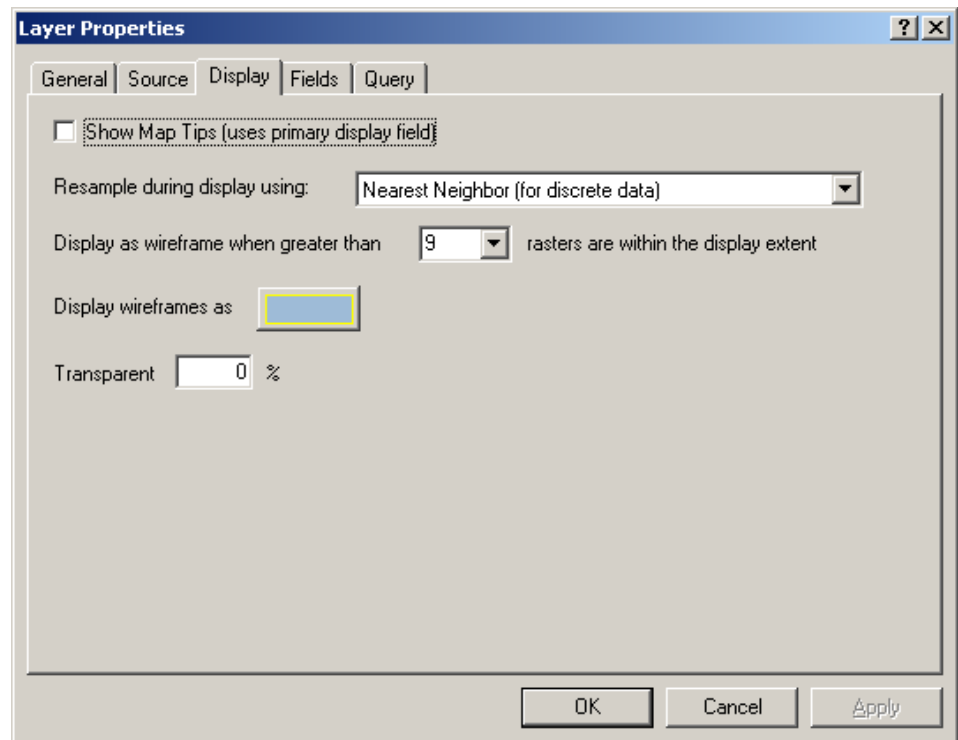
Mosaic Rasters With Color Map As a requirement, if any of the rasters intended for mosaicking operation have a color map, the operation will fail in ArcGIS. However, if the rasters are in TIFF or BSQ format, the user can use the SDERASTER command to load and mosaic the data using -N option to ignore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

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 ArcObjects or ArcGIS loading tools are used. 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`, an 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 that 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.



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.

Limitations of Raster Catalogs by Reference in a Database

Each individual raster in a raster catalog by reference 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.

Performance Example

Environment setting

Server: Oracle 8i™ with ArcSDE 8.1.2, Windows NT, 512 MB RAM (dual CPU)
 Client: ArcGIS 8.1.2, Windows 2000, 1 GB RAM
 Source data: 2000, 640 x 480, 8-bit, 1-band raster data (total extent of 32,168 x 19,680)

Test scenario

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

Table 1
Display 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
16,000 x 10,000	(Wire frame) 16 seconds	1 second
8,000 x 5,000	(Wire frame) 14 seconds	1 second
4,000 x 2,500	(Wire frame) 14 seconds	1 second
2,000 x 1,250	(Wire frame) 13 seconds	1 second
1,000 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.

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," Chapter 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 8.2 Developer Help*, ESRI. "Administrator Command References," Concepts chapter.
- *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 8.2 Developer Help*, Concepts chapter.

RASTER_COLUMNS Table

When the user adds 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 Thematic Mapper 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_<rastercolumn_id>)

The raster table, created as SDE_RAS_<rastercolumn_id> in the database, stores a record for each image stored in a raster column. The rastercolumn_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 8.2)

Raster Band Table
(SDE_BND_<rastercolumn_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_<rastercolumn_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.

<i>NAME</i>	<i>DATA TYPE</i>	<i>NULL?</i>
<code>rasterband_id</code>	<code>SE_INTEGER_TYPE</code>	NOT NULL
<code>rrd_factor</code>	<code>SE_INTEGER_TYPE</code>	NOT NULL
<code>row_nbr</code>	<code>SE_INTEGER_TYPE</code>	NOT NULL
<code>col_nbr</code>	<code>SE_INTEGER_TYPE</code>	NOT NULL
<code>block_data</code>	<code>SE_BLOB_TYPE</code>	NOT NULL

Raster blocks table schema

- `rasterband_id`—The foreign key reference to the raster band table's 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_< rastercolumn_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—SDERASTER Command

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 | grayscale]
[ {-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 | grayscale]
[{-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 | grayscale] [-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 | grayscale]
-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.

Options	-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	Converts data when loading rgb: expands the color map into a true color image. grayscale: converts one-bit data to eight-bit data
	-d	Deletes color map.
	-D	Database or data source name. Not supported by all DBMSs.
	-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.
	-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 <i>Using ArcGIS Spatial Analyst</i> .
	-I	Inverts bilevel images (WITH -o export ONLY).
	-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.
 - t DBMS table name (**WITH -o drop ONLY**).
 - 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.

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 or 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.