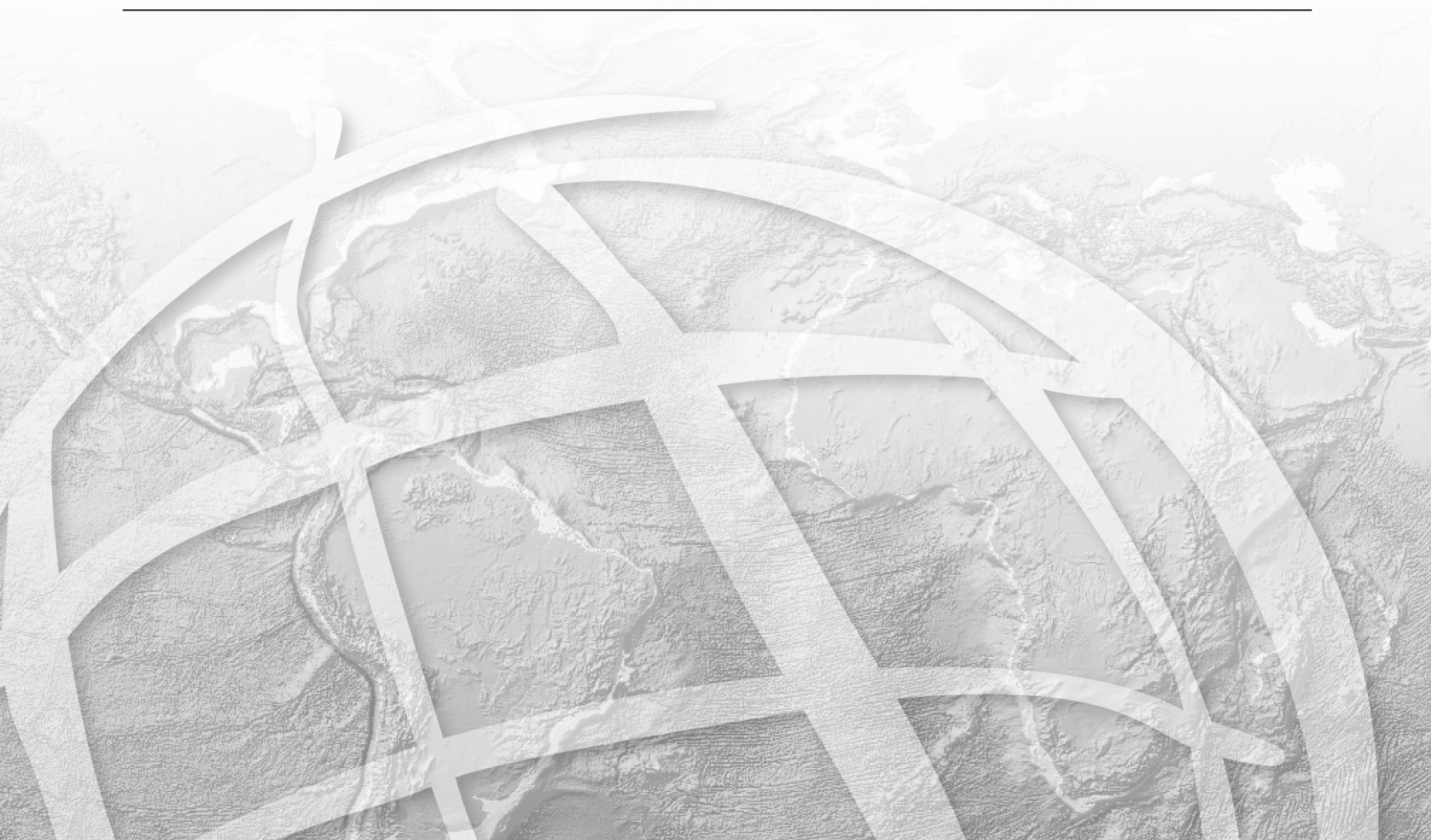


# ArcGIS® 9

---



Copyright © 1999–2005 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.

#### **DATA CREDITS**

*Graphical Editing map:* Wilson, North Carolina

*Universal Data Editor map, Editing in Data view and Layout view map:* Greeley, Colorado

*Context Menus and Shortcut Keys map:* P.F.R.A., Regina, Saskatchewan, Canada

*Quick-start tutorial data:* Wilson, North Carolina; Greeley, Colorado

#### **CONTRIBUTING WRITERS**

Andrew Perencsik, Simon Woo, Bob Booth, Scott Crosier, Jill Clark, Andy MacDonald

#### **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, ArcGIS, ArcMap, ArcCatalog, ArcInfo, ArcSDE, ArcToolbox, ArcIMS, ArcReader, ArcEditor, ArcStorm, SDE, Spatial Database Engine, ArcView, ArcObjects, GIS by ESRI, the ArcGIS logo, and www.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.

# Contents

## **1 Introduction 1**

- Creating a geodatabase from an existing design 3
- Creating a geodatabase from scratch 4
- Geodatabases and ArcCatalog 7
- Geodatabases and ArcMap 8
- The first step: creating a database 9
- Copying schema from another geodatabase 15
- Tips on learning how to build and edit geodatabases 17

## **2 Creating new items in a geodatabase 19**

- Geodatabase items 20
- ArcGIS data types 25
- Setting an appropriate geodatabase spatial domain 28
- Upgrading a geodatabase 35
- Creating tables 36
- Creating feature datasets 39
- Creating feature classes 45
- Creating indexes 52
- Granting and revoking privileges 55

## **3 Importing data 57**

- Importing data into new feature classes and tables 59
- Importing feature classes 63
- Importing tables 67
- Registering ArcSDE data with the geodatabase 70
- Loading data into existing feature classes and tables 71
- Loading data in ArcCatalog 77
- Loading data in ArcMap 81
- Copying data between geodatabases 86
- Sending data to another user 95
- Updating DBMS statistics 98

## **4 Topology 99**

- What is topology? 101
- Creating a topology 102
- Topology basics 104
- Topology and feature geometry 108
- Topologies and ArcCatalog 110
- Migrating data into a geodatabase to create topologies 111
- Creating a new topology 114
- Adding new feature classes to your topology 119
- Validating a topology 123
- Topology: defining the rules 124
- Planning for exceptions 128
- Refining topologies with subtypes 129
- Managing a topology 130
- Modifying a topology 131
- Summarizing topology errors 140
- Creating new polygons from lines 141
- Topology and versioned databases 143
- Topology and versioning 145
- Topology and disconnected editing 155

## **5 Subtypes and attribute domains 157**

- What are subtypes and attribute domains? 158
- Working with attribute domain properties 162
- Browsing the attribute domains of a geodatabase 163
- Creating new attribute domains 165
- Modifying and deleting attribute domains 168
- Associating default values and domains with tables and feature classes 169
- Creating subtypes 170
- Modifying and deleting subtypes 173



<b>6</b>	<b>Defining relationship classes</b>	<b>175</b>
	What is a relationship class?	176
	Relationship classes in ArcCatalog and ArcMap	180
	Creating a simple relationship class	182
	Creating a composite relationship class	186
	Creating an attributed relationship class	189
	Creating relationship rules	191
	Managing relationship classes	193
	Exploring related objects in ArcMap	194
	Using related fields in ArcMap	197
<b>7</b>	<b>Geometric networks</b>	<b>199</b>
	What is a geometric network?	200
	Geometric networks and ArcCatalog	204
	Creating geometric networks	205
	Creating a new geometric network	211
	Building a geometric network from existing simple feature classes	215
	Adding new feature classes to your geometric network	221
	Network connectivity: defining the rules	224
	Establishing connectivity rules	225
	Managing a geometric network	227
<b>8</b>	<b>Managing annotation</b>	<b>229</b>
	Annotation in the geodatabase	230
	Annotation and ArcCatalog	235
	Creating annotation feature classes	236
	Converting labels to annotation	241
	Importing coverage annotation	244

<b>9</b>	<b>Dimensioning</b>	<b>247</b>
	Dimensions in the geodatabase	248
	Dimensions and ArcCatalog	251
	Creating dimension feature classes	252
	Creating and managing dimension styles	257
<b>10</b>	<b>Working with a versioned geodatabase</b>	<b>267</b>
	Integrating versioning with your organization's work flow	268
	Registering data as versioned	270
	Creating and administering versions in ArcCatalog	271
	Working with versions in ArcMap	278
	Editing and conflict resolution	281
	Editing a version	286
	Versioning scenarios	290
<b>11</b>	<b>Disconnected editing</b>	<b>293</b>
	Disconnected editing	294
	Checking out data from a geodatabase	313
	Customizing a check-out	315
	Checking in data to a geodatabase	318
	Managing check-outs	321
<b>12</b>	<b>Building a raster geodatabase</b>	<b>327</b>
	Rasters and the geodatabase	328
	Importing and loading raster data	332
	Attributes of type raster	340
	Converting raster formats	341
	Mosaicking raster datasets	342
	Raster data and disconnected editing	343
	More about rasters in ArcGIS	344

**Glossary 345**

**Index 369**



# Introduction

## IN THIS CHAPTER

- **Creating a geodatabase from an existing design**
- **Creating a geodatabase from scratch**
- **Geodatabases and ArcCatalog**
- **Geodatabases and ArcMap**
- **The first step: creating a database**
- **Copying schema from another geodatabase**
- **Tips on learning how to build and edit geodatabases**

The *geodatabase* supports a model of topologically integrated *feature classes*, similar to the *coverage* model. It also extends the coverage model with support for complex networks, topologies, *relationships* among feature classes, and other object-oriented *features*. The ESRI® ArcGIS® applications (ArcMap™, ArcCatalog™, and ArcToolbox™) work with geodatabases as well as with coverages and *shapefiles*.

The ArcGIS geodatabase model is implemented on standard relational databases with the ArcSDE® application *server*. *ArcSDE* defines an open interface to *database* systems. It allows ArcInfo® or ArcEditor™ seats to manage geographic information on a variety of different database platforms including Oracle®, Microsoft® SQL Server™, IBM® DB2®, and Informix®.

The geodatabase provides a generic framework for geographic information. This framework can be used to define and work with a wide variety of different user- or application-specific models.

The geodatabase supports object-oriented vector and raster data. In this model, entities are represented as *objects* with properties, behavior, and relationships. Support for a variety of different geographic object types is built into the system. These object types include simple objects, geographic features, network features, *annotation* features, and other more specialized feature types. The model allows you to define relationships between objects and rules for maintaining referential and topological integrity between objects.

How the data is stored in the database, the applications that access it, and the client and server hardware configurations are all key factors to a successful multiuser geographic information system (GIS). Successfully implementing a GIS with ArcInfo and ArcSDE starts with a good data model design. Designing a geodatabase is a critical process that requires planning and revision until you reach a design that meets your requirements and performs well. You can either start with an existing geodatabase design or design your own from scratch. Throughout this book, guidelines for good data modeling of each aspect of the geodatabase are discussed to help you implement a successful multiuser GIS system with ArcInfo, either with ArcSDE or with a *personal geodatabase*.

Once you have a design, you can create the geodatabase and its schema by creating new database *items* with ArcCatalog, loading existing shapefile and coverage data, using Unified Modeling Language (UML) and Computer-Aided Software Engineering (CASE) tools, or a combination of these.

A critical part of a well-performing geodatabase is the tuning of the database management system (DBMS) in which it is stored. This tuning is not required for personal geodatabases; however, it is critical for ArcSDE geodatabases. For more information on tuning your database for ArcSDE and the geodatabase, see the *Configuration and Tuning Guide for <DBMS>* PDF file.

The main tools you will use to create and edit geodatabases are found in ArcCatalog and ArcMap. ArcCatalog has various tools for creating and modifying your geodatabase

*schema*, while ArcMap has tools for analyzing and editing the contents of your geodatabase.

This book teaches you how to implement a geodatabase. If you're using ArcView®, it shows you how to create a personal geodatabase, import data, set up subtypes and domains, and create standard annotation feature classes and raster catalogs. If you're using ArcEditor or ArcInfo, it shows you how to create personal and ArcSDE geodatabases; import data; create geodatabase topology, subtypes, domains, relationship classes, geometric networks, standard and feature-linked annotation classes, raster catalogs, and mosaics; and manage editing with versions and disconnected editing. This book is one of three books designed to teach you how to make the most of geodatabases.

The second book, *Editing in ArcMap*, approaches the geodatabase from the editor and data manager's perspective. It describes how to create and edit *data* within an existing geodatabase.

The third book, *Geodatabase Workbook*, contains tutorial exercises that allow you to apply the concepts developed in the first two books.

# Creating a geodatabase from an existing design

There may be a data model that already exists that may partly or wholly suit your needs. Adopting an existing design can give you a quick head start in creating a geodatabase.

## Data models distributed by ESRI

ESRI and a number of leaders in their disciplines have been actively designing a series of GIS data models using topology and other capabilities in ArcGIS. The goal is to provide a common design framework for key layers of geographic information, and promote openness and interoperability of GIS data. These efforts have resulted in a series of comprehensive design specifications for a number of thematic *layers*:

- Census and Administrative Boundaries (applied to U.S. Census geography)
- Topographic basemaps for 1:24,000-scale maps
- Hydrography
- *Raster* imagery and elevation catalogs
- Streets and comprehensive address information
- Transportation (to support linear referencing, navigation, addressing, and cartography)
- Public Land Survey System (PLSS) (to support a national database of the legal survey fabric)
- Parcels (to support both U.S. and worldwide systems)
- Water facilities
- And numerous other efforts

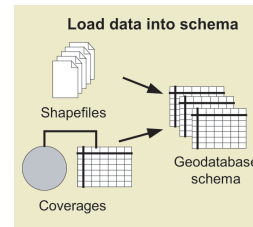
These data models provide a practical template for implementing a geodatabase. While you may find your industry-specific model to be a great starting point for your geodatabase, you may also find related models useful. For the latest information on the data models or to download one of them, see <http://support.esri.com>.

## Data models from other locations

You may know an ArcGIS user in a similar industry who has successfully implemented a geodatabase. ArcMap and ArcCatalog provide tools that allow you to copy the schema from an existing geodatabase. Once copied, the schema can be sent to someone else and adopted as a starting point for a geodatabase design. The tools that allow you to copy a schema are discussed later in this chapter.

## Loading data

Once you have obtained a model and customized the schema to suit your needs, the next step is to load data into it. You can do this by editing the database in ArcMap to create new objects or loading objects from existing shapefiles, coverages, raster datasets, *computer-aided design (CAD) feature classes*, raster catalogs, INFO™ tables, dBASE® tables, ArcStorm™, or Map LIBRARIAN.



Data creation and maintenance may involve managing *version* and topology information. ArcCatalog and ArcMap have wizards to help you with this—Simple Data Loader and Object Loader—that will be discussed in the chapter ‘Importing data’.



# Creating a geodatabase from scratch

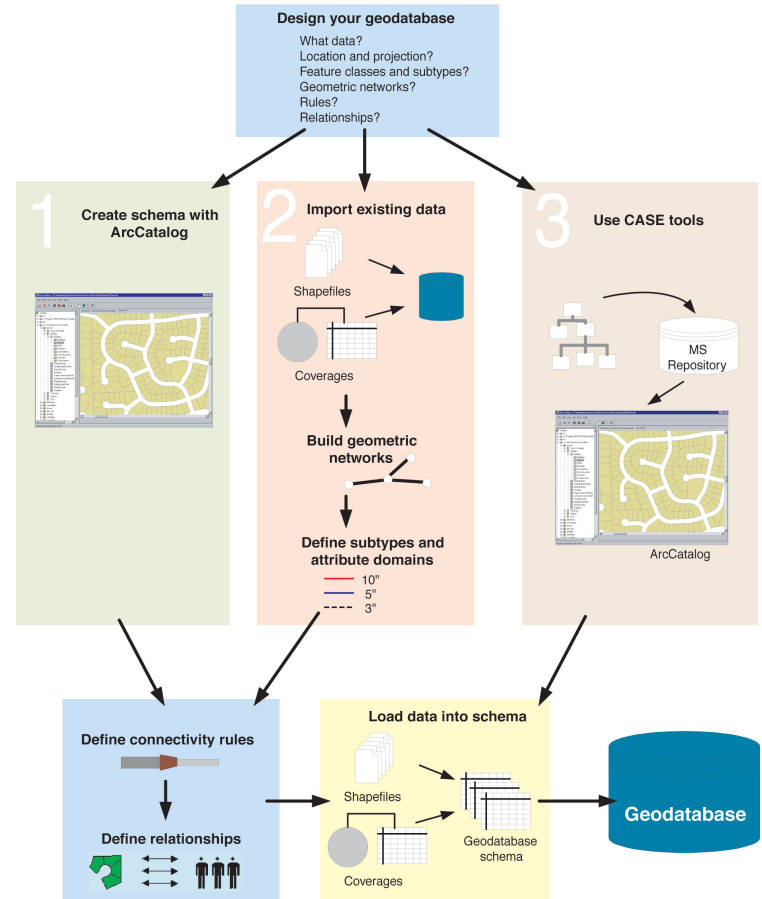
If you choose to create all or a part of your geodatabase from scratch, the first step is to design what you want to create. When designing a geodatabase you should consider such questions as:

- What kind of data will be stored in the database?
- In what *projection* do you want your data stored?
- Do you want to establish *rules* about how the data can be modified?
- How do you want to organize your *object classes* such as *tables*, feature classes, and subtypes of feature classes?
- Do you want to maintain relationships between objects of different types?
- Will your database contain geometric networks?
- Will your database contain topologically related features?
- Will your database store custom objects?

Use the data modeling guidelines in *Modeling Our World* and in this book to help you design a geodatabase that meets your requirements and performs well.

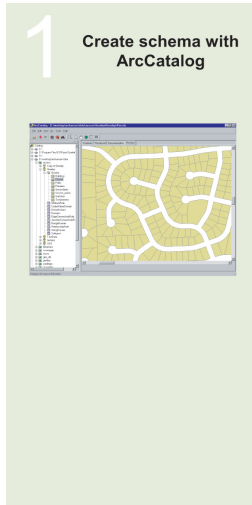
Once you have completed your design, the next step is to employ any of three methods to implement it. The method you choose depends on the source of your data and whether you will store custom objects. In practice, you will often use a combination of some or all of the methods outlined here. Subsequent chapters show you how to perform each task. If your geodatabase contains raster data, please read the chapter 'Building a raster geodatabase'.

## Three Methods to Create a Geodatabase



## Creating schema with ArcCatalog

In some cases, you may not yet have any data that you want to load into a geodatabase, or the data you have to load only accounts for part of your database design. In this case, you can use the tools provided in ArcCatalog to create the schema for *feature datasets*, feature classes, tables, *geometric networks*, topologies, and other items inside the database.

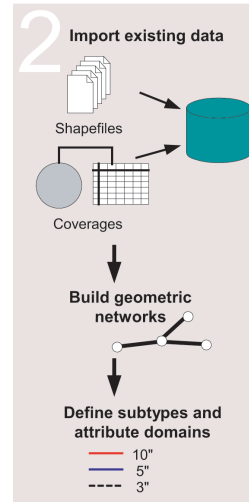


ArcCatalog provides a complete set of tools for designing and managing items you will store in the geodatabase. To learn how to create new items in the geodatabase, see the chapter ‘Creating new items in a geodatabase’.

## Importing data

It is likely that you already have data in various formats—*shapefiles*, coverages, INFO tables, raster datasets, raster catalogs, and dBASE tables—that you want to store in a geodatabase. You may also have your data stored in other multiuser geographic information system data formats such as ArcStorm, Map LIBRARIAN, and ArcSDE. Through ArcCatalog, you can convert data stored in one of these formats to a geodatabase by importing it.

When converting data from one of these formats into the geodatabase, both the spatial and nonspatial component of each



object is translated. For example, when converting a shapefile to a feature class, both the shapes (geometry) and attributes are stored in the geodatabase. Attributes can be left out or renamed. Shapefiles of the same spatial *extent* can be imported into the same feature dataset. All or some of the feature classes from a coverage can be imported into a feature dataset. *Topology rules* can be created to regulate the spatial relationships between the features and feature classes stored in geodatabase feature datasets.

Converting ArcStorm and Map LIBRARIAN data is done using tools that are similar to those used for importing coverages. However, you must use ArcSDE for Coverages before ArcCatalog or ArcToolbox can access and display ArcStorm and Map LIBRARIAN data.

If you already have your data in a Spatial Database Engine™ (SDE®) 3.x database, you do not need to reload your data. ArcCatalog contains tools that allow you to register the existing data with the geodatabase. Once registered, you can also use ArcCatalog to reorganize that data into feature datasets.

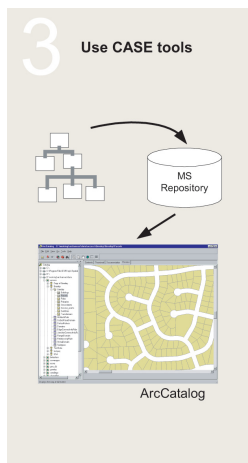
ArcGIS and geodatabases do not support multiple feature types in a single feature class (for example, *points* and lines in the same feature class). If any of your SDE 3.x layers contain multiple-entity types, those must be reorganized into single feature type layers before you can view them in ArcInfo or register them with the geodatabase.

Annotation stored with SDE 3.x is read-only in ArcGIS. If you want to use ArcMap to edit this annotation, you must convert it to geodatabase annotation. See the chapter ‘Managing annotation’ in this book for more information on converting SDE 3.x annotation to geodatabase annotation.

Once you have imported your data into the geodatabase, you can then use ArcCatalog to further define your geodatabase. ArcCatalog contains tools for building topologies and geometric networks and for establishing subtypes, *attribute domains*, and so on.

To learn how to move your existing data into the geodatabase, see the chapter ‘Importing data’.

## Building geodatabases with CASE tools



Unified Modeling Language is a graphical language used to develop software systems and database design. With UML class diagrams you can design items in a geodatabase schema, such as feature datasets, feature classes, tables, geometric networks, and relationships. Once the UML diagram is complete, you can use the Computer-Aided Software Engineering tools subsystem in ArcCatalog to generate a new geodatabase schema from the diagram.

The steps are:

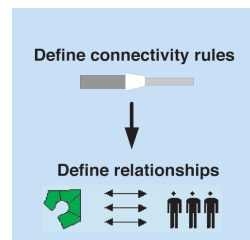
1. Design the schema in UML with Visio® or Rational Rose® and export it to XML Metadata Interchange (XMI). To learn how, see <http://support.esri.com/geodatabase/UML>.

2. Generate geodatabase schema from XMI with CASE tools. To learn how, see the online help.

## Further refining the geodatabase

Whether you load data manually or use ArcCatalog to create the geodatabase schema, you can continue to define your geodatabase by establishing how objects in the database relate to one another.

Using ArcCatalog, you can establish relationships between objects in different object classes, *connectivity rules* for objects participating in geometric networks, and topological rules for features in topologies. Some of these relationships and rules may be part of the schema that CASE tools generate, but often you will want to further refine what is generated by CASE to meet your geodatabase design. Note that topologies cannot be designed with the existing CASE tools. You can continue to use the geodatabase management tools in ArcCatalog to refine or extend a geodatabase once it has been designed and implemented.



# Geodatabases and ArcCatalog

ArcCatalog allows you to easily view and modify the contents of your geodatabase. ArcCatalog contains a full suite of tools to create and manage your geodatabase.

## Accessing geodatabases in ArcCatalog

You can manage geographic data in a variety of formats in ArcCatalog. Some of the formats that you can manage directly include personal geodatabases, shapefiles, ArcInfo coverages, rasters, TINs, and tables.

In addition to managing data on your desktop or local network, you can manage remote ArcSDE geodatabases by creating a connection to the database. Database connections to remote geodatabases behave similarly to personal geodatabases, with one important difference: when you delete a personal geodatabase, the database itself is deleted from the disk. When you delete a remote geodatabase connection, however, only the connection is deleted—the geodatabase and its data are unaffected.

## Spatial database connections

Using data stored in a DBMS, such as Oracle, requires a database connection. There are two methods for connecting to a *spatial database* from ArcInfo. One method is to connect to an ArcSDE service that spawns a process on the server to broker the connection between ArcInfo and the database *instance*.

The second method is to use a *direct connection* to the database. In this case, ArcInfo connects directly to the database server. The functionality that is managed by the server process in the first connection method is transferred to the client, thus eliminating the middle tier. The direct connect method is a two-tiered rather than three-tiered architecture.

You can use the direct connect method to connect to your geodatabase if it is stored in Oracle8i™, Oracle9i™, SQL Server, DB2, or Informix. If connecting to SQL Server, you do not require any additional software to connect to the database. If direct connecting to any of the others, you'll need to install client software on your machine. For more information about direct connect, see *Making a Direct Connection*, located on the ArcSDE CD-ROM.

When you add a new connection to an ArcSDE geodatabase service or a direct database connection in ArcCatalog, it creates a connection file on disk. This file contains the information needed to establish a connection. The *username* and *password* can be included in the connection file and are encrypted for security.

You can set up connection files for your organization and distribute these such that end users will not require any information about the geodatabase server to which they are connecting.

# Geodatabases and ArcMap

ArcMap allows you to edit the contents of your geodatabase. ArcMap contains a full suite of tools to edit *simple features*, geometric networks, and topologies in your geodatabase.

## Editing geodatabases in ArcMap

You can edit geographic data in a variety of formats in ArcMap. ArcView seats of ArcMap can be used to edit simple features in shapefiles and personal geodatabases. ArcInfo and ArcEditor seats have more extensive toolsets with special tools for creating and editing geographic data in topologies or geometric networks and performing spatial adjustment and *disconnected editing*.

## Editing topologies

Editing a feature in a *map topology* or geodatabase topology with the topology edit tools automatically updates the geometry of the shared parts of all the features. After you edit a topology, the geodatabase can discover if any of the edits violate the rules of the topology. If there are *errors*, ArcMap enables you to quickly see which features are involved and which rules have been violated and allows you to fix the error, mark it as an exception, or leave it as an error. Error reporting in ArcMap gives you a measure of how good your data is.

## Editing geometric networks

Editing features in a geometric network allows you to maintain network *connectivity* of specified types of features and lets you move connected features without breaking the connections as well as add junctions to certain types of edges without splitting them. Geometric networks are useful for modeling connected linear features, such as pipes, cables, and streams, and for modeling points that represent *nodes* in the network such as valves, switches, and stream confluences or gauging stations.

The features that participate in a geometric network become network feature classes, not simple feature classes.

## Versions

ArcEditor and ArcInfo seats of ArcMap allow you to edit versioned geodatabases. Different versions of a geodatabase can be used to model complex “what if” scenarios; create different design alternatives; or manage the stages in a multistep planning, design, and construction *work flow* cycle without modifying or making multiple copies of your base data. Versioning also allows multiple editors to work on the same large continuous GIS *dataset* without the need to lock the data or divide it into tiles.

ArcMap lets you easily switch from one version to another to see how they differ.

When you are ready to merge changes from a version into another version, ArcMap can *reconcile* the versions. When multiple editors are working on an area, there is a chance for the same feature to be modified by more than one editor—in these cases, ArcMap helps you identify and resolve the conflicts.

There are currently no versioning capabilities for raster data.

## Disconnected editing

ArcInfo and ArcEditor seats of ArcMap can be used to check out data from your geodatabase to use while disconnected from the network. This is especially useful for remote offices that need to work with a portion of the database without incurring network-related performance problems and for people who need to take a part of the geodatabase out into the field for editing or analysis on a field computer. Changes to the data made in the field can be checked back in to the main geodatabase to allow them to be used by others.

## The first step: creating a database

The first step in creating a geodatabase is to create the database itself using ArcCatalog.

There are two kinds of geodatabases: personal geodatabases and ArcSDE geodatabases. Creating a new personal geodatabase involves creating a new .mdb file on disk.

Before you can create data in an ArcSDE geodatabase, you must do some setup. Setting up the database for use as an ArcSDE geodatabase is described in *Managing ArcSDE Services* and in the *ArcSDE Installation Guide*, located on the ArcSDE CD-ROM. For information about direct connections, please see *Making a Direct Connection*, also located on the ArcSDE CD-ROM.

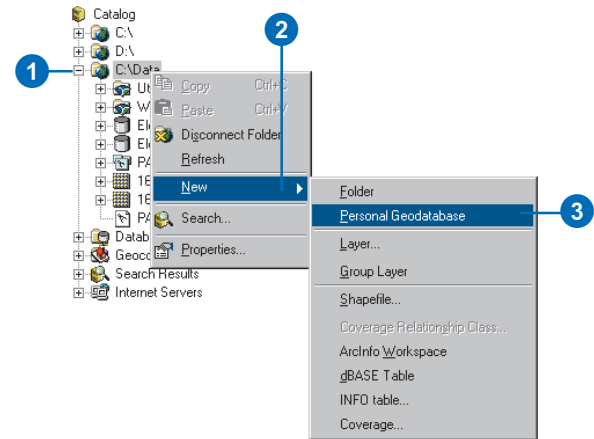
Several *versions* of an ArcSDE geodatabase can exist, although not every table or feature class in the geodatabase must be versioned. Feature editing in ArcMap requires a versioned feature class in a geodatabase.

New connections will automatically access the DEFAULT version of the database. ►

## Creating a new personal geodatabase

1. Right-click the location in the ArcCatalog tree where you want to create the new personal geodatabase.
2. Point to New.
3. Click Personal Geodatabase.
4. Type a new name for this personal geodatabase.
5. Press Enter.

ArcCatalog creates a new personal geodatabase in the location you selected and sets its name to edit mode.



To connect to an alternate version, you must provide your username and password along with the version name. If you do not specify the version, ArcCatalog connects to the DEFAULT version.

ArcSDE functionality is read-only for ArcView.

### Tip

#### Testing the connection

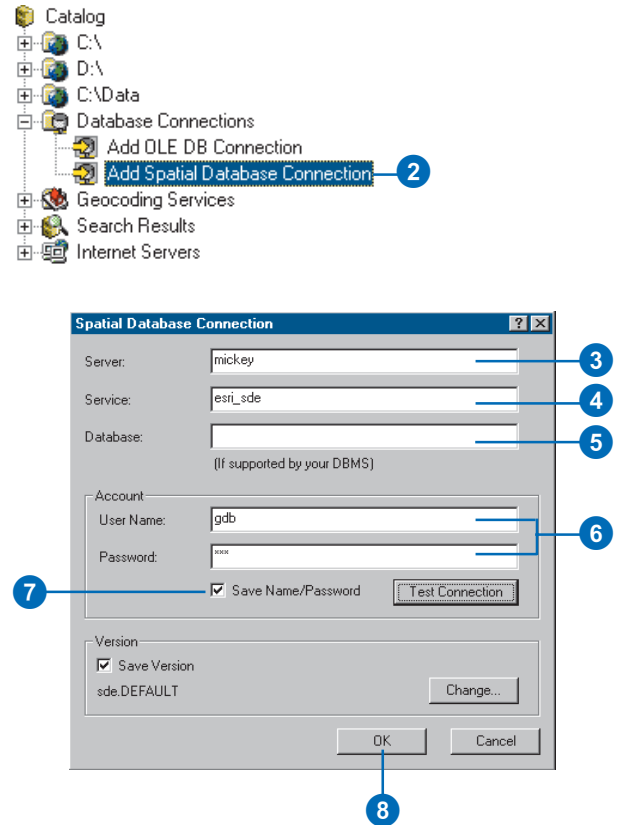
Clicking OK in the Spatial Database Connection dialog box does not actually connect to the database but creates the connection file on disk. To make sure that the connection parameters you entered are correct, you can click Test Connection.

### See Also

For more information on how to use ArcCatalog to browse your file system, see Using ArcCatalog.

## Adding a connection to an ArcSDE geodatabase service in ArcCatalog

1. Double-click Database Connections.
2. Double-click Add Spatial Database Connection.
3. Type either the name or the Internet Protocol (IP) Address of the server to which you want to connect.
4. Type either the name or the TCP/IP port number of the ArcSDE service to which you want to connect.
5. Type the name of the database to which you want to connect if your DBMS supports it; otherwise, skip to step 6.
6. Type the username and password with which you will connect to the ArcSDE geodatabase.
7. Check the check box to save the username and password in the connection file so that you can connect to the database in the future without being prompted to log in.
8. Click OK.
9. Type a new name for the spatial database connection.
10. Press Enter.





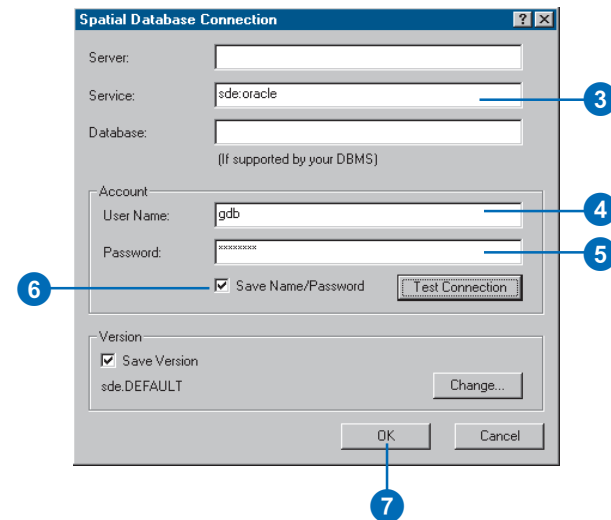
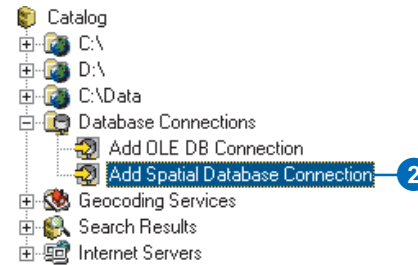
## Tip

### Oracle network service name

*You must create an Oracle network service name on your client machine before you can create a direct connection to an Oracle database.*

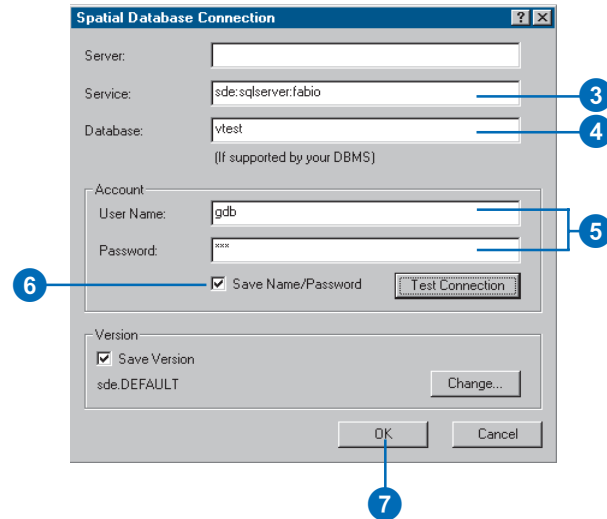
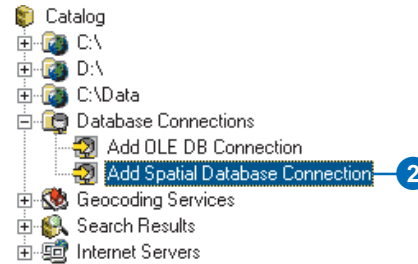
## Adding a direct connection to an Oracle geodatabase in ArcCatalog

1. Double-click Database Connections.
2. Double-click Add Spatial Database Connection.
3. If you're connecting to Oracle8i, type "sde:oracle" in the Service box. If you're connecting to Oracle9i, type "sde:oracle9i".
4. Type the username.
5. Type the password followed by "@<oracle network service name>".
6. Check the check box to save the username and password in the connection file so that you can connect to the database in the future without being prompted to log in.
7. Click OK.
8. Type a new name for the spatial database connection.
9. Press Enter.



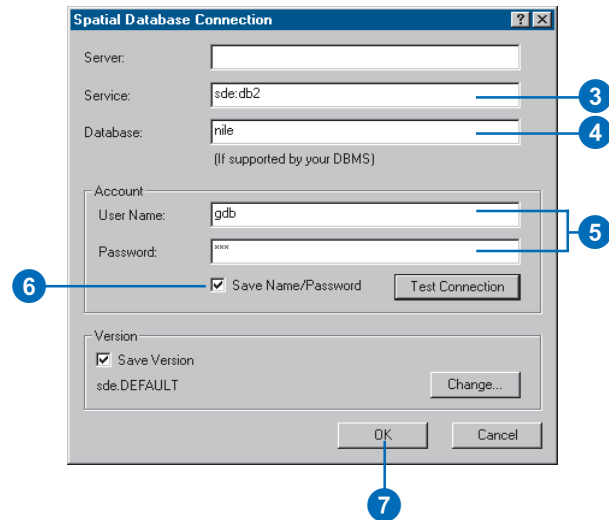
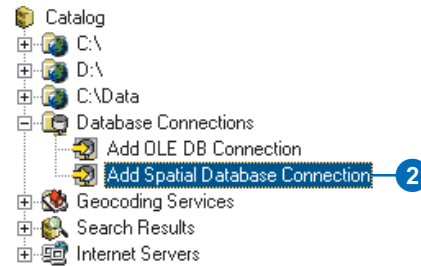
## Adding a direct connection to a SQL Server geodatabase in ArcCatalog

1. Double-click Database Connections.
2. Double-click Add Spatial Database Connection.
3. Type “sde:sqlserver:<name or the IP Address of the server>” in the Service box. In this example, the server name is fabio.
4. Type the name of the database you want to connect to.
5. Type the username and password.
6. Check the check box to save the username and password in the connection file so that you can connect to the database in the future without being prompted to log in.
7. Click OK.
8. Type a new name for the spatial database connection.
9. Press Enter.



## Adding a direct connection to a DB2 or Informix geodatabase in ArcCatalog

1. Double-click Database Connections.
2. Double-click Add Spatial Database Connection.
3. If you're connecting to DB2, type "sde:db2" in the Service box. If you're connecting to Informix, type "sde:informix".
4. Type the name of the database you want to connect to.
5. Type the username and password.
6. Check the check box to save the username and password in the connection file so that you can connect to the database in the future without being prompted to log in.
7. Click OK.
8. Type a new name for the spatial database connection.
9. Press Enter.

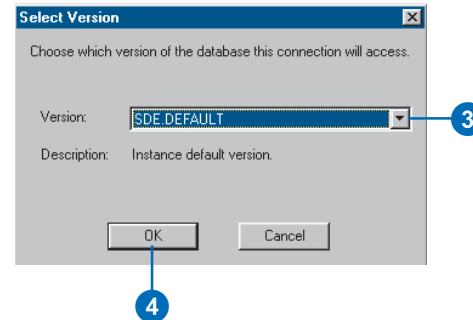
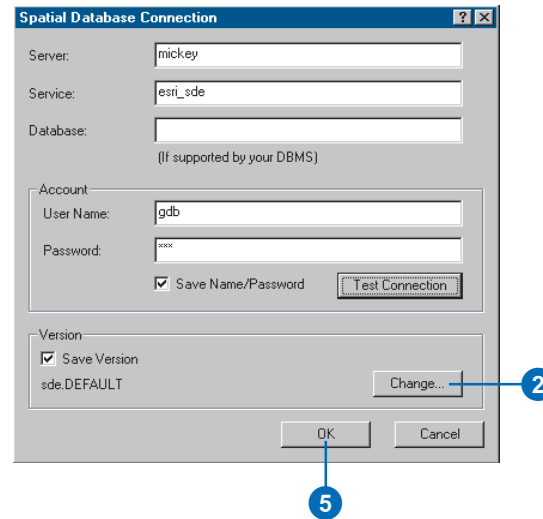


## See Also

For more information on geodatabase versions, see the chapter 'Working with a versioned geodatabase' in this book.

## Connecting to an alternative version of the database

1. Follow steps 1 through 7 for adding a connection to a spatial database geodatabase service or direct connect in ArcCatalog.
2. Click Change.
3. Click the Version dropdown arrow and click the version you want to access.
4. Click OK.
5. Click OK in the Spatial Database Connection dialog box.
6. Type a new name for the spatial database connection.
7. Press Enter.



# Copying schema from another geodatabase

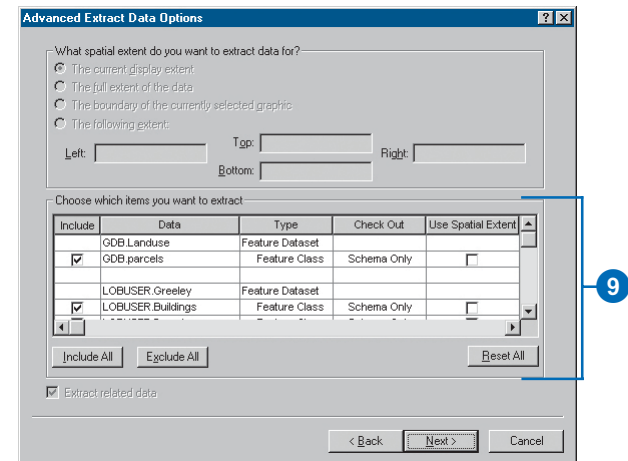
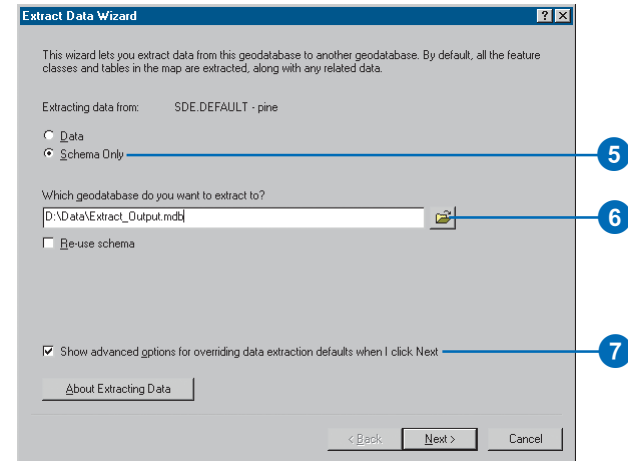
A great way to get design ideas for your new geodatabase is to look at the design of an existing geodatabase.

If you want to see someone else's design, ask them to export their schema to a ZIP file and send it to you. You can then import the schema into a geodatabase. Please see the ArcGIS online Help system for more information.

Another tool that allows you to copy schema from another geodatabase is the Extract Data wizard in ArcMap. It allows you to modify the spatial reference of the new schema you create. This is useful because the spatial reference requirements of your new geodatabase will probably be different from those of the source geodatabase.

Regardless of the method you choose, the result is a new schema with no data but with all the feature datasets, feature classes, tables, topologies, relationships, geometric networks, domains, subtypes, indexes, and field properties from the source geodatabase.

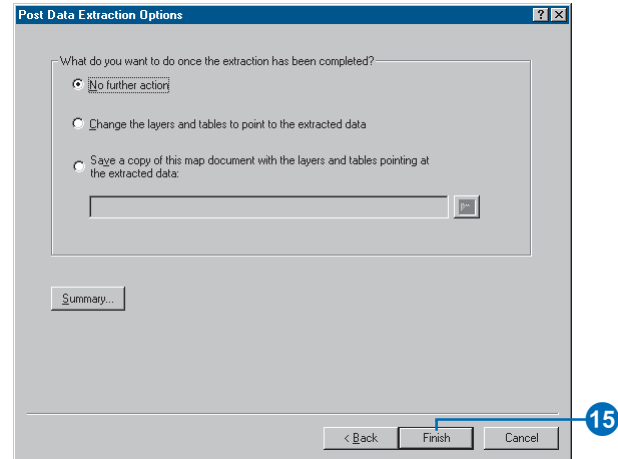
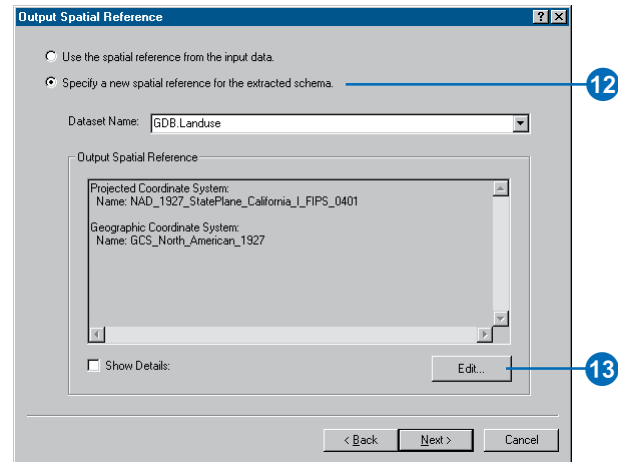
1. Start ArcMap.
2. Use the Add Data button to add the data that you want to export schema from to the data frame.
3. Click View, click Toolbars, then choose Disconnected Editing to display the Disconnected Editing toolbar.
4. Click the Extract Data command on the Disconnected Editing toolbar to start the Extract Data wizard.
5. Click Schema Only.
6. Navigate to the geodatabase you want to export the schema into or type its path. If the geodatabase doesn't already exist, it will be created.
7. Check Show advanced options for overriding data extraction defaults when I click Next.
8. Click Next.
9. The list of data to export schema from expands to include all the data in any feature dataset and any related data. For example, if you have just one feature class from a feature dataset in the data frame, all feature classes in the feature dataset are shown. ►



You can then review the schema and modify it to suit your needs with the tools discussed in the chapter ‘Creating new items in a geodatabase’.

Uncheck a check box if there is a feature class, table, or relationship class you don’t want to export schema from. If you leave a box checked for a feature class in a network or topology, the schema from all the feature classes in the network and topology will export.

10. Click Next.
11. If you want the output schema to have the same spatial reference as the source data, skip to step 14.
12. Click Specify a new spatial reference.
13. Click Edit to display the Spatial Reference Properties dialog box and set a spatial reference for the output schema. The one spatial reference you choose will become the spatial reference for the entire schema you extract.
14. Click Next.
15. Click Finish to export the schema.



# Tips on learning how to build and edit geodatabases

If you're new to GIS, you don't have to know everything about ArcCatalog, ArcMap, and geodatabases or how to extend the generic geodatabase data model to get immediate results. To learn how to create and edit geodatabases, see the *Geodatabase Workbook*. ArcGIS comes with the data used in the tutorials, so you can follow along step by step at your computer. You can also read the tutorial without using your computer.

## Finding answers to questions

If you are like most people, your goal is to complete your tasks while investing a minimum amount of time and effort on learning how to use the software. You want intuitive, easy-to-use software that gives you immediate results without having to read pages of documentation. However, when you do have a question, you want to be able to find the answer quickly so that you can complete your task. That's what this book is all about—getting you the answers you need when you need them.

This book describes how to get your existing data into a geodatabase; how to create new items in your geodatabase; and once created, how to add a variety of behavior to that data and edit it. Although you can read this book from start to finish, you will likely use it more as a reference. When you want to know how to do a particular task, such as creating a geometric network or topology, just look it up in the table of contents or index.

What you will find is a concise, step-by-step description of how to complete tasks. Some chapters also include detailed information if you want to learn more about the concepts behind the tasks. Refer to the glossary if you come across any unfamiliar GIS terms or need to refresh your memory.

## About this book

This book is designed to introduce how to build and edit a geodatabase. While this book does have some conceptual content about the different aspects of the geodatabase, it assumes that you already have a schema design that you are trying to implement. If you have not yet designed your schema or need more information on how to make the best schema design decisions, read *Modeling Our World*.

## Getting help on your computer

In addition to this book, the ArcGIS online Help system is a valuable resource for learning how to use the software.

## Contacting ESRI

If you need to contact ESRI for technical support, refer to 'Contacting Technical Support' in the 'Getting more help' section of the ArcGIS Desktop Help system. You can also visit ESRI on the Web at [www.esri.com](http://www.esri.com) and [support.esri.com](http://support.esri.com) for more information on the geodatabase and ArcGIS.

## ESRI education solutions

ESRI provides educational opportunities related to geographic information science, GIS applications, and technology. You can choose among instructor-led courses, Web-based courses, and self-study workbooks to find educational solutions that fit your learning style. For more information, go to [www.esri.com/education](http://www.esri.com/education).





# Creating new items in a geodatabase

# 2

## IN THIS CHAPTER

- **Geodatabase items**
- **ArcGIS data types**
- **Setting an appropriate geodatabase spatial domain**
- **Upgrading a geodatabase**
- **Creating tables**
- **Creating feature datasets**
- **Creating feature classes**
- **Creating indexes**
- **Granting and revoking privileges**

The first step in creating any database is to design the tables it will contain. A good design will ensure that data retrieval is fast and efficient. *Modeling Our World* discusses the considerations to take into account when you build a geodatabase.

Once your design is complete, you can start using tools in ArcCatalog to create tables, feature datasets, and feature classes. After adding data to tables and feature classes, you can build *indexes* for the appropriate fields to improve *query* performance. You can also grant and revoke privileges on your table, feature class, or feature datasets for another database user.

After creating feature classes, tables, and feature datasets, you can refer to further chapters in this book to create more advanced items such as relationship classes, topologies, and geometric networks.

To create raster-based items, such as raster datasets, raster catalogs, or raster attributes, refer to the chapter ‘Building a raster geodatabase’.

ArcView can create simple feature classes and tables in a personal geodatabase. More advanced geodatabase functionality requires an ArcEditor or ArcInfo license.

# Geodatabase items

Geodatabases organize geographic data into a hierarchy of data objects. These data objects are stored in feature classes, object classes, and feature datasets. An object class is a table in the geodatabase that stores nonspatial data. A feature class is a collection of features with the same type of geometry and the same attributes.

A feature dataset is a collection of feature classes that share the same *spatial reference*. Feature classes that store simple features can be organized either inside or outside a feature dataset. Simple feature classes that are outside a feature dataset are called standalone feature classes. Feature classes that store topological features must be contained within a feature dataset to ensure a common spatial reference.

ArcCatalog contains tools for creating object classes (tables), feature classes, and feature datasets. Once these items are created in the geodatabase, further items such as subtypes, relationship classes, geometric networks, and topologies can also be created. These geodatabase items are covered in subsequent chapters.

## Spatial reference

When creating a new feature dataset or standalone feature class, you must specify its spatial reference. The spatial reference for a feature class describes its *coordinate system* (for example, geographic, Universal Transverse Mercator [UTM], and State Plane), *spatial domain*, and *precision*. The spatial domain is best described as the allowable *coordinate* range for x,y coordinates, m (measure) values, and z-values. The precision describes the number of system units per one unit of measure. A spatial reference with a precision of 1 will store integer values, while a precision of 1,000 will store three decimal places. Once the spatial reference for a feature dataset or standalone feature class has been set, only the coordinate system can be modified—the spatial domain is fixed.

All feature classes in a feature dataset share the same spatial reference. The spatial reference is an important part of geodatabase design because its spatial domain describes the maximum spatial extent to which the data can grow. You must be careful to choose an appropriate x, y, m, and z domain. For example, if you create a feature dataset with a minimum z-value of 0 and a precision of 1,000, none of the features in the feature dataset can have z-values that are less than 0, and all z-values will be stored to three decimal places. The same rule applies to x- and y-values. The exception to the rule is m domains; feature classes within the same feature dataset can have different m domains.

The spatial domain for a feature class or feature dataset cannot be changed. If the required x-, y-, m-, or z-value ranges for your database change, the data has to be reloaded into feature classes with a spatial reference that accommodates the new value range.

A collection of predefined geographic and projected coordinate systems is installed with ArcInfo. You can create custom coordinate systems, or you can import a coordinate system from an existing feature class, feature dataset, coverage, or shapefile. You can read more about spatial references and spatial domains in *Managing ArcSDE Services* and *Understanding Map Projections*.

## Spatial index grid

Much like you use a locator grid on a city map to find a street, ArcMap uses grids to quickly locate features in feature classes. Identifying a feature, selecting features by pointing or dragging a box, and panning and zooming all require ArcMap to use the spatial index grid to locate features.

A feature class can have up to three grids. The size of each must be at least three times the previous grid size. For most feature classes, only a single grid size is required. Feature classes with features of very different sizes may require additional grids so that larger features can be queried faster.

When you create an empty feature class in ArcCatalog or import data to create a new feature class, you can choose a default grid size or specify your own. Once the feature class is created, you can change the grid size any time.

ArcMap doesn't use the spatial index grid for feature classes in Informix ArcSDE geodatabases. As other strategies are used to locate features in Informix, you can ignore the grid size. For a more detailed discussion of spatial indexes and grid sizes, see the *ArcSDE Configuration and Tuning Guide for <DBMS>* PDF file.

## Field properties

When you use ArcCatalog to create a new table or feature class, you can specify any number of *fields* to be included. You can also specify settings for fields such as the field type and the maximum size of the data that can be stored in it.

Each field type has special properties. All fields have *property* default values, domains, *aliases*, and allow nulls. The field alias property will be discussed in the next section. You can set the allow nulls property to No if you do not want the field to store *null values*. If you set the allow nulls property to Yes, then the field will allow null values.

Use the default value property if you want the field to be automatically populated with a default value when a new feature or object is created with the ArcMap editing tools. You can set a domain, which is a valid set or range of values that can be stored in the field by using the domain property. Default values and domains are discussed in detail in the 'Subtypes and attribute domains' chapter.

The exceptions are fields of type ObjectID, binary large object (BLOB), GlobalID, and Geometry which do not have a default value or domain property. Alias is the only property of the ObjectID and GlobalID fields you can modify, while BLOB and Geometry fields have special properties you can modify.

The properties of a Geometry field describe the kind of features that can be stored in a feature class, the size of the spatial index, and the spatial reference for the features.

## Field precision and scale

The precision and scale of a field describe the maximum size and precision of data that can be stored in it. The precision describes the number of digits that can be stored in the field, while the scale describes the number of decimal places for float and *double* fields. When creating a new field in a geodatabase feature class or table, you can specify the field's type, precision, and scale. When the field is actually created in the database, the field type may be changed based on the precision and scale values you specify.

Use the following guidelines for choosing the correct field type for a given precision and scale:

- When you create a float, double, or integer field and specify 0 for precision and scale, the geodatabase will attempt to create a binary type field if the underlying database supports it. Personal geodatabases support only binary type fields, and precision and scale are ignored.
- When you create float and double fields and specify a precision and scale, if your precision is greater than 6, use a double; otherwise, use a float. If you create a double field and specify a precision of 6 or less, a float field is created in the database. If you create a float field and specify a precision greater than 6, a double field is created.
- If you specify a scale of 0 and a precision of 10 or less, you should be creating integer fields. When creating integer fields, your precision should be 10 or less or your field may be created as double.

## Required fields

All tables and feature classes have a set of required fields that are necessary to record the state of any particular object in the table or feature class. These required fields are automatically created when you create a new feature class or table and they cannot be deleted. Required fields may also have required properties such as their domain property. You cannot modify the required property of a required field.

For example, in a simple feature class, OBJECTID and Shape are required fields. They do have properties you can modify, such as their aliases and geometry type, but these fields cannot be deleted.

Some types of feature classes have a number of required fields; for more information, see the chapters ‘Managing annotation’, ‘Geometric networks’, and ‘Topology’ in this book.

## Field, table, and feature class aliases

The names of feature classes and tables in a geodatabase are the same as the names of the physical tables in the DBMS in which they are stored. When storing data in a DBMS, often the names for tables and fields are cryptic, and you require a detailed data dictionary to keep track of what data each table stores and what each field in those tables represents. For example, your database may have a feature class called “Pole” that has a field called “HGT”. Without consulting your data dictionary, you may have a difficult time determining that Pole stores utility poles and HGT has values for pole heights.

The geodatabase provides the ability to create aliases for fields, tables, and feature classes. An alias is an alternative name to refer to those objects. Unlike true names, aliases do not have to adhere

to the limitations of the database, so they can contain special characters such as spaces. In the above example, you may set the alias for the Pole table to “Utility Poles” and the alias for the HGT field to “Height”. In ArcMap, when using data with aliases, the alias name is automatically used for feature classes, tables, and fields. However, in ArcCatalog these objects are always represented by their true names. You can view the alias for feature classes, tables, and fields by examining their properties.

Aliases can be specified when creating a feature class or table and can be modified at any time. Similarly, when creating new fields, the alias is set as a property of that field and can be modified at any time.

## Tracking properties of geometry

Often when working with spatial data, you may want to query your data based on properties of the geometry. For example, you may want to query the water mains feature class for all mains that are more than 50 feet in length. Doing this by examining each geometry and calculating its length can be a slow process, especially if there is a large number of water mains in the feature class. To make this more efficient, the feature class has special fields that track this kind of information about the geometry of your features.

Line feature classes have a field that tracks the features’ length, while *polygon* feature classes have fields that track both the features’ area and perimeter. When changes are made to the geometry, the values in these fields are automatically updated. These fields behave similar to other fields, except that you cannot delete them, assign default values and attribute domains to them, or assign values to them while editing in ArcMap.

When you create a new feature class in a personal geodatabase, these fields will not be displayed in the fields panel of the wizard. However, when you open the properties dialog box for a polygon



feature class, you will see fields named Shape\_Area and Shape\_Length that store the area and perimeter of the geometry. When you open the properties dialog box for a line feature class, you will see a field named Shape\_Length that stores the length of the geometry. Point and *multipoint feature* classes do not have either field.

When you create a new feature class in an ArcSDE geodatabase, these fields will be called SHAPE.AREA and SHAPE.LEN, respectively.

In ArcMap, these fields behave as any other field in the identify window, the layer properties dialog box, the attribute table dialog box, and the query builder. For more information on these aspects of ArcMap, see *Using ArcMap*.

## Feature datasets

Feature datasets exist in the geodatabase to define a scope for a particular spatial reference. All feature classes that participate in topological relationships with one another, for example, a geometric network or a topology, must have the same spatial reference. Feature datasets are a way to group feature classes with the same spatial reference so they can participate in topological relationships with each other.

Feature datasets also have a natural organizational quality, much like a folder on a file system. Since for many GIS applications the majority of the data for a particular database has the same spatial reference, it is possible to use feature datasets as organizational containers.

Bear in mind that topologically related feature classes must reside in the same feature dataset. If you are interested in organizing feature classes purely by category, you can arrange layer files into logical groups within folders at a shared location, without regard to the organization of the feature classes in the geodatabase.

## Topologies

Many vector datasets have features that could share boundaries or corners. If you create a topology in the dataset, you can set up rules defining how features share their geometry. Editing a boundary or *vertex* shared by two or more features updates the shape of each of those features. Topology rules can govern the relationships between features within a single feature class or between features in two different feature classes. For example, moving a slope boundary in one feature class could update two slope-class polygons and also update the boundary of a forest stand in another feature class.

Topologically associated feature classes in a geodatabase are stored in a feature's dataset with a *topology*. The feature classes remain simple feature classes, and the topology manages the rules that control how features can be spatially related. You can use the topological editing tools in ArcMap to maintain the topological associations of features.

To learn more about topologies, see the chapter 'Topology' in this book.

For more information about topology in general, see *Modeling Our World*.

## Geometric networks

Some vector datasets, particularly those used to model communications, material or energy flow, or transportation networks, need to support connectivity *tracing* and network connectivity rules. Geometric networks allow you to turn simple point and line features into network edge and junction features that can be used for network analysis. Connectivity rules of geometric networks let you control what types of network features may be connected together when editing the network. Geometric networks, like topologies, must be created from a set of feature classes in the same feature dataset.

To learn more about geometric networks, see the ‘Geometric networks’ chapter in this book.

For more information about geometric networks in general, see *Modeling Our World*.

## Relationship classes

Relationship classes define relationships between objects in the geodatabase. These relationships can be simple one-to-one relationships, such as you might create between a feature and a *row* in a table, or more complex one-to-many (or many-to-many) relationships between features and table rows. Some relationships specify that a given feature, row, or table is not only related to another, but that creating, editing, or deleting one will have a specified effect on the other. These are composite relationships and they can be used to ensure the links between objects in the database are maintained and up to date. Deleting a feature, such as a power pole, can trigger the deletion of other features, such as a transformer mounted on the pole or the maintenance *records* in a related table.

To learn more about relationship classes, see the ‘Defining relationship classes’ chapter in this book.

## Schema locking

In multiuser databases, more than one user may be reading and editing the same data at the same time. To be able to work with data in a geodatabase for applications such as ArcMap, the application must assume that the schema for that data is fixed while it is working with it.

For example, when you add a feature class from a geodatabase to your map, its schema cannot be altered by you or another user. Once you have removed the feature class from your map and no other users are querying or editing that feature class, its schema can be altered.

ArcMap, ArcCatalog, and other applications written with the ArcGIS Component Object Model (COM) components will automatically acquire a shared lock when editing or querying the contents of a geodatabase feature class or table. Any number of shared locks can be acquired on a single feature class or table at one time. When using ArcCatalog to modify schema—add a field, alter rules, and so on—the application will attempt to acquire an exclusive lock on the data being altered.

An exclusive lock can only be acquired if no other locks—shared or exclusive—are already on the data. If there are already other locks on the feature class or table, ArcCatalog will not be able to establish its exclusive lock, and the schema will not be editable. Once an exclusive lock has been acquired, no shared locks can be applied, so the data will not be accessible in ArcMap or ArcCatalog by another user.

Exclusive locks can only be acquired by the owner of the feature class or table being modified and, therefore, only the owner can ever modify the schema of an item in a geodatabase. Some of the items that exist in a geodatabase—which are discussed in further chapters, such as geometric networks, relationship classes, topologies, and so on—have special schema-locking behaviors. Each of these is discussed in its own respective chapter.

Schema locking in personal geodatabases works in much the same way except the locks are databasewide. Once an exclusive or shared lock is acquired on an item in a personal geodatabase, that lock applies to all items in the geodatabase.

# ArcGIS data types

When creating tables, you will need to *select* a data type for each field in your table. The available types include a variety of number types, text, date, BLOB, or globally unique identifier (GUID). Choosing the correct data type allows you to correctly store the data and will facilitate your analysis, data management, and business needs.

## Numeric data types

Numeric fields can be stored as one of four numeric *data types*. These include short integers; long integers; *single-precision* floating point numbers, often referred to as floats; and double-precision floating point numbers, commonly called doubles. Each of these numeric data types varies in the size and method of storing a numeric value.

In numeric data storage, it is important to understand the difference between decimal and binary numbers. The majority of people are accustomed to decimal numbers, a series of digits between 0 and 9 with negative or positive values and the possible placement of a decimal point. On the other hand, computers store numbers as binary numbers. A binary number is simply a series of 0s and 1s. In the different numeric data types, these 0s and 1s represent different coded values, including the positive or negative nature of the number, the actual digits involved, and the placement of a decimal point. Understanding this type of number storage will help you make the correct decision in choosing numeric data types.

The most basic numeric data type is the short integer. This type of numeric value is stored as a series of 16 0s or 1s, commonly referred to as 16 bits. Eight bits are referred to as a byte, thus a short integer takes up two bytes of data. One bit states whether the number is positive or negative, and the remaining 15 translate to a numeric value with five significant digits. The actual numeric value for a short integer is approximately between -32,000 and

+32,000. A long integer is a four-byte number. Again, one bit stores the positive or negative nature of the number, while the remaining bits translate to a numeric value with 10 significant digits. The actual range for a long integer is approximately between -2 billion and +2 billion. Both short and long integers can store only real numbers. That is to say that you cannot have fractions, or numbers, to the right of the decimal place. To store data with decimal values, you will need to use either a float or a double.

A float and double are both binary number types that store the positive or negative nature of the number, a series of significant digits, and a coded value to define the placement of a decimal point. This is referred to as the exponent value. Floats and doubles are coded in a format similar to scientific notation. For example, if you wanted to represent the number -3,125 in scientific notation, you would say  $-3.125 \times 10^3$  or  $-3.125E^3$ . The binary code would break this number apart and assign one bit to state that it is a negative number; another series of bits would define the significant digits 3125; another bit would indicate whether the exponent value is positive or negative; and the final series of bits would define the exponent value of 3. A float is a four-byte number and can store up to seven significant digits, producing an approximate range of values between  $-3.4E^{-38}$  to  $-1.2E^{38}$  for negative numbers and from  $3.4E^{-38}$  to  $1.2E^{38}$  for positive numbers. A double is an eight-byte number and can store up to 15 significant digits, producing an approximate range of values between  $-2.2E^{-308}$  to  $-1.8E^{308}$  for negative numbers and  $2.2E^{-308}$  to  $1.8E^{308}$  for positive numbers.

It is important to note, however, that floats and doubles are approximate numbers. This is due to two factors. First, the number of significant digits is a limiting factor. For example, you could not express the number 1,234,567.8 as a float because this number contains more than the permissible seven digits. In order to store the number as a float, it will be rounded to 1,234,568, a



number containing the permissible seven digits. This number could easily be expressed as a double, as it contains less than the permissible 15 significant digits. There are also some limitations to numbers a binary value can represent. One analogy that can be made would be in expressing fractions versus decimals. The fraction 1/3 represents a particular value. However, if you try to express this number as a decimal, the number will need to be rounded at some point. It could be expressed as 0.3333333, however, this is still an approximation of the actual value. Just as fractions cannot always be expressed as decimals, some numbers cannot be exactly expressed in binary code, and these numbers are replaced by approximate values. One example of such a number is 0.1. This number cannot be expressed as a binary number. However, the number 0.099999 can be expressed in binary. Thus 0.1 would be replaced with an approximate value of 0.099999.

In choosing the numeric data type, there are two things to consider. First, it is always best to use the smallest byte size data type needed. This will not only minimize the amount of storage required for your geodatabase but will also improve the performance. You should also consider the need for exact numbers versus approximate numbers. For example, if you need to express a fractional number and seven significant digits will suffice, use a float. However, if the number must be more precise, choose a double. If the field values will not include fractional numbers, choose either a short or long integer.

## **Text fields**

A text field represents a series of alphanumeric symbols. This can include street names, attribute properties, or other textual descriptions. An alternative to using repeating textual attributes is to establish a coded value. A textual description would be coded with a numeric value. For example, you might code road types with numeric values assigning a 1 to paved improved

roads, a 2 to gravel roads, and so on. This has the advantage of using less storage space in the geodatabase; however, the coded values must be understood by the data user. If you define your coded values in a coded value domain in the geodatabase and associate the domain with the integer field storing your codes, the geodatabase will display the textual description when the table is viewed in ArcMap or ArcCatalog. For more information on coded value domains, see the chapter ‘Subtypes and attribute domains’ in this book.

## **Date fields**

The date data type can store dates, times, or date and times. The default format in which the information is presented is mm/dd/yyyy hh:mm:ss and a specification of AM or PM. When you enter date fields in the table, they will be converted to this format.

## **BLOB fields**

A binary large object is simply some data stored in the geodatabase as a long sequence of binary numbers. Items such as images, multimedia, or bits of code can be stored in this type of field.

## **Global identifier fields**

GlobalID and GUID data types store registry style strings that uniquely identify a feature or table row within a geodatabase and across geodatabases. Developers can use them in relationships or in any application requiring globally unique identifiers. In a relationship, if a GlobalID field is the origin key, a GUID field must be the destination key. You must add the GlobalID field programmatically; however, once you add it ArcGIS maintains its values. You can create the GUID field in ArcCatalog, but you must maintain its values.

Name	Specific range, length, or format	Size (Bytes)	Applications
Short integer	-32,768 to 32,767	2	numbers without fractions within specific range; coded values
Long integer	-2,147,483,648 to 2,147,483,647	4	numbers without fractions within specific range
Single-precision floating point number (Float)	approx. $-3.4E^{-38}$ to $1.2E^{38}$	4	numbers with fractions within specific range
Double-precision floating point number (Double)	approx. $-2.2E^{-308}$ to $1.8E^{308}$	8	numbers with fractions within specific range
Text	up to 64,000 characters	varies	names or other textual qualities
Date	mm/dd/yyyy hh:mm:ss AM/PM	8	date and/or time
BLOB	varies	varies	images or other multimedia
GUID	36 characters enclosed in curly brackets	16 or 38	customized applications requiring global identifiers

Databases with a native GUID data type, such as personal geodatabases and ArcSDE geodatabases in Microsoft SQL Server, store Global ID and GUID values as 16 bytes. Databases without a native GUID data type store them as 38 bytes.

## Data types outside ArcGIS

The data types explained in this section include all the data types available when creating a table using ArcMap or ArcCatalog. However, you might choose to store your tables in another DBMS such as Oracle or dBASE. When this is done, the data types between ArcGIS and your DBMS might not match directly. The types are matched to the closest data type available in the DBMS. This process is referred to as data type mapping. In this process, it is possible that the values will be stored in the DBMS as a different type, applying different criteria to the data attribute. To learn more about the data type mapping process with your database management system, see the *Configuration and Tuning Guide for <DBMS>* PDF file.

# Setting an appropriate geodatabase spatial domain

For spatial data to be appropriately stored and referenced to a location on the earth, it must have a spatial reference composed of a coordinate system and precision. The coordinate system (geographic or projected) defines the location of the spatial data on the earth. For example, using the GCS\_North\_American\_1983 geographic coordinate system, ESRI headquarters are located at -117.195533 longitude and 34.057058 latitude. Precision defines the level of detail that is maintained when data values are stored in the geographic database. For example, if you store the above coordinates and your precision only maintains two decimal places, the values -117.20, 34.06 (rounded) would be stored in the geographic database. If these coordinates are rounded to two decimal places, the point would represent an ellipse on the earth's surface of 1,109 by 923 meters. Therefore, careful consideration should be given to choosing an appropriate data precision to maintain the precision of your data collection.

For information on choosing an appropriate coordinate system, see the Map projections topic in the ArcGIS Desktop Help. The rest of this topic discusses how to set the geodatabase precision aspect of the spatial reference. The first section discusses the fundamentals of geodatabase precision. The second section discusses different approaches for calculating precision appropriate for your data.

## About geodatabase precision

The geodatabase stores coordinates as positive 4-byte integers that have a maximum value of 2,147,483,648. This range of integers is called a spatial domain. It may seem that you are limited to storing one foot or one meter precision with an integer, but that is not the case; you decide what your 4-byte integer units represent. If you need to store meter precision, then you have 2.14 billion meters to work with (approximately 53 times the circumference of the earth). Or you could decide to store centimeters, in which case you would have 2.14 billion

centimeters to work with (approximately one-half the circumference of the earth). The units that the 4-byte integer represents are called storage units. Storage units are the smallest measurable unit that can be stored for a dataset. The geodatabase converts storage units to units of the coordinate system on the fly, and vice versa, so you always work with decimals, even if you are using the lowest-level ArcObjects application programming interface (API). The geodatabase uses precision to convert between coordinate system units and storage units using the equation:

$$\text{Storage Units} = \text{Coordinate System Units} \div \text{Precision}$$

The following table shows examples of equivalent precision, coordinate system units, and storage units.

Storage units	Coordinate system units	Precision
1 cm	Meters	100
1 mm	Meters	1000
2 cm	Meters	50
1 inch	Feet	12

The geodatabase actually does a little more to convert between storage units and coordinate system units. The coordinates are also shifted during conversion. You only need to be concerned about this shift if you are manually calculating your spatial domain.

## Spatial domain extent

The relationship is inverse between the precision and the domain extent (the area you can store). Because you have 2.14 billion integers, there is an outer edge for the spatial domain. As your storage units get smaller (and precision gets larger) the extent of your spatial domain also gets smaller. If you attempt to add features outside the spatial domain, you will get the following error: “The coordinates or measures are out of bounds.” So it is important that you do not make your storage units so small, and therefore, your precision so large, that you will not be able to add features for your entire study area. With approximately 2.14 billion storage units to work with, you can avoid this problem by simply setting the precision appropriately. For example, you can store the entire world with 1-meter storage units but only half the world with 1-cm storage units. Using a decimal degree-based geographic coordinate system such as NAD 1983, you could use 1.9-cm storage units for the entire world in a single feature class.

## Benefits of storing integers

Performance is the reason the geodatabase uses integer storage instead of floating point storage. Internally, integer coordinates allow the geodatabase to perform spatial operations several orders of magnitude faster than operations using decimal coordinates. Also, integers can be compressed to consume fewer storage resources, producing better performance. Only enterprise (ArcSDE) geodatabases take advantage of integer compression, so this storage benefit does not apply to personal geodatabases. ArcSDE uses relative offset compression of the integer coordinate values to minimize storage resources. As the precision becomes larger, the relative offsets between coordinates will get larger, thereby increasing the storage requirements.

## How to set the spatial domain

Before you create a spatial domain, there are three things to consider:

1. Will the precision of the data maintain the precision of your data collection?
2. Will the spatial domain cover the entire extent of your study area?
3. Is the precision small enough to maximize integer compression (enterprise geodatabases only)?

You don't always need to worry about all these issues. Many times you can let the default settings generated by the software deal with these issues for you. Below are three different approaches to creating a spatial domain. Choose the one that is most appropriate for your application.

- A. Accept the defaults when importing data.
- B. Define a spatial domain by setting the extent and maximizing the precision.
- C. Define a spatial domain by manually calculating your precision and extent.

### Approach A: Accepting the defaults when importing data

This is the easiest of the approaches. You simply take the default spatial domain generated by the software.

Use this approach if you

- Have at least one vector dataset or a group of tiled datasets that covers the entire extent of your study area.
- Want the most precision possible within your study area.

If you have a dataset that covers the entire study area, import the dataset first and accept the default values for the spatial domain. The defaults will create a domain that encompasses all of the features with a little room to grow. If you have tiled datasets that together cover the entire study area, calculate a spatial domain that encompasses all of the datasets using the Create Spatial Reference tool. Then, create an empty feature class with this spatial domain and load the tiled data into it.

Using this method, the precision will be maximized within the default extent. Because the resulting precision could be very large, this would not be the best approach if you are trying to maximize your ArcSDE geodatabase's performance. However, this approach will ensure that all your data will fit inside the spatial domain and that you are using the highest precision possible for your data.

As you create or import subsequent datasets to the geodatabase, use the spatial reference calculated from this original feature class. You can do this by importing the spatial reference from this feature class whenever you create new feature classes or feature datasets. You can also set your geoprocessing settings to use the spatial reference from this feature class.

### **Setting the geoprocessing environment to use a specific spatial reference**

1. In ArcCatalog or ArcMap, from the Tools menu, click Options.
2. Click the Geoprocessing tab.
3. Click the Environments button.
4. Expand General Settings.
5. For Output Spatial Reference, click As Specified Below.
6. Next to the following input box, click the folder icon.
7. On the Coordinate System tab, click Import.

8. Navigate to and select the first feature class that you imported into the geodatabase.
9. Click Add.
10. Click OK on all the open dialog boxes.

Now all subsequent geoprocessing operations, including new data imports performed by the current user on this machine, will use this spatial reference.

### **Approach B: Defining a spatial domain by setting the extent and maximizing the precision**

This approach helps you determine an extent for your study area, then maximizes the precision within that study area.

Use this approach if you

- Do not have a single vector dataset that covers the extent of your study area, but you can define your study area on a map.
- Want the most precision possible within your study area.

The result of this approach will be exactly the same as Approach A; therefore, this approach has the same strengths and weaknesses. Before you can begin, you must know which coordinate system you plan to use. For information on choosing a coordinate system, see the Map projections topic in the ArcGIS Desktop Help. If you plan to use the State Plane or UTM coordinate systems, you can find data defining the zone locations at <ArcGIS Installation location>\ArcGIS\Reference Systems in the usstpln83 and utm shapefiles.

## Determining the extent of your study area

1. Start ArcMap and add reference data for the world or your area of interest. Look for reference data in the following locations:
  - *ESRI Data and Maps* (included with ArcGIS)
  - <ArcGIS Installation location>\ArcGIS\Metadata\Data
  - Geography Network
2. Set the coordinate system of the data frame to the one that you want to use for the new dataset.
  - Open the data frame properties.
  - Click the Coordinate System tab.
  - Expand the Predefined folder and navigate to the coordinate system you plan to use.
  - Click OK.
3. Zoom in to the part of the world you plan to use as a study area.
4. Use the New Rectangle tool on the Draw toolbar to draw a rectangle on the map to define your new study area.
5. Right-click on that new rectangle and click Properties.
6. Click the Size and Position tab.
7. Under Position for Anchor Point, click the lower left box.
8. Copy and paste the coordinates in the X and Y text boxes into a text file. Delete the unit measure at the end of the coordinates. These coordinates represent the lower left corner of your study area.
9. Under Position for Anchor Point, click the upper right box.

10. Copy and paste the coordinates in the X and Y text boxes into a text file. Be sure to delete the unit measure at the end of the coordinates. These coordinates represent the upper right corner of your study area.

## Applying the calculated extent when creating a new feature class

1. In ArcCatalog, navigate to your geodatabase, right-click, and click New > Feature Class.
2. For name, type an appropriate name such as “StudyArea”.
3. Click Next.
4. If necessary, specify a configuration keyword and click Next.
5. In the Fields list, click the SHAPE field.
6. In the Field Properties table, click the Browse button next to the Spatial Reference property.
7. On the Coordinate System tab, click Select and select your coordinate system.
8. Click the X/Y Domain tab.
9. Copy and paste your coordinates from the text file into the appropriate text boxes on the X/Y Domain tab. Notice that your precision adjusts as you change your extent.
10. Click OK on the Spatial Reference Properties dialog box.
11. Click Finish on the New Feature Class wizard.

Now you can import the spatial reference from the StudyArea feature class you just created for all other data that you create in that study area. You can also set your geoprocessing environment so that all new data created from geoprocessing operations uses this spatial reference. See Approach A for how to set the geoprocessing environment to use a spatial reference from a feature class.

## Approach C: Defining the spatial domain by manually calculating your precision and extent

For this approach, you calculate the spatial domain parameters manually. Use this approach if you want to optimize performance in an ArcSDE geodatabase.

### Calculating precision

First, you must calculate an appropriate precision by choosing your storage units and calculating your precision accordingly. Set your storage units to be ten times smaller than the best precision of your data collection. This will ensure that the precision of your data collection is maintained in the geodatabase regardless of how you manipulate the data with ArcGIS (geoprocessing, topology cluster tolerance, geometry operations, and so on). Consider the following examples.

Data collection method	Coordinate system units	Equipment precision	Recommended storage unit
Digitize 1:250,000 map	Feet	+/-416 feet	1 foot
Professional GPS	Meters	+/-0.5 meter	.05 meter
Survey with theodolite	Meters	+/-5 millimeters	.5 millimeter

Precision is the multiplier that converts coordinate system units into storage units. Mathematically, precision equals one coordinate system unit divided by one storage unit. The following table shows appropriate precision values using the examples above.

Data collection method	Coordinate system units	Recommended storage unit	Precision
Digitize 1:250,000 map	Feet	1 foot	1
Professional GPS	Meters	.05 meters	20
Survey with theodolite	Meters	.5 millimeters	2,000

### Calculating precision with a geographic coordinate system

Calculating precision based on data that uses a geographic coordinate system (GCS) is slightly more difficult because angular units (degrees) are not consistent throughout the data. As the latitude changes, each degree represents a different amount of area on the ground. If you want to use a linear storage unit with data in a GCS, you will have to perform some calculations. If you calculate an appropriate precision when your angular units are at their largest, you will maintain even more precision in areas where angular units are smaller. For example, if you are maintaining 1m precision where one degree equals one hundred miles on the ground, your geodatabase will maintain 1cm precision where one degree equals one mile on the ground. In a geographic coordinate system, angular units are largest at the equator. Precision will equal the number of storage units in one degree at the equator. As mentioned above, this precision value should be multiplied by ten to account for any ArcGIS processing operations. The following equation can be used:



$$\text{Precision} = 10 * \text{GCS Equatorial circumference} \div 360 \div \text{Storage units}$$

For example, GCS\_WGS\_1984 has a circumference of 40075016.7 meters. With storage units of 1 meter, the equation would look like this:

$$\text{Precision} = 10 * 40075016.7 \div 360 \div 1 \text{ meter} = 1113195$$

Another option is to multiply the semimajor axis of the GCS by the number of radians per angular unit which is the equivalent of the circumference (semimajor axis \*  $2\pi$ )  $\div$  360

$$\text{Precision} = 10 * \text{Semimajor axis} * \text{radians per unit} \div \text{Storage units}$$

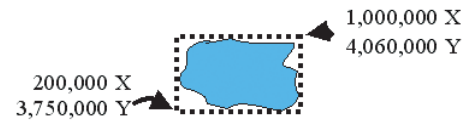
You can find this technical information about your GCS by opening its properties in ArcCatalog. If you don't see the Coordinate Systems folder in ArcCatalog, you can make coordinate systems visible from the General tab of the ArcCatalog Options dialog box.

If you are not working with a global dataset, you could calculate your precision based on your lowest latitude. This would allow you to create smaller precision values. However, if your study area ever expanded to lower latitude values, the coordinates stored in those locations would be less precise.

### Checking your precision against your study area

To validate that your precision will work given your study area, multiply the greater of the width or height (range) of your study area by the precision. If the result is less than 2,147,483,648, your data can fit inside a spatial domain with your chosen precision.

Even though your data can fit inside a spatial domain, your coordinates may fall outside the coordinate system boundary. Consider the following fictitious dataset with map units of meters.



A range of 800,000 (the width) multiplied by a precision of 1,000 equals 800,000,000, which is less than 2.14 billion; therefore, the data will fit. However, the upper right corner of the study area will be 1,000,000,000 x, 4,060,000,000 y in the spatial domain, or (1,000,000 x) \* 1,000 and (4,060,000 y) \* 1,000. Notice that the y value is outside the 0 to 2.14 billion coordinate system by approximately 1.9 billion units. To store these coordinates inside the geodatabase, you must shift the spatial domain to surround the data.

### Calculating an appropriate min x,y

Before you can shift the spatial domain to surround your data, you must identify the center of your spatial domain in map units. The goal is to place your data in the center of the spatial domain so the data can expand in all directions if necessary. All the calculations for shifting the coordinate system are in coordinate system units rather than storage units.

First, find the center of the spatial domain in storage units:

$$2,147,483,648 / 2 = 1,073,741,824$$

Next, convert the center in storage units to coordinate system units by dividing by the precision. This example uses a precision of 1,000.

$$1,073,741,824 / 1000 = 1,073,741.824$$

Now that you have found the center of the spatial domain in coordinate system units, you need to calculate a new minimum x and y of your spatial domain. The formula for calculating the minimum x and y of your spatial domain is:



$\text{Min X} = ([\text{DataMinX} + \text{DataMaxX}] / 2) - \text{Domain center in coordinate system units}$

$\text{Min Y} = ([\text{DataMinY} + \text{DataMaxY}] / 2) - \text{Domain center in coordinate system units}$

This equation finds the minimum coordinates of your spatial domain to locate the center of your data at the center of the domain. Remember, all these calculations are in coordinate system units. Examine this equation for the x dimension given the example data:

First, find the center of your data.

$(\text{DataMinX} + \text{DataMaxX}) \div 2$

$(200,000 + 1,000,000) \div 2 = 600,000$

Next, find the difference between the center of your data and the center of the geodatabase space.

$\text{Min X} = 600,000 - 1,073,741.824 = -473,741.824$

Because this is a negative number, the spatial domain will shift to the left. Remember, the shift is applied to the spatial domain, not the data. The shift is calculated for both dimensions, so you would need to repeat this process for the y coordinates.

## Using your calculated spatial domain in ArcCatalog

Once you have calculated an appropriate spatial domain, you are ready to create spatial data in the geodatabase. When creating your first dataset, navigate to the X/Y Domain tab of the spatial reference properties and enter the Min X, Min Y, and precision values that you calculated. The maximum x and y values will be calculated automatically. For all subsequent data that you import or create, you can simply import this spatial reference. You can also set your geoprocessing environment so that all new data created from geoprocessing operations uses this spatial reference. See Approach A for how to set the geoprocessing environment to use a spatial reference from a feature class.

## Defining Z and M spatial domains

The z and m domains are easier to calculate than the x and y domains. Examine your data and enter the lowest value for the minimum value and the precision to support its accuracy. You can calculate z and m precision the same way you calculated precision for the x and y coordinates. Just like x and y coordinates, you have 2,147,483,648 storage units to work with. Generally it is not necessary to center the z and m domains about the data as you can set an absolute minimum based on your data.

When calculating the minimum for a z domain, you could use the lowest point on the earth (-11,033 meters, located at the Mariana Trench off the coast of Japan). Generally m coordinates are positive numbers, so a minimum value of 0 may be appropriate. You may also set the minimum m to have a slight negative offset to account for negative values that could be produced by the extrapolation of measures during operations like Calibrate. These negative values could be corrected later instead of rejected during the extrapolation.

## Upgrading a geodatabase

Geodatabases built using previous versions of ArcGIS do not support some of the newer functions of ArcGIS.

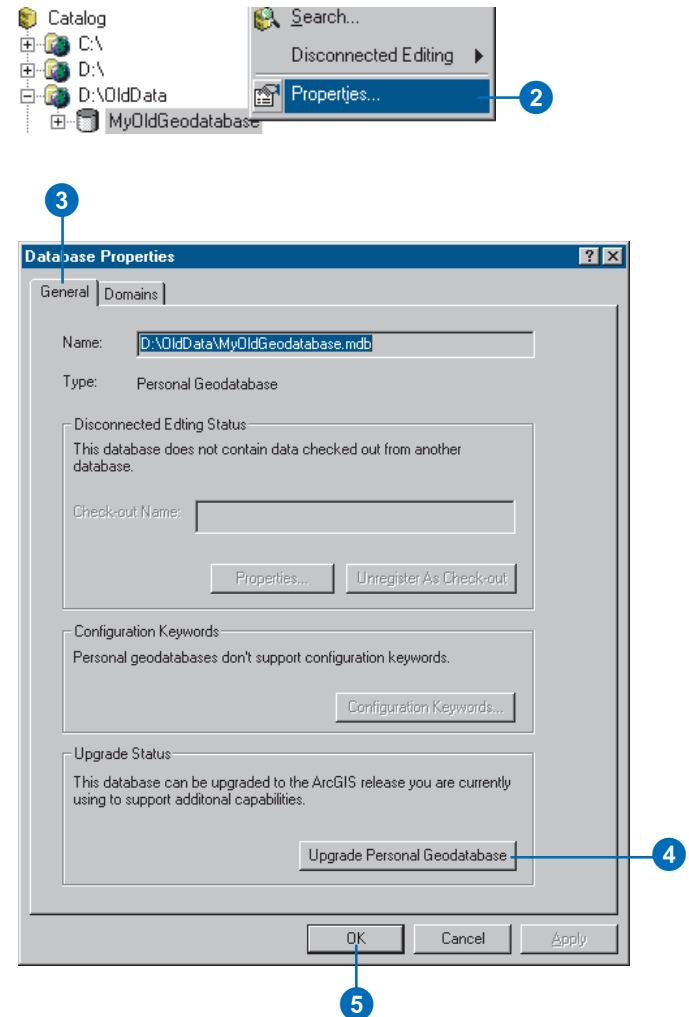
If your geodatabase was developed using a previous version of ArcGIS, you may wish to upgrade your geodatabase.

### Tip

#### Creating a backup copy of your geodatabase

*Bear in mind that once a geodatabase is upgraded, previous versions of ArcGIS cannot view or edit the geodatabase. For this reason, you may wish to make a copy of the geodatabase and upgrade the copy, thus leaving you with both an original and an upgraded geodatabase.*

1. Start ArcCatalog.
2. Right-click the geodatabase you want to upgrade and click Properties.
3. Click the General tab.
4. Click Upgrade Personal Geodatabase.
5. Click OK.



## Creating tables

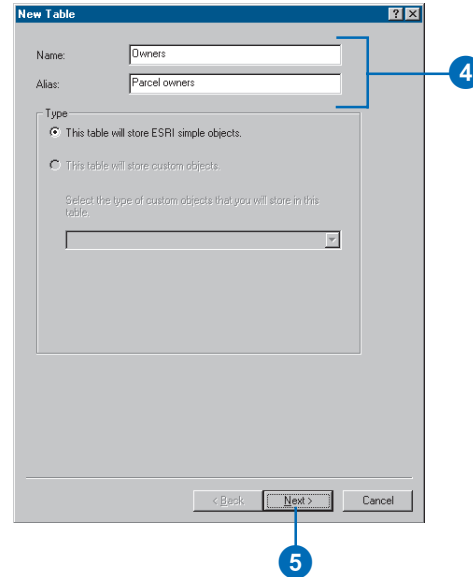
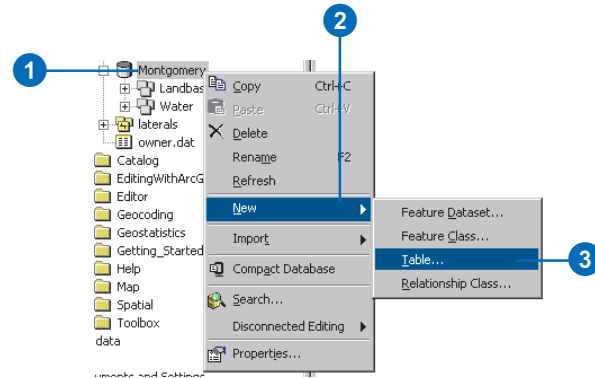
You can create tables in a geodatabase with an easy-to-use table designer.

When defining a table's fields, be aware that each database has its own rules defining what names and characters are permitted. The designer checks the names you type using a set of common rules, but each database is slightly different. If you want more control over a field's data types or structure, create the table directly in the database.

### See Also

For details about using configuration keywords with ArcSDE, see the ArcSDE Configuration and Tuning Guide for <DBMS> PDF file.

1. Right-click the database in the ArcCatalog tree in which you want to create a new table.
2. Point to New.
3. Click Table.
4. Type a name for the table. To create an alias for this table, type the alias.
5. Click Next. ▶



## Tip

### Using another table as a template

When creating a new table, you can use another table as a template. Click **Import**, navigate to the table whose field definitions you want to copy, then click **OK**. Now you can edit the field names and their data types.

## Tip

### Deleting a field

If you have entered a field that you do not want to include in the new table, select it by clicking its tab in the grid, then press **Delete**.

If your geodatabase does not use ArcSDE, skip to step 8.

6. If you want to create the table using a custom storage keyword, click **Use configuration keyword** and type the keyword you want to use.
7. Click **Next**.
8. Click the next blank row in the **Field Name** column and type a name to add a field to the table.
9. Click in the **Data Type** column next to the new field's name and click its data type. ▶

**New Table**

Specify the database storage configuration.

Configuration Keyword:

**Default**  
This option uses the default storage parameters for the new table/feature class.

Use configuration keyword  
This option allows you to specify a configuration keyword which references the database storage parameters for the new table/feature class.

< Back   **Next >**   Cancel

**New Table**

Field Name	Data Type
OBJECTID	Object ID
FirstName	Text
LastName	Text
DOB	Date
MemberType	Text
	Short Integer
	Long Integer
	Float
	Double
	Text
	Date
	Blob

Click any field to see its properties.

Field Properties:

Alias	
Allow NULL values	Yes
Default Value	
Domain	
Length	255

Import...

To add a new field, type the name into an empty row in the Field Name column, click in the Data Type column to choose the data type, then edit the Field Properties.

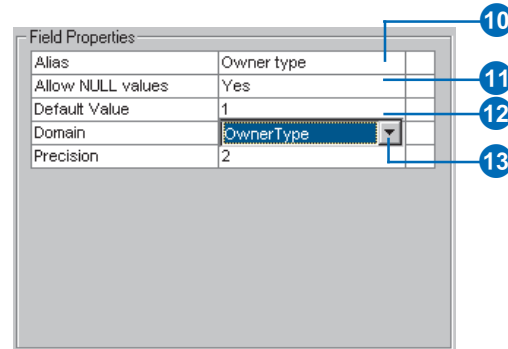
< Back   **Finish**   Cancel

## Tip

### OBJECTID field

Tables in the geodatabase require an OBJECTID type field. It uniquely identifies each object stored in the table in the database.

10. Click the field next to Alias and type the alias for this field.
11. Click the field next to Allow NULL values, click the dropdown arrow, then click No to prevent nulls from being stored in this field.
12. Click the field next to Default Value and type the value to associate a default value with this field.
13. Click the field next to Domain, click the dropdown arrow to see a list of the domains that apply to this field type, then click the domain to associate a domain with this field.
14. Set field properties by either clicking the property in the dropdown list or typing the property value, specific to the type of field.
15. Repeat steps 8 through 14 until all the table's fields have been defined.
16. Click Finish.



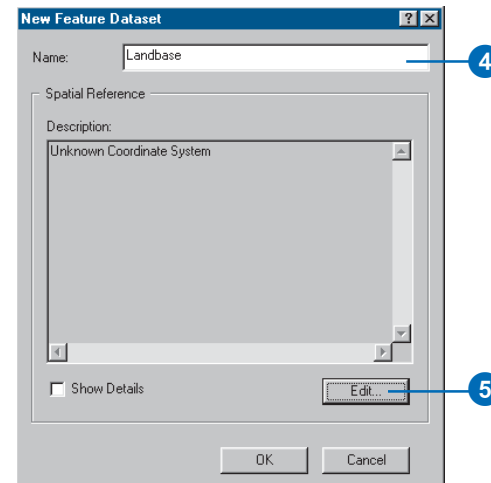
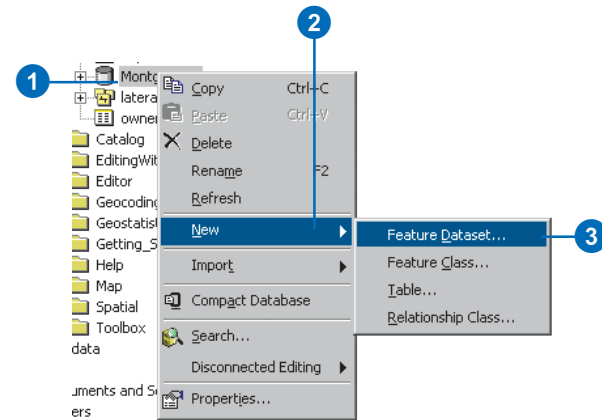
## Creating feature datasets

When creating a new feature dataset, you must define its spatial reference. This includes its coordinate system, either geographic or a specific projection, and the coordinate domains—the minimum x-, y-, z-, and m-values and their precision. All feature classes in the dataset must use the same coordinate system, and each coordinate of every feature in all feature classes must fall within the coordinate domains. The exception to the rule is m domains; feature classes in the same feature dataset can have different m domains.

When defining the coordinate system, you can choose a predefined coordinate system, use an existing feature dataset or standalone feature class as a template, or define a custom geographic or projected coordinate system.

### Creating a feature dataset with a predefined coordinate system

1. Right-click the database in the ArcCatalog tree in which you want to create a new feature dataset.
2. Point to New.
3. Click Feature Dataset.
4. Type a name for the feature dataset.
5. Click Edit to define the feature dataset's spatial reference. ▶

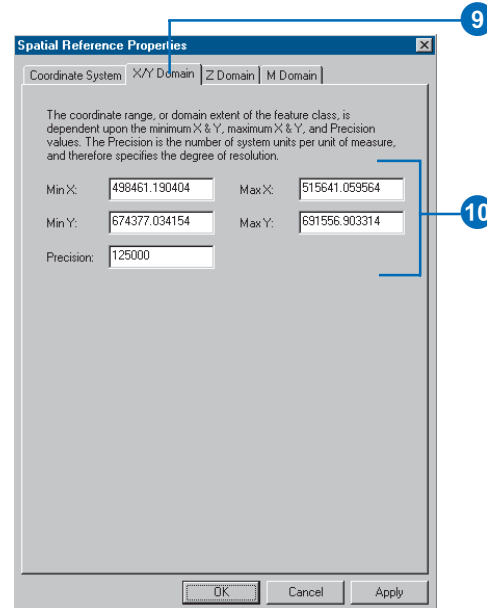
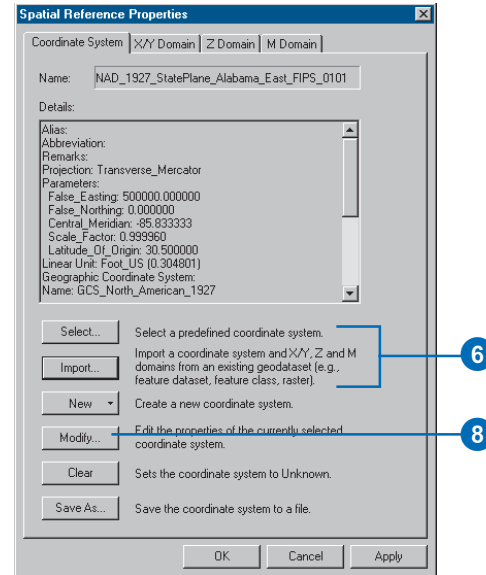


## Tip

### Saving the coordinate system

You can click *Save As* to save the coordinate system as a .prj file.

- Click Select or Import to set the feature dataset's spatial reference.
- Navigate to the spatial reference you want to use or navigate to the feature class or feature dataset whose spatial reference you want to use as a template.
- Click Modify if you want to change any parameters in the coordinate system you have chosen. Edit the coordinate system's parameters and click OK.
- Click the X/Y Domain tab.
- Type the minimum x, minimum y, maximum x, and maximum y coordinate values for the dataset and type the required precision for the coordinate values. ►



## Tip

### Precision

*Since the size of the spatial domain is dependent on the value of precision, when the precision is changed, the maximum m- or z-value will change to fit within the size of the spatial extent. Similarly, when the maximum m- or z-value is changed, the precision will also change to fit the domain extent.*

11. Click the Z Domain tab.
12. Type the minimum z-value and maximum z-value for the dataset and type the precision required for the z coordinates if any feature class in the feature dataset will have z-values.
13. Click the M Domain tab.
14. Type the minimum m-value and maximum m-value for the dataset and type the precision required for the m values if any feature class in the feature dataset will have m-values.
15. Click OK. ►

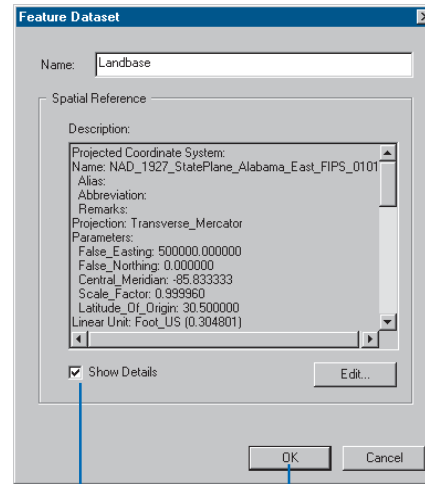
The screenshot shows the 'Spatial Reference Properties' dialog box with the 'Z Domain' tab selected. The 'Min' field is set to 0 and the 'Max' field is set to 2147483.645. The 'Precision' field is set to 1000. A blue callout line labeled '11' points to the 'Z Domain' tab. Another blue callout line labeled '12' points to the 'Min', 'Max', and 'Precision' input fields.

The screenshot shows the 'Spatial Reference Properties' dialog box with the 'M Domain' tab selected. The 'Min' field is set to 0 and the 'Max' field is set to 2147483.645. The 'Precision' field is set to 1000. A blue callout line labeled '13' points to the 'M Domain' tab. Another blue callout line labeled '14' points to the 'Min', 'Max', and 'Precision' input fields. A third blue callout line labeled '15' points to the 'OK' button.



16. Check Show Details to see the details of your new dataset's spatial reference.

17. Click OK.



## Tip

### Editing predefined parameters

You can easily create variations of a predefined coordinate system. For example, choose a predefined datum from the dropdown list; the text boxes now contain the selected datum's parameters and their contents are read-only. Now choose <custom> from the datum dropdown list. The contents of the text boxes do not change, but you can now edit their values. Type a name for your datum in place of <custom>.

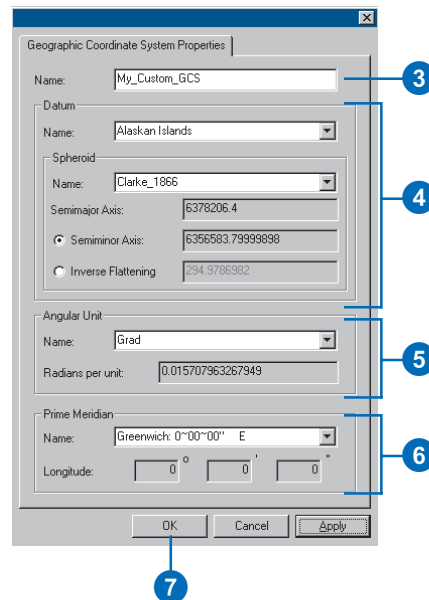
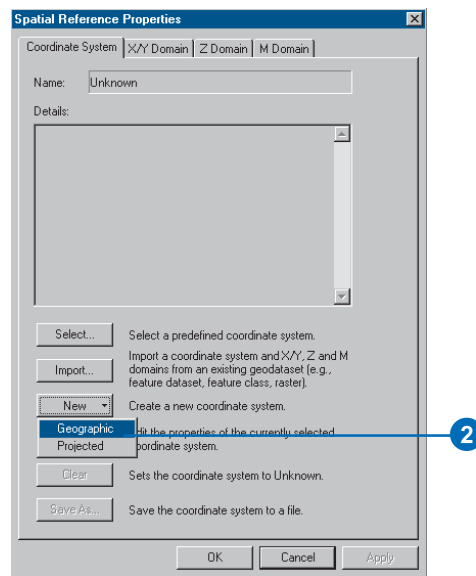
## Tip

### Saving the coordinate system

You can click Save As to save the coordinate system as a .prj file.

## Defining new geographic coordinate systems

1. Follow steps 1 through 5 for 'Creating a feature dataset with a predefined coordinate system'.
2. Click New and click Geographic.
3. Type a name for the coordinate system.
4. Type the parameters for a custom datum or choose a predefined datum from the dropdown list.
5. Type the angular unit or choose a predefined angular unit from the dropdown list.
6. Type the degrees, minutes, and seconds defining the prime meridian's longitude, or choose a predefined prime meridian from the dropdown list.
7. Click OK.
8. Follow steps 9 through 16 for 'Creating a feature dataset with a predefined coordinate system'.

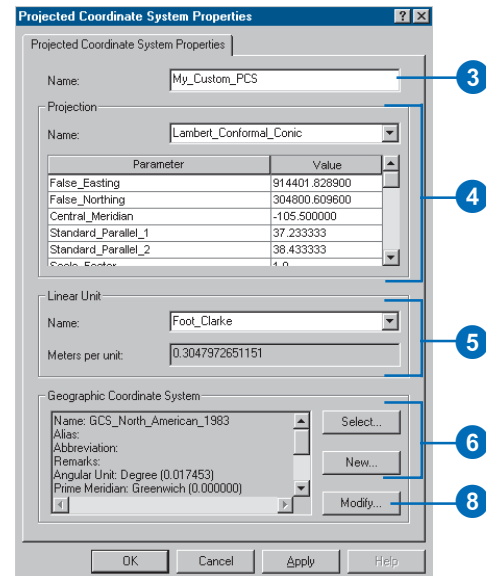
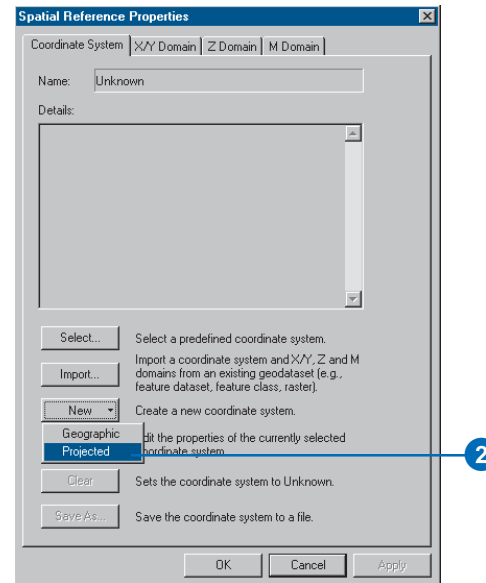


## See Also

For information about which parameters are appropriate for which projection, see Understanding Map Projections.

## Defining new projected coordinate systems

1. Follow steps 1 through 5 for 'Creating a feature dataset with a predefined coordinate system'.
2. Click New and click Projected.
3. Type a name for this coordinate system.
4. Choose a projection from the dropdown list and type the appropriate parameter values for that projection.
5. Type the linear unit or choose a predefined linear unit from the dropdown list.
6. Click Select or New to set the geographic coordinate system.
7. Navigate to the geographic coordinate system or navigate to the feature class or feature dataset whose geographic coordinate system you want to use as a template.
8. Click Modify if you want to change any parameters in the geographic coordinate system you have selected.
9. Click OK.
10. Follow steps 9 through 16 for 'Creating a feature dataset with a predefined coordinate system'.



## Creating feature classes

You create empty feature classes in ArcCatalog. When creating a feature class, you choose whether to create one that stores simple features (points, lines, or polygons) or one that stores annotation, network features, dimension features, or raster catalogs. You also define the fields it will contain and the geometry field's properties such as its spatial index and geometry type.

All feature classes in a feature dataset must use the same spatial reference, which was defined when the feature dataset was created. The exceptions to the rule are m domains; feature classes in the same feature dataset can have different m domains. When creating a standalone feature class, you must define its spatial reference.

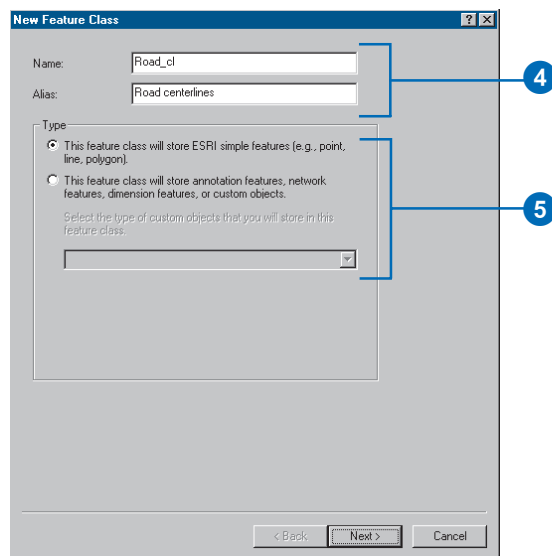
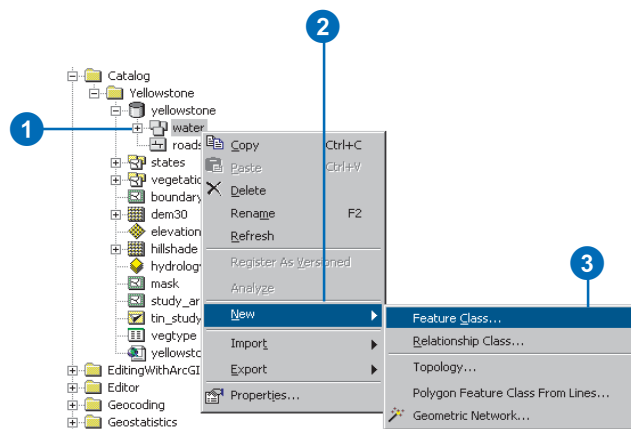
### See Also

For details about using configuration keywords with ArcSDE, see the ArcSDE Configuration and Tuning Guide for <DBMS> PDF file.

## Creating a feature class in a feature dataset

1. Right-click the feature dataset in the ArcCatalog tree in which you want to create a new feature class.
2. Point to New.
3. Click Feature Class.
4. Type a name for the feature class. To create an alias for this feature class, type the alias.
5. Specify the type of features the feature class will contain.

Click Next. ►

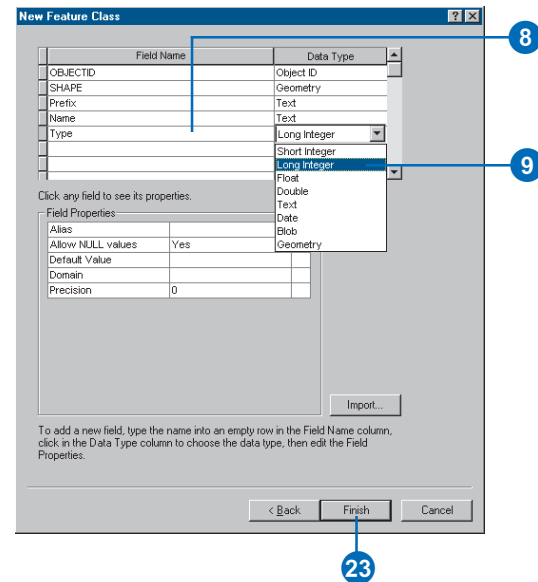
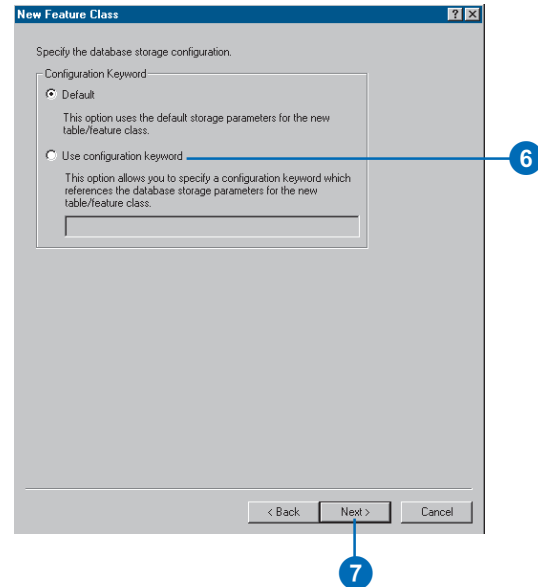


## Tip

### Using another feature class as a template

When creating a new feature class, you can use another feature class as a template. Click **Import**, navigate to the feature class whose field definitions you want to copy, and click **OK**. Now you can edit the field names and their data types.

- If your geodatabase does not use ArcSDE, skip to step 8.
- Click **Use configuration keyword** and type the keyword you want to use if you want to create the table using a custom storage keyword.
  - Click **Next**.
  - Click the next blank row in the **Field Name** column and type a name to add a field to the feature class.
  - Click in the **Data Type** column next to the new field's name and click its data type.

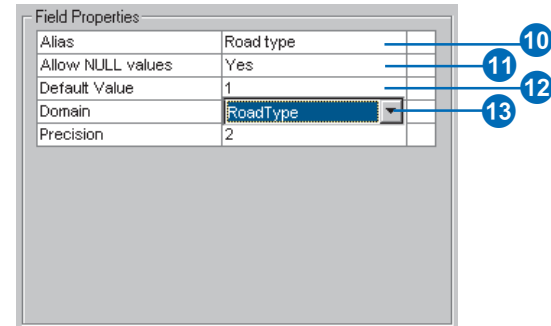


## Tip

### OBJECTID and Shape fields

All simple feature classes in the geodatabase require an OBJECTID and geometry type fields. The default OBJECTID and geometry fields cannot be deleted in this wizard.

10. Click the field next to Alias and type the alias for this field.
11. Click the field next to Allow NULL values, click the dropdown arrow, then click No to prevent nulls from being stored in this field.
12. Click the field next to Default Value and type the value to associate a default value with this field.
13. Click the field next to Domain, click the dropdown arrow to see a list of the domains that apply to this field type, then click the domain to associate a domain with this field.
14. Set field properties by either clicking the property in the dropdown list or typing the property value, specific to the type of field.
15. Repeat steps 8 through 14 until all the table's fields have been defined. ►

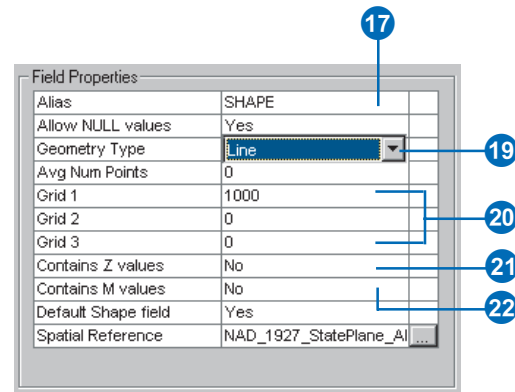


## Tip

### The Spatial Reference button

When adding a feature class to a feature dataset, this button lets you review the feature dataset's spatial reference parameters; however, you cannot change them.

- Click the name of the geometry field in the Field Name column.
- Click the field next to Alias and type the alias to create an alias for the geometry field.
- Click the field next to Allow NULL values, click the dropdown arrow, then click No to prevent null shapes from being stored.
- Click the field next to Geometry Type, click the dropdown arrow, and click the type of features you want to store in this feature class.
- Click the fields next to the grid size you want to specify and type the grid value to set the spatial index grid parameters for the feature class.
- Click the field next to Contains Z values, click the dropdown arrow, and click Yes if you want the shapes in this feature class to store z-values.
- Click the field next to Contains M values, click the dropdown arrow, and click Yes if you want the shapes in this feature class to store m-values.
- Click Finish.





## Tip

### Saving the coordinate system

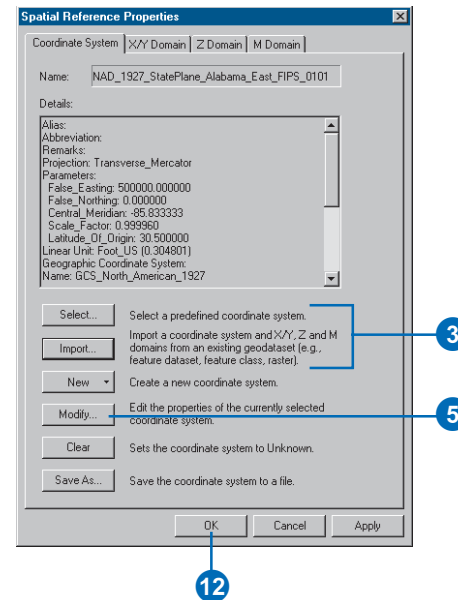
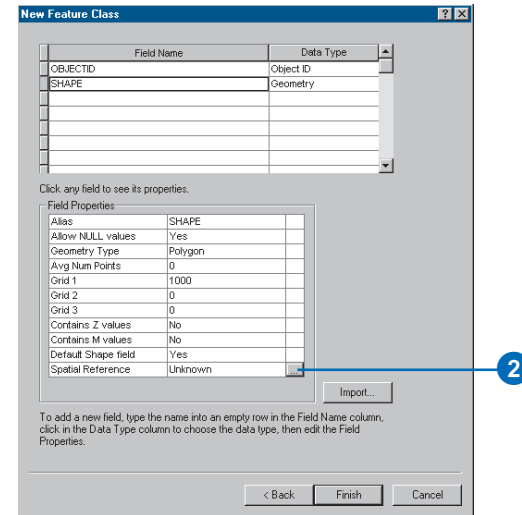
You can click *Save As* to save the coordinate system as a .prj file.

## See Also

For examples of how to define new geographic and projected coordinate systems, see 'Creating feature datasets' in this chapter.

## Creating a standalone feature class

1. Follow steps 1 through 22 for 'Creating a feature class in a feature dataset'.
2. Click the Spatial Reference properties button to define the feature class's coordinate system.
3. Click Select or Import to set the feature dataset's spatial reference.
4. Navigate to the spatial reference you want to use or navigate to the feature class or feature dataset whose spatial reference you want to use as a template. Click Add.
5. Click Modify if you want to change any parameters in the coordinate system you have chosen. Edit the coordinate system's parameters and click OK. ▶



## Tip

### Using another feature class as a template for the spatial reference only

Click *Import* to populate the *Spatial Reference Properties* dialog box with information from another feature class. You can then customize the template's spatial reference.

## Tip

### Specifying a custom coordinate system

To modify a predefined (or a template's) coordinate system or to define a custom coordinate system from scratch, click *Custom* on the *Coordinate System* dialog box.

- Click the X/Y Domain tab.
- Type the minimum x, y and maximum x, y coordinate values for the dataset and type the required precision for the coordinate values.
- Click the Z Domain tab, if present. If your feature class does not store z-values, skip to step 10.
- Type the minimum z-value and maximum z-value for the dataset, then type the precision required for the z coordinates. ▶

The screenshot shows the *Spatial Reference Properties* dialog box with the *X/Y Domain* tab selected. The dialog box has a title bar with a close button. Below the title bar is a tabbed interface with three tabs: *Coordinate System*, *X/Y Domain*, *Z Domain*, and *M Domain*. The *X/Y Domain* tab is active. Below the tabs is a text box containing the following text: "The coordinate range, or domain extent of the feature class, is dependent upon the minimum X & Y, maximum X & Y, and Precision values. The Precision is the number of system units per unit of measure, and therefore specifies the degree of resolution." Below this text are four input fields: *Min X:* 498461.190404, *Max X:* 515641.059564, *Min Y:* 674377.034154, and *Max Y:* 691556.903314. Below these fields is a *Precision:* field with the value 125000. At the bottom of the dialog box are three buttons: *OK*, *Cancel*, and *Apply*. A blue circle with the number 6 is positioned to the right of the *X/Y Domain* tab, and a blue circle with the number 7 is positioned to the right of the *Max Y* field.

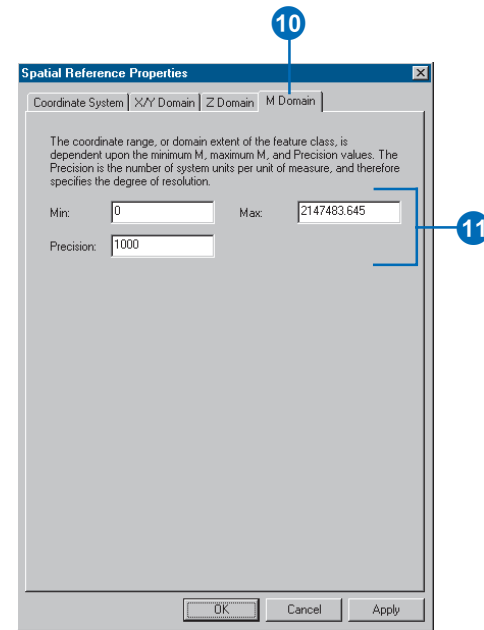
The screenshot shows the *Spatial Reference Properties* dialog box with the *Z Domain* tab selected. The dialog box has a title bar with a close button. Below the title bar is a tabbed interface with three tabs: *Coordinate System*, *X/Y Domain*, *Z Domain*, and *M Domain*. The *Z Domain* tab is active. Below the tabs is a text box containing the following text: "The coordinate range, or domain extent of the feature class, is dependent upon the minimum Z, maximum Z, and Precision values. The Precision is the number of system units per unit of measure, and therefore specifies the degree of resolution." Below this text are three input fields: *Min:* 0, *Max:* 2147483.645, and *Precision:* 1000. At the bottom of the dialog box are three buttons: *OK*, *Cancel*, and *Apply*. A blue circle with the number 8 is positioned to the right of the *Z Domain* tab, and a blue circle with the number 9 is positioned to the right of the *Max* field.

## Tip

### Precision

*Since the size of the spatial domain is dependent on the value of precision, when the precision is changed, the maximum z-value will change to fit within the size of the spatial extent. Similarly, when the maximum z-value is changed, the precision will change to fit the domain extent.*

10. Click the M Domain tab, if present. If your feature class does not store m-values, skip to step 12.
11. Type the minimum m-value and maximum m-value for the dataset, then type the precision required for the m-values.
12. Click OK.



# Creating indexes

Once you have data in a table or feature class, you may want to create attribute indexes to make your queries faster. Spatial indexes increase the selection speed of graphical queries on spatial features. An attribute index is an alternate path used by the DBMS to retrieve a record from a table. It is much faster to first look up the index and go to the appropriate record than to start at the first record and search through the entire table.

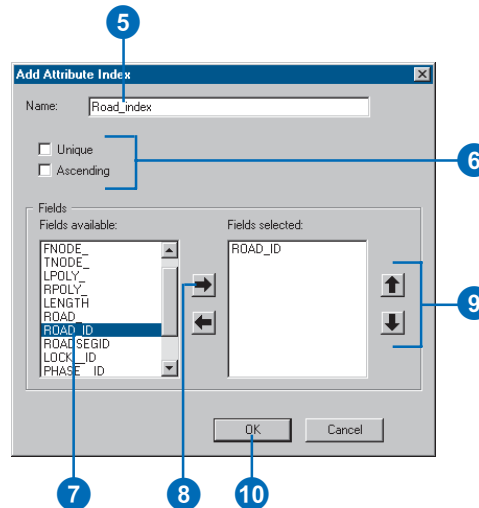
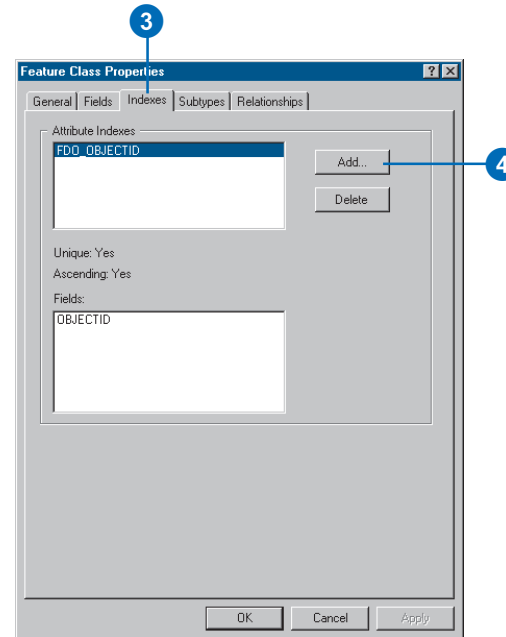
Attribute indexes can be created for single or multiple fields on the feature class and table property pages. Once an index has been added, it can be deleted and added again at any point in the lifetime of the feature class or table.

You can use the same property page to delete a spatial index from and add a spatial index to your feature class.

You can modify the spatial index for an ArcSDE feature class by deleting the index and readding it. You cannot access the features stored in an ArcSDE feature class if it doesn't have a spatial index. ►

## Creating a new attribute index

1. Right-click the table or feature class in the ArcCatalog tree for which you want to create an index.
2. Click Properties.
3. Click the Indexes tab.
4. Click Add.
5. Type the name for the new index.
6. Check the Unique check box if your field values are unique. Check the Ascending check box to create an ascending index. Data in an ascending index is returned in ascending order.
7. Click the field or fields for which you want to build this index.
8. Click the arrow button to move the fields to the Fields selected list.
9. Use the up and down arrows to change the order of the fields in the index.
10. Click OK. ►



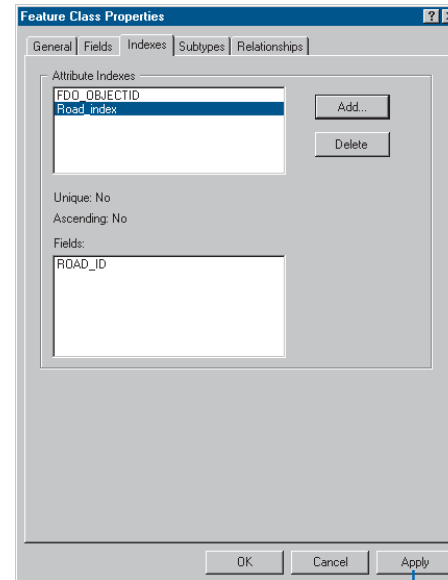
Building a new spatial index for an ArcSDE feature class is a server-intensive operation—it should not be done on very large feature classes when a large number of users are logged in to the server.

### Tip

#### Deleting an index

*You can delete an index by clicking it in the Attribute Indexes list and clicking Delete.*

11. Click Apply to build the index.



11

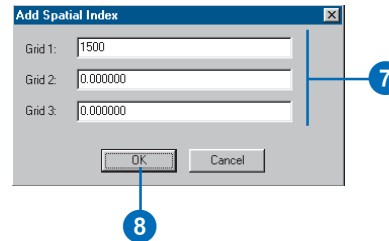
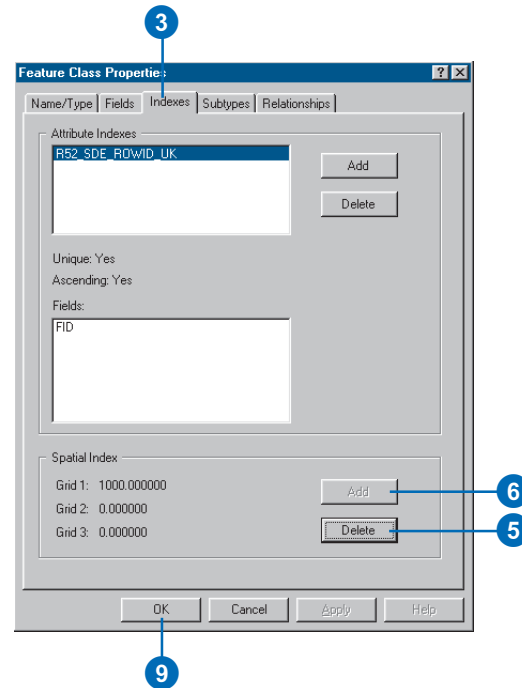
## Tip

### ArcSDE spatial indexes

For more information on what it means to have an ArcSDE feature class with no spatial index, see Managing ArcSDE Services.

## Modifying the spatial index

1. Right-click the feature class in the ArcCatalog tree whose spatial index you want to modify.
2. Click Properties.
3. Click the Indexes tab.
4. Delete the spatial index first if there is one. If there is no spatial index, skip to step 6.
5. Click Delete.
6. Click Add.
7. Type new index parameters if you do not want to use the ones already in the settings for this feature class.
8. Click OK.
9. Click OK to build the spatial index and close the property page.



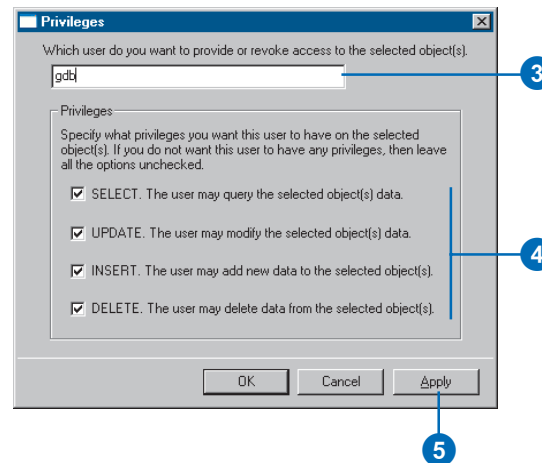
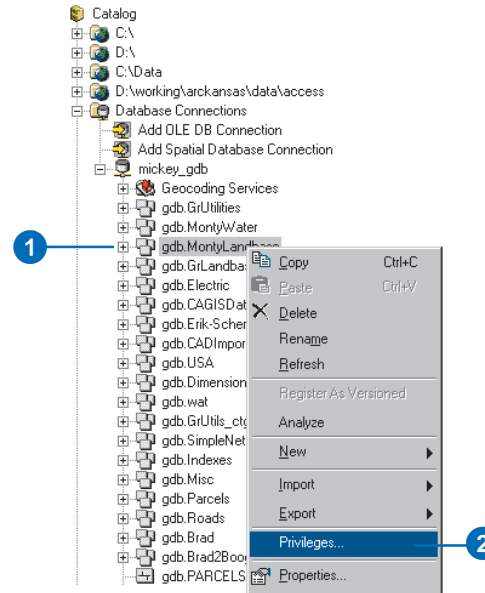
# Granting and revoking privileges

If you want to let other database users view and/or modify the contents of any items, you must grant them the privilege to do so. The same tool for granting privileges can also be used to revoke privileges from a particular database user.

You have several options when granting privileges. You can specify that a user has no privileges. You can grant “select” privileges, meaning that they can view, but not modify, the contents of an item. Alternatively, you can grant a user full privileges (Select, Update, Insert, and Delete) to both view and modify the contents of an item.

Granting or revoking privileges on a feature dataset causes all of its contents to have the same privilege changes. When you add new items to a feature dataset or build a geometric network (see the chapter ‘Geometric networks’ in this book), you will need to grant privileges on the feature dataset again.

1. Right-click the item or items in the ArcCatalog tree for which you want to grant privileges.
2. Click Privileges.
3. Type the name of the user whose privileges you want to change.
4. Click the privileges you want them to have. UPDATE, INSERT, and DELETE will only be active if SELECT is checked. If you leave all options unchecked, the user will have all access privileges revoked.
5. Click Apply to enable the privileges.







# Importing data

## IN THIS CHAPTER

- **Importing data into new feature classes and tables**
- **Registering ArcSDE data with the geodatabase**
- **Loading data into existing feature classes and tables**
- **Copying data between geodatabases**
- **Updating DBMS statistics**

With ArcCatalog, you can import shapefiles, coverages, CAD data, and INFO and dBASE tables into a geodatabase. For each feature class and table you import, you create a new feature class or table in the geodatabase. You can also load data from any of these formats into an existing feature class or table.

For data in formats other than those listed above or discussed in this chapter, you use other processes to migrate the data. To import coverage annotation, please see the chapter ‘Managing annotation’, and to import raster data, please see the chapter ‘Building a raster geodatabase’. If your data is stored in TIGER® files or in another format, ArcCatalog has the tools you need to convert the data into a format that can be imported or loaded into the geodatabase.

In other cases, you may already have data in a geodatabase but want to copy it to another geodatabase. You can import and load data from one geodatabase feature class or table to another the same way you import and load other formats.

ArcGIS provides additional tools that copy data between geodatabases. In ArcCatalog, you can copy and paste feature datasets, classes, and tables between geodatabases. In ArcMap, you can select features or records using any selection method and export them to another geodatabase. If you want to share a small amount of data with someone, you can export to a ZIP file and send it to them.

Importing or copying large amounts of data into an ArcSDE geodatabase requires careful planning to ensure good performance. Before you begin, work with your database administrator to review the data sources and the new feature classes and tables you will create. Your database administrator will provide you with configuration keywords defining the optimum way to create and store the new feature classes and tables in your DBMS. You specify the keywords when you use the tools outlined in this chapter.

Once you've finished creating feature classes and tables and loading data, update the DBMS statistics for them. This too is essential to maximize performance.

You can import and load data into a personal geodatabase with ArcView. You must have an ArcEditor or ArcInfo license to import or load into an ArcSDE geodatabase.

# Importing data into new feature classes and tables

ArcCatalog contains tools that import a coverage, shapefile, CAD file, INFO table, dBASE table, geodatabase feature class, or table. The tools also allow you to import more than one feature class and table at the same time. For each feature class or table you import, you create a new feature class or table in the geodatabase.

As the geodatabase stores data different from the formats you're importing, ArcCatalog will automatically convert the geometry and fields you import to types used by the geodatabase.

## Importing a feature class

You use the Feature Class to Feature Class tool to import a shapefile, coverage, or a CAD feature class into a new geodatabase feature class. You can also import a feature class from another geodatabase. You can specify which fields to import and how to name them, and limit which features import by specifying a query.

The feature class you create can stand alone or import into an existing feature dataset.

When you create a standalone feature class, the new feature class is created with the same coordinate system as the feature class you're importing. It's also created with the same spatial domain but expanded by a factor of 5 percent.

If you're creating a feature class in an existing feature dataset, the new feature class will automatically take on the same coordinate system, spatial domain, and precision as the feature dataset. For more information about spatial references, see the chapter 'Creating new items in a geodatabase' in this book.

If you're importing into an ArcSDE geodatabase, you can specify a configuration keyword to customize how the feature class is created and stored.

You can specify one spatial index grid size if you're importing into a personal geodatabase, and up to three if you're importing into an ArcSDE geodatabase. A poorly defined grid size can reduce spatial query performance in ArcMap, so if you're unfamiliar with creating spatial grids, use the defaults. For information on the spatial index grid, see the chapter 'Creating new items in a geodatabase'.

## Importing a table

Importing tables is a similar process to importing feature classes. When you import a table, you can choose which fields to import and how to name them, and you can choose which records to import by specifying a query.

## Importing several feature classes or tables

If you're importing a number of feature classes or tables into a geodatabase and they require the same settings at import, you can use the Feature Class to Geodatabase or Table to Geodatabase tool to import them at the same time. One feature class or table will be created for each feature class or table you import.

On the other hand, many of the feature classes and tables you import may require individual settings at import, such as for the destination geodatabase, custom configuration keyword, or spatial index. If this is the case, you can create and run a model instead of manually repeating the steps outlined in this chapter. A model helps automate importing by allowing you to save and reuse environment settings and tool parameters. Once you've created a model, you can import data, edit the model to specify other input data, modify any parameters you wish, then rerun the model with a single click. For more information on models, see *Geoprocessing in ArcGIS*.

## Importing coverages and INFO tables

Coverages contain several fields that are relevant to the coverage data model only and are not maintained by the geodatabase; therefore, you shouldn't import them.

When importing polygon or point coverages, don't import <cover#>, AREA, or PERIMETER. When importing line coverages, don't import <cover#>, RPOLY#, LPOLY#, FNODE, TNODE, or LENGTH.

Also, if the coverage you're importing doesn't use the <cover-ID> field to relate to another table, you shouldn't import it.

All feature class types in coverages convert to geometry types in the geodatabase. However, more than one feature class type in a coverage will convert to a single geometry type in the geodatabase. For example, points, *tics*, and nodes all convert to points. Table 1 illustrates how feature class types convert to geodatabase geometry types.

**Table 1: Coverage feature type to geodatabase geometry conversion**

This feature type	Converts to this geometry
point	point
arc	line (polyline)
polygon	polygon
node	point
tic	point
region	polygon
route	line (polyline) with measures

Annotation in the geodatabase is not a geometry type; therefore, you cannot import coverage annotation into geodatabase annotation with the Feature Class to Feature Class or Feature Class to Geodatabase tool. To learn how to import coverage annotation, see the chapter 'Managing annotation' in this book.

All attribute types in coverages and INFO tables convert to field types in the geodatabase. Coverage and INFO table items convert based on a combination of their type and their width. For example, an item of type I can map to a short, long, or double integer, depending on its width. Table 2 summarizes how items convert.

**Table 2: Coverage, INFO item to geodatabase field conversion**

This item type and width	Converts to this field type
B 4	long integer
C 1–320	text
D 8	date
F 4	float
F 8	double
I 1–4	short integer
I 5–9	long integer
I 10–16	double
N 1–9	float
N 10–16	double

## Importing shapefiles and dBASE tables

All feature types in shapefiles convert to geometry types in the geodatabase. Unlike coverages, shapefile feature types are similar to the geometry types stored in a geodatabase, so conversion is more straightforward. This is illustrated in Table 3.

**Table 3: Shapefile to geodatabase geometry conversion**

This feature type	Converts to this geometry
point	point
point M	point with measures
point Z	point with Zs
polyline	line (polyline)
polyline M	line (polyline) with measures
polyline Z	line (polyline) with Zs
polygon	polygon
polygon M	polygon with measures
polygon Z	polygon with Zs
multipoint	multipoint
multipoint M	multipoint with measures
multipoint Z	multipoint with Zs
multipatch	multipatch

Each shapefile and dBASE field type converts to a single geodatabase field type, except for the Number type field. Table 4 summarizes how shapefile and dBASE field types convert.

**Table 4: Shapefile, dBASE field to geodatabase field conversion**

This field type and width	Converts to this field type
date	date
string	text
boolean	short integer
number	short integer
number	long integer
number	double
float	float
float	double
number	float
number	double

## Importing CAD files

The Feature Class to Feature Class and Feature Class to Geodatabase tools import CAD feature classes from AutoCAD® DWG, Drawing Exchange Format (DXF), and MicroStation® DGN formats to geodatabase feature classes. The geometry converts as summarized in Table 5.

---

**Table 5: CAD to geodatabase geometry conversion**

This feature type	Converts to this geometry
point	point
polyline with Zs	line (polyline) with Zs
polygon with Zs	polygon with Zs

The properties that are inherent to CAD features are preserved in the output feature class's attribute table. These attributes include entity type, layer, color, and line type as well as complex information such as tag data, block attributes, and database linkage values.

The CAD field type to geodatabase field type conversion is summarized in Table 6.

---

**Table 6: CAD field to geodatabase field conversion**

This field type	Converts to this field type
string	text
integer	long integer
double	double

If you would like to import a CAD drawing file instead of CAD feature classes, use the Import from CAD tool in your system toolbox. The Import from CAD tool imports the drawing file into a temporary geodatabase known as a staging geodatabase. You can then customize your data conversion process by choosing what to import from the staging geodatabase.

## Importing ArcStorm and Map LIBRARIAN data

ArcMap and ArcCatalog display and query ArcStorm and Map LIBRARIAN data that is served by ArcSDE for Coverages. ArcSDE for Coverages layers are treated in the same way as ArcSDE 8 layers in that they can be displayed and queried, but they can't be edited. To import ArcStorm and Map LIBRARIAN data into a geodatabase, use the Feature Class to Feature Class or Feature Class to Geodatabase tool or copy and paste the data. To learn how to copy and paste, see 'Copying data between geodatabases', later in this chapter.

## Importing a geodatabase feature class or table

You can use the Feature class to Geodatabase or the Table to Geodatabase tool to import a feature class or table from another geodatabase. For more information on importing data from other geodatabases, see 'Copying data between geodatabases', later in this chapter.



# Importing feature classes

You use the Feature Class To Feature Class and Feature Class to Geodatabase tools to import coverages, shapefiles, and CAD files into a geodatabase. You can also import feature classes from another geodatabase.

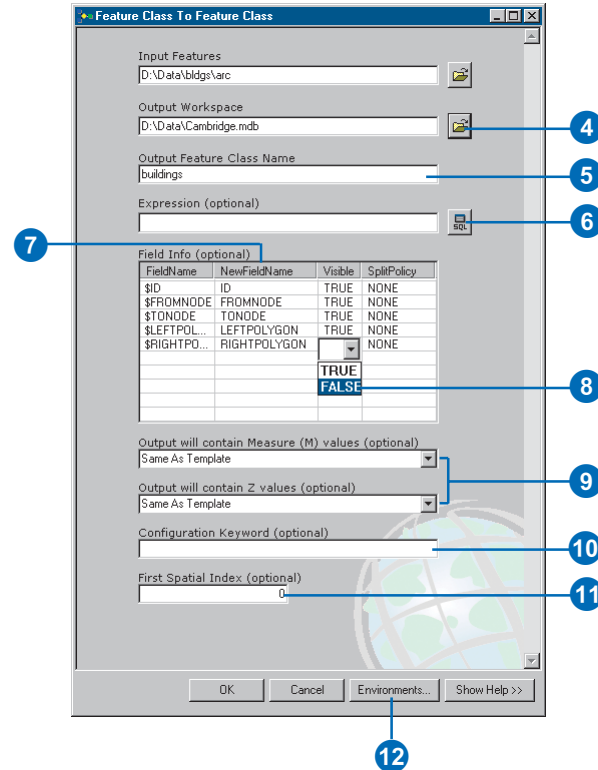
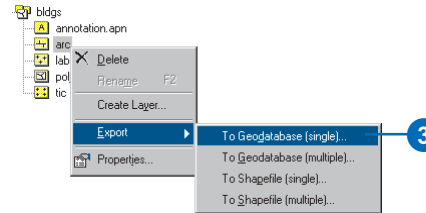
When you import several feature classes at the same time with the Feature Class to Geodatabase tool, each feature class imports into a separate feature class.

## Importing a feature class

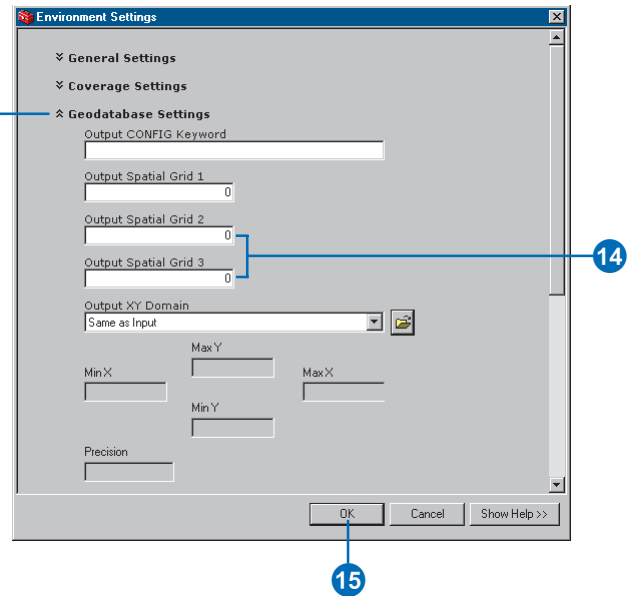
1. In the ArcCatalog tree, right-click the feature class you want to import.
2. Point to Export.
3. Click To Geodatabase (single).
4. Navigate to the geodatabase or ArcSDE connection you want to import to.

If you want to import to an existing feature dataset, navigate to the feature dataset.

5. Type a name for the new feature class.
6. If you want to create a query to limit the features you're importing, open the Query Builder dialog box and create a query.
7. Review the names in the NewFieldName column. If you do not want to use a default, click a name and type a new one.
8. If you do not want to import one of the fields, click TRUE in the Visible column and change it to FALSE.
9. Click the dropdown arrows to specify whether the new feature class will contain z- and m-values.



10. If you're importing to ArcSDE and you want to create the feature class using a custom storage class configuration keyword, type the keyword.
11. If you know an optimal spatial index grid size for your data, specify it in map units.
12. Click Environments.
13. Expand Geodatabase Settings.
14. If you're importing to an ArcSDE geodatabase and have additional grid sizes, type them in.
15. Click OK.
16. On the Feature Class To Feature Class tool, click OK to import the feature class.





## Tip

### Importing several feature classes

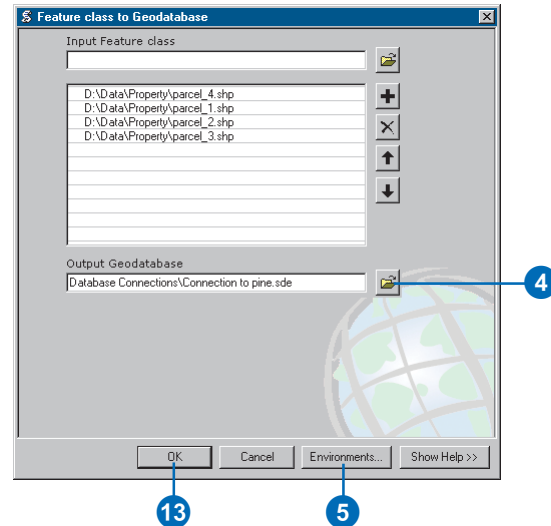
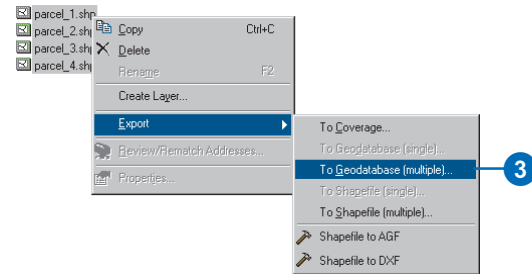
*The fields you create in the new feature classes are named the same as the fields you're importing. However, any invalid characters in the field names are automatically replaced—for example, a hyphen is replaced with an underscore.*

### Importing several feature classes

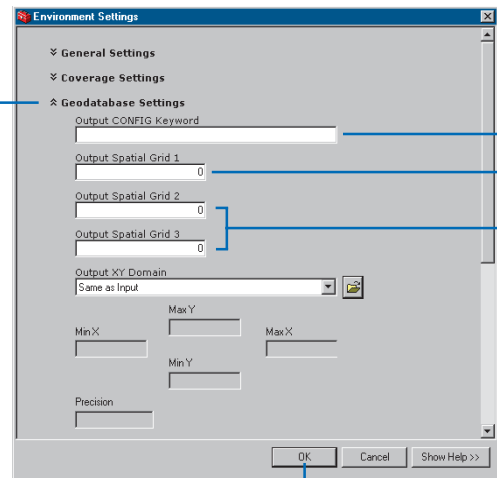
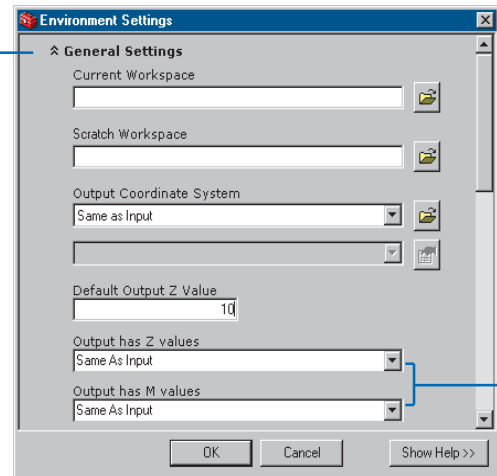
1. In the ArcCatalog contents view, select the feature classes you want to import and right-click.

Or, in the ArcCatalog tree view, right-click the geodatabase or feature dataset containing the feature classes you want to import.

2. Point to Export.
3. Click To Geodatabase (multiple).
4. Navigate to the geodatabase or ArcSDE connection you want to import to.  
  
If you want to import to an existing feature dataset, navigate to the feature dataset.
5. Click Environments. ▶



6. Expand General Settings.
7. Click the dropdown arrows to specify whether the new feature classes will contain Z and M values.
8. Expand Geodatabase Settings.
9. If you're importing to ArcSDE and you want to create the feature classes using a custom storage keyword, type the keyword.
10. If you know an optimal spatial index grid size for your data, specify it in map units.
11. If you're importing to an ArcSDE geodatabase and have additional grid sizes, type them in.
12. Click OK.
13. Click OK to import the feature classes.



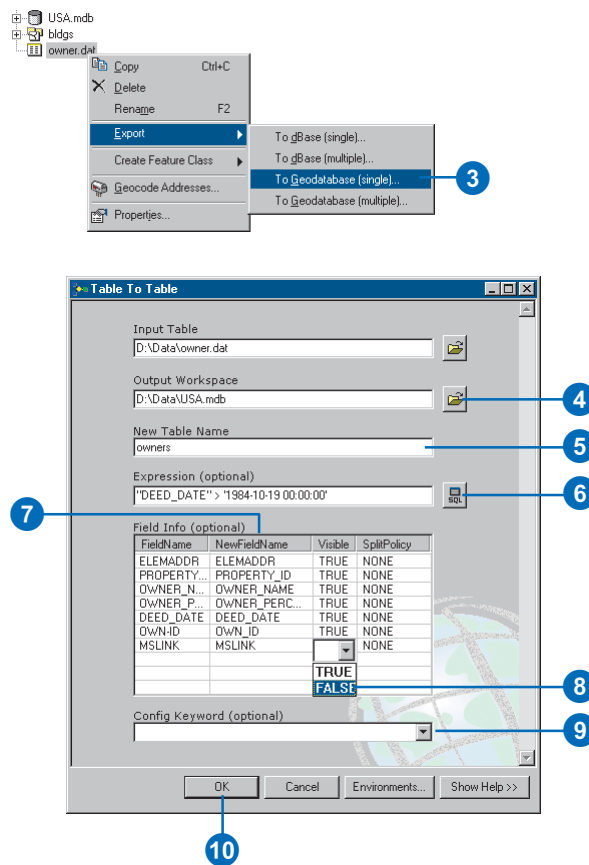
## Importing tables

You use the Table To Table and Table to Geodatabase tools to import dBASE and INFO attribute tables into a geodatabase. You can also import tables from another geodatabase.

When you import several tables at the same time with the Table to Geodatabase tool, each table imports into a new table. The tool will automatically correct any illegal or duplicate field names.

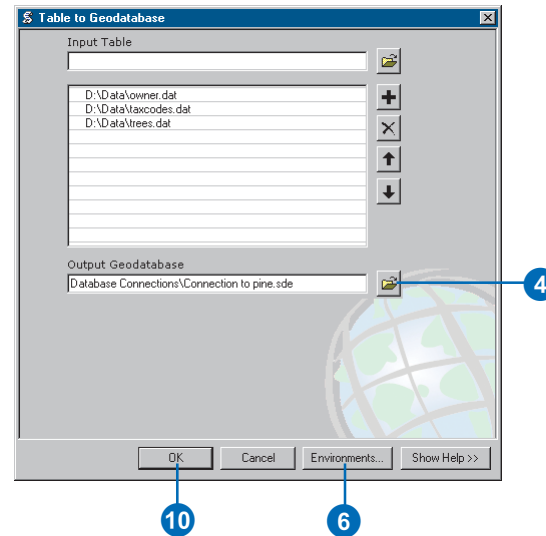
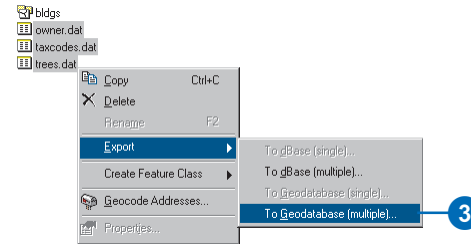
### Importing a table

1. Right-click the table in the ArcCatalog tree you want to import.
2. Point to Export.
3. Click To Geodatabase (single).
4. Navigate to the geodatabase or ArcSDE connection you want to import to.
5. Type a name for the new table.
6. If you want to create a query to limit the records you're importing, open the Query Builder dialog box and create a query.
7. Review the names in the NewFieldName column. If you do not want to use a default, click a name and type a new one.
8. If you do not want to import a field, click TRUE in the Visible column and change it to FALSE.
9. If you're importing to an ArcSDE geodatabase and want to create the table using a custom configuration keyword, choose the keyword.
10. Click OK.

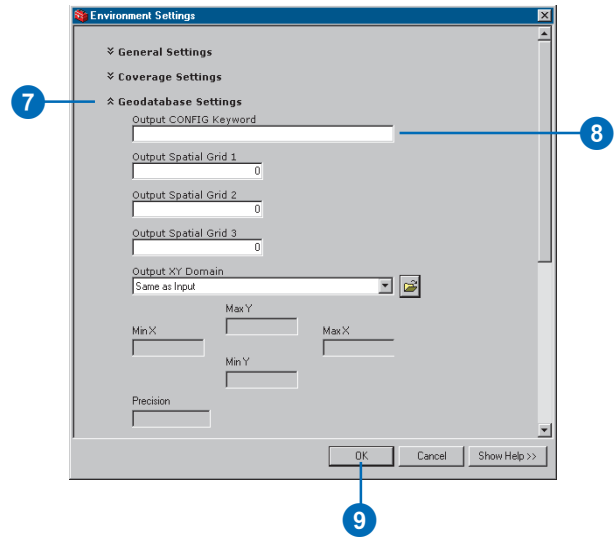


## Importing several tables

1. In the ArcCatalog contents view, select the tables you want to import and right-click.
2. Point to Export.
3. Click To Geodatabase (multiple).
4. Navigate to the geodatabase or ArcSDE connection you want to import to.
5. If you're not importing to an ArcSDE geodatabase and don't need to create the tables using a custom configuration keyword, skip to step 10.
6. Click Environments. ►



7. Expand Geodatabase Settings.
8. Type the configuration keyword.
9. Click OK to close the Environment Settings dialog box.
10. Click OK to import the tables.



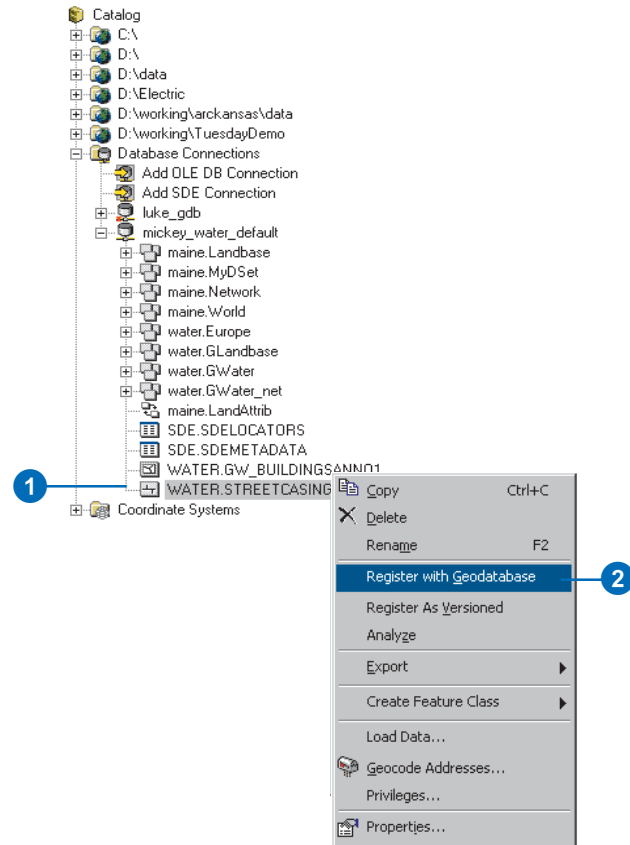
## Registering ArcSDE data with the geodatabase

To tailor the import process to your particular needs, you may have chosen to import shapefiles, coverages, CAD feature classes, or tables into an ArcSDE geodatabase with the ArcSDE C API or with an ArcSDE command such as `shp2sde`. Once imported, the ArcSDE layers and tables appear in the ArcCatalog tree as feature classes and tables. They can be displayed and queried, and if versioned, they can also be edited.

However importing this way does not register the new feature classes and tables with the geodatabase system tables. If they are to participate in relationships, geometric networks, or topology or have subtypes, default values, domains, or *validation rules*, you must register them in ArcCatalog.

Registering an ArcSDE layer or table adds an OBJECTID field to the table. This field will be called OID for tables and FID for layers. If a field called OID or FID already exists on the table or layer, then another name is chosen.

1. Right-click the table or feature class in the ArcCatalog tree you want to register.
2. Click Register with Geodatabase.



# Loading data into existing feature classes and tables

You have empty feature classes and tables, and would like to load data into them. Or, perhaps your feature classes and tables already contain data but you want to add more. You can load coverage, shapefile, CAD, or geodatabase feature class data into an existing feature class, providing it falls within the x/y domain of the feature class you're loading into. You can load INFO, dBASE, or geodatabase table data into an existing table.

You can load data with the Object Loader in ArcMap or the Simple Data Loader in ArcCatalog.

You load data with the Object Loader after you start an edit session in ArcMap. This provides the following functionality:

- If the feature coordinates you're loading are not precisely located, you can choose to honor the current snapping environment, snapping coordinates as they load.
- If you're loading into a feature class that has validation rules, such as attribute domain or geometric network connectivity rules, you can create a selection of the loaded features that are in violation of these rules.
- If you're loading into a network feature class, ArcMap builds connectivity as each feature is added.
- If you're loading into a relationship feature class or table that has messaging, ArcMap adds a record to the related table as each feature or record is added.
- If you're loading into a feature class that has feature-linked annotation, ArcMap adds a record to the linked annotation feature class as each feature is added.

Because you're loading during an edit session in ArcMap, once you've finished loading with the Object Loader, you can undo the changes if needed.

With the Object Loader, you can load into feature classes in a geometric network, feature classes in a relationship with messag-

ing, or feature classes that have feature-linked annotation. You cannot load into these types of feature classes with the Simple Data Loader.

If you don't require any of the above capabilities, you can load with the Simple Data Loader. The Simple Data Loader is faster because it doesn't validate or process data as it loads.

If you're loading into a feature class that has topology, you can load with either the Object Loader or the Simple Data Loader. However, neither tool validates topology as the features load, so the end result is the same—you will need to validate the topology yourself once you've finished loading.

## Loading into versioned feature classes and tables

If you're migrating data to a geodatabase, you should load the data before registering your data as versioned. Loading into a versioned feature class or table is slower than loading into feature classes or tables that aren't versioned. Once you've completed migrating your data and applications to the geodatabase, register your feature classes and tables as versioned. You can then load any updates into the versioned feature classes and tables.

You can load into any version with the Object Loader or Simple Data Loader. However, if you need to load into a version that is being edited by someone else, load with the Object Loader. Loading in an ArcMap edit session ensures changes will merge, and allows you to review other edits before you save the newly loaded features. It also allows you to take advantage of ArcMap's conflict resolution capabilities, should you need them.

When you load into a versioned ArcSDE feature class or table with either the Simple Data Loader or Object Loader, the data loads into the delta tables. Therefore, after you've finished loading, run Compress on your database to push all the records from the delta tables to the base tables. Having your data in the base tables will result in better query speed than if you have large

amounts of data in your delta tables. For more details on compressing your database to improve performance, see the chapter ‘Working with a versioned geodatabase’ in this book.

## Loading into network feature classes

Loading a lot of data into a network feature class with the Object Loader can take a long time, especially if the network is large and consists of several feature classes. So if you’re creating a network from scratch, you should load all of the data with the Simple Data Loader before you build the network. If you’ve already built the network, instead of using the Object Loader, you can save time by deleting the network class, loading with the Simple Data Loader, then rebuilding the network. These strategies are discussed below.

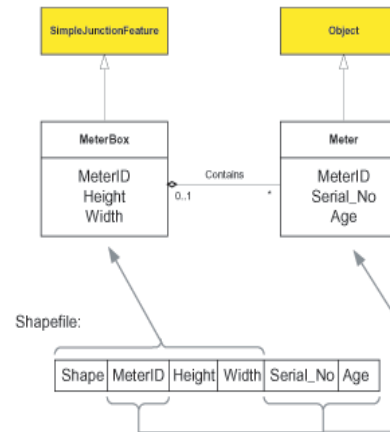
## Loading data: an example

You’ve generated a schema with CASE tools and you have a simple junction feature class called MeterBox with the attributes MeterID, Height, and Width. You also have a table called Meter with the attributes Serial\_No and Age and the foreign key MeterID, which relates the meter to its meter box. MeterBox and Meter participate in a one-to-many relationship class.

You have maintained meter boxes and meters in a single shapefile that has the attributes MeterID, Height, Width, Serial\_No, and Age. You would like to load the data from the shapefile into the MeterBox feature class and the Meter table while maintaining the relationships between the meter and its meter box.

The first step would be to use the Simple Data Loader or the Object Loader to load the geometry and the attributes MeterID, Height, and Width into the MeterBox feature class. You would then load the attributes MeterID, Serial\_No, and Age into the Meter table. Since the features in MeterBox are related to the

records in Meter by the foreign key MeterID, you maintain the relationship between the meter and its meter box.



*This example shows how you could load a shapefile into a feature class and a related table.*

## Loading into empty feature classes and tables

If you need to load a lot of data into empty feature classes and tables, you can use one of two strategies. The following steps incorporate a work flow in which UML and CASE tools were used to generate the empty feature classes and tables, but the same general work flow can be followed if you have arrived at the empty feature classes and tables in another way.

*Strategy 1—Using the Simple Data Loader:*

1. Use the Schema Generation wizard to create the empty feature classes and tables from your UML model.
2. Delete any networks that were created. This will also delete any associated connectivity rules and class extensions.



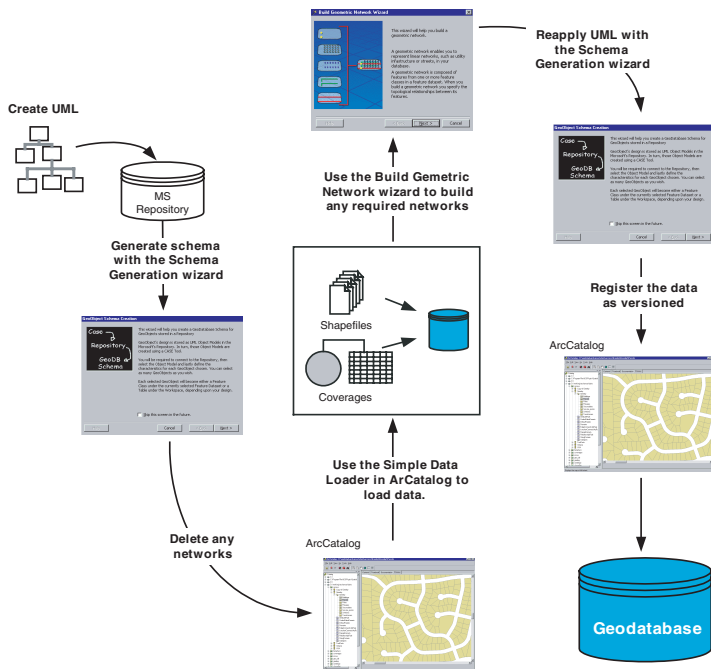
3. Load all of your data into your database using the Simple Data Loader in ArcCatalog.
4. Build any required networks using the Build Geometric Network wizard in ArcCatalog or ArcToolbox.
5. Use the Schema Generation wizard to reapply the UML to the existing data to re-create the network connectivity rules and to assign any class extensions.
6. Create and *validate* any topologies.
7. Register your data as versioned.

This strategy has a number of advantages. Without a network, your data will load much faster. Since the data is not versioned, all of the data will be loaded directly into the base tables, and you won't be required to compress your database. If your data model includes geometric networks, deleting the network in step 2 will automatically delete all connectivity rules associated with that network, and all of its participant feature classes will revert to simple feature classes. By reapplying the UML model after the network is built, your connectivity rules are reapplied and any class extensions described by the model are also reassociated with their corresponding classes.

Loading your data before creating topologies will eliminate the overhead of creating *dirty areas* for each new feature that you insert into a participating feature class. If the topology is created after the data is loaded, then a single dirty area spanning all of the features is created, which can then be validated as described in the 'Topology' chapter of this book.

This method's only limitation has to do with custom objects that have custom object creation behavior. Using this strategy, custom creation behavior will not be executed. In this case, you may want to do a combination of the first and second method (see below): load all noncustom features, build networks, apply your model from which to create the custom object classes, then version your data and use the Object Loader to populate the custom classes.

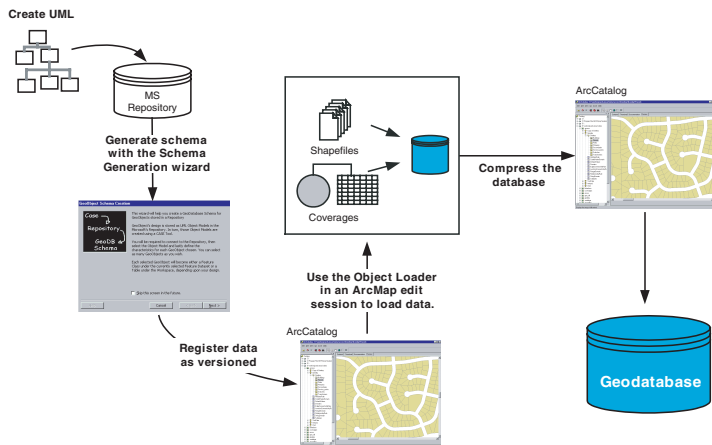
To learn more about geometric networks and connectivity rules, see the chapter 'Geometric networks' in this book. To learn more about versioning, see the chapter 'Working with a versioned geodatabase' in this book. To learn more about class extensions, see *Exploring ArcObjects*.



Loading data into an existing geodatabase schema: Strategy 1

### Strategy 2—Using the Object Loader:

1. Use the Schema Generation wizard to create the empty geodatabase schema in your database.
2. Use the Simple Data Loader in ArcCatalog to load your existing data into your simple feature classes and tables.
3. Create and validate any topologies.
4. Register your data as versioned.
5. Use the Object Loader in ArcMap to load your existing data into your network feature classes. This step automatically builds network topology within an edit session.
6. Run Compress to push all the new records from the delta tables to the base tables.
7. Use the Analyze command in ArcCatalog to update the database statistics for each feature class into which you loaded data.



Loading data into an existing geodatabase schema: Strategy 2

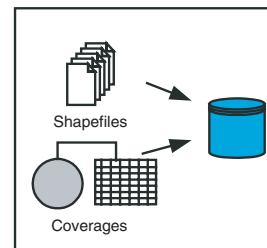
Loading data into network feature classes is a slow process that can make this method impractical for loading a large number of network features.

### Loading into feature classes and tables with data

The most straightforward process is as follows:

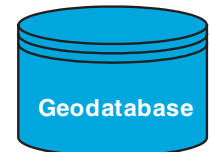
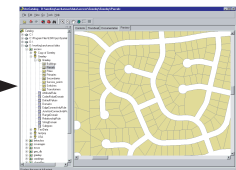
1. Use the Object Loader in ArcMap to load the new data into your feature classes.
2. Run Compress to push all the new records from the delta tables to the base tables.
3. Use the Analyze command in ArcCatalog to update the database statistics for each feature class into which you loaded data.

Use the Object Loader in an ArcMap edit session to load data.



Compress the database

ArcCatalog

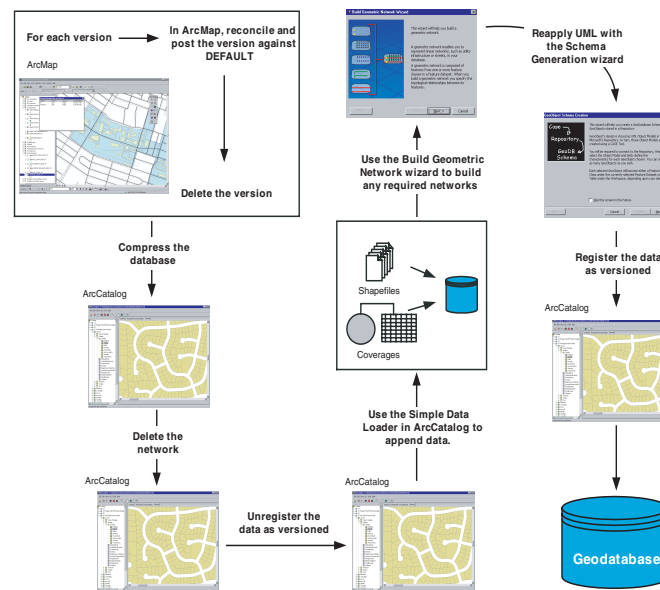


Loading data with the Object Loader

This method will work fine for simple features and features in a topology. If you're loading data into network feature classes, then the above method will work too. However, since the entire network cannot be cached when you're loading, it will be a slow process—up to several seconds per feature, depending on the number of feature classes in the network. If you are loading more than a few thousand features, this method may be too time-consuming, and you should consider the following method.

This method involves unregistering the data as versioned and dropping the network before you load the data. Before unregistering the data as versioned, make sure you compress the database first; otherwise, all edits that are not in the base table will be lost.

1. Reconcile and *post* each outstanding version in the database against the DEFAULT version. After posting, delete each version.
2. Run compress to compress the database.
3. Unregister the data as versioned. Note: If you have not completed steps 1 and 2 before unregistering your data as versioned, then you will lose any edits that those versions contain.
4. Delete the geometric network.
5. Use the Simple Data Loader in ArcCatalog to load the new data to your existing feature classes.
6. Rebuild the geometric network using the Build Geometric Network wizard in ArcCatalog or ArcToolbox.
7. If you created your geodatabase schema using CASE tools, use the Schema wizard to reapply the UML to the existing data in order to re-create the network connectivity rules and to assign any class extensions. If you are not using CASE tools, then you will need to use ArcCatalog to re-create your connectivity rules.



*Appending data to a geodatabase using the Simple Data Loader*

8. Register your data as versioned and continue with production. Registering the data as versioned automatically updates the database statistics for the feature classes.

There are a number of limitations to this method that may make it necessary for you to use the first method:

- You cannot use this method if your network has any complex junction features with connection points and custom topology since the process of batch rebuilding the network will not re-create the custom topology.
- The process of rebuilding the network will reconnect all network features you may have disconnected from the network.

- If any of the feature classes you're loading data into have feature-linked annotation, you cannot use the Simple Data Loader. In this case you must use the Object Loader.
- This method is incompatible with some work flows. If you have outstanding versions that cannot be reconciled and posted to DEFAULT, then you cannot use this method. Such versions include outstanding design versions that are not complete, are not ready for posting, or are historical versions. If this is the case, you need to use the Object Loader and append your data as part of an edit session.

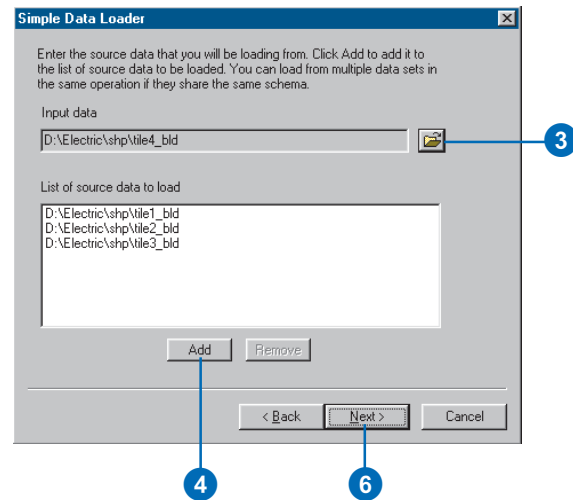
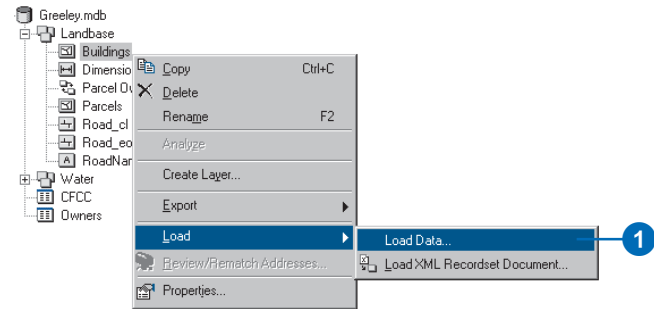
To learn more about geometric networks, complex junctions, enabled and disabled network features, and connectivity rules, see the chapter 'Geometric networks'. To learn more about reconcile, post, compress, and versioning, see the chapter 'Working with a versioned geodatabase'.

## Loading data in ArcCatalog

The Simple Data Loader wizard in ArcCatalog allows you to specify a number of source tables and feature classes, provided their schema match. It also allows you to specify which fields in the input data are loaded into which fields of the target feature class or table.

The wizard also gives you the option of loading all of the source data into a *subtype* of the target and lets you specify a query to limit the features you load.

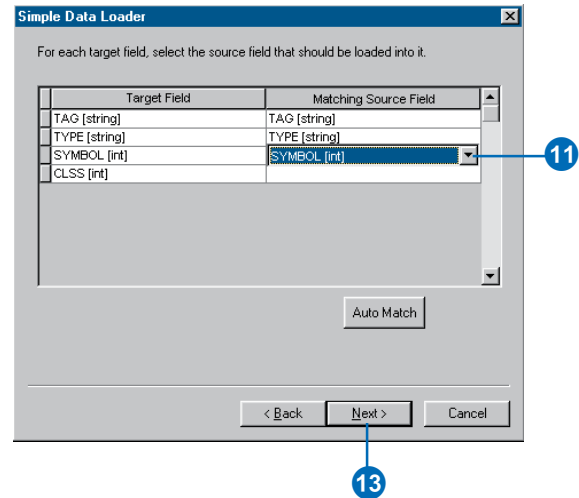
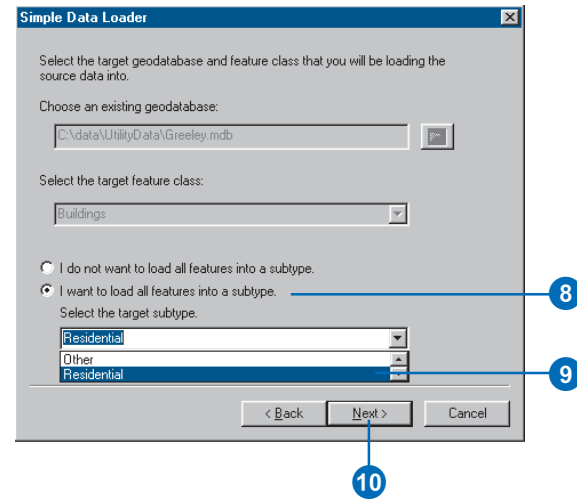
1. Right-click the table or feature class in the ArcCatalog tree that you want to load data into, point to Load, and click Load Data.
2. Click Next on the introductory panel.
3. Browse to the input feature class or table.
4. Click Add to add the table of feature classes to the list of source data.
5. Repeat steps 3 and 4 until you have specified all of the source data.
6. Click Next. ▶



7. Click the first option and skip to step 10 if you do not want to load data into a specific subtype of the target.
8. Click the second option if you want to load data into a specific subtype.
9. Click the dropdown arrow and click the subtype into which you want to load the source data.
10. Click Next.
11. Click the dropdown arrow in the Matching Source Field list and click the field from the source data that you want to match to the target field.

Leave the Matching Source Field as <None> if you don't want data from a field in the source data to be loaded into the target data.

12. Repeat step 11 until you have matched all the fields you want to load from your source data.
13. Click Next. ►



## Tip

### Source data

When matching fields, you can browse the source data's field values to help you correctly match the source and target fields.

## Tip

### Relationships

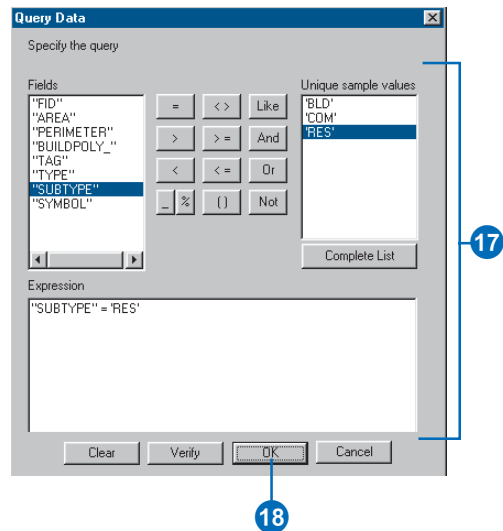
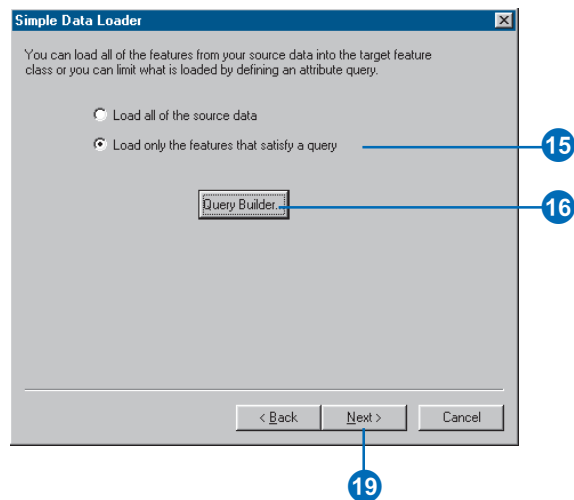
If the feature class or table you want to load data into participates in a relationship class with messaging (such as a composite relationship class), the data is considered nonsimple and the Load Data command will be unavailable.

To load data into these feature classes and tables, you can either delete the relationship or use the Object Loader.

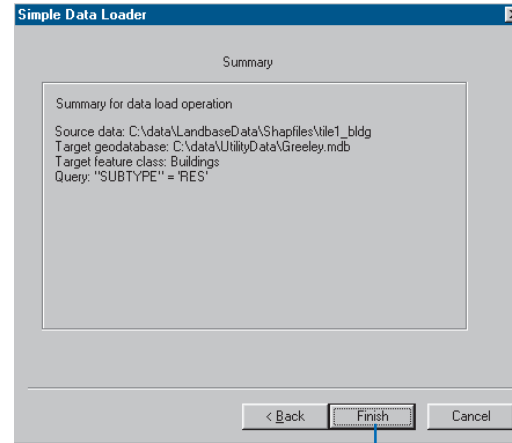
## See Also

To learn more about using the Query Builder to query your data, see Using ArcMap.

- Click the first option and skip to step 19 if you want to load all of the source data.
- Click the second option if you want to limit the features from the source data loaded into the target using an attribute query.
- Click Query Builder to open the query builder dialog box.
- Use the query builder to create a query to limit the features or rows from the source data that are going to be loaded into the target.
- Click OK.
- Click Next. ▶



20. Review the options you have specified for loading your data. If you want to change something, you can go back through the wizard by clicking Back.
21. Click Finish to load your data when satisfied with your options.





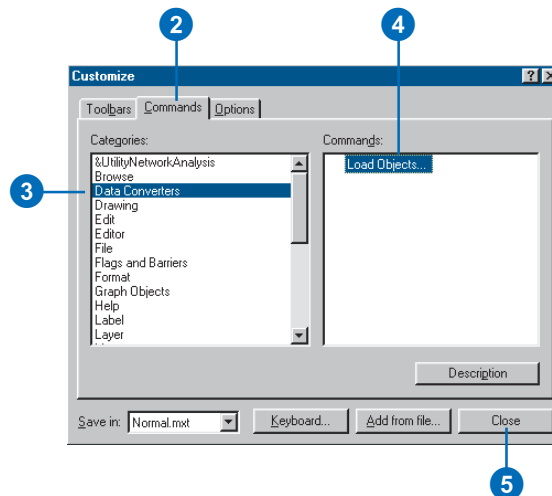
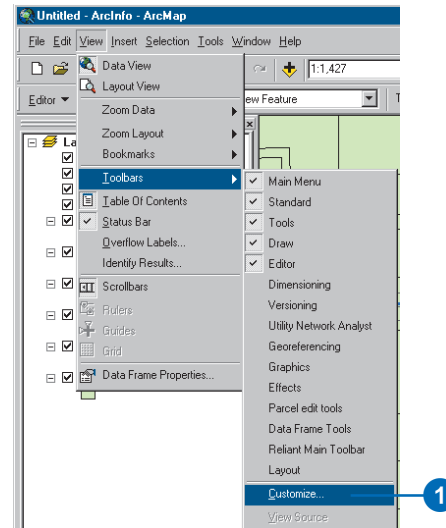
## Loading data in ArcMap

The Object Loader wizard in ArcMap allows you to specify a number of source tables and feature classes, provided their schema match. It also allows you to specify which fields in the input data are loaded into which fields of the target feature class or table.

The wizard also lets you specify a query to limit the features loaded.

## Adding the Load Objects command to ArcMap

1. Click View, point to Toolbars, and click Customize.
  2. Click the Commands tab.
  3. Click Data Converters.
  4. Click and drag the Load Objects command from the Commands list and drop it on the Editor toolbar.
- The command appears on the toolbar.
5. Click Close.



The command appears on the toolbar.

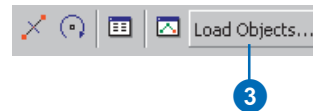
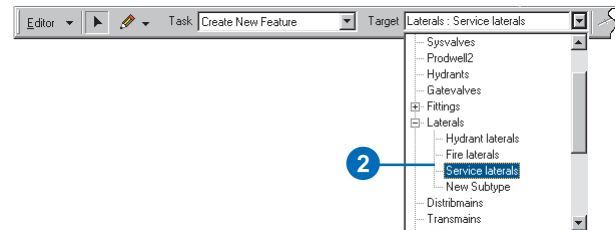
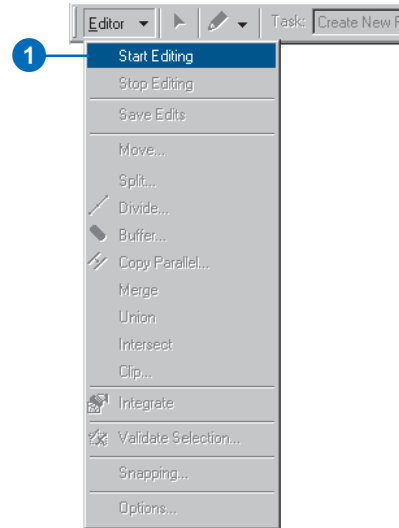
## Tip

### Versioned data

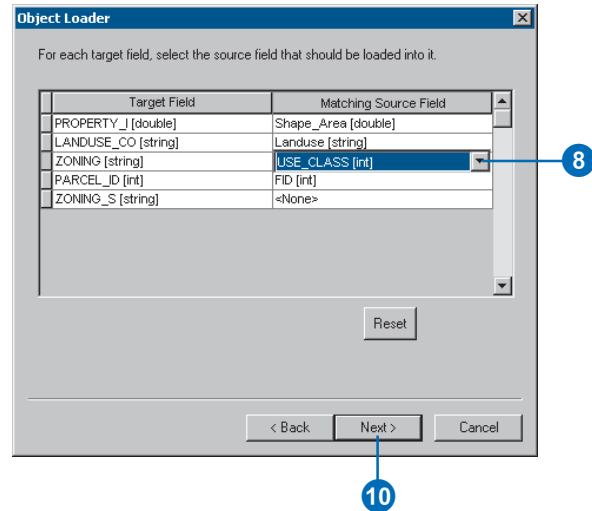
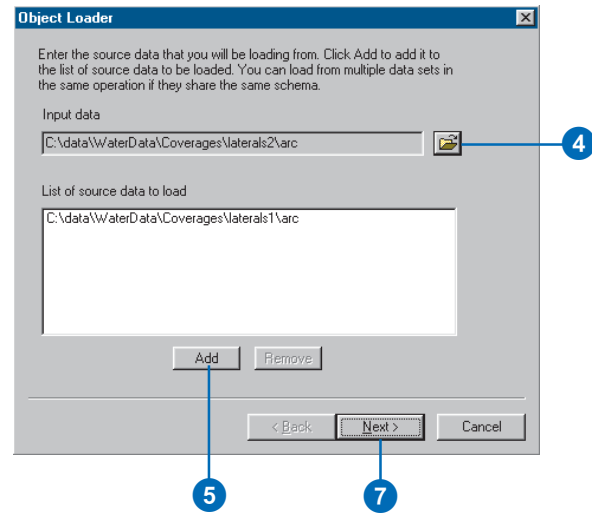
When you load data into a versioned feature class, the new features are only visible in the version you are working with.

## Loading data with the Load Objects command

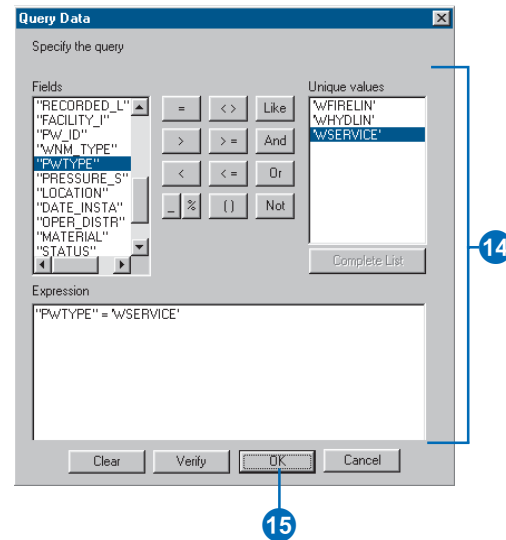
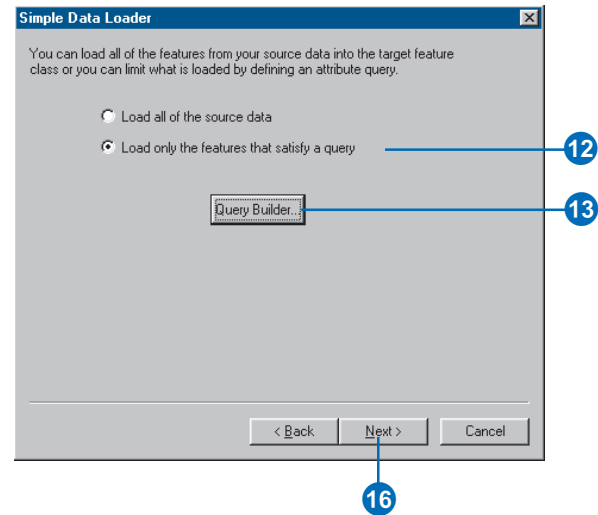
1. Add your data to ArcMap, click Editor, then click Start Editing.
2. Click the Target layer dropdown and click the feature class or subtype into which you want to load data.
3. Click Load Objects. ►



4. Browse to the source feature class.
5. Click Add to add it to the list of source data.
6. Repeat steps 4 and 5 until you have specified all of the source data.
7. Click Next.
8. Click the dropdown arrow in the Matching Source Field list and click the field from the source data you want to match to the target field.  
Leave the Matching Source Field as <None> if you don't want data from a field in the source data to be loaded into the target data.
9. Repeat step 8 until you have matched the fields you want loaded from your source data.
10. Click Next. ►



11. Click the first option and skip to step 16 if you want to load all of the source data.
12. Click the second option if you want to limit the features from the source data to load into the target using an attribute query.
13. Click Query Builder to open the Query Data dialog box.
14. Create a query to limit the features or rows from the source data to be loaded into the target.
15. Click OK.
16. Click Next. ►



## Tip

### Network features

When loading data into an edge network feature class, the network connectivity is maintained, and default junctions are used as described earlier in this chapter.

## Tip

### Feature class validation

The option to validate the new features applies only to the geodatabase validation rules and does not validate the topology. For more information on topologies, see the chapter 'Topology' in this book.

## See Also

For more information on the ArcMap snapping environment, see Editing in ArcMap.

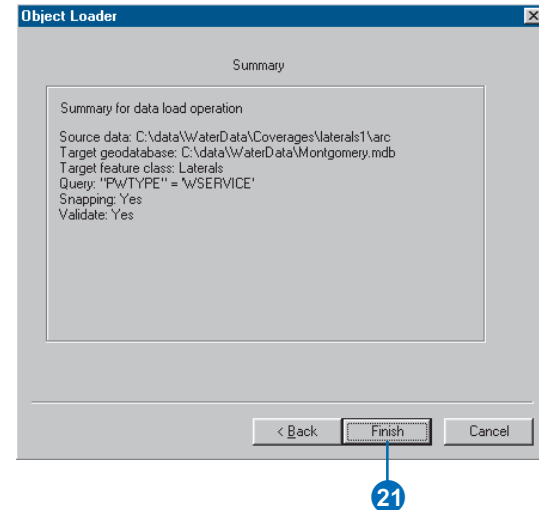
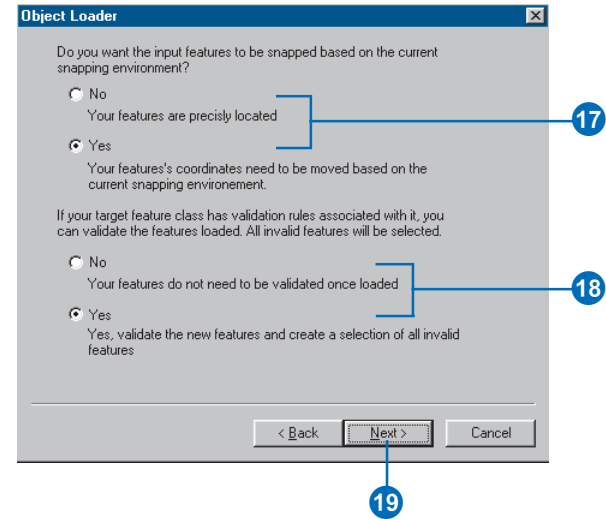
- Click No if you don't want your features to be snapped to existing features in your edit session.

Click Yes if you want to use the current Editor snapping environment to snap the new features as they are loaded.

- Click No if you don't want your new features to be validated after they are loaded.

Click Yes if the feature class or subtype into which you are loading data has rules associated with it and you want any new invalid features to be selected after the loading process.

- Click Next.
- Review the options you have specified for loading your data. If you want to change something, go back through the wizard by clicking Back.
- Click Finish to load your data when satisfied with your options.



# Copying data between geodatabases

This chapter has shown you how to use the Feature Class to Geodatabase and Table to Geodatabase tools as well as the Load Data and Load Object commands. While these tools and commands work with data in a variety of formats, they also allow you to import or load data from a geodatabase feature class or table. The end result is like copying data from one geodatabase feature class or table to another.

ArcCatalog and ArcMap contain additional tools that copy data between geodatabases. As with the above tools, these tools copy data between personal geodatabases, between ArcSDE geodatabases, or between personal and ArcSDE geodatabases.

Because these tools copy data into new feature classes and tables you create during the copying process, you must use the Load Data or Load Objects command to load data into an existing feature class or table.

## Copying feature datasets, classes, and tables

ArcCatalog allows you to copy data from the tree view or contents view and paste it to another location. You can copy entire feature datasets or individual feature classes and tables. For every feature dataset, feature class, and table you copy and paste, the result is a new feature dataset, feature class, and table in the destination geodatabase with all the features or records from the source data. If you're copying into an ArcSDE geodatabase, you can specify a configuration keyword to control how the new feature classes and tables are stored.

When you copy and paste, you copy any dependent data as well. Therefore if you copy a geometric network or topology class, then all the feature classes in the network or topology are also copied. If you copy a feature class or table in a relationship, the relationship class, along with the feature class or table it relates to, is also copied. The same goes for a feature class that has feature-linked annotation; the feature-linked annotation is copied

too. For a feature class that has a domain, subtype, or index, the domain, subtype, or index also copies.

If you are copying a feature class into an existing feature dataset, either in the same geodatabase or in another geodatabase, the spatial reference of the feature class and feature dataset must match. If they don't, you will not be able to copy the data.

## Copying selected features and records

Suppose you don't want to copy all of the features or records from a feature class or table, but only a selection. In ArcMap you can select features or records using any selection method, such as selecting features by dragging a box around them or by specifying an attribute query. You can then export them to a new feature class or table using either the Extract Data wizard or the Export Data command.

If you want to export dependent data with the features or records, like Copy and Paste does, use the Extract Data wizard. Because it can export data from any layer or table in the ArcMap document, it also allows you to export from several feature classes at the same time.

If you are exporting to an ArcSDE geodatabase, the Extract Data wizard doesn't allow you to specify a configuration keyword. One way to get around this would be to use Extract Data, then copy and paste the new data to specify one or more configuration keywords.

If you want to export selected features or records from a single feature class or table, you can use the Export Data command. Unlike Copy and Paste, it exports attributes and records only, without any dependent data. For example, if you export features from a feature class that uses a domain or has linked annotation, the domain or annotation doesn't export with the features. Export Data does preserve field properties though, such as the alias, whether to allow null values, and the default value.

If you are exporting to an ArcSDE geodatabase, the Export Data command doesn't allow you to specify a configuration keyword. So if you want to control how the new feature class or table is created and stored, use the Feature Class to Feature Class or Table to Table tool. They copy data the same way Export Data does, only they apply the configuration keyword you specify in the geoprocessing environment. For more information on these tools, see the tasks earlier in this chapter.

## **Sending data to another user**

Suppose you have a small amount of data, less than a few hundred thousand features, you want to send to another user. One option is to copy the data into a personal geodatabase and either send it to the other user or have it transferred to the other user's computer. However if you're working with enough data, the geodatabase can be a large file to send or transfer.

Another option is to export all or any part of your geodatabase to a smaller ZIP file. You can then send or transfer the ZIP file to another user, who can import the data into a different geodatabase to create new feature datasets, feature classes, or tables.

This chapter shows you how to export feature datasets, feature classes, and tables to a ZIP file. When you export entire feature classes and tables, you export any dependent data as well. Once the data has been imported into another geodatabase, the end result is like copying and pasting data from one geodatabase to another.

If instead you want to send features or records to someone who wants to import them into an existing feature class or table, you can export the features or records to a ZIP file. However, unlike when you export entire feature classes and tables, any dependent data doesn't export. So once the data has been sent to someone and loaded into an existing feature class or table, the end result is

like using the Load Data command in ArcCatalog or the Load Objects command in ArcMap. For more information on exporting features and records, please see the ArcGIS online Help system.

## Tip

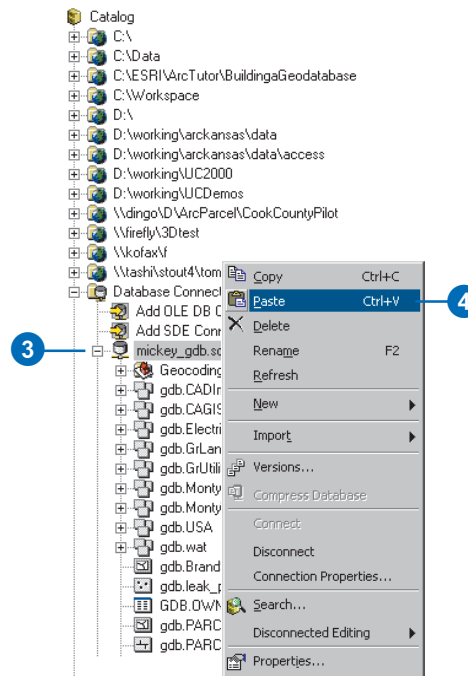
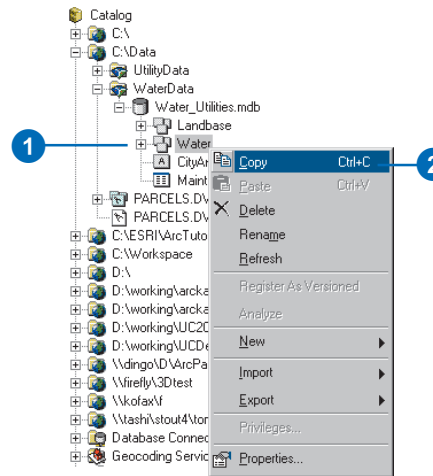
### Copying a geometric network or topology class

To copy a geometric network or a topology class and all the participating feature classes, copy and paste the network or topology class only. This will copy all the participating feature classes along with it. You cannot copy and paste individual feature classes participating in a network or topology.

### Copying feature datasets, classes, and tables

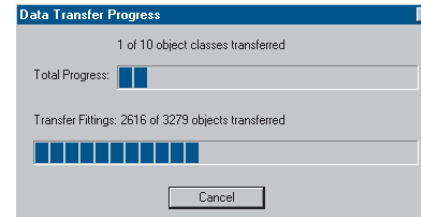
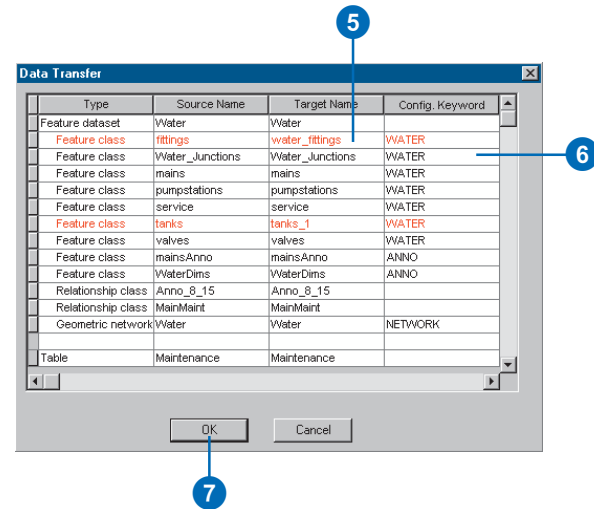
1. In the ArcCatalog tree, right-click the feature dataset, feature class, or table you want to copy.
2. Click Copy.
3. Right-click the geodatabase you want to copy the data to.
4. Click Paste.

A dialog box appears that indicates what data is being copied. Any name conflicts are automatically resolved and highlighted in red. ►





5. Type over the target name to change any of the resolved names.
6. If you're copying into an ArcSDE geodatabase and want to control how the new feature classes and tables are created and stored, click a keyword and choose a new one from the dropdown list.
7. Click OK to copy the data into new feature classes and tables.



*A progress indicator will appear to indicate the progress of the data copy.*

## Exporting selected features with the Extract Data wizard

1. In ArcMap on the main menu, click View, point to Toolbars, then choose Disconnected Editing.

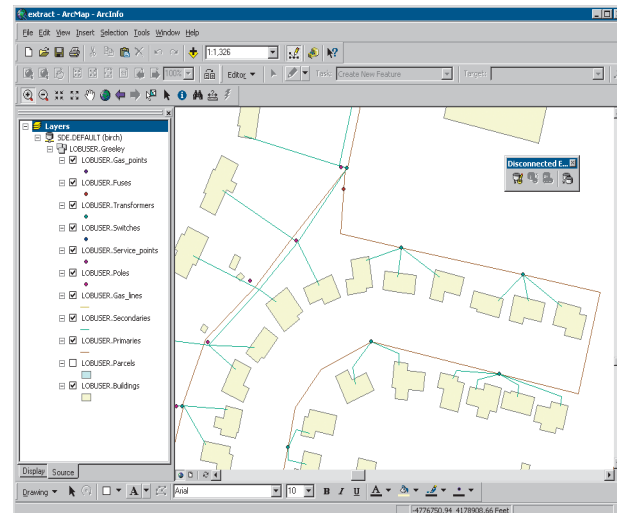
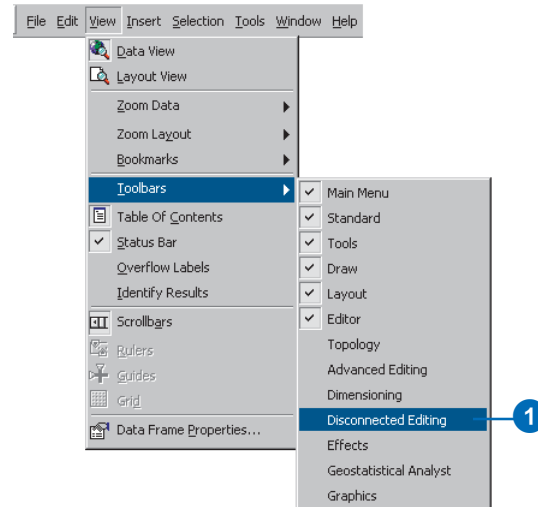
The Disconnected Editing toolbar will now appear on your map document.

2. Specify the features you want to export by setting the current view extent to include them, drawing a graphic around them, selecting the features, or applying a definition query.

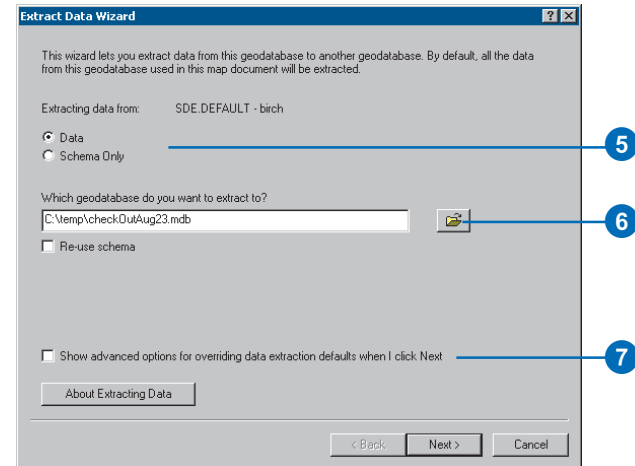
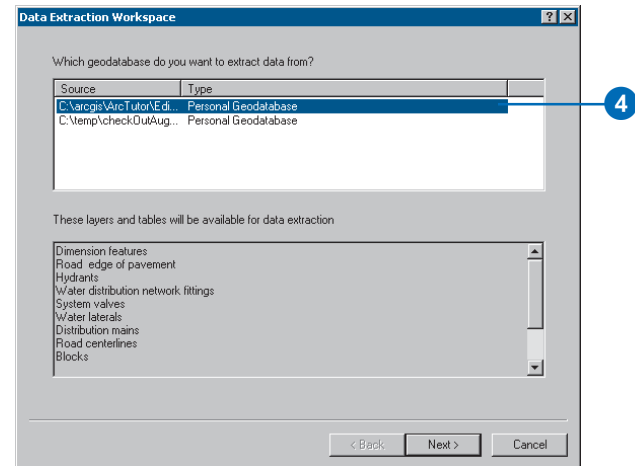
Or, you can specify the bounding coordinate values later in the wizard.

If you use more than one of these methods, the features in common will export.

3. Click the Extract Data command. ▶

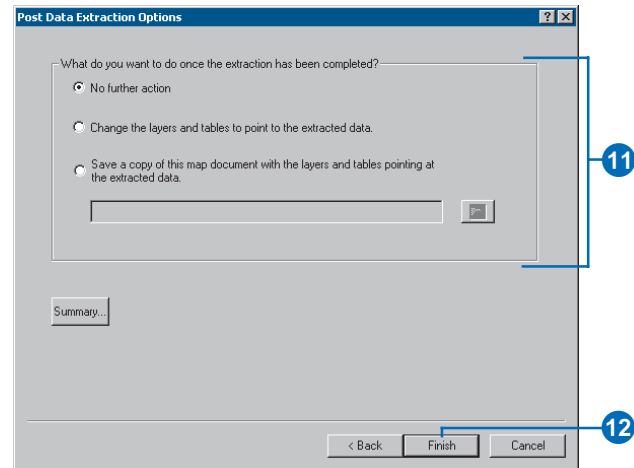
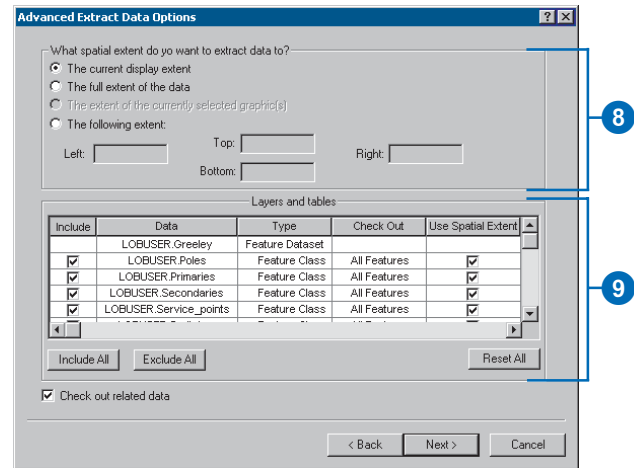


4. If in step 2 you specified features from more than one geodatabase, click the geodatabase containing the data you want to export and click Next.
5. Click Data.
6. Specify the destination geodatabase or type the path and name of a new geodatabase you wish to create.
7. Check the Show advanced options box and click Next. To accept the defaults, leave the check box empty, click Next, and proceed to step 11. ►



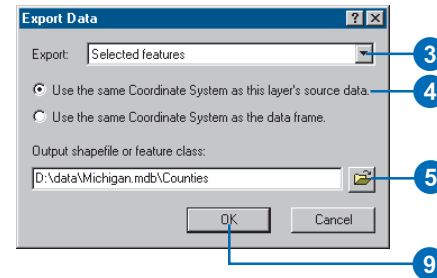
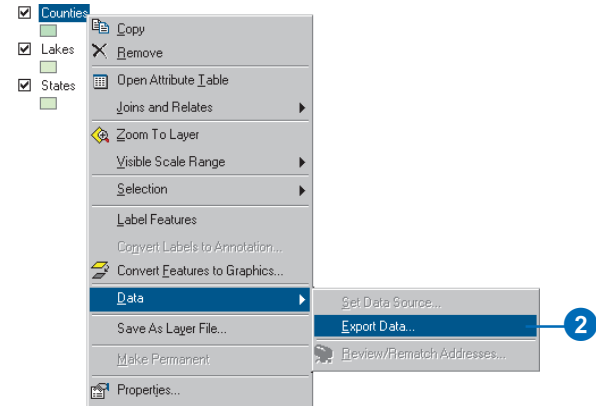
8. Specify the method you chose in step 2.
9. The list of data to export expands to include all the data in any feature dataset and any related data. For example, if you selected features from just one feature class in a feature dataset, all feature classes in the feature dataset list. Uncheck a box if there is a feature class or table you don't want to export. You can also override selections, queries, and spatial extents.

10. Click Next.
11. Choose a post data extraction option.
12. Click Finish.



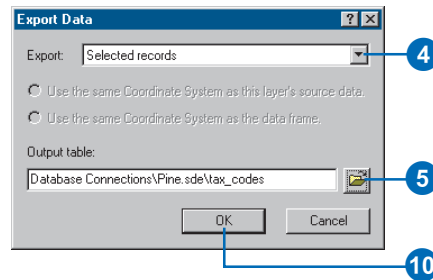
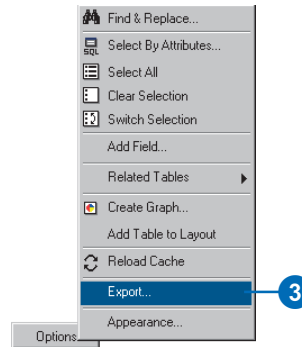
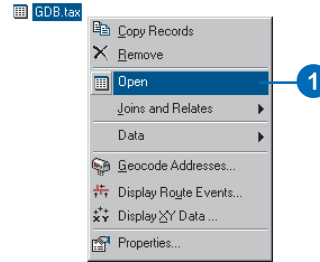
## Exporting selected features with the Export Data command

1. Select features using any selection method.
2. Right-click the layer that contains the selected features, point to Data, and click Export Data.
3. Click the Export dropdown arrow and click Selected features.
4. Click Use the same coordinate system as this layer's source data.
5. Click the Browse button and navigate to the existing geodatabase you will export to.
6. Type the name for the new feature class you will create.
7. Click the Save as type dropdown arrow and choose the output type.
8. Click Save.
9. Click OK.



## Exporting selected records from a table

1. In the ArcMap table of contents, right-click a table and click Open.
2. Select records using any selection method.
3. Click Options and click Export.
4. Click the Export dropdown arrow and click Selected Records.
5. Click the Browse button.
6. Click the Save as type dropdown arrow and click Personal Geodatabase or SDE tables.
7. Navigate to the existing geodatabase you want to export to.
8. Type a name for the new table you will create.
9. Click Save.
10. Click OK.



## Sending data to another user

These steps show you how to export a geodatabase, feature dataset, feature class, or table to a file and how to import them into another geodatabase.

As the geodatabase export format is comprised of text, exporting even moderate amounts of data can result in a large file. For this reason when you export, you should always compress the data by choosing to export to a ZIP file. This allows you to save space and transfer the data more easily. You or another user can then import the data directly from the ZIP file.

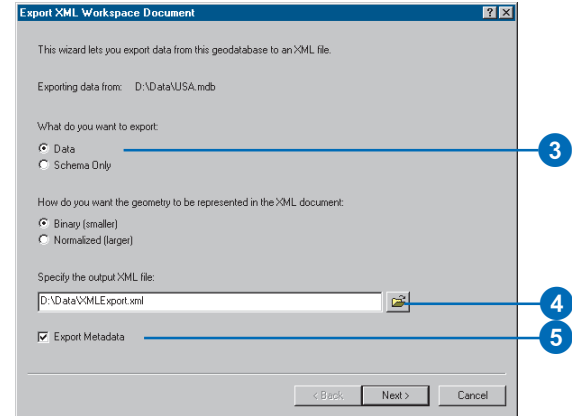
## Exporting feature datasets, classes, and tables

1. In the ArcCatalog tree, right-click the geodatabase, feature dataset, feature class, or table you want to export and point to Export.
2. Click XML Workspace Document.
3. Click Data.
4. Specify the path and name of the new ZIP file you will create.

If you specify the path and name by typing it into the text box, give the file a .zip extension.

If you specify the path and name by browsing to a folder with the Save As dialog box, save the file as a ZIP file.

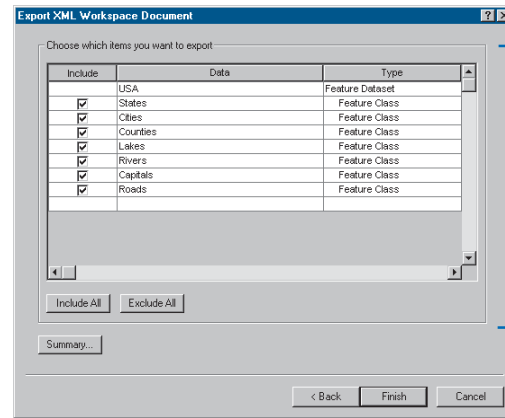
5. If the data you're exporting has metadata you want to export, check the Export Metadata check box.
6. Click Next. ►



- The list of data to export expands to include any dependent data. For example, if you had right-clicked just one of the feature classes participating in a network in step 1, all feature classes in the network would be listed.

Uncheck a check box if there is a feature class, table, or relationship class you don't want to export. If you leave a box checked for a feature class in a network or topology, all the feature classes in the network and topology will export.

- Click Finish to create the ZIP file.





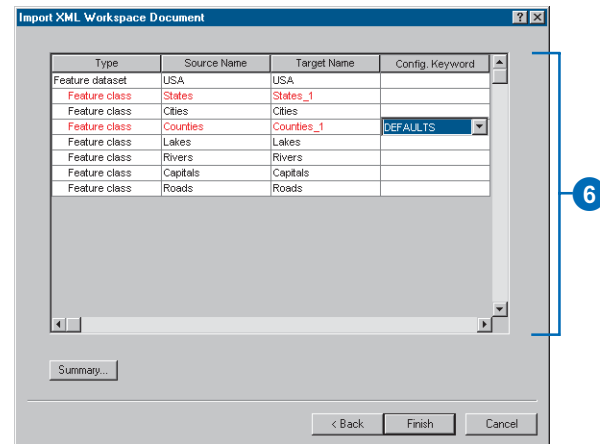
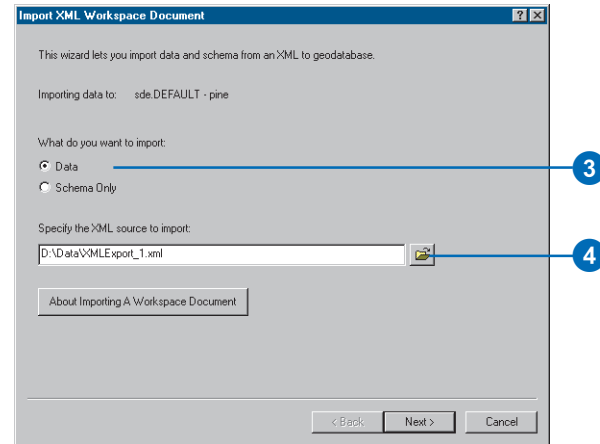
## Importing feature datasets, classes, and tables

1. In the ArcCatalog tree, right-click the geodatabase you want to import into and point to Import.
2. Click XML Workspace Document.
3. Click Data.
4. Navigate to the ZIP file you or someone else created by following the steps in 'Exporting feature datasets, classes, and tables'.
5. Click Next.

6. Any naming conflicts display in red. To change a suggested target name, type over it.

If you're importing into an ArcSDE geodatabase and want to use a configuration keyword, select one from the dropdown list.

8. Click Finish to import the data.

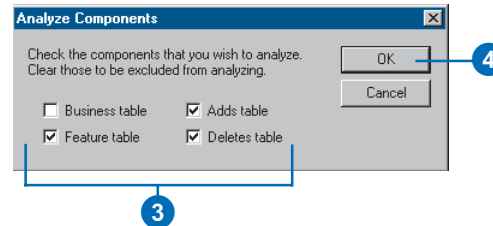
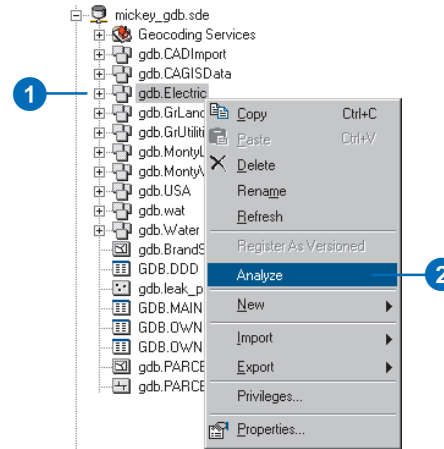


## Updating DBMS statistics

Once you've finished importing, loading, or copying data into an ArcSDE geodatabase, use the Analyze command in ArcCatalog to help maximize query performance. The Analyze command updates the statistics of business tables, feature tables, and delta tables along with the statistics on those tables' indexes.

When you update the statistics for a feature dataset, statistics for all the feature classes in that feature dataset update. If the feature dataset contains a geometric network, then the network tables also update. For details on which statistics the Analyze command updates, see the ArcSDE online documentation.

1. Right-click the dataset in the ArcCatalog tree that you want to update statistics for. This can be a feature dataset, feature class, or table.
2. Click Analyze.
3. Check those tables you want analyzed.
4. Click OK.



# Topology

# 4

## IN THIS CHAPTER

- **What is topology?**
- **Creating a topology**
- **Topology and feature geometry**
- **Adding new feature classes to your topology**
- **Topology: defining the rules**
- **Refining topologies with subtypes**
- **Managing and modifying a topology**
- **Creating polygons from lines**
- **Topology and versioning**
- **Topology and disconnected editing**

When you model geographic features, you may find that you want to model some features that have spatial relationships with other features around them. Countries might be modeled such that adjacent countries meet without gaps along a common border line but never overlap. States or provinces could be modeled such that they fall exclusively within one country. Streets may be modeled such that two streets always meet at an *intersection* and never share a *segment*. Bus stops may be modeled such that they must always occur on a street. These relationships are maintained in the geodatabase through an association called a topology.

In GIS technology, topology is the model used to describe how features share geometry, and it is also the mechanism for establishing and maintaining topological relationships between features. ArcGIS implements topology through a set of validation rules that define how features may share a geographic space and a set of editing tools that work with features that share geometry in an integrated fashion.

A topology is stored in a geodatabase as one or more relationships that define how the features in one or more feature classes share geometry. The features participating in a topology are still simple feature classes—rather than modifying the definition of the feature class, a topology serves as a description of how the features can be spatially related. ArcInfo and ArcEditor provide tools for creating, evaluating, and managing the quality of those topological relationships. This chapter describes the physical geodatabase data model for topology, how you create and configure a topology, and the editing tools in ArcGIS that help you to maintain the quality

of the *integrated features* in a topology. Another type of *topological association*, the geometric network, is discussed in the chapter ‘Geometric networks’ in this book.

You can create simple temporary topological relationships between features in ArcView. Creating or editing geodatabase topology requires an ArcEditor or ArcInfo license.

# What is topology?

Topology has historically been viewed as a spatial data structure that is used primarily to ensure that the associated data forms a consistent and clean topological fabric. With advances in object-oriented GIS development, an alternative view of topology has evolved. The geodatabase supports an approach to modeling geography that integrates the behavior of different feature types and supports different types of key relationships. In this context, topology is a collection of rules and relationships that, coupled with a set of editing tools and techniques, enables the geodatabase to more accurately model geometric relationships found in the world.

Topology, implemented as feature behavior and rules, allows a more flexible set of geometric relationships to be modeled than topology implemented as a data structure. It also allows topological relationships to exist between more discrete types of features within a feature dataset. In this alternative view, topology may still be employed to ensure that the data forms a clean and consistent topological fabric, but also more broadly, it is used to ensure that the features obey the key geometric rules defined for their role in the database.

## Why use topology?

Topology is used most fundamentally to ensure data quality and to allow your geodatabase to more realistically represent geographic features. A geodatabase provides a framework within which features can have behavior, such as subtypes, default values, attribute domains, validation rules, and structured relationships, to tables or other features. This behavior enables you to more accurately model the world and maintain referential integrity between objects in the geodatabase. Topology may be considered an extension of this framework for behavior that allows you to control the geometric relationships between features and to maintain their geometric integrity. Unlike other

feature behavior, topology rules are managed at the level of the topology and dataset, not for individual feature classes.

## How do I work with topology?

Different people work with topology in different ways, depending on their role in an organization and its GIS design and management work flow.

Initially, creating a topology requires a geodatabase designer. A topology organizes the spatial relationships between features in a set of feature classes. The designer analyzes an organization's data modeling needs, identifies the key topological relationships required in the geodatabase, and defines the rules that will constrain different features' topological relationships.

Once the participating feature classes have been added to the topology and the rules defined, the topology is validated. Data quality managers use the topology tools to analyze, visualize, report, and where necessary, repair the spatial integrity of the database after it is initially created as well as after editing. Topology provides these users with a set of validation rules for the topologically related features. It also provides a set of editing tools that lets users find and fix integrity violations.

As the geodatabase is used and maintained, new features are added and existing features are modified. Data editors update features in the geodatabase and use the topology tools to construct and maintain relationships between features within the *constraints* imposed by the database designer. Depending on the work flow of the organization, the topology may be validated after each edit session or on a schedule.

# Creating a topology

A topology consists of a set of rules structuring the relationship between a collection of features in one or more feature classes in a feature dataset.

To create a topology, you must specify which feature classes will participate in the topology and what rules will govern the interaction of features. All feature classes participating in a topology must be in the same feature dataset.

Because creating topological relationships involves *snapping* feature vertices together to make them *coincident*, a *cluster tolerance* must be specified for the topology. Vertices within the cluster tolerance may move slightly in the snapping process. The default cluster tolerance is the minimum possible cluster tolerance and is based on the precision defined for the dataset. The cluster tolerance should be small, so only close vertices are snapped together. A typical cluster tolerance is at least an order of magnitude smaller than the accuracy of your data. For example, if your features are accurate to 2 meters, your cluster tolerance should be no more than 0.2 meters.

Often, you will want to be able to control which feature classes are more likely to be moved in the *clustering* process. For example, when features in one feature class are known to have more reliable positions than another set of features, you may want the less reliable features to snap to the more reliable ones. *Ranks* are assigned to the feature classes in the topology to accommodate this common situation. Vertices of lower ranking features within the cluster tolerance will be snapped to nearby vertices of higher ranking features. Vertices of features of equal rank that lie within the cluster tolerance will be geometrically averaged together.

## Building a topology

You may already have data from which you want to create a topology in your geodatabase. ArcCatalog contains tools to create a topology from that data.

The process of building a topology from existing data can be summarized in the following steps:

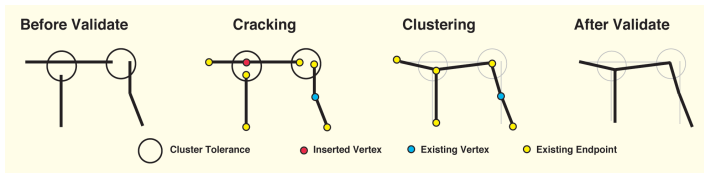
1. Use ArcCatalog or ArcToolbox tools to convert and load your data into a feature dataset in a geodatabase.
2. Use the ArcCatalog Topology wizard to build a topology from existing simple feature classes. In the wizard, you do steps 3–9.
3. Name the topology.
4. Set a *cluster tolerance* for the topology.
5. Choose the feature classes that will participate in the topology.
6. Choose the number of ranks to use in the topology.
7. Rank the feature classes in the topology.
8. Add topology rules to structure the spatial relationships between features.
9. Create the topology.
10. Use ArcCatalog or ArcMap to validate the topology.
11. Use the ArcMap Topology Error Inspector to identify topology errors.
12. Use ArcMap to fix topology errors or mark them as exceptions.

## How topologies are built

Building topologies from existing features is a computationally intense operation that may take a considerable amount of time and system resources, depending on the number of input features.

If those features require snapping, the validating operation will spend most of its time in the clustering (feature snapping) phase. The validating process proceeds in the following sequence:

1. *Cracking* features
2. Clustering vertices



*Validating topology passes through two stages: cracking and clustering. In the cracking stage, vertices are created along edges that fall within the cluster tolerance of an existing edge, vertex, or endpoint. In the clustering stage, the endpoints and vertices that fall within the cluster tolerance are snapped together.*

When a vertex of one feature in the topology is within the cluster tolerance of an *edge* of any other feature in the topology, the topology engine creates a new vertex on the edge to allow the features to be geometrically integrated in the clustering process.

When clustering, or snapping, features during topology validation, it is important to understand how the geometry of

features is adjusted. All vertices of any feature in a feature class that participates in a topology can potentially be moved, if they fall within the cluster tolerance of another vertex. Vertices of higher ranking features will not move to lower ranking features, but vertices of equally ranked features will be geometrically averaged.

## Schema locking

An exclusive lock is required on all of the input feature classes when building a topology. If any of the input feature classes has a shared lock, the topology will not be built.

If any of the feature classes in a topology have a shared or exclusive lock, that lock is propagated to all of the other feature classes in the topology. For more information on exclusive locks and schema locking, see the ‘Creating new items in a geodatabase’ chapter in this book.

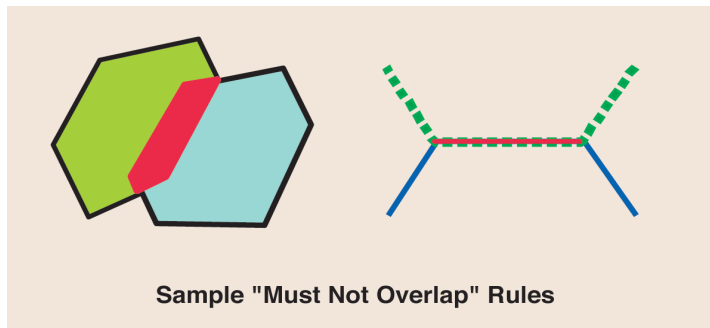


# Topology basics

Topologies store several sets of parameters—rules, ranks, and cluster tolerances. They also maintain internal feature layers that contain dirty areas, errors, and exceptions. These parameters and special features in a topology are discussed in more detail in the next few sections.

## Rules

The rules you define for a topology control the allowable relationships of features within a feature class, between features in different feature classes, or between subtypes of features.



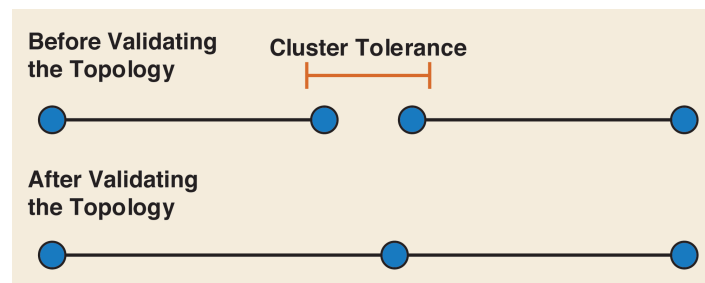
*Example of a "Must not overlap" rule applied to polygons and lines. The red polygon and line mark the places where the rule is violated. These are stored in the topology as error features. Such rules can apply to features within the same feature class, to pairs of feature classes, or to subtypes of features.*

The initial validation of the topology checks all the features against all the rules. This initial check can take some time, but subsequent checks are performed only on the areas that have been edited—the dirty areas.

## The x,y cluster tolerance and ranks

The x,y cluster tolerance is the minimum horizontal distance between vertices of features that are not coincident. Vertices that fall within the cluster tolerance are defined as coincident and snapped together.

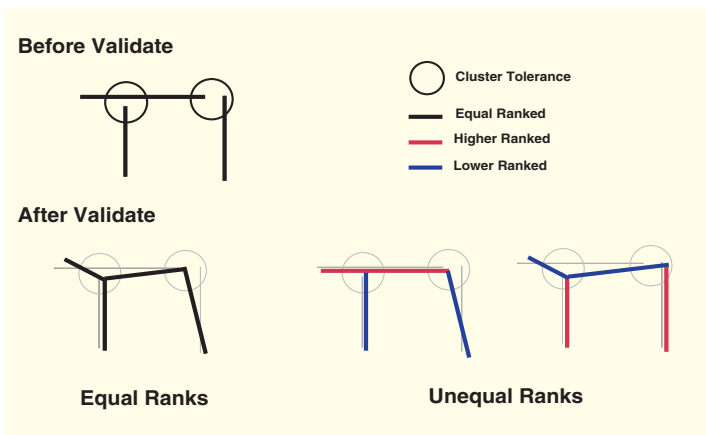
The cluster tolerance is typically a small actual distance, to minimize the movement of correctly placed features. The default cluster tolerance is the smallest cluster tolerance possible for a dataset and is calculated based on the dataset's precision and extent. The precision describes the number of system units per one unit of measure in the dataset; so it defines the smallest storable distance between coordinates in the dataset. A spatial reference with a precision of 1 will store integer values, while a precision of 1,000 will store three decimal places. The extent defines the maximum geographic extent that can be stored in the dataset.



*When you validate a topology, features within the x,y cluster tolerance are snapped together.*



The ranks you specify for feature classes in the topology control the features that will be moved when snapping coincident vertices during the initial validation of the topology and during subsequent validations. When different feature classes have different levels of accuracy, such as when one was collected by survey or a differential global positioning system (GPS) and another was digitized from less accurate source material or collected with uncorrected GPS, ranks can allow you to ensure that reliably placed vertices are not snapped to the location of less reliable ones. Lower ranked features' vertices will be snapped to the location of higher ranked vertices if they fall within the cluster tolerance. The location of equally ranked vertices are geometrically averaged when they are within the cluster tolerance.

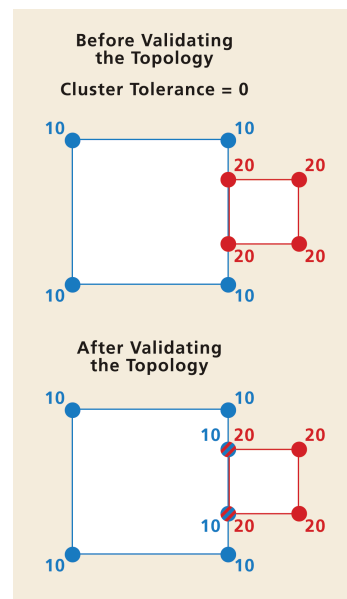


When you validate a topology, the ranks of the feature classes in the topology control how features are snapped together. Lower ranking features snap to higher ranking features. Equally ranked features snap to the geometric average of their position.

## Z cluster tolerance and ranks

Feature classes that model terrain or buildings three dimensionally have a z-value representing elevation for each vertex. Just like you control how features are snapped horizontally with x,y cluster tolerance and ranks, if a topology has feature classes that model elevation, you can control how coincident vertices are snapped vertically with the z cluster tolerance and ranks.

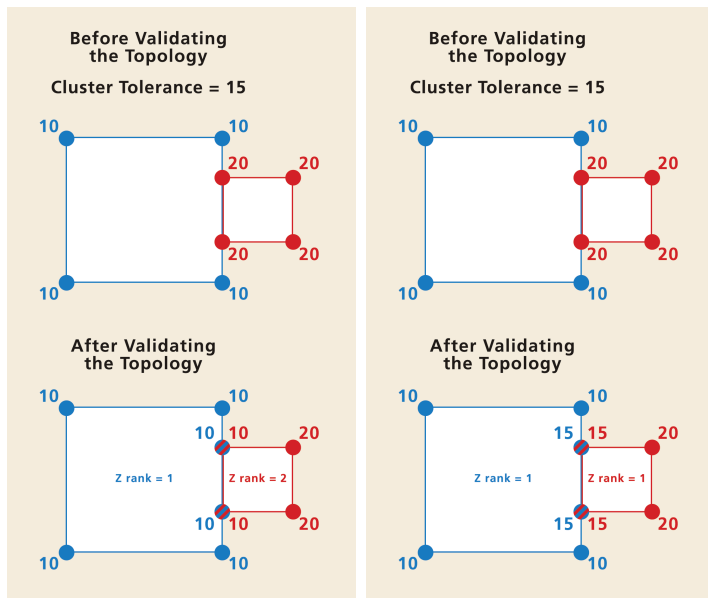
The z cluster tolerance defines the minimum difference in elevation or z-value between coincident vertices. Vertices with z-values that are within the z cluster tolerance are snapped together during the Validate Topology process.



New vertices inserted by the topology validation process get interpolated along the feature. If the cluster tolerance is zero, z-values don't change.

If you're modeling city buildings, two buildings may be adjacent to one another and appear to share a common edge in the x,y domain. If elevation values for building corners were collected using photogrammetry, you should be concerned about maintaining the relative height of each building structure during the topology validation process. By setting the z cluster tolerance to a value of zero, you can prevent z-values from clustering when you validate topology.

If you're modeling terrain, you may have datasets collected with different x,y and z accuracies. In this case, you may want to set a z cluster tolerance greater than zero to allow snapping. To avoid z-values collected with a high level of accuracy snapping to less accurate z-values, you can assign each feature class a rank. Lower ranked features' z-values snap to the elevation of higher ranked vertices if they fall within the cluster tolerance. Z-values of vertices belonging to feature classes of the same rank are averaged if they fall within the cluster tolerance. For more information, see the ArcGIS online Help system.



The z-values of vertices belonging to the feature class of the lower rank change to those of the higher rank when they fall within the z cluster tolerance. The z-values of vertices belonging to feature classes of the same rank are averaged when they fall within the z cluster tolerance.

## Feature layers maintained by a topology

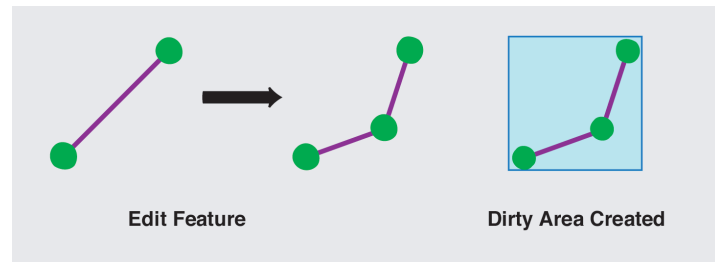
Instead of storing topological information with the feature classes, the topology discovers those relationships when the information is requested, such as when you are editing using the shared geometry tool. To help you manage the process of creating and editing a logically consistent topology, the topology internally stores two additional types of feature classes: dirty areas and error features.

### Dirty areas

Dirty areas let the topology efficiently track the places where topology rules may have been violated during editing. The dirty areas allow selected parts, rather than the whole extent of the topology, to be validated after editing.

Dirty areas are created when:

- A feature is created or deleted.
- A feature's geometry is modified.
- A feature's subtype is changed.
- Versions are reconciled.
- The topology properties are modified.

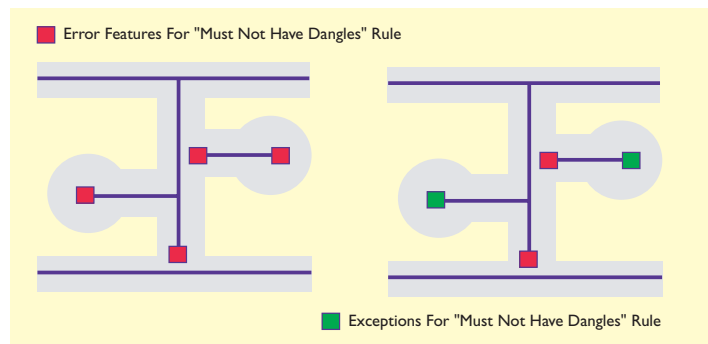


When you edit features in a topology, the topology creates a dirty area to mark the area that should be checked for violations of the topology rules.

Dirty areas are stored in the topology as a single feature, with each new dirty area united with the existing dirty area and each area that has been validated removed from the dirty area.

## Errors and exceptions

Topologies also store error features, which record where topological errors were discovered during validation. Certain errors may be acceptable, in which case the error features can be marked as exceptions.



*When you validate a topology, features that violate the rules are marked as error features. You can edit the features to fix the errors, or you can mark the errors as exceptions. In this example, the street line features cannot have dangles, which are endpoints that do not connect to other street features. Because cul-de-sac streets are a legitimate exception to this rule, they may be marked as exceptions in the topology. The remaining errors should be fixed by editing the street features.*

ArcMap and ArcCatalog allow you to create a report of the total number of errors and exceptions for the feature classes in your topology. You can use the report of the number of error features

as a measure of the data quality of a topological dataset. In addition, the error inspector in ArcMap lets you select different types of errors and zoom to individual errors. You can correct topology errors by editing the features that violate the topology's rules. After you validate the edits, the error is deleted from the topology.

## Topology review

Rules define the permissible relationships between features. Ranks control which features may be moved to other features when snapping the topology together in the initial validation and during subsequent validations of the topology. The x,y and z cluster tolerances define how close vertices must be to each other in order to be considered coincident. Dirty areas are areas that have been edited or affected by the addition, update, or deletion of features. Dirty areas allow the topology to limit the area that must be checked for topology errors during topology validation. Errors and exceptions are stored as features in the topology and allow you to render and manage the places where features do not obey the topology rules you specified.

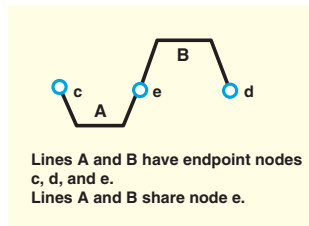
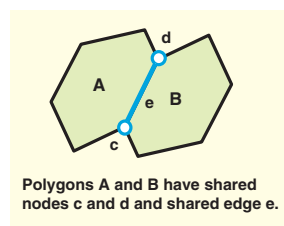
# Topology and feature geometry

## Geometries involved in a topology

The features participating in a topology belong to simple feature classes in the same dataset. Rather than modifying the definition of the feature class, a topology serves as a description of how the features in a feature dataset can be spatially related. Annotation, dimension, and geometric network features are not simple features and cannot participate in a topology. Feature classes outside of the topology's feature dataset cannot participate in the topology, and feature classes cannot participate in more than one topology at a time.

At the geometry level, topologies are about simple relationships such as coincidence, covering, and crossing between the geometric primitives that make up features. While all simple feature class geometries (point, line, polygon) may participate in topologies, internally, the types of geometry that are acted on when editing a topology are:

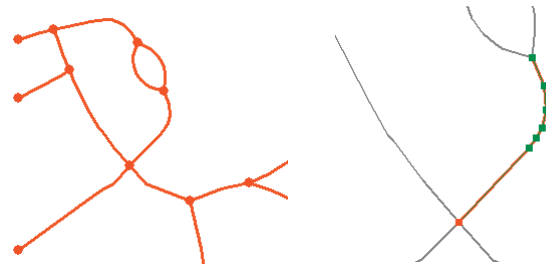
- Edges—Line segments that define lines or polygons.
- Nodes—Points at the end of an edge.
- *Pseudonodes*—A node connecting only two edges or a logical split defined in the topology cache while editing. Pseudonodes of the latter sort become a vertex after editing.



## Ways of sharing geometry

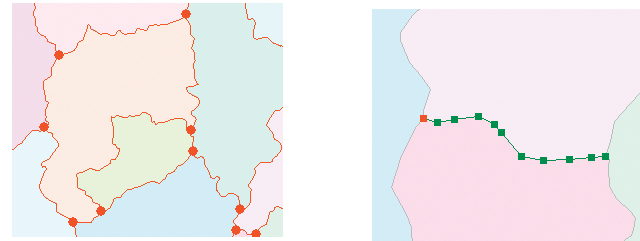
Features can share geometry within a topology in a number of ways:

- Line features can share endpoints (*arc-node topology*).



Line features can share edges and nodes, in red. Vertices define the shape of the edges, in green.

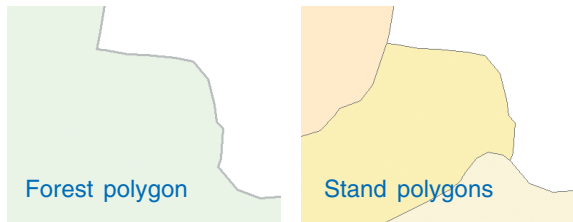
- Area features can share boundaries (polygon topology).



Polygon features share edges and nodes, in red. Vertices define the shape of the edges, in green.

- Line features can share segments with other line features (route topology).

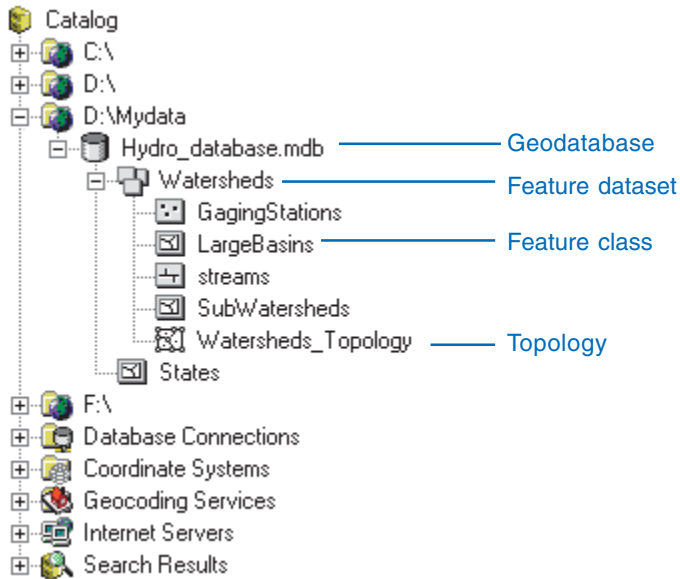
- Area features can be coincident with other area features (region topology).



- Line features can share endpoint vertices with point features (node topology).
- Point features can be coincident with line features (point events).

# Topologies and ArcCatalog

In ArcCatalog, you can view and manage topologies in geodatabases. Because all topologies must be inside a feature dataset, they appear in the ArcCatalog tree under their feature dataset.



It is not immediately evident in the ArcCatalog tree which feature classes in the dataset participate in the topology, which feature classes participate in which topology (if there is more than one topology in the dataset), and which participate in none. However, by examining the properties of a topology, you can identify its constituent feature classes.

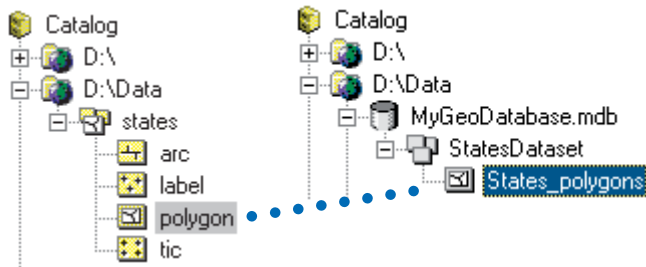
ArcCatalog also contains various tools to create, delete, and manage both topologies and the feature classes that participate in topologies. These tools are discussed in more detail later in this chapter.

Once you have created a topology using the Topology wizard in ArcCatalog, you can validate it in ArcMap or ArcCatalog. Validating in ArcCatalog is typically faster.

# Migrating data into a geodatabase to create topologies

Before you create a topology, you should look at the data that you will be using to see what feature classes and topological rules you will need. In some cases, you will be migrating non-topological data, such as shapefiles, into a geodatabase and creating new topological relationships. In other cases, you will be moving data from the coverage topological data model. The coverage data model allows, and can be used to enforce, certain topological relationships. Some of these relationships may be useful for your database design and should be re-created using topology rules. Others may not be needed, in which case you may choose not to re-create them.

Coverages may also contain feature classes needed for the coverage data model that aren't needed in a geodatabase. You may choose not to import these feature classes.



*Polygon coverage with polygon and supporting feature classes, compared to personal geodatabase polygon feature class. You do not typically need to import all of the coverage feature classes into your geodatabase.*

Similarly, coverages maintain topological information in a number of attribute fields. Since geodatabase topology is not stored with the features, this information may be redundant. You may choose to drop some of these attributes when loading coverage data into a geodatabase.

OBJECTID	Shape*	AREA	PERIMETER	STATES	STATES_ID	Shape_Area	Shape_Length	STATE_NAME
1	Polygon	1796232500	270761.75	2	34	17962323017201289	2870761.639158487	Washington
2	Polygon	144819120	59689.63871875	3	38	144819034.0177	59693.6256835659	Washington
3	Polygon	504790692	16585.234375	4	39	504790695.951825	165265.225458305	Washington

Annotations below the table:

- Columns managed by the geodatabase: OBJECTID, STATE\_NAME
- Columns managed by coverage topology: AREA, PERIMETER, STATES
- User assigned ID attribute: STATES\_ID
- Columns managed by the geodatabase: Shape\_Area, Shape\_Length
- User assigned attribute: STATE\_NAME

*Attribute table for a polygon feature class imported from a coverage. Polygon coverage feature classes have AREA, PERIMETER, and <COVER># fields (circled in red, above) that are managed by the ArcInfo coverage topology model. These are not updated by the geodatabase topology tools. You do not typically need to import these fields into your geodatabase. Certain attributes that are managed by the geodatabase are added to the attribute table during the import process. Shape stores the geometry, while Shape\_Area and Shape\_Length store these attributes of the geometry.*

## Migrating point features to a geodatabase

Point feature classes from shapefiles or coverages can migrate into point feature classes in a geodatabase. In a coverage, *label point* features may be related to an arc feature class to form polygons. The polygons' attributes are stored with the label points. In the geodatabase, the attributes of polygon features are stored with the polygons, so label points are unnecessary. You can use point features as an input feature class to supply attributes for new polygon features when creating polygons from line features. The attributes of the points are copied to the polygons; the points do not need to be stored with the polygon features in the geodatabase.

## Geometry

Point features are not constructed from other feature classes in the coverage data model, so there are no supporting feature classes to keep or drop when importing coverage point features to a geodatabase. The Area and Perimeter items may be dropped, as they are used to manage polygon coverage topology.

## Attributes

Node feature classes in an arc coverage may carry attributes and can be migrated to point feature classes in the geodatabase. The ARC# item in a node attribute table may be redundant, as it is primarily used to manage coverage arc–node topology.

## Topology

In a geodatabase topology, point features can be topologically related to line features, the endpoints of line features, and polygon features. Points can be constrained to fall along lines, at the endpoints of lines, within polygons, or on the edges of polygons. See the ‘Topology: defining the rules’ section of this chapter for a more detailed discussion of the main topological relationships supported by the geodatabase for point features.

## Migrating linear features to a geodatabase

Line features are represented by arcs in a coverage and by polylines in a shapefile. Arc and polyline feature classes migrate into line feature classes in a geodatabase.

## Geometry

Arcs may be topologically related to nodes, polygons, or other arcs (routes) in the coverage data model. When migrating arc coverages into a geodatabase, you may want to import the node features as point features if you use the nodes to store attribute values. If you do not store attributes with nodes, you can choose

not to migrate the node features. When migrating route features, you can choose whether or not to migrate the supporting arc feature class to another line feature class.

## Attributes

Coverage arc feature classes have a Length field that is superseded by the geodatabase’s Shape\_length field. The geodatabase will not update this field, so it may be deleted when you load the feature class. Coverage arc feature classes also have FNODE#, TNODE#, LPOLY#, RPOLY#, and <cover name># fields that are not managed by geodatabase topology. You can drop these fields when you import the coverage.

## Topology

Polyline features in shapefiles do not have topology rules or topological relationships to other features.

Coverage arc feature classes have built-in topology rules that specify that they must split at a node when they cross other arcs. Arcs have an additional topology rule that they must connect to more than one other arc at a node. Exceptions to this rule are called dangles—where an end of an arc fails to connect to another arc—and pseudonodes—where an arc connects to itself or one other arc.

In a geodatabase topology, line features can be topologically related to point features, other line features, and polygon features. Lines can be constrained to fall along other lines, to connect to other lines at their endpoints, or not to touch or intersect themselves or other lines. They can be constrained not to have dangles or pseudonodes. They can outline the edges of polygons or not touch polygons. See the ‘Topology: defining the rules’ section of this chapter for a more detailed discussion of the main topological relationships supported by the geodatabase for line features.



## Migrating area features to a geodatabase

Area features are represented by polygon feature classes in coverages and in shapefiles. They may also be represented by region feature classes in a coverage. Coverage polygon topology has certain inherent rules, requiring coverage polygon edges to be defined by the arcs' feature class, to completely tile the extent of the coverage, and not to overlap other polygons. Coverage polygons may also be topologically related to a label point feature class.

Shapefile polygons do not have any topology rules and are not topologically related to other feature classes.

### Geometry

Coverage polygon feature classes are topologically related to two supporting feature classes, the label and arc feature classes. They may also be topologically related to other polygons to form regions, which are coverage area features that can overlap and which do not have to tile the whole extent of the coverage. Regions are somewhat like shapefile polygons. They migrate to polygon features in a geodatabase. You may choose not to migrate the supporting polygon feature classes if the regions store the attributes you use.

### Attributes

Coverage polygon feature classes have Perimeter and Area fields that are superseded by the geodatabase's Shape\_length and Shape\_area fields. Perimeter and Area fields can be dropped when polygon coverages are migrated to the geodatabase. The supporting arc and node feature classes can also be dropped as geodatabase polygons do not depend upon these feature classes.

## Topology

In a geodatabase topology, polygon features can be topologically related to point features, line features, and other polygon features. Points can be constrained to fall inside or on the edges of polygons. Lines can be constrained to fall along the edges of polygons, within polygons, outside or not touching polygons, or not crossing the edges of polygons. Polygons can be constrained to not overlap or to be allowed to overlap. Polygons from one feature class may tile within the polygons of another polygon feature class, may exactly cover the other feature class, or may not overlap polygons of another feature class. See the 'Topology: defining the rules' section of this chapter for a more detailed discussion of the main topological relationships supported by the geodatabase for polygon features.

## Creating a new topology

Topologies are created inside feature datasets. When you create a topology, you must specify which feature classes in the feature dataset participate in the topology and define the topological rules that they must obey.

New feature classes can be added to a topology at any time. Only simple feature classes can participate in topologies.

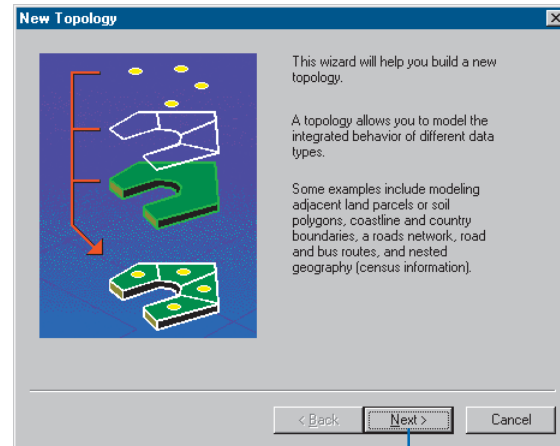
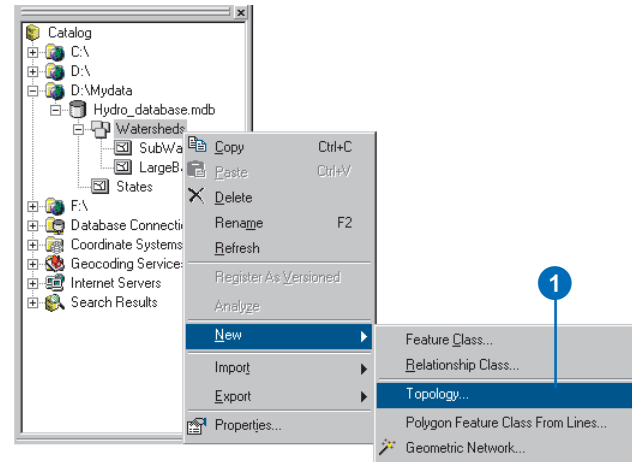
### See Also

*For more information on creating feature datasets and feature classes, see the chapter 'Creating new items in a geodatabase' in this book.*

1. Right-click the feature dataset that will contain the topology, point to New, and click Topology.

The New Topology wizard starts.

2. Read the information on the first panel and click Next.



## Tip

### Cluster tolerance

The cluster tolerance defines the minimum distance between vertices in the topology. Vertices that fall within the cluster tolerance will be snapped together during the Validate Topology process. The default cluster tolerance is the minimum possible cluster tolerance, based on the precision and extent defined for the spatial reference of the dataset.

If you change the cluster tolerance, you should choose one that is an order of magnitude smaller than the accuracy of the most accurate feature class in your dataset. For example, if all of your features are accurate to 2 meters, you would set a cluster tolerance of 0.2 meters. If you have another feature class in the same dataset that is accurate to 0.25 meters, you should set the cluster tolerance to 0.025 meters in order to avoid degrading the higher accuracy data. You will also want to assign a higher rank to the more accurate features in order to prevent them from being snapped to less accurate features. More information about ranking feature classes appears on the following page.

## Tip

### Feature classes not in the topology

Only feature classes in the same dataset can participate in a topology, but not all of the feature classes in a dataset are required to participate in the topology.

3. Type a name for the topology.

4. Type a cluster tolerance for the topology.

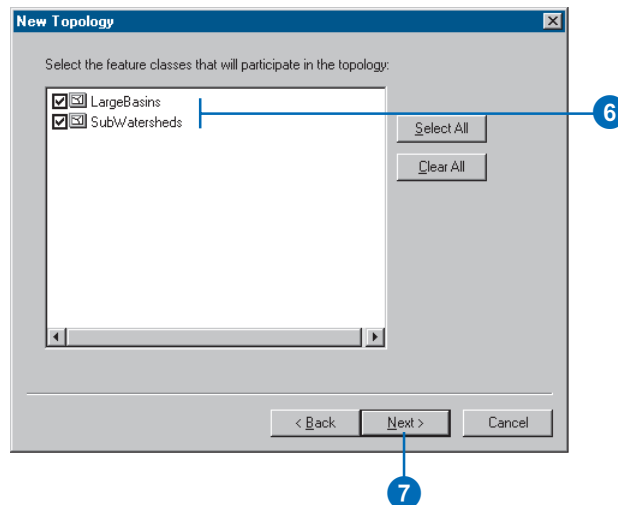
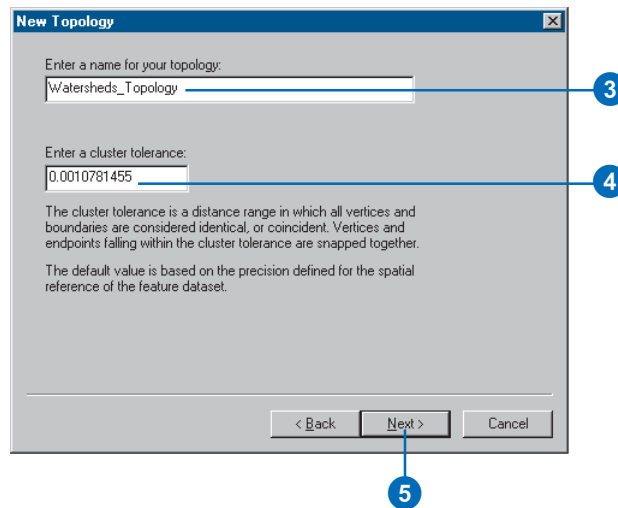
The default cluster tolerance is based on the precision of the dataset and is the minimum possible cluster tolerance.

5. Click Next.

6. Check the feature classes that will participate in the topology.

You will not see relationship, annotation, or dimension classes; feature classes registered as versioned; or feature classes that participate in another topology or geometric network.

7. Click Next. ▶

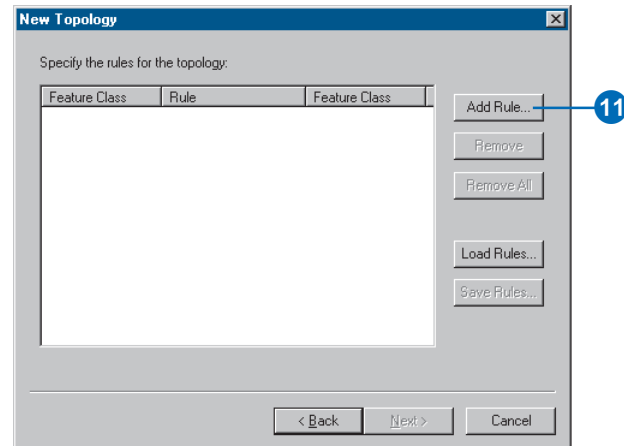
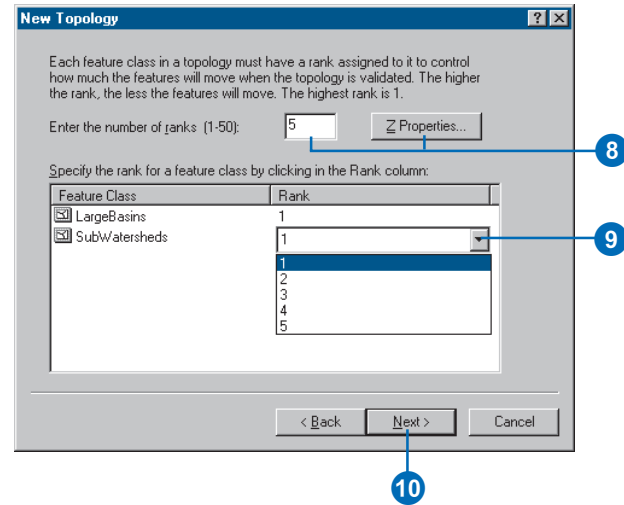


## Tip

### Feature class ranks

When a topology is validated, all the vertices of each feature class are evaluated against the cluster tolerance. Vertices that are within the cluster tolerance of each other are snapped together. To avoid having vertices from a feature class collected with a high level of accuracy being snapped to vertices from a less accurate feature class, you assign each feature class a rank. Vertices from higher ranking feature classes will not be moved when snapping to vertices with lower ranked feature classes. The highest rank is 1, and you can assign up to 50 different ranks. The position of vertices belonging to feature classes of the same rank will be geometrically averaged when they fall within the cluster tolerance.

8. Type the number of ranks you want to allow in the topology.  
  
Optionally, if your features have z-values embedded in their geometries, click the Z Properties button to set the z cluster tolerance and ranks.
9. Click in the Rank column and assign each feature class a rank.
10. Click Next.
11. Click Add Rule. ▶



12. Select a feature class that you want to participate in a topology rule.

13. Select a topology rule.

The rule description for the rule you picked appears in the Rule Description panel.

14. Select the second feature class if the rule relates the feature class to another feature class.

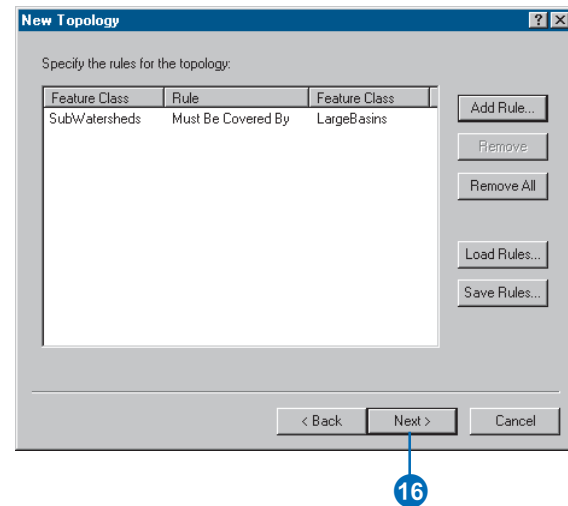
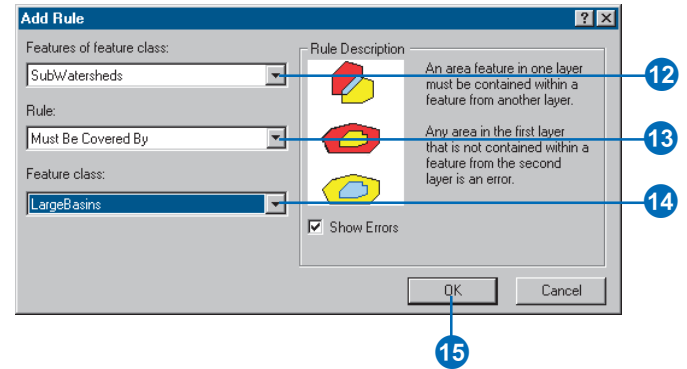
Optionally, click the Show Errors check box to hide or show which geometric relationships are considered errors for the rule.

15. Click OK.

Optionally, add additional topology rules, repeating steps 11 through 15.

Optionally, click Save rules. This will save the rules you've specified as a rule set. Saving a rule set is useful when you know you will be setting up another topology using the same or similar rules.

16. Click Next. ►



## See Also

For details about using storage keywords with ArcSDE, see *Managing ArcSDE Services*.

## Tip

### Validating topology

The first time you create a topology, you will be asked to validate it. Validating a topology evaluates the rules and creates error features where the rules are violated. Validating the topology also starts the cracking and clustering process, which can take some time and is irreversible.

During cracking, vertices are created at the intersection of feature edges. During clustering, vertices that fall within the cluster tolerance are snapped together. The rank of a feature class determines whether or not its vertices will move when they fall within the cluster tolerance of vertices of another feature. Where vertices belong to features with the same rank—within the same feature class, for example—the clustering process geometrically averages the position of the vertices.

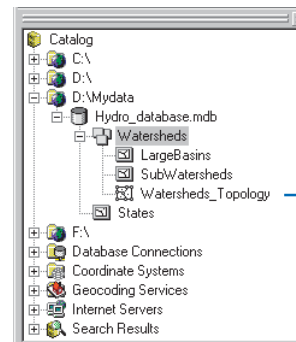
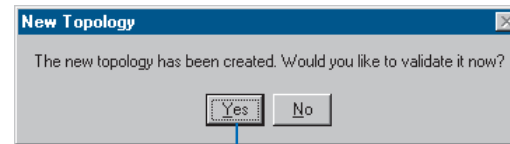
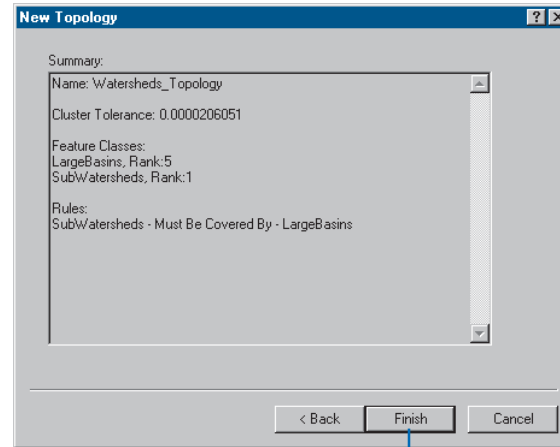
17. Click Yes, click the configuration keyword dropdown list, and click the keyword to use if your geodatabase is stored in an ArcSDE database and you have a configuration keyword for the topology storage. If not, skip to step 18.
18. Review the parameters and rules you've defined for the topology.
19. Click Finish.

The wizard begins creating the new topology and a progress bar appears. You can cancel the process by clicking Cancel.

Once the topology is created, you will be asked if you want to validate the topology.

20. Click Yes.

The topology is validated and appears in the feature dataset.



New topology added to dataset in ArcCatalog

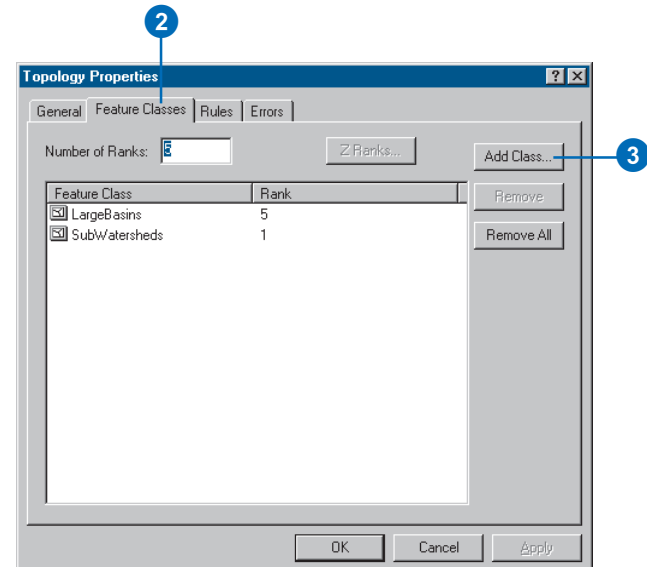
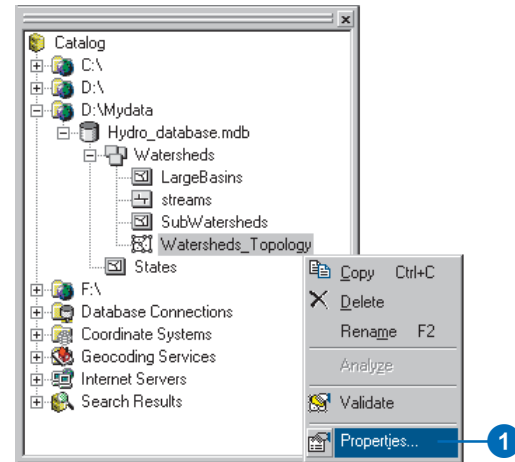
## Adding new feature classes to your topology

At any time, you can add new feature classes to a topology. These new feature classes can be empty or may contain existing features. The new feature class must be in the same feature dataset as the topology. Versioned feature classes cannot be added to a topology.

When adding a feature class to a topology, you must specify the rules that govern the feature classes' spatial relationships. Adding new rules to a topology automatically makes the entire topology dirty, so when you finish adding rules you will need to revalidate the topology. The new features may create error conditions, depending on the rules that you add.

### Adding a new feature class to a topology

1. Right-click the topology and click Properties.
2. Click the Feature Classes tab.
3. Click Add Class. ▶

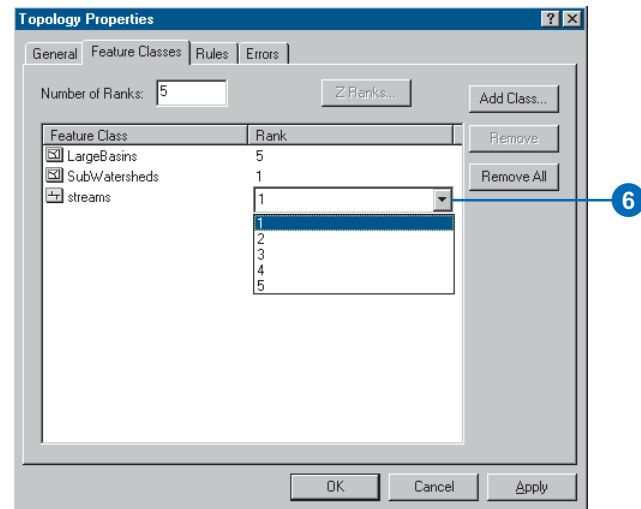
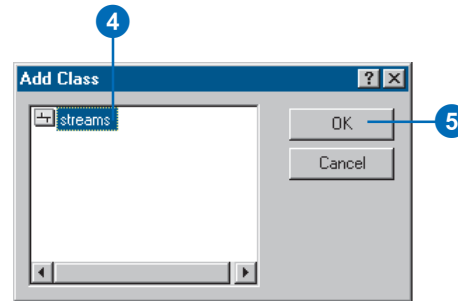


- Click the feature class that you want to add to the topology.

Only feature classes that are within the dataset, and that are not currently participating in a topology or geometric network, can be added.

You will not see relationship, annotation, or dimension classes; feature classes registered as versioned; or feature classes that participate in another topology or geometric network.

- Click OK.
- Click in the Rank column and set a rank for the new feature class. ►



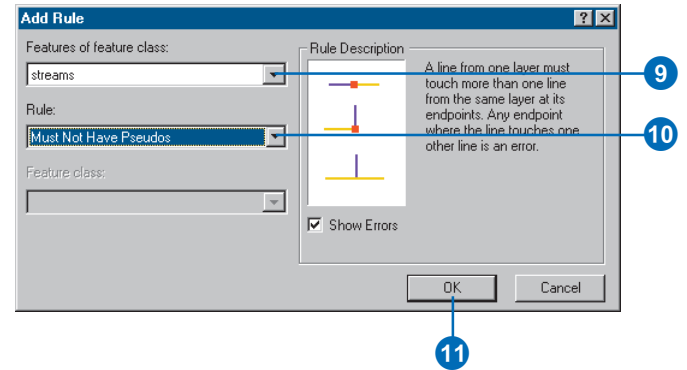
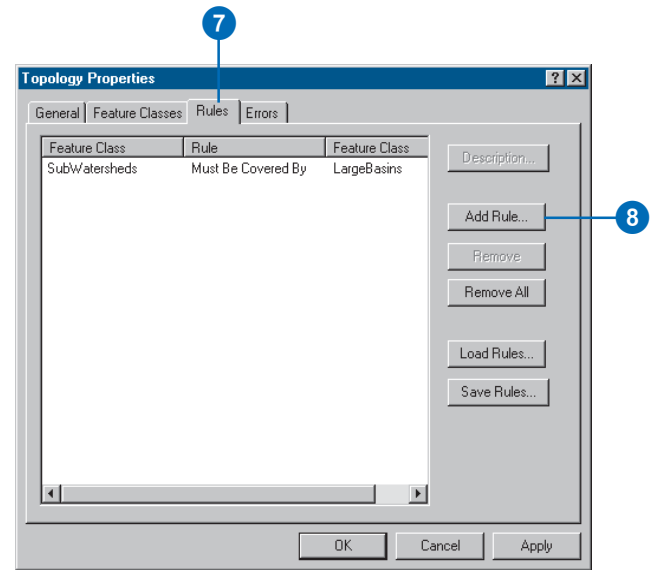


7. Click the Rules tab.
8. Click Add Rule.
9. Choose the feature class that will participate in the rule.
10. Choose the rule.

Optionally, choose the other feature class that will participate in the rule. Some rules only apply to the features in one feature class, while others apply to features in two different classes.

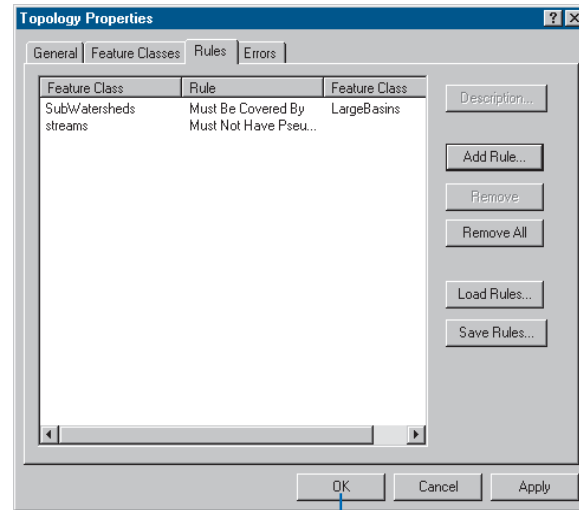
11. Click OK.

Optionally, repeat steps 8–10 to define other rules involving the new feature class. ►



12. Click OK.

The new feature class and rules have been added to the topology. The topology will need to be validated again.



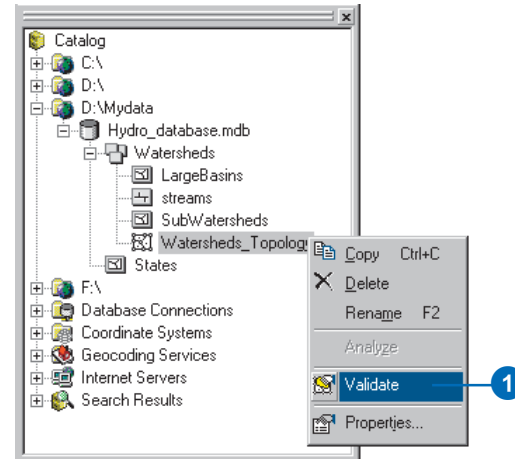
## Validating a topology

When the rules or other properties of a topology are changed, the topology will need to be validated again. Validating the topology evaluates the features against the rules and finds any new errors related to new rules or feature classes. It will also remove errors related to rules or feature classes you've removed.

Validating the topology starts the irreversible *cracking* and *clustering* process, which can take some time. During cracking, vertices are created at the intersection of feature edges. During clustering, vertices that fall within the cluster tolerance snap together. The rank of a feature class determines whether or not its vertices will move when they fall within the cluster tolerance of vertices of another feature. Where vertices belong to features with the same rank (within the same feature class, for example), the clustering process geometrically averages the position of the vertices.

Once a topology has been validated, it will typically only experience cracking and clustering again where new features are added, unless you change the cluster tolerance or add feature classes.

1. Right-click the topology and click Validate.



# Topology: defining the rules

Some topology rules govern the relationship between features in a single feature class, while others govern the relationship between features in two feature classes. The rules can also be between two subtypes in a feature class or between subtypes in two different feature classes.

Some of the key topological rules are discussed in the following pages. The first step is to choose the most important ones for your data, then plan how you will implement and maintain each one.

## Polygon rules

### Must Not Overlap

This rule requires that the interior of polygons in the feature class not overlap. The polygons can share edges or vertices. This rule is used when an area cannot belong to two or more polygons. It is useful for modeling administrative boundaries, such as ZIP Codes or voting districts, and mutually exclusive area classifications such as land cover or landform type.

### Must Not Have Gaps

This rule requires that there are no voids within a single polygon or between adjacent polygons. All polygons must form a continuous surface. An error will always exist on the perimeter of the surface. You can either ignore this error or mark it as an exception. Use this rule on data that must completely cover an area. For example, soil polygons cannot include gaps nor form voids—they must cover an entire area.

### Must Not Overlap With

This rule requires that the interior of polygons in one feature class must not overlap with the interior of polygons in another feature class. Polygons of two feature classes can share edges or vertices or be completely disjointed. This rule is used when an

area cannot belong to two separate feature classes. It is useful for combining two mutually exclusive systems of area classification, such as zoning and water body type, where areas defined within the zoning class cannot also be defined in the water body class and vice versa.

### Must Be Covered By Feature Class Of

This rule requires that a polygon in one feature class must share all of its area with polygons in another feature class. An area in the first feature class that is not covered by polygons from the other feature class is an error. This rule is used when an area of one type, such as a state, should be completely covered by areas of another type, such as counties.

### Must Cover Each Other

This rule requires that the polygons of one feature class must share all of their area with the polygons of another feature class. Polygons may share edges or vertices. Any area defined in either feature class that is not shared with the other is an error. This rule is used when two systems of classification are used for the same geographic area and any given point defined in one system must also be defined in the other. One such case occurs with nested hierarchical datasets, such as census blocks and block groups or small watersheds and large drainage basins. The rule can also be applied to nonhierarchically related polygon feature classes, such as soil type and slope class.

### Must Be Covered By

This rule requires that polygons of one feature class must be contained within polygons of another feature class. Polygons may share edges or vertices. Any area defined in the contained feature class must be covered by an area in the covering feature class. This rule is used when area features of a given type must be located within features of another type. This rule is useful

when modeling areas that are subsets of a larger surrounding area, such as management units within forests or blocks within block groups.

### **Boundary Must Be Covered By**

This rule requires that boundaries of polygon features must be covered by lines in another feature class. This rule is used when area features need to have line features that mark the boundaries of the areas. This is usually when the areas have one set of attributes, and their boundaries have other attributes. For example, parcels might be stored in the geodatabase along with their boundaries. Each parcel might be defined by one or more line features that store information about their length or the date surveyed, and every parcel should exactly match its boundaries.

### **Area Boundary Must Be Covered By Boundary Of**

This rule requires that boundaries of polygon features in one feature class be covered by boundaries of polygon features in another feature class. This is useful when polygon features in one feature class, such as subdivisions, are composed of multiple polygons in another class, such as parcels, and the shared boundaries must be aligned.

### **Contains Point**

This rule requires that a polygon in one feature class contain at least one point from another feature class. Points must be within the polygon, not on the boundary. This is useful when every polygon should have at least one associated point, such as when parcels must have an address point.

## **Line rules**

### **Must Not Overlap**

This rule requires that lines not overlap with lines in the same feature class. This rule is used where line segments should not be

duplicated, for example, in a stream feature class. Lines can cross or intersect but cannot share segments.

### **Must Not Intersect**

This rule requires that line features from the same feature class not cross or overlap each other. Lines can share endpoints. This rule is used for contour lines that should never cross each other or in cases where the intersection of lines should only occur at endpoints such as street segments and intersections.

### **Must Not Have Dangles**

This rule requires that a line feature must touch lines from the same feature class at both endpoints. An endpoint that is not connected to another line is called a dangle. This rule is used when line features must form closed loops such as when they are defining the boundaries of polygon features. It may also be used in cases where lines typically connect to other lines as with streets. In this case, exceptions can be used where the rule is occasionally violated as with cul-de-sac or dead-end street segments.

### **Must Not Have Pseudos**

This rule requires that a line connect to at least two other lines at each endpoint. Lines that connect to one other line (or to themselves) are said to have pseudo-nodes. This rule is used where line features must form closed loops, such as when they define the boundaries of polygons, or when line features logically must connect to two other line features at each end, as with segments in a stream network, with exceptions being marked for the originating ends of first-order streams.

### **Must Not Intersect Or Touch Interior**

This rule requires that a line in one feature class must only touch other lines of the same feature class at endpoints. Any line segment in which features overlap or any intersection not at an endpoint is an error. This rule is useful where lines must only be

connected at endpoints, such as in the case of lot lines, which must split (only connect to the endpoints of) back lot lines and which cannot overlap each other.

### **Must Not Overlap With**

This rule requires that a line from one feature class not overlap with line features in another feature class. This rule is used when line features cannot share the same space. For example, roads must not overlap with railroads, or depression subtype of contour lines cannot overlap with other contour lines.

### **Must Be Covered By Feature Class Of**

This rule requires that lines from one feature class must be covered by the lines in another feature class. This is useful for modeling logically different but spatially coincident lines such as routes and streets. A bus route feature class must not depart from the streets defined in the street feature class.

### **Must Be Covered By Boundary Of**

This rule requires that lines be covered by the boundaries of area features. This is useful for modeling lines, such as lot lines, that must coincide with the edge of polygon features, such as lots.

### **Endpoint Must Be Covered By**

This rule requires that the endpoints of line features must be covered by point features in another feature class. This is useful for modeling cases where a fitting must connect two pipes or a street intersection must be found at the junction of two streets.

### **Must Not Self Overlap**

This rule requires that line features not overlap themselves. They can cross or touch themselves, but must not have coincident segments. This rule is useful for such features as streets, where segments might touch, in a loop, but where the same street should not follow the same course twice.

### **Must Not Self Intersect**

This rule requires that line features not cross or overlap themselves. This rule is useful for lines, such as contour lines, that cannot cross themselves.

### **Must Be Single Part**

This rule requires that lines must have only one part. This rule is useful where line features, such as highways, may not have multiple parts.

## **Point rules**

### **Must Be Covered By Boundary Of**

This rule requires that points fall on the boundaries of area features. This is useful when the point features help support the boundary system, such as boundary markers, which must be found on the edges of certain areas.

### **Must Be Properly Inside Polygons**

This rule requires that points fall within area features. This is useful when the point features are related to polygons, such as wells and well pads or address points and parcels.

### **Must Be Covered By Endpoint Of**

This rule requires that points in one feature class must be covered by the endpoints of lines in another feature class. This rule is similar to the line rule, Endpoint Must Be Covered By, except that, in cases where the rule is violated, it is the point feature that is marked as an error rather than the line. Boundary corner markers might be constrained to be covered by the endpoints of boundary lines.

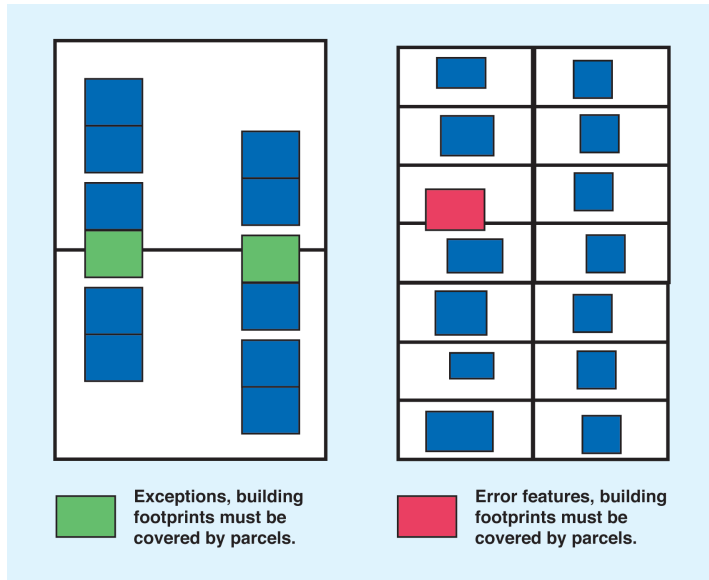
### **Must Be Covered By Line**

This rule requires that points in one feature class must be covered by lines in another feature class. It does not constrain the covering portion of the line to be an endpoint. This rule is useful for points that fall along a set of lines, such as highway signs that fall along highways.

# Planning for exceptions

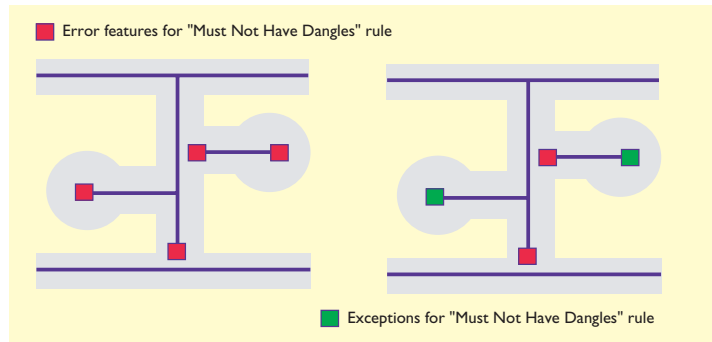
Topology rules may represent an ideal situation, but geodatabases are flexible enough to handle exceptions to the rules found in real-world data. Violations of topology rules are stored as errors in the topology but, where appropriate, you can mark them as exceptions. Exceptions are thereafter ignored when the error inspector searches for errors, though you can return them to error status if you decide that they are actually errors and that the features should be modified to comply with the topology rules.

Here are a couple of cases where you might implement topology rules that you know will have exceptions. If you manage a database of parcels and you want to *digitize* a feature class of buildings, you might implement a topology rule requiring that parcels cover building features (i.e., that building features not cross parcel lines) as a quality control for the building digitizing



effort. This rule might be true for 90 percent of the features in your management area, but it could be violated by some high-density housing and commercial buildings. You can designate the buildings that represent acceptable violations of the rule as exceptions.

Similarly, if you manage a street database for a city, you might create a rule that centerlines must not have dangles (i.e., that they connect at both ends to other centerlines). This rule would ensure that street segments are correctly snapped to other street segments when the streets are edited. However, some streets are cul-de-sacs, one end of which does not snap to other centerlines. These cases could be marked as exceptions, and you would still be able to use the rule to find cases where streets were incorrectly digitized or edited.



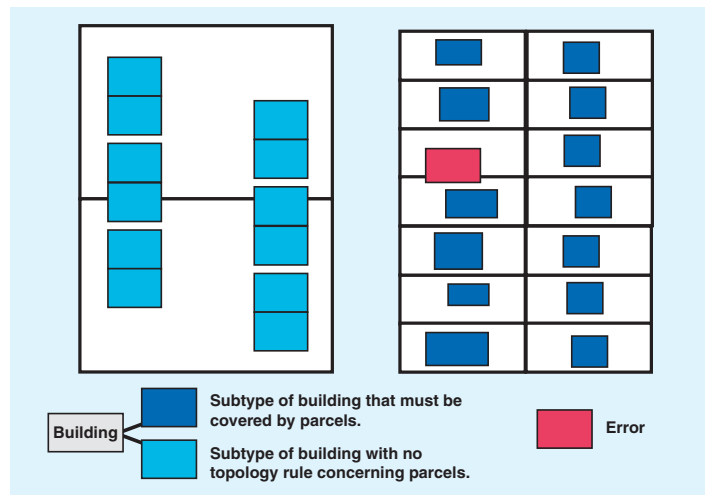


## Refining topologies with subtypes

When you design a geodatabase, you should be aware of the option of creating topological relationships between subtypes of features classes. Subtypes allow you to model real-world objects more effectively by creating specialized default values and domains for specific subtypes of features. Subtypes also allow you to represent a variety of real-world objects in a given feature class instead of in several feature classes, which provides some performance benefits for the geodatabase.

Subtypes extend your design options for topology rules. In some cases, you will want a topology rule to apply to all features in a feature class, except for a certain type of feature. One way to handle this design requirement is to create the rule for the whole feature class and systematically mark the exceptions. Another is to use subtypes, rather than feature classes, in your topology rules. This allows you to create rules that apply only to specific subtypes.

Using the building footprint example from the previous page, you could solve the problem of a small percentage of buildings that can legitimately cross parcel boundaries by creating subtypes of buildings and only creating the Must Be Covered By parcels topology rule for the subtypes that cannot extend across a parcel.



*Subtypes allow you finer control in setting up topology rules.*

# Managing a topology

You can manage topologies using ArcCatalog. Unlike most items that appear in ArcCatalog, the topology does not represent a single entity, such as a table, shapefile, or feature class. A topology is actually an association among several feature classes and is represented by several tables in the database. Managing a topology is different from managing other items in ArcCatalog.

## Managing the topology itself

Some of the standard operations on the topology are handled the same way as other items in ArcCatalog. A topology can be renamed or deleted. Renaming the topology doesn't affect any of its member feature classes or the structure of the topology. Deleting a topology does not affect the participating feature classes; it merely removes the rules governing their spatial relationships.

You can delete a topology in two ways. The first is to delete the entire feature dataset that contains the topology. This action deletes from the geodatabase all of the participating feature classes, all of the topology rules, and any other objects stored inside that feature dataset. The second method is to simply delete the topology itself and leave the rest of the feature dataset intact.

You can also copy and paste a topology from one feature dataset to another. Copying a topology also copies the feature classes that participate in the topology.

## Managing topologically related feature classes

Managing feature classes in a topology is more restricted than managing feature classes that do not participate in a topology. You must remove the feature class from the topology or delete the topology before you can rename or delete a feature class that participates in a topology. You can still add fields, alter subtypes, and modify domains of fields for feature classes in a topology.

## Schema locking

An exclusive lock is required to modify a topology's rules or to rename or delete a topology. An exclusive lock can only be acquired for a topology if the feature classes that participate in the topology can also be locked. Therefore, if another user has an exclusive or shared lock on any of the feature classes in a topology, then the properties of the topology cannot be edited.

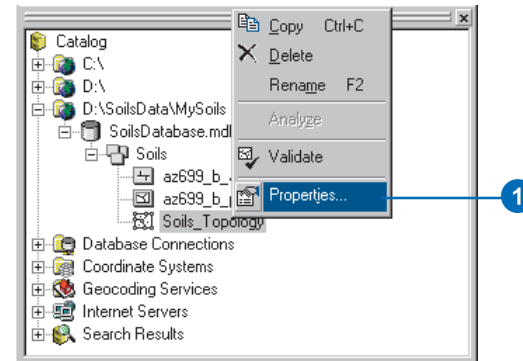
For more information on exclusive locks and schema locking, see the chapter 'Creating new items in a geodatabase' in this book.

## Modifying a topology

You can change the properties of a topology not registered as versioned. In some cases, such as when renaming a topology, the change has no effect upon the state of the topology. In other cases, the change may require that the topology be validated again. Some changes, such as adding new feature classes or new rules or changing the cluster tolerance, may create new dirty areas, new error features, and necessitate that features undergo cracking and clustering.

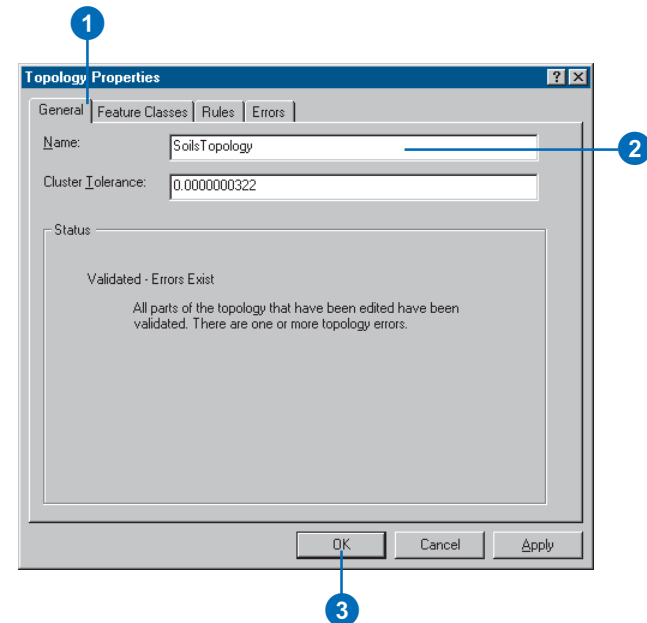
### Getting the properties of a topology

1. Right-click the topology and click Properties.



### Renaming a topology

1. Click the General tab on the Topology Properties dialog box.
2. Click in the Name text box and type a new name.
3. Click OK.



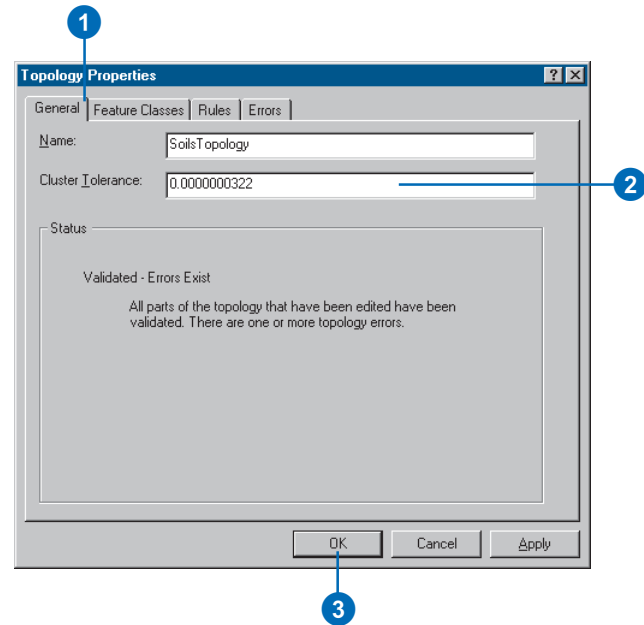
## Tip

### Changing the cluster tolerance

*Changing the cluster tolerance of a topology will require the topology to be validated again. The larger the cluster tolerance, the greater the likelihood that features in your data will be moved from their current positions or have their shape changed.*

### Changing the cluster tolerance of a topology

1. Click the General tab on the Topology Properties dialog box.
2. Click in the Cluster Tolerance text box and type a new cluster tolerance.
3. Click OK.



## Tip

### Adding a feature class

*Adding a feature class will require the topology to be validated again.*

*Before you validate the topology, you may want to assign topology rules to the new feature class.*

## Adding a feature class to a topology

1. Click the Feature Classes tab on the Topology Properties dialog box.

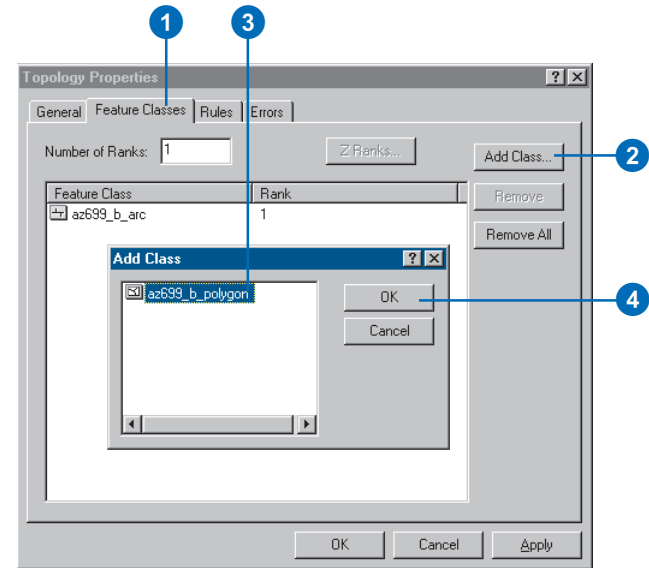
2. Click Add Class.

You see a list of the simple feature classes in the dataset that do not yet participate in a topology.

3. Click a class that you want to add.

4. Click OK.

You will need to add topology rules for this feature class.



## Tip

### Removing a feature class

*Removing a feature class also removes all of the topology rules associated with that feature class.*

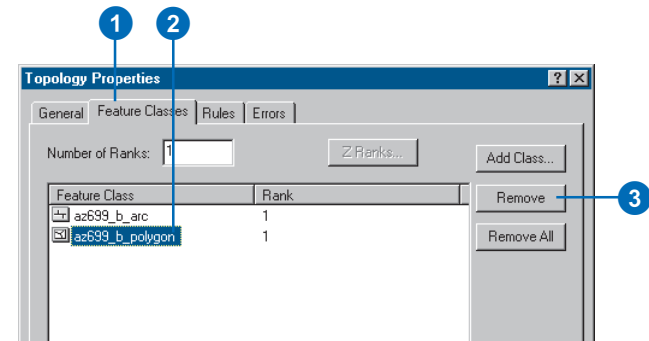
*Removing a feature class will require the topology to be validated again.*

## Removing a feature class from a topology

1. Click the Feature Classes tab on the Topology Properties dialog box.

2. Click the feature class you want to remove.

3. Click Remove.



## Tip

### Changing the number of ranks

*Changing the number of ranks will not require the topology to be validated again.*

## Changing the number of ranks in a topology

1. Click the Feature Classes tab on the Topology Properties dialog box.
2. Click the Number of Ranks textbox and type a number of ranks.

A topology can support up to 50 ranks to which feature classes may be assigned.



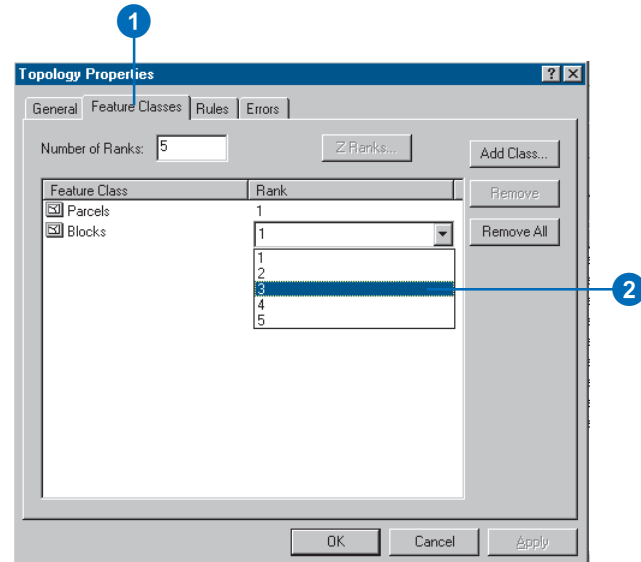
## Tip

### Changing the rank of a feature class

*Changing the rank of a feature class will require the topology to be validated again.*

## Changing the rank of a feature class in a topology

1. Click the Feature Classes tab on the Topology Properties dialog box.
2. Click the current rank of the feature class.



## Tip

### Adding a rule

*Adding a rule will require the topology to be validated again.*

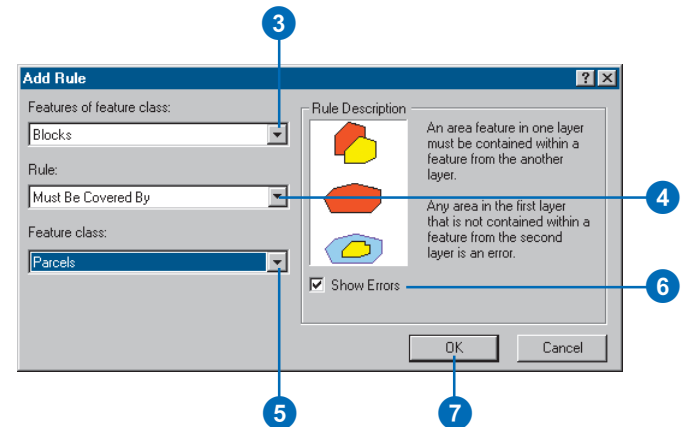
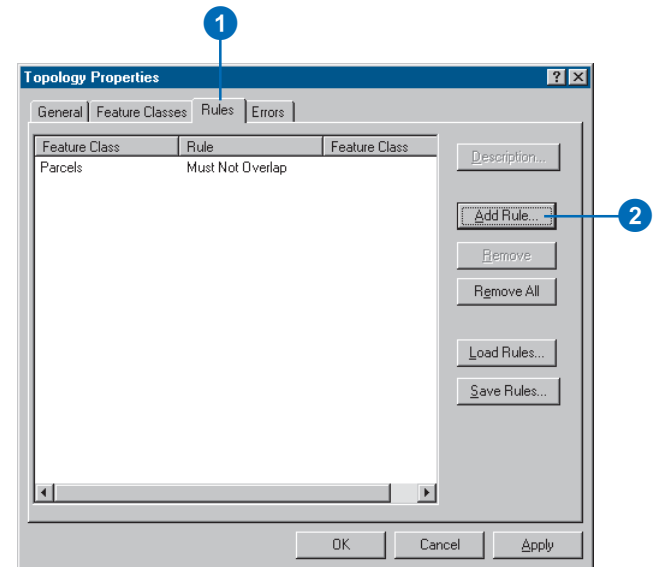
## Tip

### Adding feature classes before adding rules

*You must add feature classes to the topology before you can specify rules.*

## Adding a rule to a topology

1. Click the Rules tab on the Topology Properties dialog box.
2. Click Add Rule.
3. Click the dropdown list to select the feature class or subtype that the rule will apply to.
4. Click the dropdown list to select the rule that you want to apply.
5. Click the dropdown list to select the feature class or subtype if the rule involves a topological relationship with another feature class.
6. Optionally, check the Show Errors textbox to see simplified graphics of what constitutes a violation of this rule.
7. Click OK.



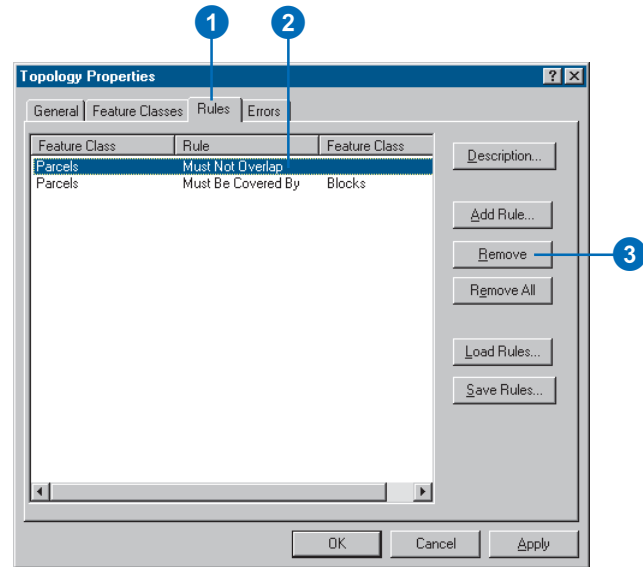
## Tip

### Removing a rule

*Removing a rule will require the topology to be validated again.*

## Removing a rule from a topology

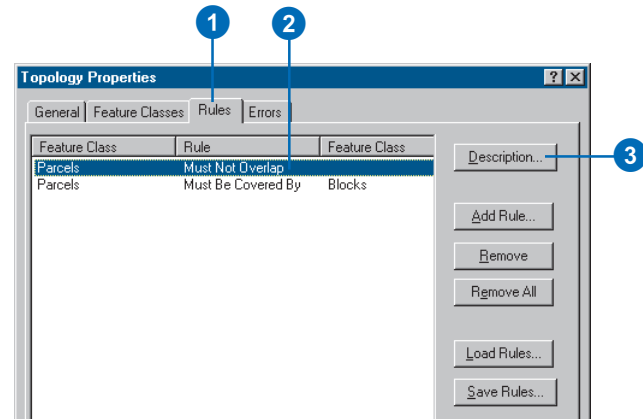
1. Click the Rules tab on the Topology Properties dialog box.
2. Click the rule that you want to remove.
3. Click Remove.



## Getting the description of a topology rule

1. Click the Rules tab on the Topology Properties dialog box.
2. Click the rule that you want to see a description of.
3. Click Description.

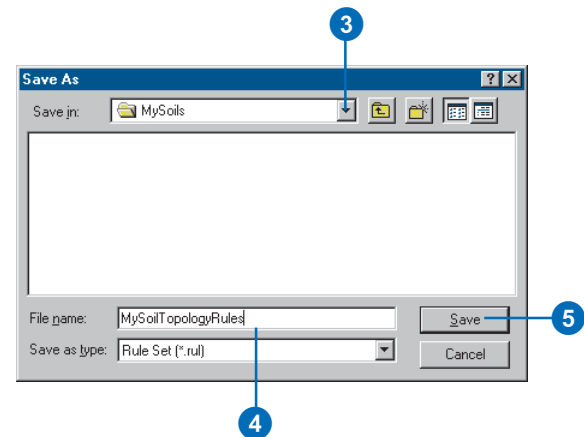
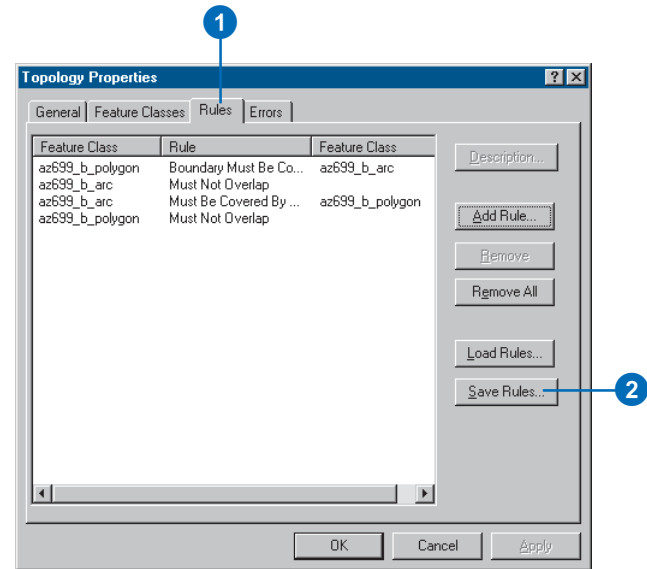
You can also view the description of a topology rule for which errors have been detected using the Show Rule Description item in the Fix Topology Error context menu.





## Saving rules for a topology into a Rule Set file

1. Click the Rules tab on the Topology Properties dialog box.
2. Click Save Rules.
3. Navigate to the place where you want to save the rules that you've defined for the topology.
4. Type a name for the Rule Set file.
5. Click Save to save all the rules for the topology to the file.



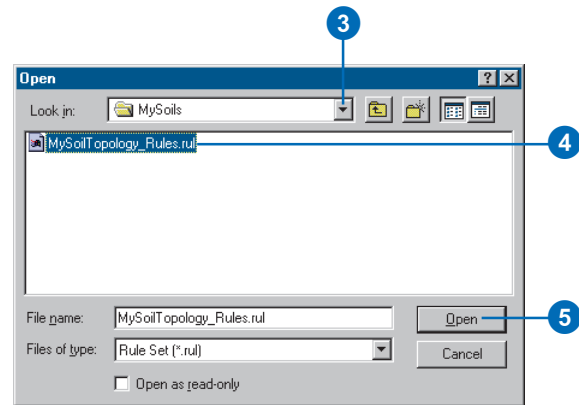
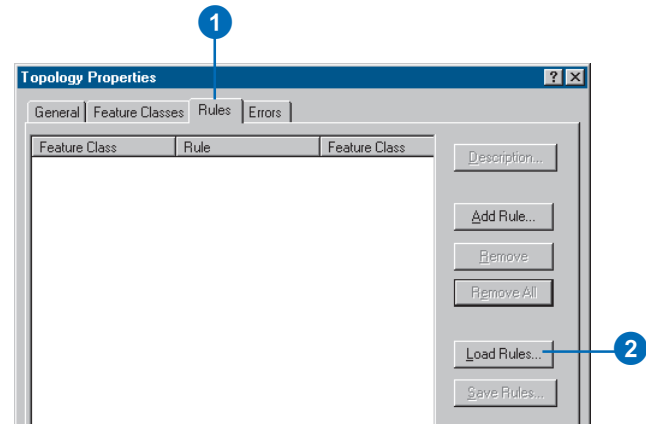
## Tip

### Loading rules

*Loading a Rule Set doesn't remove existing rules. Once you've loaded a rule set be sure to validate the topology.*

## Loading rules for a topology from a Rule Set file

1. Click the Rules tab on the Topology Properties dialog box.
2. Click Load Rules.
3. Navigate to the place where the Rule Set that you want to load is saved.
4. Click the Rule Set.
5. Click Open. ▶



## Tip

### Loading a rule set when some feature classes cannot be matched

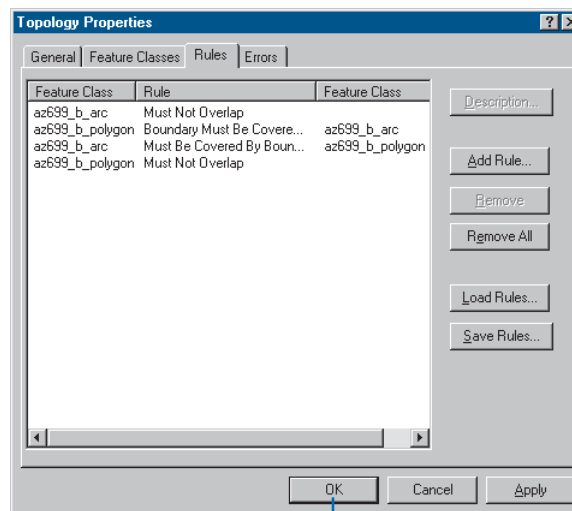
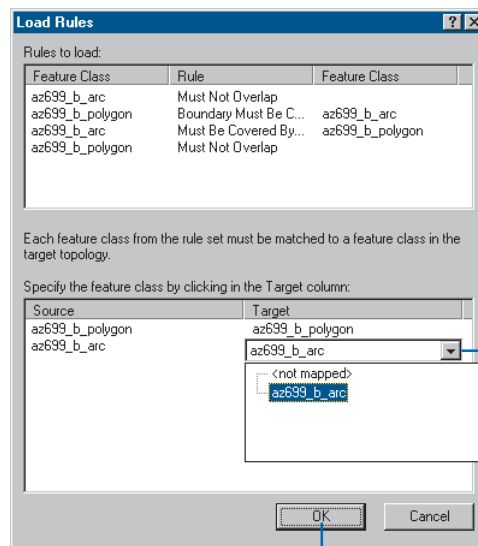
*If there are feature classes specified in a rule set that cannot be matched to feature classes in the new topology, rules involving the unmatched feature classes will not be loaded.*

The Load Rules feature class mapping dialog box appears.

If the rule set was created from a topology that had the same feature class names as the feature classes in the new topology you're defining, the feature classes named in the rule set should be correctly matched to the feature classes in the new topology.

If the names are different, you will need to match the feature classes mentioned in the rule set to their corresponding feature classes in the new topology.

6. Click in the Target column and click the feature class that it corresponds to in the new topology for each Source feature class that is not mapped.
7. Click OK.
8. Click OK on the Topology Properties dialog box.

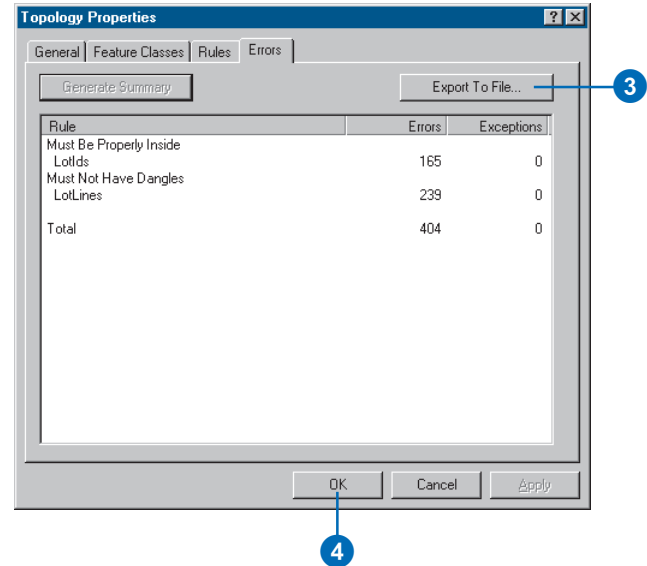
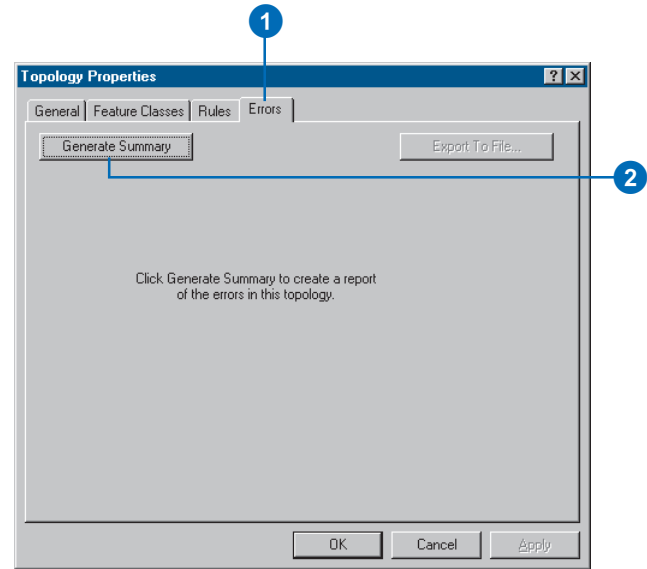


## Summarizing topology errors

You can view a summary of the number of errors in a topology from the Topology Properties dialog box. The summary tells you how many errors and exceptions exist for each of the topology rules.

You can save the summary as a text file to create a record of the state of the topology at a given time. This can be a useful way to document and monitor progress on a large topology editing project.

1. Click the Errors tab on the Topology Properties dialog box.
2. Click Generate Summary.
3. Optionally, click Export To File if you want to save the contents of the summary to a text file.
4. Click OK.



## Creating new polygons from lines

Sometimes you need to create polygon features from line feature data. For example, you might have digitized the boundaries of a set of features into a line feature class, or you may have only been able to obtain line features from a data provider.

The Polygon Feature Class From Lines tool lets you create new polygon features from line and polygon features in one or more feature classes within a given dataset in a geodatabase. You can specify a point feature class that will supply attributes for the new polygon features.

### Tip

#### Make sure your line work forms closed areas

The Create Polygon feature class tool creates polygons from areas that are completely enclosed by lines or polygon edges. If there are gaps in the line work defining a polygon's boundary, a polygon will not be created. Small gaps can be spanned by increasing the cluster tolerance.

1. Right-click the dataset, point to New, and click Polygon Feature Class From Lines in ArcCatalog.
2. Type a name for the new polygon feature class.
3. Optionally, change the cluster tolerance.

You can increase the cluster tolerance to span small gaps in the line work, but a large cluster tolerance may create polygons that you do not want. It is generally best to clean up the line work before creating polygons.

4. Check the feature classes that you want to be considered when finding closed areas.

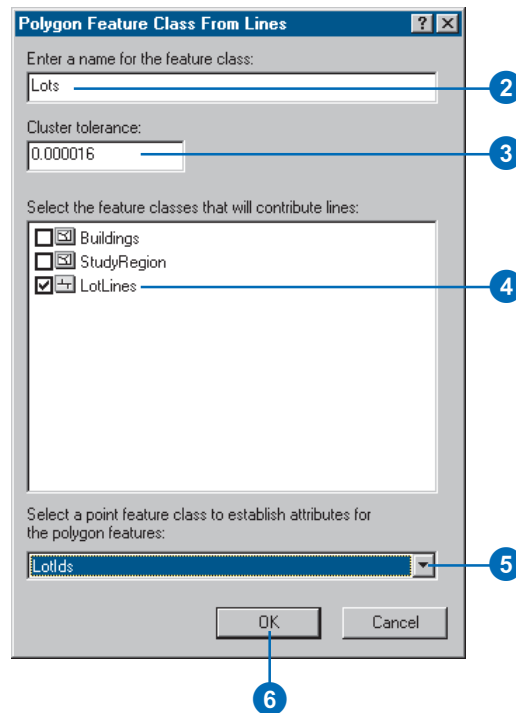
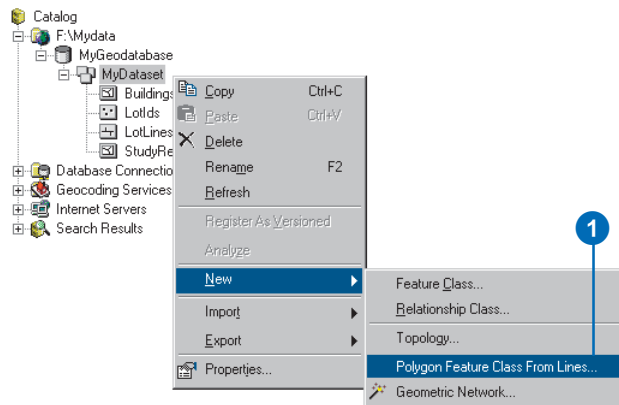
If you check more than one feature class, then areas that are enclosed by lines from any of the feature classes will become polygon features.

If you check a polygon feature class, the boundaries of the polygons will be considered as lines.

5. Optionally, select a point feature class that will be used to assign attributes to the new polygons.

Polygons that enclose a point will be given the attributes of the point.

6. Click OK. ▶



## Tip

### Creating polygons from lines in ArcMap

You can also use the *Construct Features* tool while editing in ArcMap to create new features in a target feature class from selected features. The *Construct Features* tool is on the *Topology* toolbar and works when you are editing data with map topology or a geodatabase topology.

The new feature class is created in the feature dataset.



## Topology and versioned databases

You can use geodatabase topologies in a versioned geodatabase but you must have appropriate permission to edit the geodatabase.

You can only create a topology and edit the properties of a topology in a nonversioned dataset. After the topology has been created, you can register the dataset as versioned and work with the topology in any version.

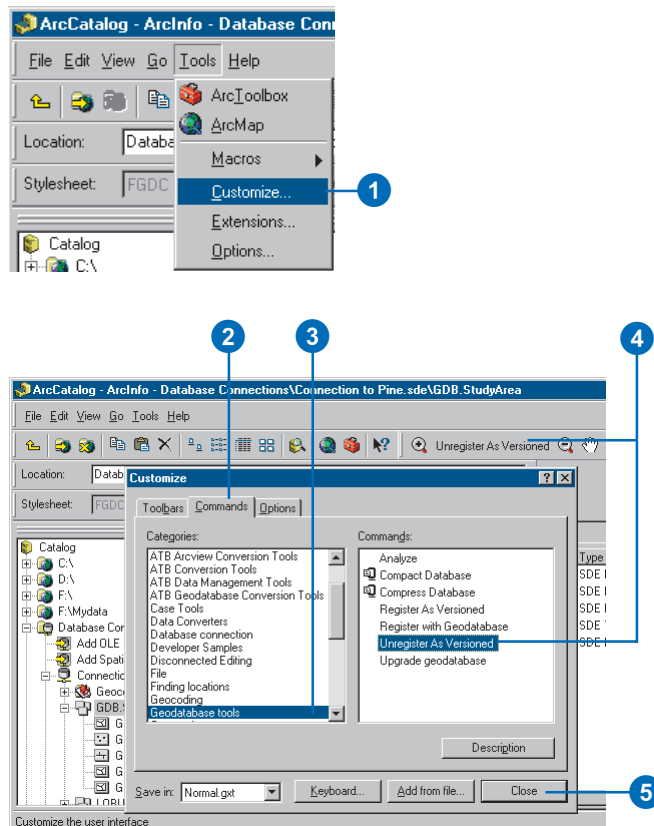
If you want to add a topology to a versioned dataset, you will first need to unregister the dataset as versioned. To unregister the dataset as versioned, you may need to add the Unregister As Versioned command to ArcCatalog.

For more information about how versioned data behaves with topology, see the section 'Topology and versioning' in this chapter.

### Creating topology in a versioned database

If you have edits in existing versions, they will be lost when you unregister the data as versioned. To preserve the edits, you must compress the database before you unregister it as versioned.

1. Click Tools and click Customize.
2. Click the Commands tab.
3. Click Geodatabase tools.
4. Click and drag Unregister As Versioned onto a toolbar.
5. Click Close.
6. Reconcile and post each outstanding version in the database against the DEFAULT version. After posting, delete each version.
7. Run compress to compress the database. ►

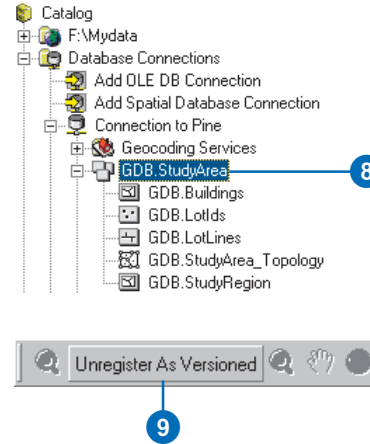


## Tip

### Removing a customization

After you've added a new command or tool to the interface and used it, you can remove it by opening the Customize dialog box and dragging the command or tool from the interface back onto the Customize dialog box.

8. Click the dataset.
9. Click Unregister As Versioned.
10. Unregister the data as versioned. Note: If you have not completed steps 6 and 7 before unregistering your data as versioned, then you will lose any edits that those versions contain. A warning dialog box will appear informing you that outstanding edits still remain in existing versions.
11. Create the topology.
12. Register the dataset as versioned.





# Topology and versioning

It is helpful to understand how the versioning process works in geodatabases before planning your work flow. For more information on how versioning works in a geodatabase, see the chapter ‘Working with a versioned geodatabase’ in this book.

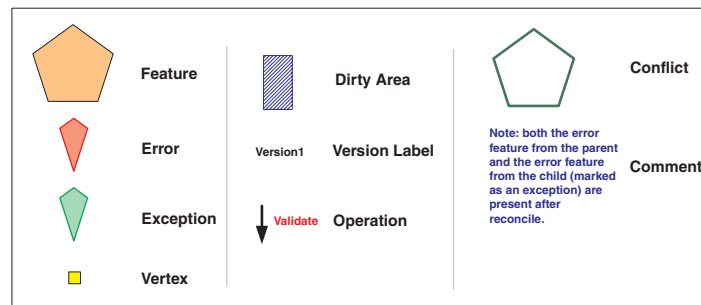
In a versioned geodatabase, feature classes that participate in a topology do not have any special version reconciliation or *conflict* detection and resolution behavior. However, the dirty areas, error features, and exceptions that are maintained by the topology itself do have special behavior in the version reconciliation and conflict detection process to ensure the integrity of the topology.

The nature of topological edits and the cracking and clustering modifications that the validate process makes to feature geometry may result in conflicts during version reconciliation.

You should consider the behavior of dirty areas, error features, and the types of conflicts that may result from topological edits when you are planning your work flow for managing versioned topological feature classes.

The following sections describe the results of a reconcile on dirty areas, errors, exceptions, and potential conflicts. In each case, the results are based on a reconcile in which a parent and child version have both been edited since the child version was created. If the parent version is not edited before the child version is reconciled, the results of the reconcile will be the contents of the child version. In each example, Version2 is created as a child of Version1. Both versions are then edited in the

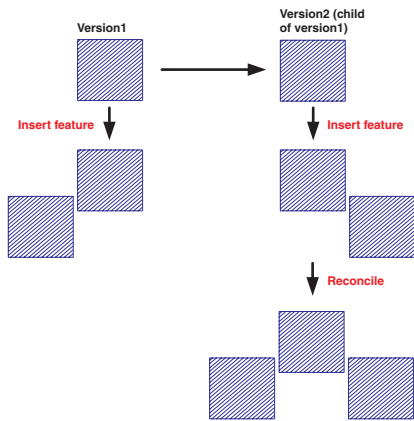
manner described in the example; then Version2 is reconciled against Version1. For the illustrations in the following examples, use the following as a legend:



## Dirty areas

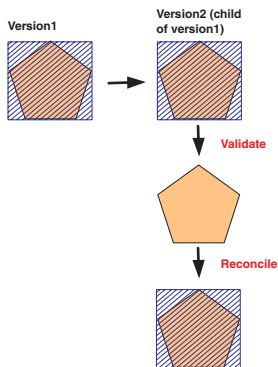
New topology errors may occur when edited parent and child versions are reconciled, even when the dirty areas within each version have been validated and are free of errors. In order to detect such topology errors, dirty areas are treated in a special way by the reconcile process. Results of the reconcile on dirty areas can be summarized as:

- Any dirty areas present in the parent or child version that did not exist before the parent and child version were created will remain dirty as a result of the reconcile:



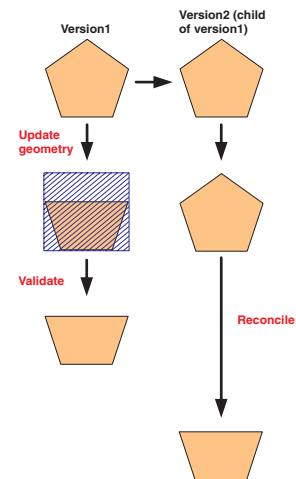
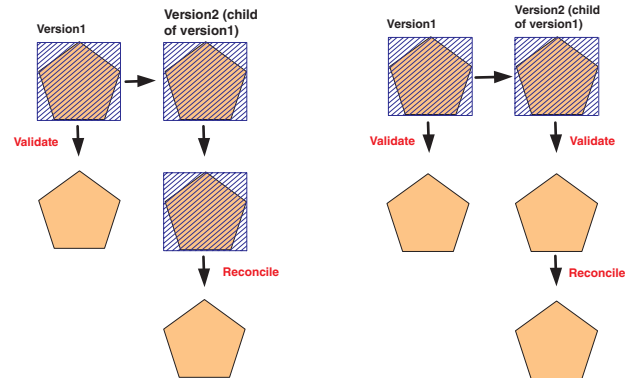
*Any dirty areas created in either the parent or child version will remain dirty as a result of the reconcile.*

- Any dirty area that was present in the parent version and validated in the child version will become dirty as a result of the reconcile:



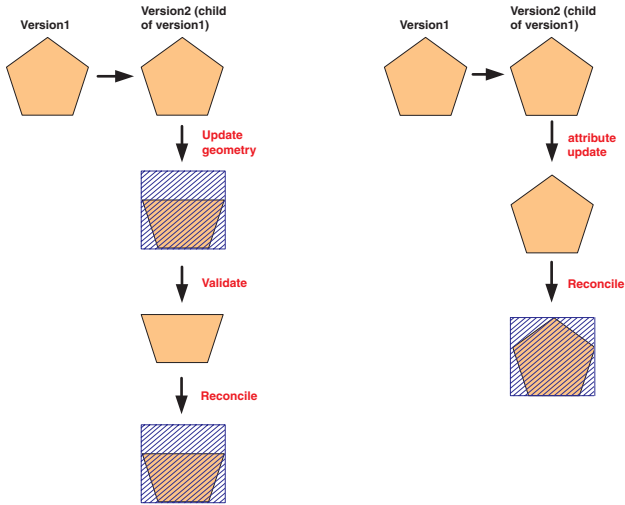
*A dirty area in the parent version, validated in the child version, will become dirty as a result of the reconcile.*

- Any dirty area validated in the parent version, whether it is validated in the child version, will remain validated as a result of the reconcile:



*Dirty areas introduced and validated in the parent version remain validated as a result of the reconcile.*

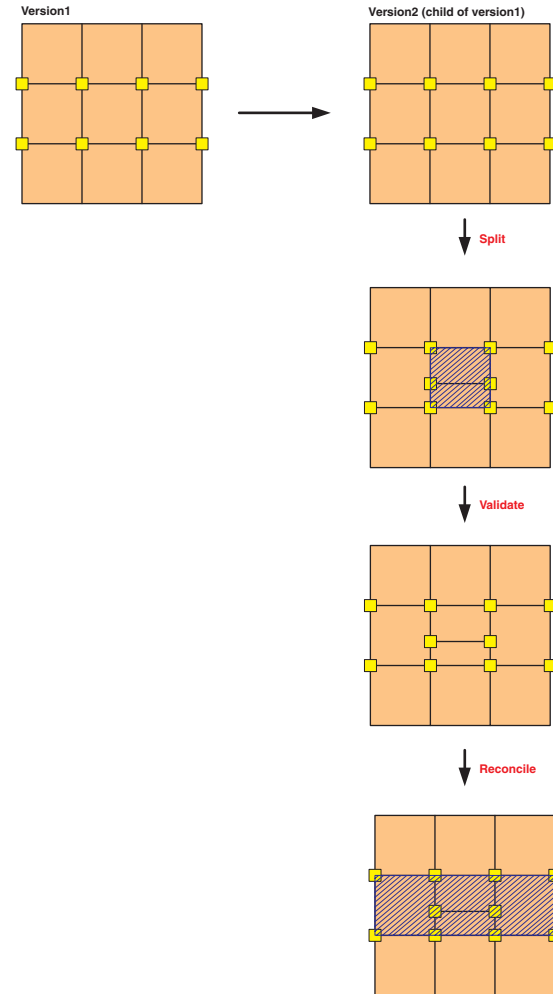
- Any edits made to topology features in the child version will result in a dirty area on the reconcile, even if the dirty area resulting from the edit is validated in the child version. This is also the case when the original edit did not result in a dirty area (e.g., an attribute update):



*Edits made to topology features in the child version result in dirty areas after the reconcile.*

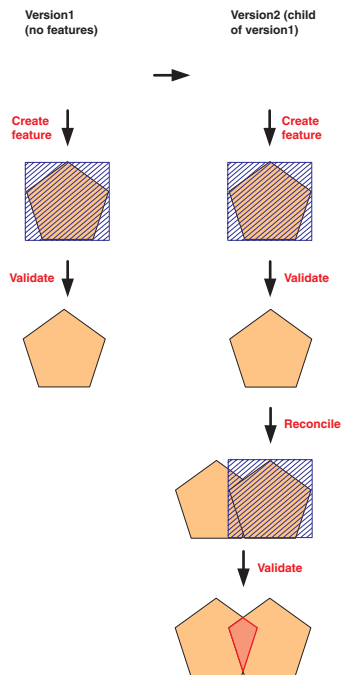
There are a number of scenarios where a reconcile can result in new dirty areas that did not exist in the parent or child, because of cracking and clustering during the validate process. In the following example, both versions contain polygons that share edges in a topology. A polygon is split in the child version and the dirty area is validated. Splitting the polygon deletes the original feature and replaces it with two new ones. When the dirty area is validated, cracking and clustering introduces new vertices into the shared edges of the adjacent polygons. When the versions are reconciled, all of the features that have been modified in the child version—the split polygons and the

polygons modified by cracking and clustering—are covered by dirty areas:



*Insertion of new vertices during cracking and clustering may result in additional dirty areas on the reconcile.*

The following examples illustrate why this is necessary. In the following example, new features were created in each version, and the resulting dirty areas were validated, resulting in no errors. On a reconcile, dirty areas must be re-created such that errors introduced by merging the changes from the two versions together can be discovered. In this example, features added in Version1 and Version2 overlap each other, which is a violation of the “no overlaps” rule:



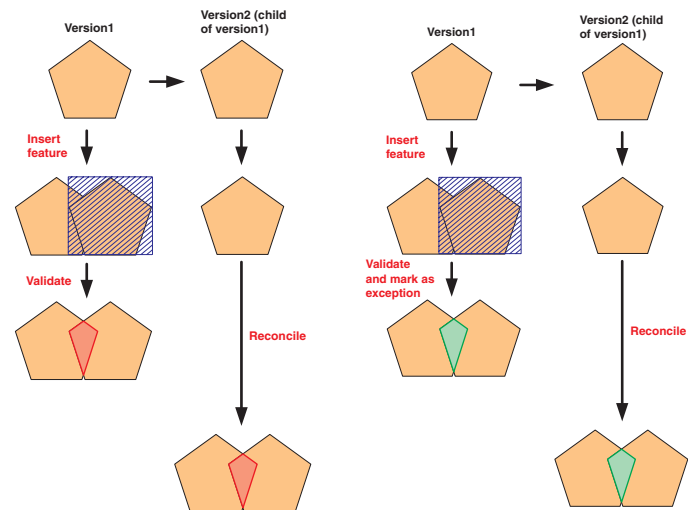
*It is necessary to reactivate dirty areas from the child version after a reconcile to discover errors that may result from additional edits made in the parent version.*

## Errors and exceptions

Error features and error features marked as exceptions have special logic with respect to how they are treated during the version reconciliation process. Since errors and exceptions cannot be edited directly, the system will not report conflicts for them between two versions. Errors are only created by the topology validation process and can be deleted by correcting the error using the topology error correction tools in ArcMap or by editing features and using the validation process. Error features can only be updated by marking an error as an exception or by marking an exception as an error.

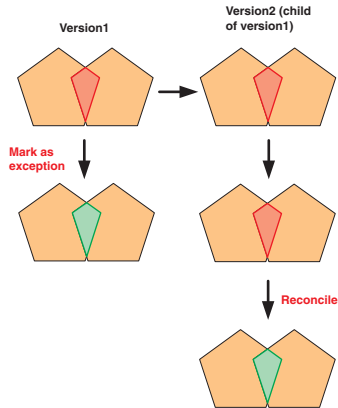
The results of a reconcile on errors and exceptions in the parent version can be summarized as:

- Any error created in the parent version, whether or not it is marked as an exception, will be brought into the child version as a result of the reconcile:



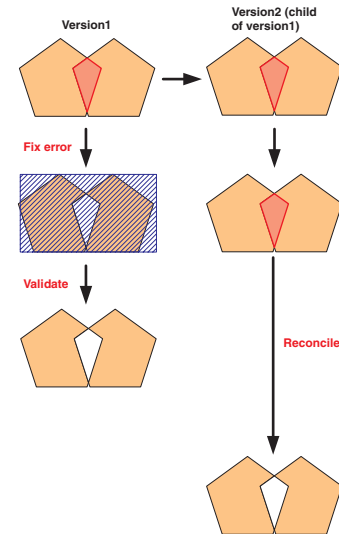
*Errors and exceptions created in the parent version are brought into the child version as a result of the reconcile.*

- Any pre-existing error marked as an exception in the parent version will be marked as an exception in the child version after the reconcile:



*Errors marked as exceptions in the parent version remain exceptions after the reconcile.*

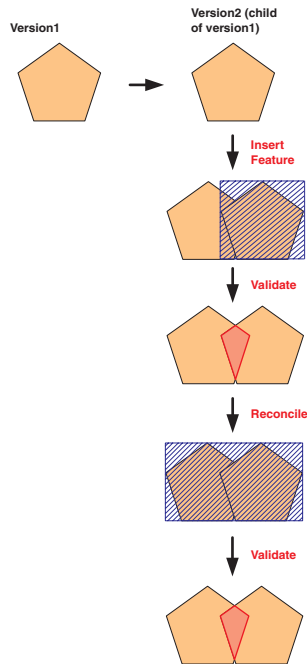
- Any error or exception that is deleted in the parent version—either by fixing the error or by the validation process—will be deleted from the child version as a result of the reconcile.



*Errors and exceptions deleted in the parent version remain deleted in the child version as a result of the reconcile.*

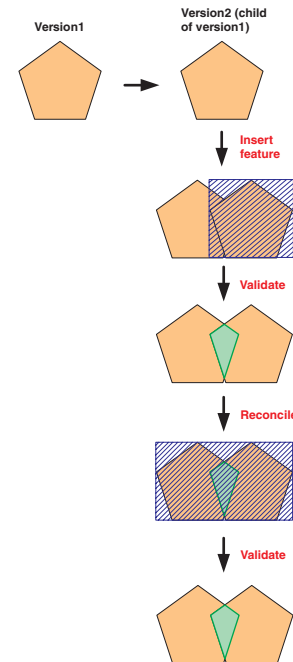
The results of the reconcile on errors and exceptions in the child version can be summarized as:

- Any error created in the child version will be deleted as a result of the reconcile and, by definition, will be covered by a dirty area (see the section 'Dirty areas' in this chapter). The error can then be rediscovered by validating the dirty area:



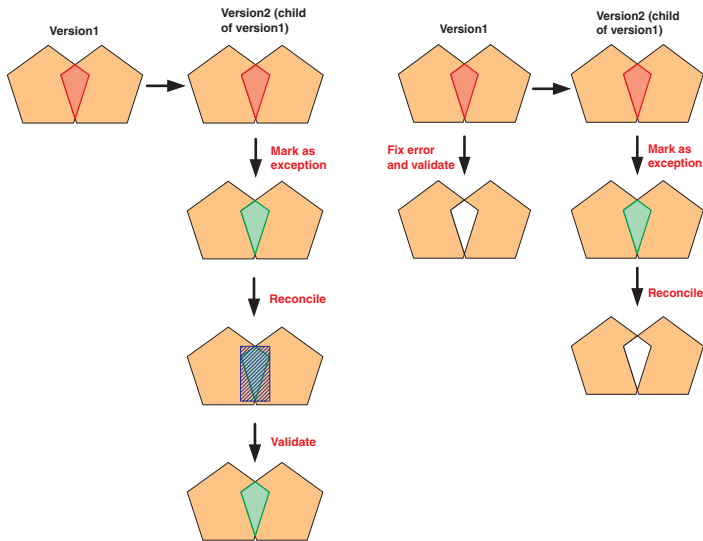
*Errors created in the child version are deleted as a result of the reconcile.*

- Any error created in the child version and marked as an exception will remain an exception as a result of the reconcile. By definition, it will be covered by a dirty area:



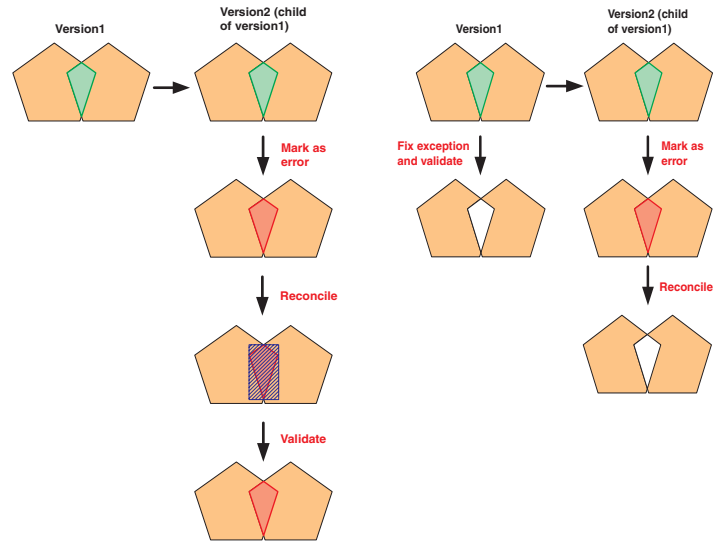
*Errors created in the child version and marked as exceptions remain as a result of the reconcile and are always covered by a dirty area.*

- An error that existed in the parent version and is marked as an exception in the child version will remain an exception as a result of the reconcile and will be covered by a dirty area. However, if the error was fixed in the parent version, then it will remain fixed in the child version:



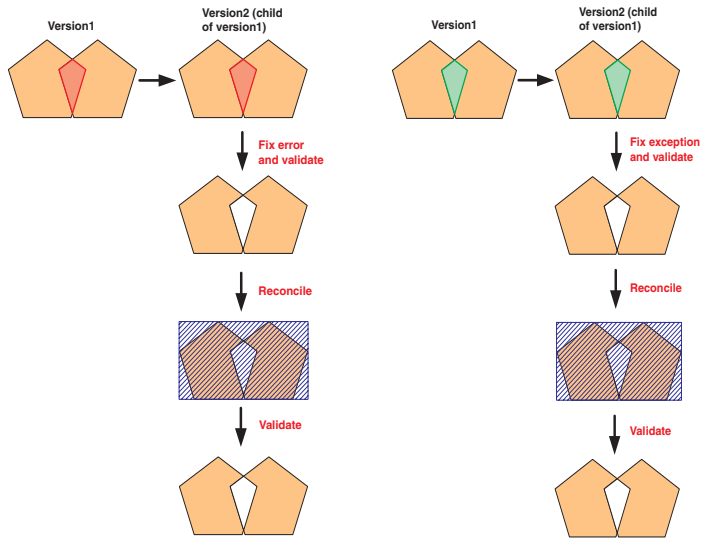
*Errors that existed in the parent version and are marked as exceptions in the child version will remain as a result of the reconcile but will be covered by a dirty area. If the same error is fixed in the parent version, it will be fixed in the child version as a result of the reconcile.*

- An exception that existed in the parent version and is marked as an error in the child version will remain an error as a result of the reconcile and will be covered by a dirty area. However, if the exception was fixed in the parent version, then it will remain fixed in the child version:



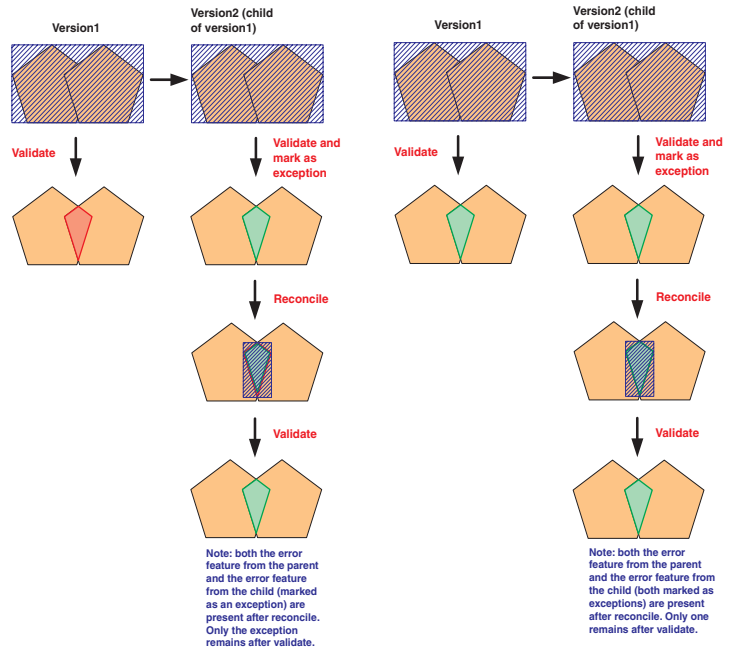
*Exceptions that existed in the parent version and are marked as errors in the child version will remain as a result of the reconcile but will be covered by a dirty area. If the same exception is fixed in the parent version, it will be fixed in the child version as a result of the reconcile.*

- An error or exception that existed in the parent version and is fixed in the child version will remain fixed as a result of the reconcile:



*Errors and exceptions that existed in the parent version and are fixed in the child version will remain fixed as a result of the reconcile.*

There are cases in which the same error can be created in both versions by validating a dirty area that existed in the parent version when the child version was created. If this error is marked as an exception in either the parent or child version, the reconcile will result in duplicate error features. In these cases, the error features will be covered by a dirty area and will be reduced to a single error or exception when the dirty area is validated.



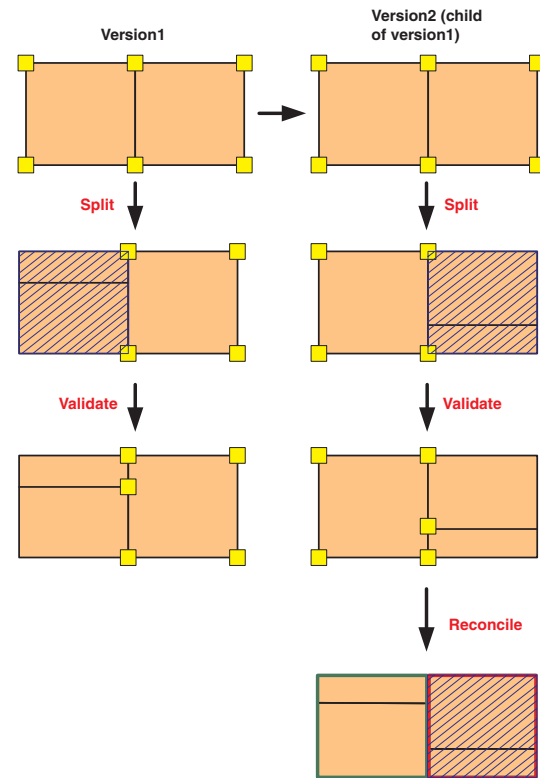
*There are cases in which the same error can be discovered in both the parent and child. If the error is marked as an exception in the parent version, child version, or both, there will be duplicate errors as a result of the reconcile. Validating the resulting dirty area will eliminate the duplicate.*



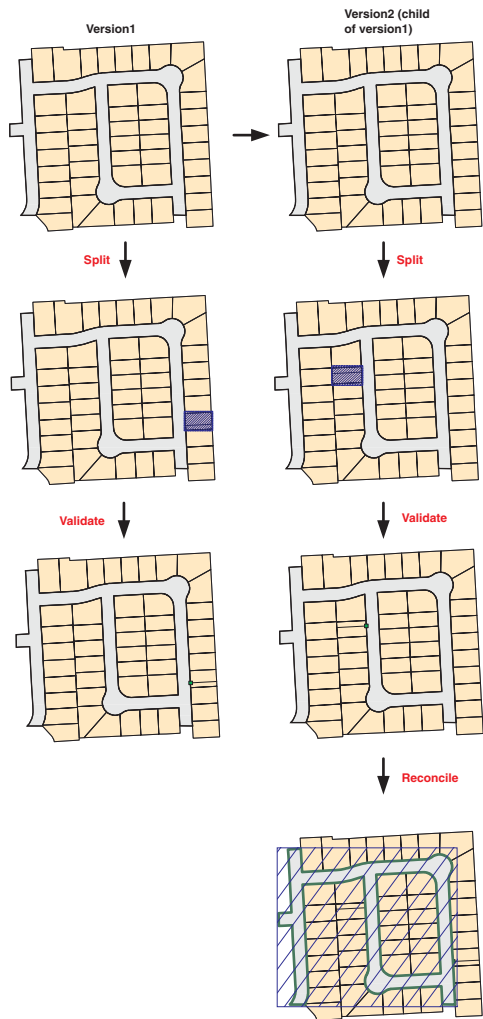
## Conflicts and topology features

Features that participate in topology do not have any special behavior with respect to conflicts resulting from version reconciliation. If the same feature is edited in two separate versions and those versions are reconciled, they will be in conflict. The type of edits that are made to topologies, and the effect of clustering and cracking during the validation process, can result in a large number of conflicts. It is important to understand the potential for conflicts resulting from topological edits when you design your data model and your work flow.

The most common source of conflicts resulting from the validation process with topologies is the introduction of vertices due to the integration of features during the cracking and clustering phase of validation. The following examples illustrate how conflicts can result from the validate process.



*Polygons that share edges in a topology in the parent version are inherited by the child version. One polygon is split in the parent version, an adjacent polygon is split in the child version, and the dirty areas are validated. Splitting the polygons deletes the original features and replaces each of them with two new ones. When the versions are reconciled, both original polygons are reported as update–delete conflicts. In other words, the feature that was deleted in the parent version was updated by the cracking and clustering process in the child version, and the feature that was deleted in the child version was updated by the cracking and clustering process in the parent version.*



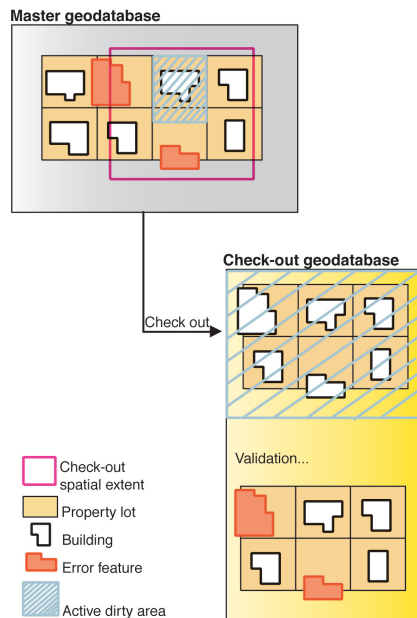
Conflicts like the ones described here can be avoided by structuring your work flow so that editors are working in different areas, or by using disconnected editing to control where users can and cannot edit. Additionally, data model design can help reduce the types of conflicts illustrated in the second example. These types of conflicts can be reduced by subdividing the right-of-way polygon into smaller chunks.

*Another example where the introduction of vertices by the validate process can introduce conflicts in a land records database. In this case, the parcel polygons that are split share edges with a large right-of-way polygon.*

# Topology and disconnected editing

As in the case of versioned databases, features in feature classes that participate in a topology do not have any special behavior for disconnected editing.

When a topology is checked out of a master geodatabase, all of the feature classes that participate in the topology are checked out as well. The entire extent of the checked out data is marked as dirty in the check-out geodatabase. An editor of the data in the check-out geodatabase can then validate the dirty area, make edits, and find and correct errors, or mark errors as exceptions.



When the edited data is checked back into the master geodatabase, all of the feature classes are checked back in, and any differences between the checked-in data and the original version in the master are marked as dirty areas. These dirty areas

in the master geodatabase can then be validated to identify topology errors. Errors are not checked in to the master; they must be discovered by validating the checked-in area.

For more information about check-out geodatabases and disconnected editing, see the chapter 'Disconnected editing' in this book.





# Subtypes and attribute domains

# 5

## IN THIS CHAPTER

- **What are subtypes and attribute domains?**
- **Working with attribute domain properties**
- **Browsing the attribute domains of a geodatabase**
- **Creating new attribute domains**
- **Modifying and deleting attribute domains**
- **Associating default values and domains with tables and feature classes**
- **Creating subtypes**
- **Modifying and deleting subtypes**

When maintaining your geographic database, care must be taken to ensure that when you edit the data you do so in a manner that is consistent with the system you are modeling. The geodatabase together with the ArcMap Editor provides mechanisms to ensure that the data you store in your geodatabase is consistent with your data model.

The geodatabase has several data integrity and data management capabilities, including validation rules, subtypes, relationship classes, geometric networks, and so on. Each of these capabilities and how you use them are covered throughout this book. This chapter describes the subtypes and the first class of validation rules—attribute domains.

Subtypes can be used to add finer control of spatial relationships in topologies and in connectivity rules for geometric networks. See the chapter ‘Topology’ in this book for more information about using subtypes in topologies. See the chapter ‘Geometric networks’ in this book for more information about using subtypes in geometric networks.

You can add and modify subtypes and attribute domains with any license, including ArcView.

# What are subtypes and attribute domains?

The geodatabase stores objects. These objects may represent nonspatial real-world entities, such as manufacturers, or they may represent spatial objects such as pipes in a water network. Objects in the geodatabase are stored in feature classes (spatial) and tables (nonspatial).

The objects stored in a feature class or table may be organized into subtypes and can have a set of validation rules associated with them. The ArcInfo system uses these validation rules to help you maintain a geodatabase that contains valid objects. This chapter outlines how to create subtypes for your feature classes and tables and how to establish attribute validation rules for the objects stored in them. These validation rules are distinct from the validation rules established in a topology.

## Subtypes and validation rules

Tables and feature classes store objects of the same type—that is, that have the same behavior and attributes. For example, a feature class called *WaterMains* may store pressurized water mains. All of the water mains behave alike and have the attributes *ReferenceID*, *Depth*, *Material*, *GroundSurfaceType*, *Size*, and *PressureRating*.

You may be modeling a system in which water mains can only be made of cast iron, ductile iron, or copper. They can only be a certain size based on their type, and there are four possible ground surface types. When you create a new water main object using ArcMap, you may want these attributes to take on certain default values. Similarly, when ArcMap changes values for an attribute, you may want to make sure that only legal or valid values are inserted into the attributes for that lateral.

When an object in a feature class or table has valid values for all of its attributes, it is considered to be a valid object. If one of its attributes contains an invalid value, it is considered to be an invalid object. When designing your geodatabase, you can

specify what makes any particular object in a feature class or table a valid feature by establishing one or more validation rules.





In addition to topology rules, there are four broad classes of validation rules: attribute domains, connectivity rules, relationship rules, and custom rules. Connectivity rules are discussed further in the chapter ‘Geometric networks’ and relationship rules are discussed further in the chapter ‘Defining relationship classes’ in this book; custom rules are discussed further in *Exploring ArcObjects*. This chapter focuses on attribute domains.

Attribute domains are rules that describe the permissible values of a field type. Multiple feature classes and tables can share attribute domains stored in the database. This allows the water mains feature class to use the same domain for the ground surface type field as a feature class that stores water laterals.

Although all objects in a feature class or table must have the same behavior and attributes, not all objects must share the same attribute domains. For example, it may be true in a water network that only transmission water mains can have a pressure between 40 and 100 psi, while distribution water mains can have a pressure between 50 and 75 psi. You would use an attribute domain to enforce this restriction. To implement this kind of validation rule, you do not have to create separate feature classes for transmission and distribution water mains, but you would want to distinguish these types of water mains from each other to establish a separate set of domains and default values. You can do this using subtypes.

Feature classes and tables can contain subtypes. An object’s subtype is determined by its subtype code value. The subtype code is stored in an integer field in the feature class or table. Each subtype can have its own set of default values and attribute domains for a given field, different connectivity rules, and different topology rules.



Geometry	Unique identifier	Attributes				Subtype code
Shape	ObjectID	ReferencID	GroundSurfaceType	Material	PressureRating	TypeCode
	1	M1111-TM	1	CI	25	1
	2	M1112-TM	1	CI	25	1
	3	M1111-DM	3	CO	10	2
	4	M1111-BM	2	CO	15	3

*Features in the geodatabase have behavior, geometry, a system-managed unique identifier, attributes, and can also have subtypes. Different subtypes can have different sets of valid values for their attributes.*

## When to use subtypes

An important geodatabase design issue arises when you must decide where it is appropriate to use subtypes and where additional feature classes are required. When you are trying to distinguish objects by their default values, attribute domains, connectivity rules, and relationship rules, it is recommended that you create separate subtypes for a single feature class or table.

You can also specify subtypes instead of feature classes in topology rules, which allows you to more precisely specify the spatial relationships that are appropriate for a given subtype.

When you want to distinguish objects based on different behaviors, attributes, access privileges, or whether the objects are multiversions, you must create additional feature classes.

## Attribute domains

Attribute domains are used to constrain the values allowed in any particular attribute for a table, feature class, or subtype. Each feature class or table has a set of attribute domains that apply to different attributes and/or subtypes. These attribute domains can be shared across feature classes and tables in a geodatabase.

There are two different types of attribute domains: range domains and coded value domains. Each domain has a name, a description, and a specific attribute type to which it can apply.

A range domain specifies a valid range of values for a numeric attribute. In the water mains example, you could have subtypes transmission, distribution, and bypass water mains. Distribution water mains can have a pressure between 50 and 75 psi. For a distribution water main object to be valid, its pressure value must be between 50 and 75 psi. A range domain specifies this range of values.

A coded value domain can apply to any type of attribute—text, numeric, date, and so on. Coded value domains specify a valid set of values for an attribute. The GroundSurfaceType field on the water mains feature class stores the type of material above the water main. Water mains may be buried under different types of surfaces: pavement, gravel, sand, or none (for exposed water mains). The coded value domain includes both the actual value that is stored in the database (for example, 1 for pavement) and a more user-friendly description of what that value actually means.

When editing your feature classes and tables, you can enforce these rules by validating individual or sets of objects. For details on editing objects with subtypes and validation rules, see *Editing in ArcMap*.

Attribute domains do not have a property that allows or disallows null values in an associated field. When a table or feature class is created in a geodatabase, each field has a property that indicates whether or not null values are permissible values. The database itself will not permit null values to be inserted into columns that do not support them. Therefore, all domains treat null values as valid values.

## Splitting and merging features

Often when editing data, a single feature is split into two features, or two separate features are combined, or merged, into a single feature. For example, in a land base database, a land parcel may be split into two separate land parcels due to rezoning. Similar zoning changes may require two adjacent parcels to be merged into a single parcel.

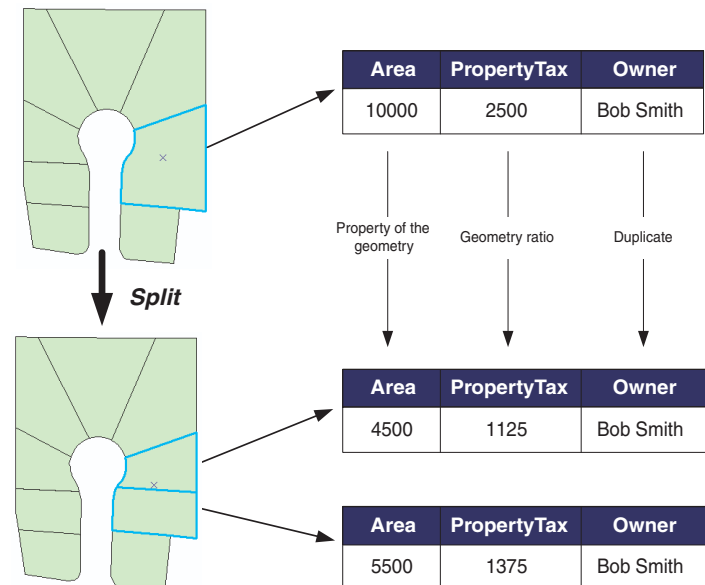
While the results of these types of edit operations on the feature's geometry are easily predictable, their effects on the attribute values are not. The behavior of an attribute's values when a feature is split is controlled by its *split policy*. When two features are merged, an attribute's value is controlled by its *merge policy*.

Each attribute domain has both a split policy and a merge policy. When a feature is split or merged, the ArcInfo system looks to these policies to determine what values the resulting feature(s) have for a particular attribute.

An attribute for any given table, feature class, or subtype can have one of three split policies that control the value of an attribute in the output object:

- **Default value:** the attribute of the two resulting features takes on the default value for the attribute of the given feature class or subtype.
- **Duplicate:** the attribute of the two resulting features takes on a copy of the original object's attribute value.
- **Geometry ratio:** the attribute of resulting features is a ratio of the original feature's value. The ratio is based on the ratio in which the original geometry is divided. If the geometry is divided equally, each new feature's attribute gets half of the value of the original object's attribute. Geometry ratio policies only apply to domains for numeric field types.

In the parcel example, when a parcel is split, the Area attribute is automatically assigned as a property of the resulting geometry. The value for Owner is copied to the new objects (in this database, splitting a parcel does not affect its ownership). The PropertyTax is calculated based on the area, or size, of a parcel. To calculate the PropertyTax for each of the new objects, the split policy divides the PropertyTax of the original parcel proportionally among the new features according to their area.



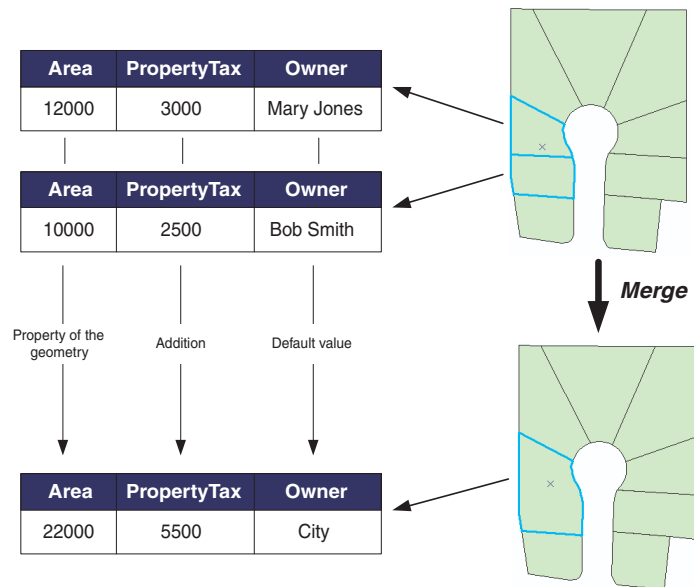
*This example shows how split policies can be applied to attributes of a parcel object.*



When two features are merged into a single feature, merge policies control the value of attributes in the new feature. An attribute for any given feature class or subtype can have one of three merge policies:

- **Default value:** the attribute of the resulting feature takes on the default value for the attribute of the given feature class or subtype. This is the only merge policy that applies to nonnumeric fields and coded value domains.
- **Sum values:** the attribute of the resulting feature takes on the sum of the values from the original features' attribute.
- **Geometry weighted:** the attribute of the resulting feature is the weighted average of the values of the attribute from the original features. This average is based on the original features' geometry.

In the parcel example, when two parcels are merged, the Area attribute is automatically assigned as a property of the resulting geometry. Owner is assigned its default value. As the PropertyTax for the merged feature is the sum of the original feature's PropertyTax, its merge policy is to sum the values.



*This example shows how merge policies can be applied to attributes of a Parcel object.*

This chapter shows you how to use ArcCatalog to create attribute domains for a geodatabase and how to create subtypes for a feature class or table. It then discusses how to create an attribute validation rule by associating an attribute domain to an attribute for a table, feature class, or subtype.

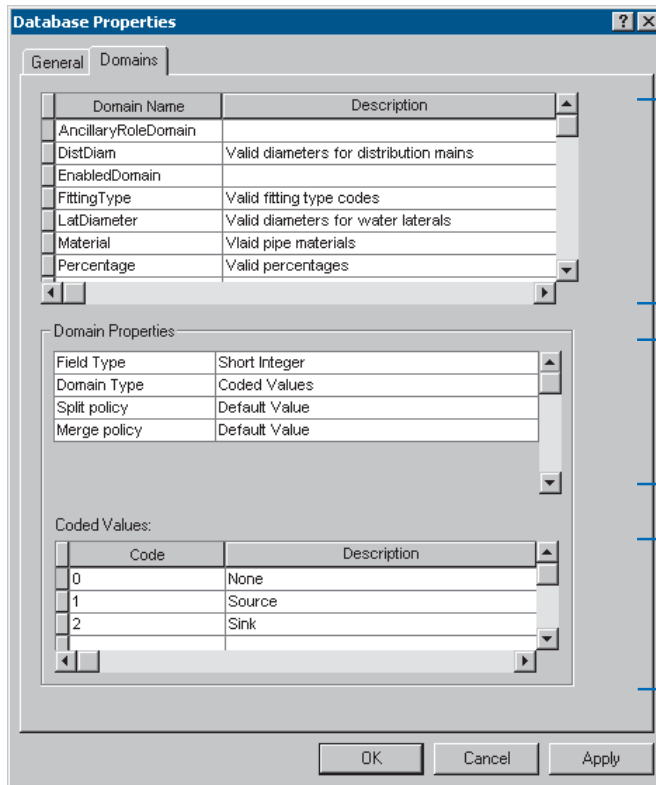
## Schema locking

An exclusive lock is required on a feature class or table to modify its subtypes, default values, and attribute domains. For more information on exclusive locks and schema locking, see the chapter 'Creating new items in a geodatabase' in this book.

# Working with attribute domain properties

The Domains tab of the Database Properties dialog box includes a list of all the domains that exist in a geodatabase. Each domain's name, description, properties, and valid set of values are displayed.

From this dialog box, you can also add, remove, and modify domains. An explanation of how to use this property page to manage your geodatabase's domains comes later in this chapter.



A list of the domains and their descriptions.

Displays the properties of the selected domain such as its field type, domain type, minimum and maximum values (for range domains), and the domain's split and merge policies.

If a coded value domain is selected, the list of valid values and their user-friendly descriptions is displayed.

## Browsing the attribute domains of a geodatabase

Attribute domains are stored geodatabase-wide. Once a user creates a new attribute domain, that user and all other users can view the properties of that domain and use the domain in a feature class or table.

Attribute domains are managed using the Domains tab on the Database Properties dialog box. This dialog box can be displayed as part of the geodatabase's properties or from the feature class or table properties dialog box.

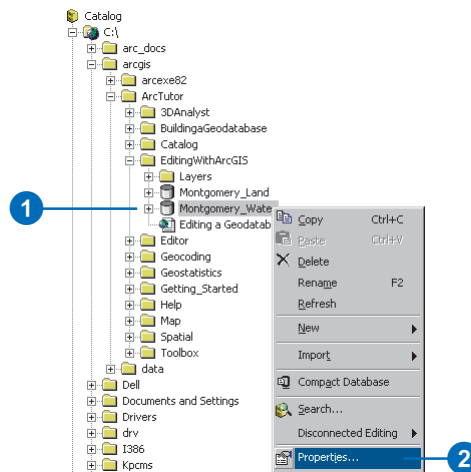
Attribute domains can be added, deleted, and modified with the domain properties dialog box.

## Browsing the domains of a personal geodatabase

1. Right-click the personal geodatabase in the ArcCatalog tree whose domains you want to browse.

2. Click Properties.

The Database Properties dialog box appears.

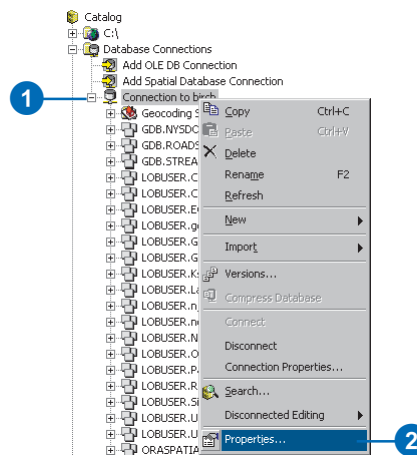


## Browsing the domains of an ArcSDE geodatabase

1. Right-click the ArcSDE connection, in the ArcCatalog tree, for the geodatabase whose domains you want to browse.

2. Click Properties.

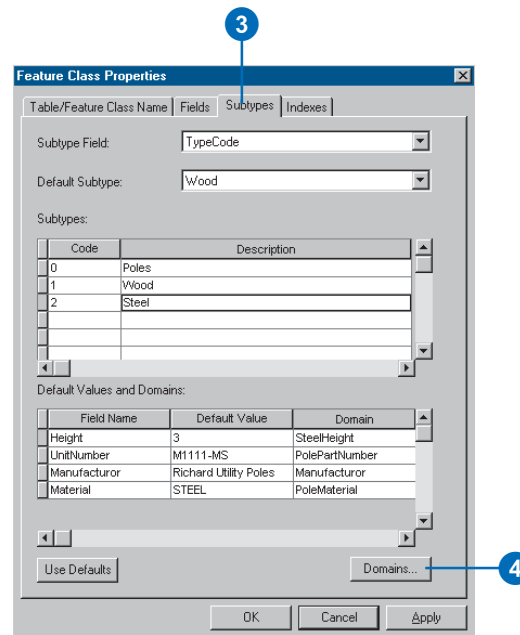
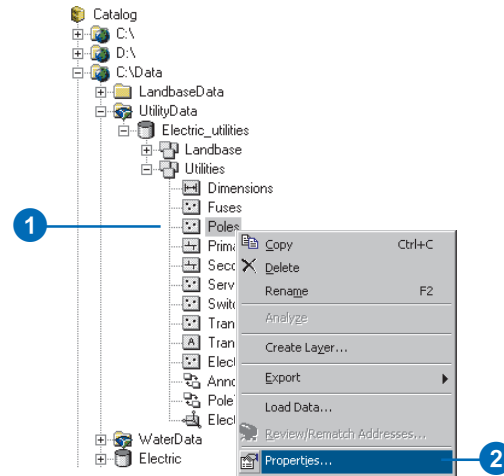
The Database Properties dialog box appears.



## Browsing the attribute domains of a geodatabase from a feature class or table

1. Right-click a feature class or table in the ArcCatalog tree.
2. Click Properties.
3. Click the Subtypes tab.
4. Click Domains.

The Database Properties dialog box appears.



## Creating new attribute domains

At any time in the life of a geodatabase, a new attribute domain can be created using the Domains tab on the Database Properties dialog box.

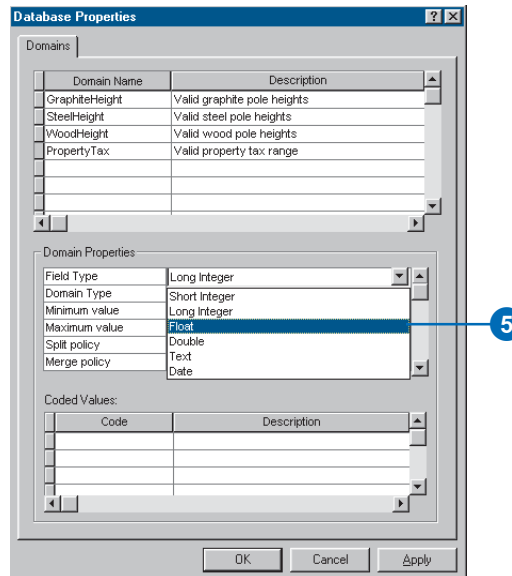
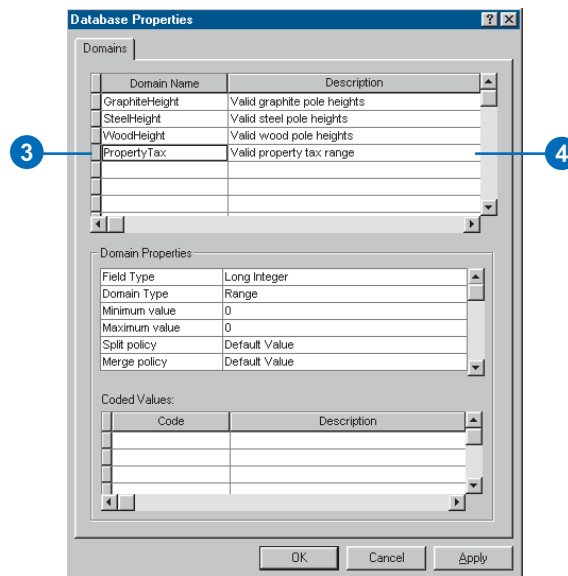
You can create new attribute range domains or coded value domains.

### See Also

*To learn how to apply an attribute domain to a field in a feature class or table, see 'Associating default values and domains with tables and feature classes' in this chapter.*

## Creating a new attribute range domain

1. Right-click the geodatabase in the ArcCatalog tree and click Properties.
2. Click the Domains tab.
3. Click the first empty field under Domain Name and type a name for the new domain.
4. Press the Tab key or click the new domain's Description field and type a description for the domain.
5. Click the field next to Field Type in Domain Properties, click the dropdown arrow, and click the type of attribute field to which this domain will be applied. ▶



## Tip

### Range domains

Range domains can't be created for text fields. They can only be applied to numeric and date fields.

- Click the field next to Domain Type, click the dropdown arrow, and click Range from the list of domain types.
- Click the field with the minimum value for the range domain and type the minimum value. Do the same for the maximum value.
- Click the field next to Split policy, click the dropdown arrow, and click the split policy for the new domain. Do the same for the merge policy.
- Click Apply to create the new domain in the geodatabase or OK to create the domain and close the dialog box.

Database Properties

Domains

Domain Name	Description
GraphiteHeight	Valid graphite pole heights
SteelHeight	Valid steel pole heights
WoodHeight	Valid wood pole heights
PropertyTax	Valid property tax range

Domain Properties

Field Type: Float

Domain Type: Range

Minimum value: Range

Maximum value: Coded Values

Split policy: Default Value

Merge policy: Default Value

Coded Values:

Code	Description
------	-------------

OK Cancel Apply

Database Properties

Domains

Domain Name	Description
GraphiteHeight	Valid graphite pole heights
SteelHeight	Valid steel pole heights
WoodHeight	Valid wood pole heights
PropertyTax	Valid property tax range

Domain Properties

Field Type: Float

Domain Type: Range

Minimum value: 500

Maximum value: 1000

Split policy: Geometry Ratio

Merge policy: Default Value

Coded Values:

Code	Description
Sum Values	Weighted Average

OK Cancel Apply

## Tip

### Coded value descriptions

When you add a new value to a domain's coded value list, you must also add a more user-friendly description. When you edit an attribute value for a field that has this domain, the user-friendly values appear in the ArcMap Editor. Descriptions help you select the right value.

## Tip

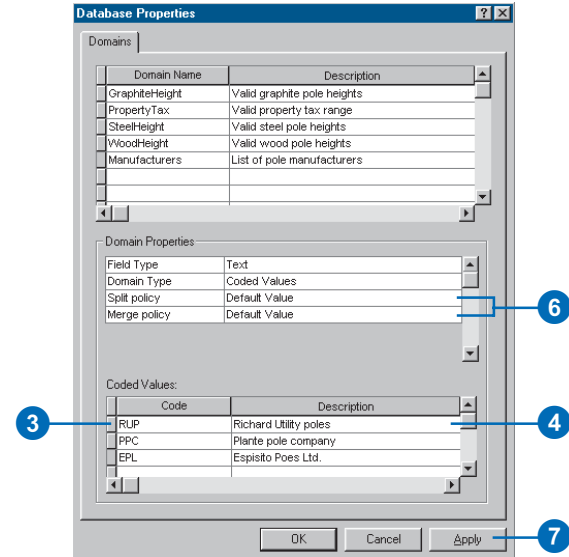
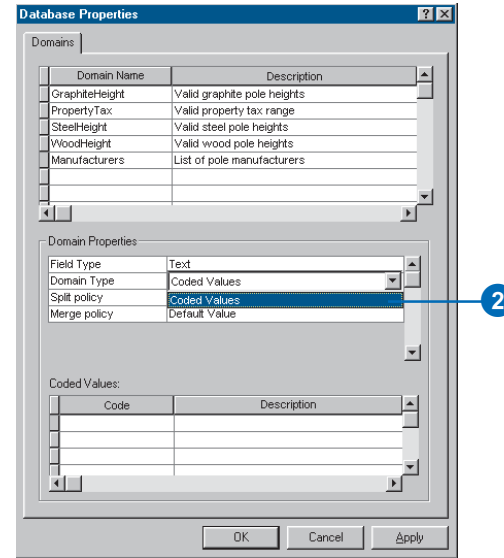
### Coded value split/merge policies

Coded value domains support only default value and duplicate split policies.

Coded value domains support the default value merge policy only.

## Creating a new coded value domain

1. Follow steps 1 through 4 for 'Creating a new attribute range domain'.
2. Click the field next to Domain Type, click the dropdown arrow, and click Coded Values from the list of domain types.
3. Click the first empty field under Coded Values and type the first valid code.
4. Press the Tab key or click the new coded value's Description field. Type a user-friendly description for this coded value.
5. Repeat steps 3 and 4 until all valid values and their descriptions have been typed.
6. Click the field next to Split policy, click the dropdown arrow, and click the split policy for the new domain. Do the same for the merge policy.
7. Click Apply to create the new domain in the geodatabase or OK to create the domain and close the dialog box.



## Modifying and deleting attribute domains

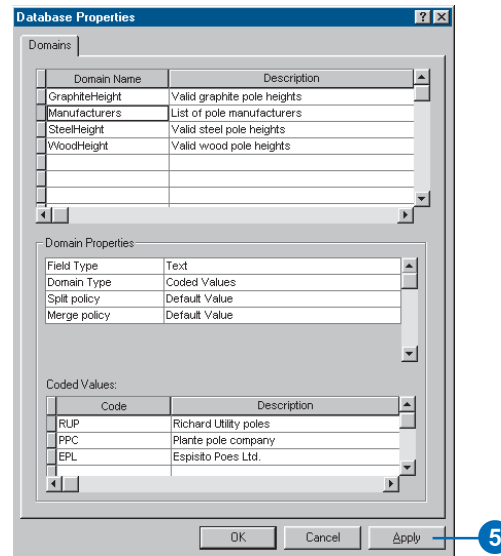
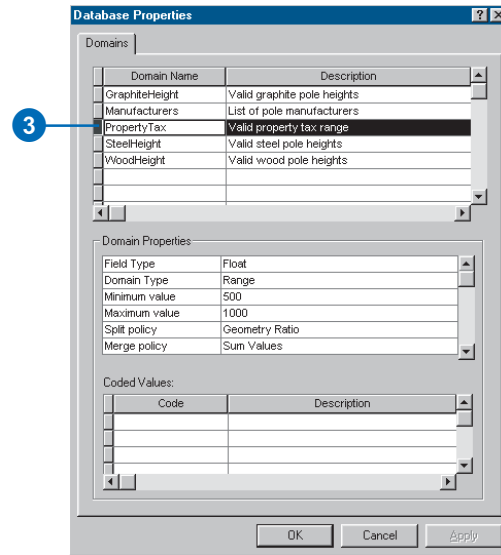
The Domains property page can be used to delete an attribute domain from the geodatabase or to modify an existing domain.

When a new domain is created, the owner of that domain—that is, the user who created it—is recorded. Only the owner of an attribute domain can delete or modify it.

As you will see later in this chapter, domains can be associated with particular fields for a feature class or table or for a subtype of a feature class or table. While a domain is being used by a table or feature class, it cannot be deleted or modified.

You can modify domains simply by selecting them on the domain properties dialog box and changing anything from their name to their type and valid values. This is done in the same manner as when you create a new domain.

1. Right-click the geodatabase in the ArcCatalog tree and click Properties.
2. Click the Domains tab.
3. Click the domain you want to delete by clicking the left tab in the grid.
4. Press the Delete key.
5. Click Apply to delete the domain from the geodatabase or OK to delete the domain and close the dialog box.





# Associating default values and domains with tables and feature classes

Once you have created an attribute domain, or domains, you can associate them and their default values with fields in a table or feature class. Once a domain is associated with a feature class or table, an attribute validation rule is created in the database.

The same attribute domain can be associated with multiple fields of the same table, feature class, or subtype and can be associated with multiple fields in multiple tables and feature classes.

## Tip

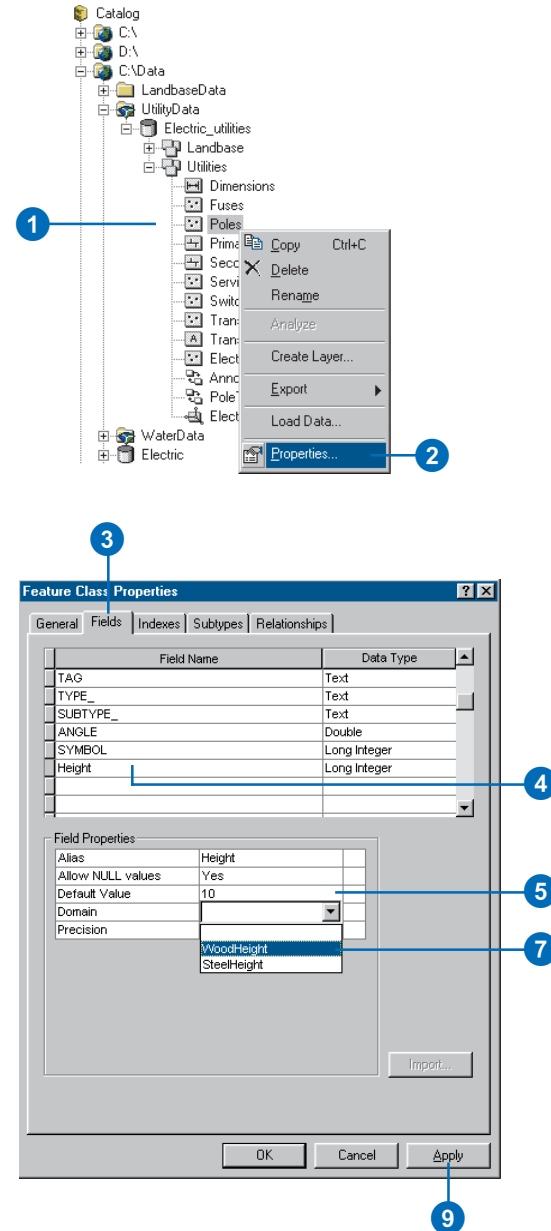
### Subtypes

Not all of the objects in a table or feature class must have the same domains or default values applied to the same fields. To apply different domains and default values to the same field in a single table or feature class, you must create subtypes.

You'll learn how to create subtypes and associate domains and default values to a subtype's fields later in this chapter.

1. Right-click the table or feature class in the ArcCatalog tree with which you want to associate domains.
2. Click Properties.
3. Click the Fields tab.
4. Click the field for which you want to create a default value and associate a domain.
5. Click the field next to Default Value and type the default value.
6. Skip to step 9 if you don't want to associate a domain to the field.
7. Click the field next to Domain, click the dropdown arrow, and click the domain you want to associate with the field.
8. Repeat steps 4 through 7 until you have associated default values and domains for all fields that you want to have these properties.
9. Click Apply.

Only those domains that apply to the field type are displayed in the list.



# Creating subtypes

You can use ArcCatalog to add subtypes and to set default values and attribute domains for the fields of each subtype.

You can manage subtypes using the properties dialog box for each table or feature class. You can define the subtype field, add new subtypes, and remove or modify existing subtypes.

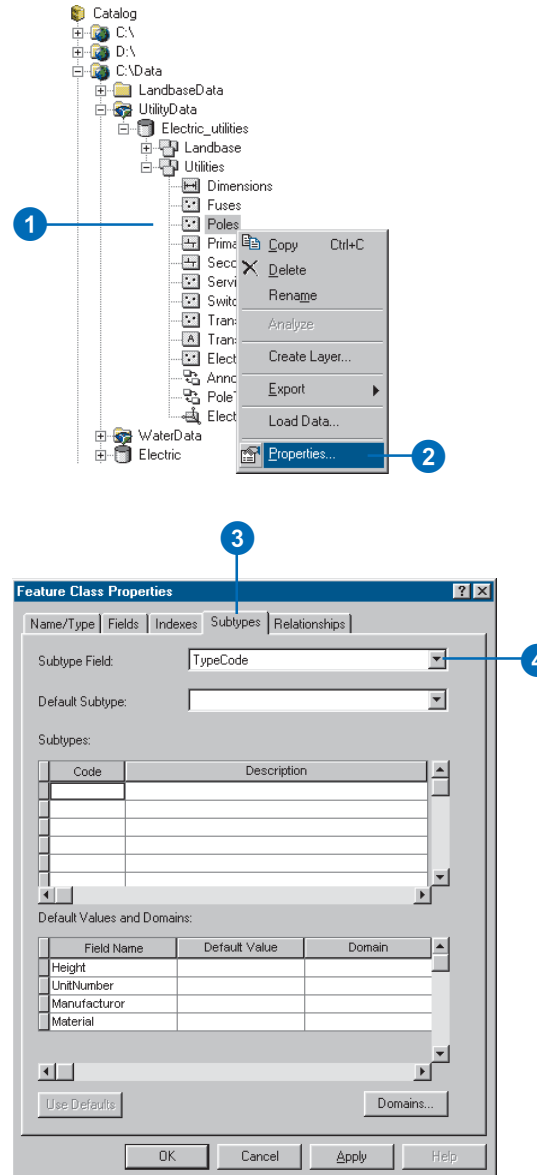
## Tip

### Subtype field

*The subtype field must be a long integer or short integer field. If you have no subtype field selected, you will not be able to add subtypes.*

## Creating new subtypes for a feature class or table

1. Right-click the feature class or table in the ArcCatalog tree to which you want to add subtypes.
2. Click Properties.
3. Click the Subtypes tab.
4. Click the dropdown arrow and click the subtype field from the list of available long integer and short integer fields. ▶



## Tip

### The default subtype

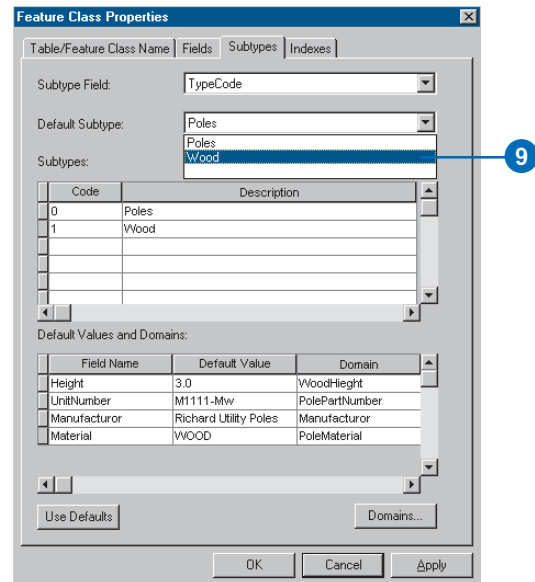
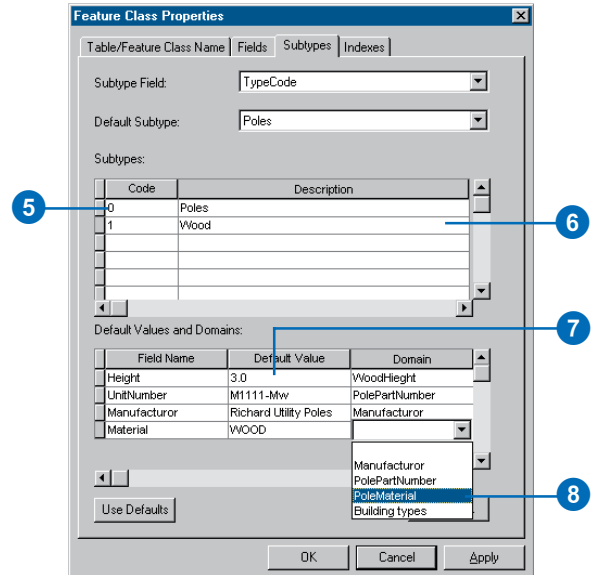
The default subtype serves two purposes. When you create a new subtype, click *Use Defaults* and the subtype will inherit all of the default values and domains for its fields from the default subtype. These can then be modified to meet the requirements for the new subtype. As you add additional subtypes, the default subtype can be changed at any time.

When you create a new feature in the feature class without specifying a subtype, it will automatically be assigned the default subtype.

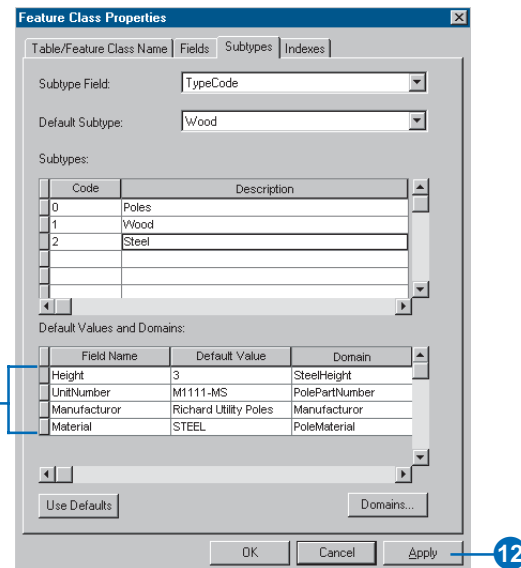
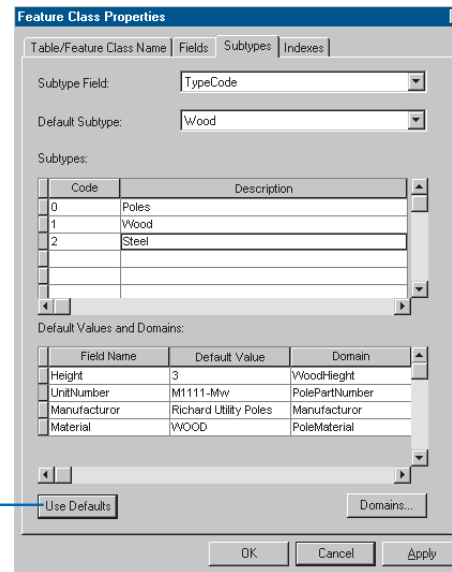
- Click the first empty field under Code and type an integer value that will be the code for that subtype to add a new subtype.
- Press the Tab key or click the Description field and type a description for the subtype.
- Type a default value in the appropriate field in the table for each field.
- Click the Domain field, click the dropdown arrow, and click the domain from the list of domains to associate an attribute domain with a field for the new subtype.

Only those domains that apply to the field type are displayed in the list.

- Click the dropdown arrow and click it from the list of subtypes to set this subtype as the default subtype. ►



10. Repeat steps 5 through 8 to add additional subtypes. You can reset the default subtype at any time.
11. Click Use Defaults to have your new subtype take all of the default values and domains from the default subtype when adding a new subtype. You can then modify all or some of these.
12. Click Apply to create the new subtypes in the geodatabase or OK to create the subtypes and close the dialog box when you are finished creating your subtypes and have selected the default subtype.

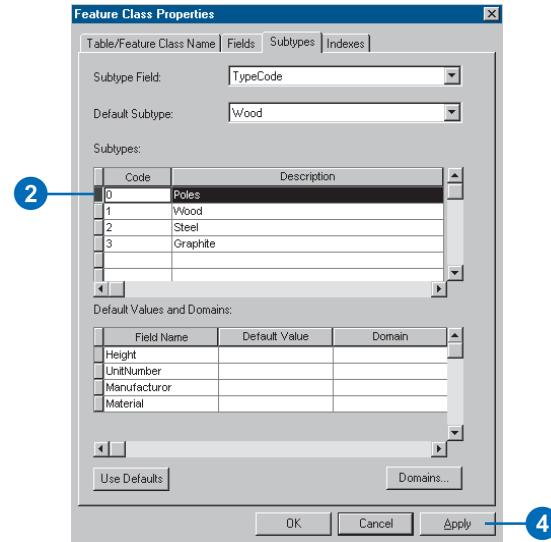


# Modifying and deleting subtypes

Subtypes for a feature class or table can also be modified or deleted using the properties dialog box for the table or feature class. You can modify any aspect of a subtype including its description, its default values, and its domains. Modifying each aspect of a subtype is done the same way as creating a new subtype.

You cannot delete a subtype if it is currently referenced by a topology rule.

1. Follow steps 1 through 3 for 'Creating new subtypes for a feature class or table'.
2. Click the left tab next to the subtype you want to delete.
3. Press the Delete key.
4. Click Apply to delete the subtype from the geodatabase or OK to delete the subtype and close the dialog box.





# Defining relationship classes

# 6

## IN THIS CHAPTER

- **What is a relationship class?**
- **Relationship classes in ArcCatalog and ArcMap**
- **Creating a simple relationship class**
- **Creating a composite relationship class**
- **Creating an attributed relationship class**
- **Creating relationship rules**
- **Managing relationship classes**
- **Exploring related objects in ArcMap**
- **Using related fields in ArcMap**

Objects in a real-world system often have particular associations with other objects in the database. These kinds of associations between objects in the geodatabase are called relationships. Relationships can exist between spatial objects (features in feature classes), nonspatial objects (rows in a table), or spatial and nonspatial objects. While spatial objects are stored in the geodatabase in feature classes and nonspatial objects are stored in tables, relationships are stored in relationship classes.

ArcCatalog contains tools to create, modify, and manage relationship classes in your geodatabase, while ArcMap provides tools to create, delete, and use relationships to find objects that are associated with other objects in the geodatabase. This chapter describes how to use ArcCatalog to manage these relationship classes and how to use relationships in ArcMap. *Editing in ArcMap* discusses how to create and delete relationships.

ArcView can view feature classes that use advanced geodatabase functionality. To create or edit feature classes that take advantage of advanced geodatabase functionality, you need an ArcEditor or ArcInfo license.

# What is a relationship class?

Objects in a real-world system, such as an electrical network or a parcel database, often have particular associations with other objects in the database. For example, in an electrical network, poles support transformers. In a parcel database, a parcel will have one or many owners.

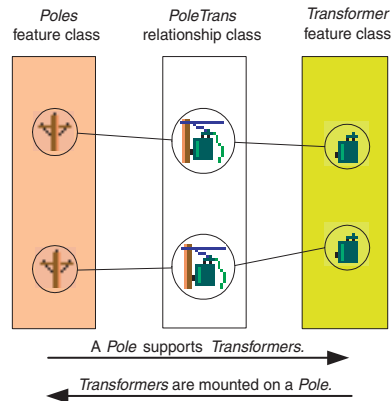
These kinds of associations between objects in the geodatabase are called relationships. Relationships can exist between spatial objects (features in feature classes), nonspatial objects (rows in a table), or spatial and nonspatial objects. While spatial objects are stored in the geodatabase in feature classes and nonspatial objects are stored in tables, relationships are stored in relationship classes.

To store relationships, such as between electric transformers and poles, you must create a relationship class. If, in your geodatabase, transformers also have relationships to transformer attribute objects, then a second relationship class is required to store those relationships.

## The anatomy of a relationship

As with any association, relationships have particular characteristics. One obvious characteristic is the notion of cardinality. Cardinality describes how many objects of one type are related to an object of another type. In the pole–transformer example, a single pole may support more than one transformer, but a transformer can only be mounted on a single pole. The relationship between poles and transformers is one to many: one pole, which is an object in the origin class of the relationship, to many transformers, which is an object in the destination class of the relationship.

In general, relationships can have one-to-one, one-to-many, and many-to-many cardinalities. As you will see later in this chapter, certain types of relationships support certain cardinalities, and



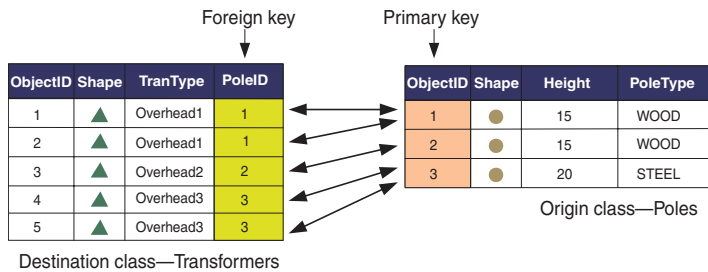
*A relationship is an association between two or more objects in two feature classes or tables in a geodatabase. Relationships are stored in relationship classes.*

you can control cardinalities for any relationship class when you define relationship rules.

A relationship between two objects is maintained through attribute values for key fields. In the pole–transformer example, the unit number of the pole that supports the transformer may be included in the attributes of the transformer object. This is referred to as an embedded foreign key. It tells us what object in the pole feature class this particular transformer is related to.

Relationship classes can have attributes. Any relationship class that has attributes must be stored as a table in the database and have a pair of foreign keys referencing the origin and destination classes of the relationship class. In this case, each relationship is stored as a row in the relationship classes table. Similarly, any





*A relationship between two objects is maintained through attribute values for key fields.*

many-to-many relationship classes require a table in the database to store at least the foreign keys.

Relationship classes have path labels. Forward and backward path labels describe the relationship when navigating from one object to another. The forward path label describes the relationship navigated from the origin class to the destination class; the backward path label describes the relationship when navigating from the destination to the origin class. In the pole-transformer example, when navigating from the transformer to the pole, the relationship path label may be “is mounted on”. The same relationship, when navigated from the pole to the transformer, may have a path label of “supports”.

Relationship classes can also be used to propagate standard messages between related objects. Messaging is the mechanism that objects related to each other use to notify each other when they change. For example, in a relationship between poles and transformers, when the pole is deleted, it sends a message to its related transformer objects informing them it was deleted. As transformers can’t exist without a pole to support them, these transformer objects could respond to the message by deleting themselves.

The geodatabase supports two kinds of relationships: simple (or peer-to-peer) relationships and composite relationships. Each is discussed briefly below.

## Simple relationships

Simple, or peer-to-peer, relationships are relationships between two or more objects in the database that exist independently of each other.

In a relationship between object A and object B, if object A is deleted from the database, object B continues to exist. For example, in a railroad network you may have railroad crossings that have one or more related signal lamps. However, a railroad crossing can exist without a signal lamp, and signal lamps exist on the railroad network where there are no railroad crossings.

Simple relationships can have one-to-one, one-to-many, or many-to-many cardinality.

## Composite relationships

The geodatabase also supports the notion of a composite relationship, where the lifetime of the origin object controls the lifetime of its related objects. The pole-transformer relationship is an example of a composite relationship. Once a pole is deleted, a delete message is sent to its related transformers, which are deleted from the transformer’s feature class.

Composite relationships are always one-to-many but can be constrained to be one-to-one using relationship rules.

## Attributed relationship classes

One-to-one and one-to-many relationship classes do not require a new table in the geodatabase to be created to store the relationships. However, many-to-many relationship classes do require a

new table in the database to store the foreign keys from the origin and destination classes to make the relationship. This table can also have other fields to store attributes of the relationship itself that are not attributes of either the origin or destination class.

For example, in a parcel database you may have a relationship class between parcels and owners, where owners “own” parcels and parcels are “owned by” owners. An attribute of that relationship may be the percentage of ownership.

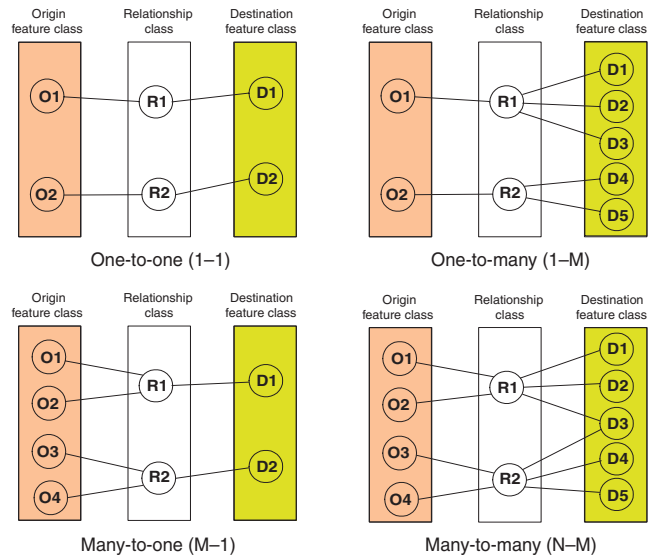
One-to-one and one-to-many relationship classes may also have attributes; in this case, a table would be created to store the relationships.

## Relationship rules

Relationship classes can have an associated set of relationship rules. Relationship rules control which object *subtypes* from the origin class can be related to which object subtypes in the destination class. They can also be used to specify a valid cardinality range for all permissible subtype pairs.

For example, the subtype wood pole may be able to support from 0 to 3 transformers, whereas the subtype steel pole may support 0 to 5 transformers. In the first case, the cardinality range would be 0–3; in the second case, it would be 0–5.

You can establish a relationship between two or more objects in the geodatabase by using tools available in ArcMap. Once the relationship is established, use ArcMap tools to navigate it. You can find all objects that have a relationship with a particular object through any particular relationship class.



*Relationships have cardinality. Cardinality describes how many objects of type A are associated with objects of type B. Relationships can have 1–1, 1–M, M–1, or N–M cardinality.*

## Performance considerations

When editing features that have relationships with messaging, edits, such as move, rotate, and delete, also affect the related objects through the relationship class. There is a cost when navigating these relationships. The cost is minimized when indexes are maintained for the primary and foreign keys for the relationship class. When a new relationship class is created with ArcCatalog, the primary and foreign keys are automatically indexed if they do not already have indexes.

It is important to realize that when a feature class participates in a relationship class, that feature class utilizes messaging. When editing that feature class in ArcMap, the related class must be

opened so it can respond to the message—either by moving, deleting itself, or implementing some *custom behavior*. If the related class is not already in the map you are working with, it will automatically be opened to respond to the message, then closed. Each time it responds to a message, it will need to be reopened.

In general, when working with a class in ArcMap, have all related classes also in the map. This way, the related classes are opened once when they are added to ArcMap. If they are not in the map, then each time you access related objects, the class must be opened.

With many ArcInfo coverage data models, the feature–attribute table contained as few items as possible, and many of the attributes for a feature class were contained in a related table. This can be done with geodatabase feature classes; however, navigating a relationship in the geodatabase is a more costly operation than navigating relates in INFO. In the INFO environment, it was common to store the *symbolology* for a feature in an external, related table called a lookup table. This can still be done in the geodatabase using relationship classes and joining the two tables; however, for large data, symbolizing this way will be slow, even with indexes on the primary and foreign keys. Try to keep attributes for symbolization on the feature class’s table. For performance considerations, it is recommended that symbolology information be stored in the feature class itself.

## Schema locking

An exclusive lock is required when modifying a relationship class’s relationship rules or when renaming or deleting a relationship class. An exclusive lock can only be acquired for a relationship class if the feature classes or tables that participate in the relationship can also be locked. Therefore, if another user has an exclusive or shared lock on either the origin, destination, or both classes, the properties of the relationship class cannot be edited.

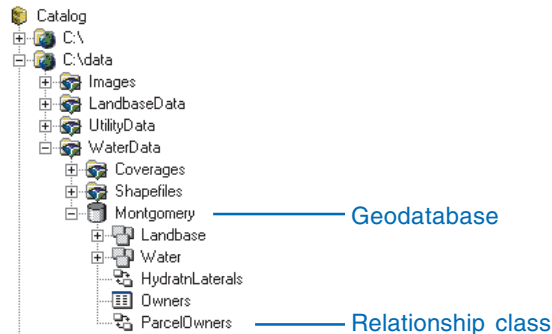
For more information on exclusive locks and schema locking, see the chapter ‘Creating new items in a geodatabase’ in this book.

# Relationship classes in ArcCatalog and ArcMap

## Relationship classes in ArcCatalog

You can work with relationship classes in ArcCatalog. Relationship classes can exist both inside feature datasets and at the root level of the geodatabase.

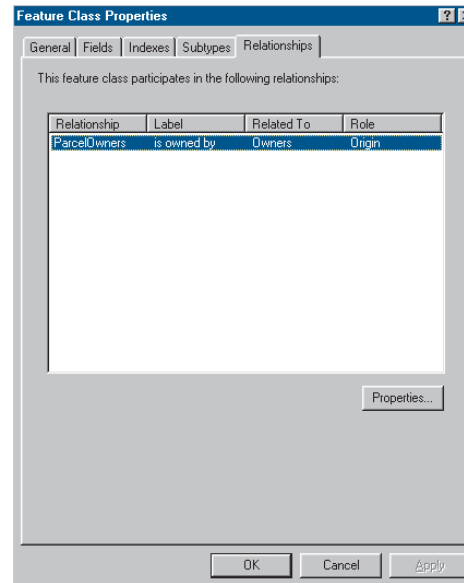
When you look at a particular relationship class in the ArcCatalog tree, it is not immediately evident how feature classes or tables are related. However, by examining the properties of both the feature class or table and the relationship class, you can achieve a clear picture of this.



*Relationship classes appear in the ArcCatalog tree either at the geodatabase level or inside a feature dataset.*

In the Feature Class Properties or Table Properties dialog box, the Relationships tab displays the relationship classes, if any, in which a feature class or table participates. For each relationship class, it displays the path label, the other feature class or table, and its role in the relationship. You can click Properties to view the properties for the selected relationship class.

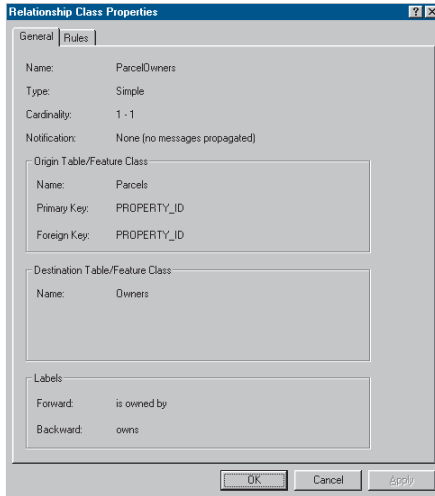
The properties dialog box for each relationship class—whether opened from the Feature Class Properties or Table Properties dialog box tree—contains more detailed information about the



*By clicking the Relationships tab on the properties dialog box for a feature class or table, you can view what relationship classes the feature class or table participates in along with what role the feature class or table plays within the relationship class. To get more details about the relationship class, click Properties.*

relationship class. It also lets you establish relationship rules. The procedure for creating and modifying relationship rules is discussed later in this chapter.

ArcCatalog also contains various tools to create, delete, and manage relationship classes. The tools will also be discussed in more detail in this chapter.



*The Relationship Class Properties dialog box, whether opened from the feature class or table property page, contains detailed information about the relationship class.*

## Relationship classes in ArcMap

Once you have established a relationship class between feature classes or tables, you can use these relationships in ArcMap. For example, when you identify a feature in your map, you can see all of the objects related to that feature. When working with tables, you can select one or more rows or features and open the related table to see the selected related objects.

You may want to use fields on a related table or feature class to symbolize or label your map. Once you have added a feature class to your map that participates in a relationship class, you can do this by establishing a join between the feature class and its related feature class or table. You can use these joined fields like you use other fields in your feature layer.

For more information on maps, feature layers, symbolizing, and labeling your features, see *Using ArcMap*.

There are also a number of tools in ArcMap for editing relationships and related objects. For example, when you select a feature, you can edit the properties of its related objects. You can also use ArcMap to add new relationships and delete existing relationships. For more information on editing relationships, see *Editing in ArcMap*.

## Deciding between relationship classes and joins and relates

Geodatabase relationship classes are usually created to establish an enduring business process relationship between a feature class and another table or feature class. ArcMap joins and relates are useful in data building, data exploration, or analysis.

Relationship classes have many advantages over joins and relates. The relationship class is stored with the data in the geodatabase. This makes the relationship class accessible to anyone who uses the geodatabase. A relationship class allows greater interaction among related objects when editing the feature classes that participate in the relationship class. Relationship classes allow you to build behavior into the relationship. For example, the deletion or modification of one feature could delete or alter another related feature.

You may find, however, that a join or relate will perform the desired function. Joins and relates are independent of the geodatabase. This offers several advantages. First, a join or relate can be performed by a user without influencing the data in the geodatabase. Further, joins and relates are stored with the map document and are not geodatabase specific. They can establish relationships among data found in different geodatabases or data that is not in a geodatabase. For more information on joins and relates, see *Using ArcMap*.

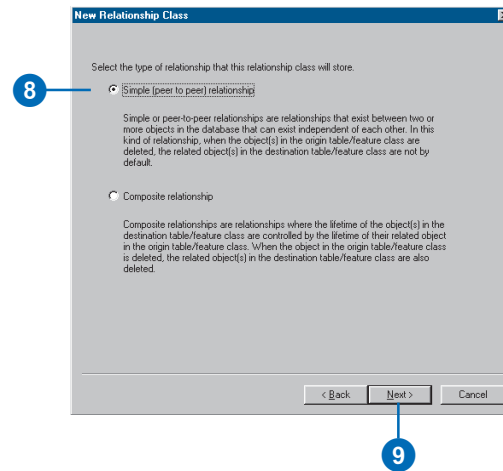
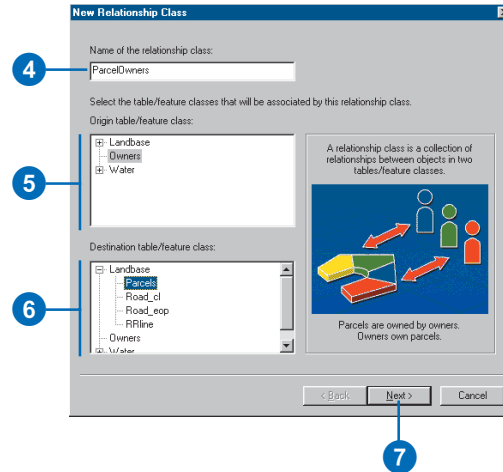
# Creating a simple relationship class

You can create new relationship classes between any feature class or table within your geodatabase using tools in ArcCatalog. These tools can be used to create simple, composite, and attributed relationship classes.

Relationship classes appear in the Catalog tree, and you can inspect their properties as well as the relationships for any particular feature class.

The example in this task shows how to create a relationship class between a feature class that stores parcel objects and a table that stores owner objects. It is a simple, nonattributed relationship. In the database, a parcel can be owned by a single owner, and an owner can own a single parcel, so it is a one-to-one (1-1) relationship.

1. Right-click the geodatabase or feature dataset, in the ArcCatalog tree, in which you want to create the new relationship class.
2. Point to New.
3. Click Relationship Class.
4. Type the name for the new relationship class.
5. Click the origin table or feature class.
6. Click the destination table or feature class.
7. Click Next.
8. Click Simple (peer-to-peer) relationship.
9. Click Next. ►



## Tip

### M–N relationship classes

*Many-to-many (M–N) relationship classes require the relationship class to have its own table in the database. You can optionally add attributes to this table, or you can allow ArcGIS to manage the schema of the table for you.*

## Tip

### Notification direction

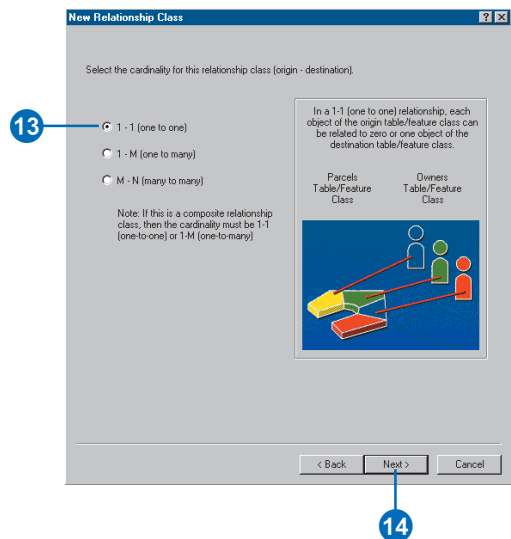
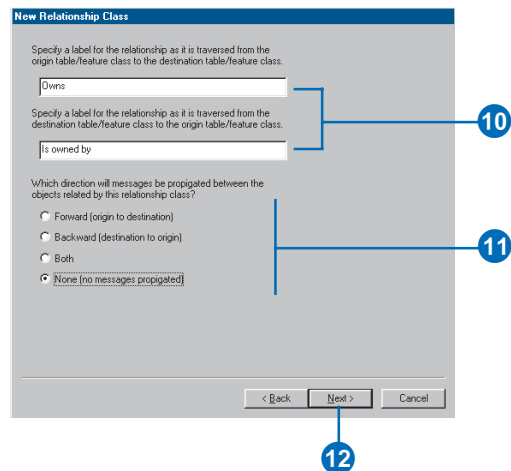
*By default, the notification direction for a simple relationship is None.*

## Tip

### Turning on messaging

*Turning on messaging causes messages to be generated for each edit, slowing performance. Unless you are responding to messages and reacting accordingly, you should not turn on messaging.*

10. Type the forward and backward path labels.
11. Click the message notification direction.
12. Click Next.
13. Click the first cardinality option. In this example, an owner can own a single parcel and a parcel can be owned by a single owner, so this is a one-to-one (1–1) relationship.
14. Click Next. ►



## See Also

*In this first example, you are not adding attributes to your relationship class, although any relationship class can have attributes. For more information on how to create an attributed relationship class, see 'Creating an attributed relationship class' in this chapter.*

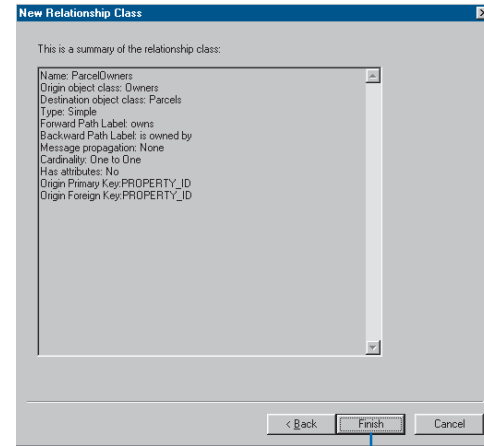
15. Click No. The relationship class does not require attributes in this example.
16. Click Next.
17. Click the dropdown arrow to see a list of fields from the origin table or feature class. Click the primary key for this feature class or table.
18. Click the dropdown arrow to see a list of fields from the destination table or feature class. Only those fields that are the same type as selected in step 17 are displayed. Click the foreign key that refers to the primary key selected in step 17.
19. Click Next. ►

The dialog box is titled "New Relationship Class". It contains the following text: "You have specified that this is a M-N (many to many) relationship. A new table in the database will be created for this relationship to store the foreign keys for these two participating tables/feature classes." Below this, it asks: "Do you wish to add additional attributes to this relationship class?". There are two radio button options: "Yes, I would like to add attributes to this relationship class." and "No, I do not want to add attributes to this relationship class.". The "No" option is selected. At the bottom, there are three buttons: "< Back", "Next >", and "Cancel".

The dialog box is titled "New Relationship Class". It contains the following text: "Select the primary key in the origin table/feature class (generally, this will be the object identifier field). If this is a 1 - M (one to many) relationship, you will also need to select the foreign key in the destination table/feature class." Below this, there are two sections. The first section is "Select the primary key field in the origin table/feature class:" with a dropdown menu showing "PROPERTY\_ID". The second section is "Select the foreign key field in the destination table/feature class that refers to the primary key field in the origin table/feature class:" with a dropdown menu showing a list of fields: "CALC\_AREA", "ELEMADDR", "PROPERTY\_ID", and "SEQUENCE\_NUMBER". The "PROPERTY\_ID" field is selected. At the bottom, there are three buttons: "< Back", "Next >", and "Cancel".



20. Review the options you specified for your new relationship class. If you want to change something, you can go back through the wizard by clicking Back.
21. Click Finish to create the new relationship class when satisfied with your options.



21

# Creating a composite relationship class

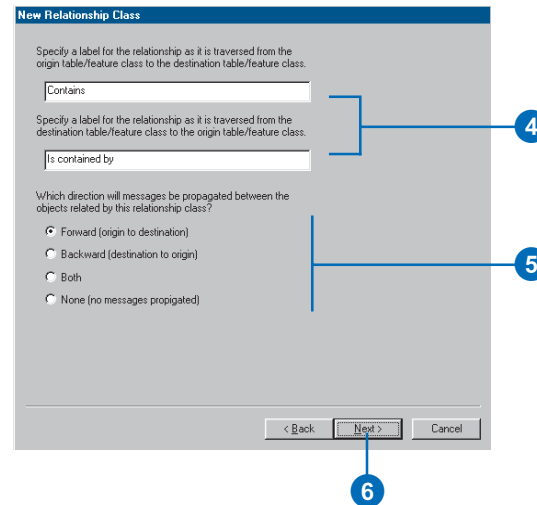
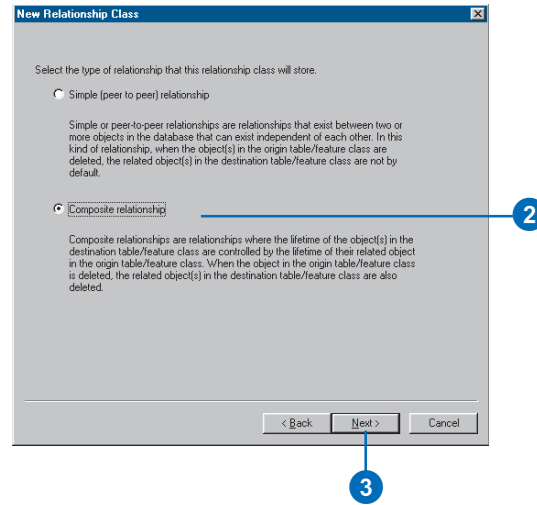
You can use a wizard to create a composite relationship class. The example in this subtask shows how to create a relationship class between a feature class that stores transformer banks and one that stores transformer units.

The existence of a transformer unit in the database is dependent on the presence of a transformer bank. This relationship class is a composite relationship with the transformer bank as the origin feature class.

The relationship will be nonattributed; composite relationships are by definition one-to-many (1-M) relationships.

Creating a composite relationship involves many of the same steps used in the task for creating a simple relationship. The steps outlined here reflect the differences between the two tasks including using different origin and destination classes.

1. Follow steps 1 through 7 for 'Creating a simple relationship class'.
2. Click Composite relationship.
3. Click Next.
4. Type the forward and backward path labels.
5. If you need to turn on messaging, click the message notification direction.
6. Click Next. ▶

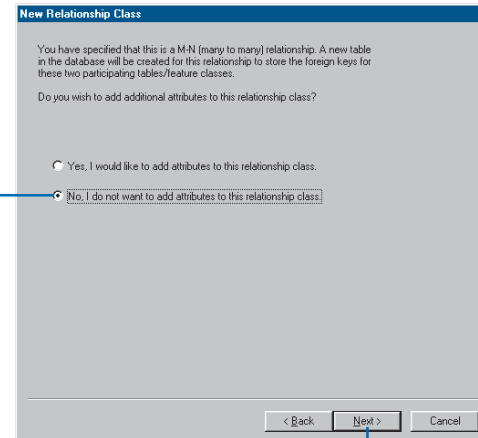
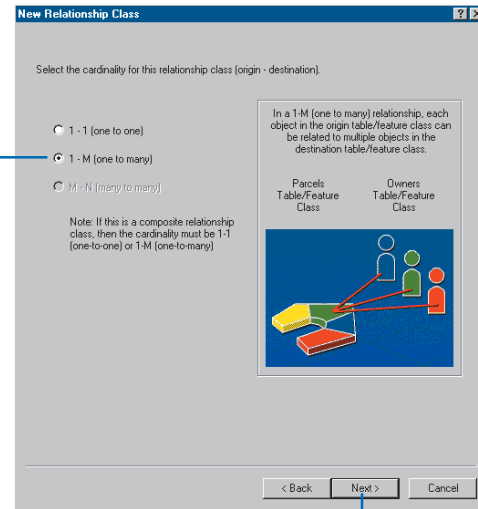


## Tip

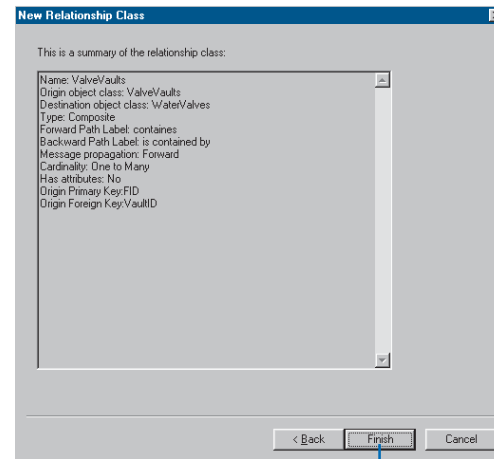
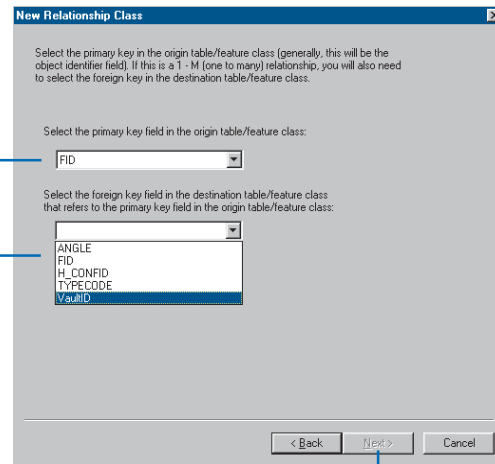
### One-to-many relationships

When creating a one-to-many relationship, whether simple or composite, the one side must be the origin class. The many side must always be the destination class.

- Click the second cardinality option. A composite relationship is, by definition, a 1–M or 1–1 relationship.
- Click Next.
- Click No. The relationship class does not require attributes in this example.
- Click Next. ►



11. Click the dropdown arrow to see a list of fields from the origin table or feature class. Click the primary key for this feature class or table.
12. Click the dropdown arrow to see a list of fields in the destination table or feature class. Only those fields that are the same type as selected in step 11 are displayed. Click the foreign key that refers to the primary key selected in step 11.
13. Click Next.
14. Review the options you specified for your new relationship class. If you want to change something, you can go back through the wizard by clicking Back.
15. Click Finish to create the new relationship class when satisfied with your options.



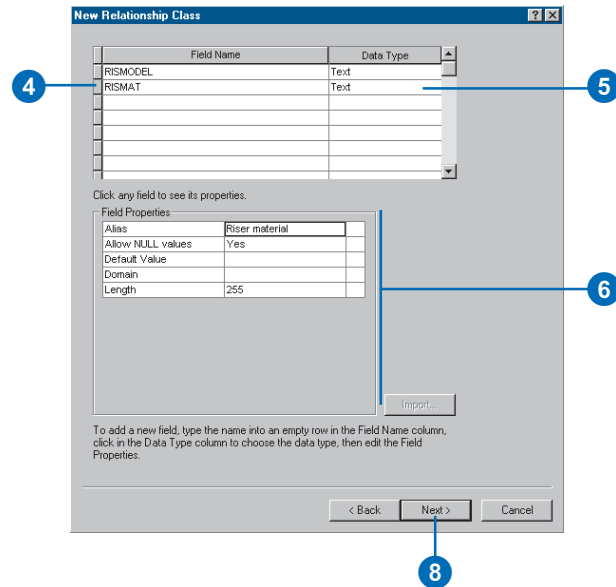
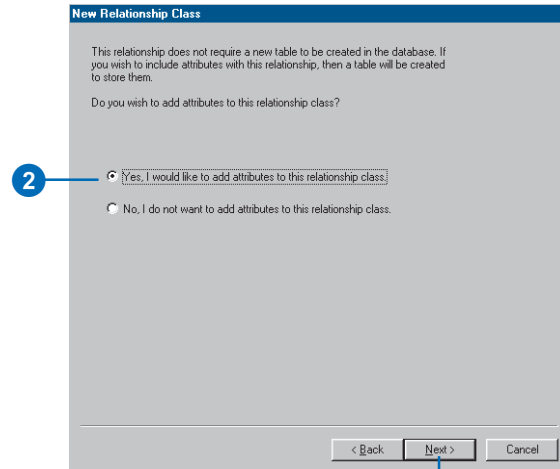
# Creating an attributed relationship class

Any relationship class—whether simple or composite, of any particular cardinality—can have attributes. Relationship classes with attributes are stored in a table in the database. This table contains at least the foreign key to the origin feature class or table and the foreign key to the destination feature class or table.

An attributed relationship can also contain any other attribute. The example in this subtask shows how to create a simple relationship between a feature class that stores water laterals and a feature class that stores hydrants.

Water lateral objects have their own attributes, and hydrant objects have their own attributes. The relationship class in this example describes which water laterals feed which hydrants. Because you want to store some kind of information about that relationship, such as the type of riser connecting the two, you can store this information as attributes in the relationship class.

1. Follow steps 1 through 14 for 'Creating a simple relationship class' or steps 1 through 8 for 'Creating a composite relationship class'.
2. Click the first option to add attributes to the relationship class.
3. Click Next.
4. Click the next row in the Field Name column and type a name to add a field.
5. Click in the Data Type field next to the new field's name, then click its data type.
6. Set the new field's properties in the property sheet.
7. Repeat steps 4 through 6 until all the relationship class's fields have been defined.
8. Click Next. ▶



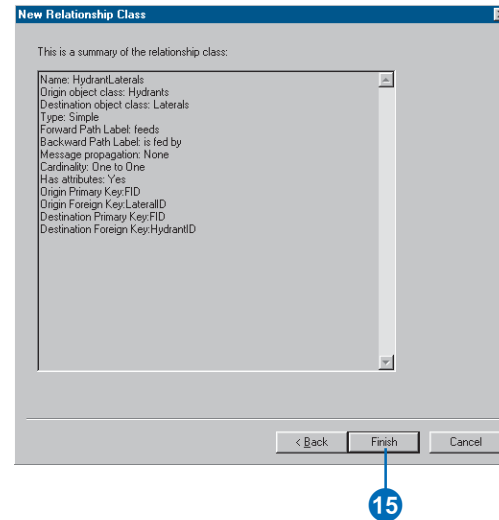
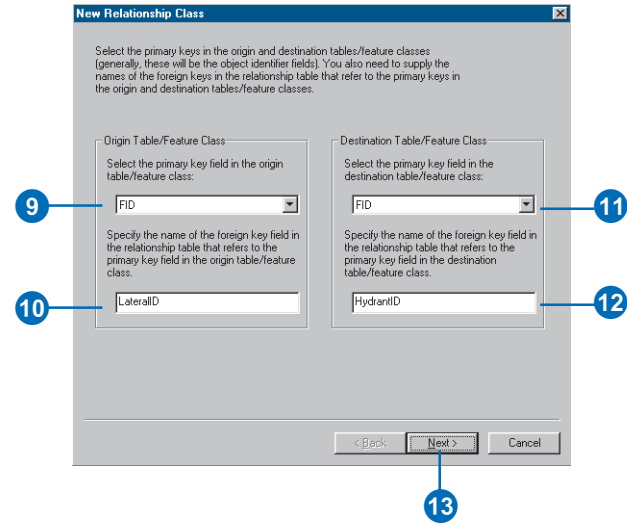
## Tip

### Relationship table foreign keys

*In an attributed relationship, the relationship table must have fields that act as foreign keys to the origin and destination feature classes or tables.*

*These foreign keys relate to the primary keys on the origin and destination feature class or table primary keys.*

- Click the dropdown arrow to see a list of fields from the origin table or feature class. Click the primary key for this feature class or table.
- Type the name of the foreign key field for the origin table or feature class.
- Click the dropdown arrow to see a list of fields from the destination table or feature class. Click the primary key for this feature class or table.
- Type the name of the foreign key field for the destination table or feature class.
- Click Next.
- Review the options you specified for your new relationship class. If you want to change something, you can go back through the wizard by clicking Back.
- Click Finish to create the new relationship class when satisfied with your options.



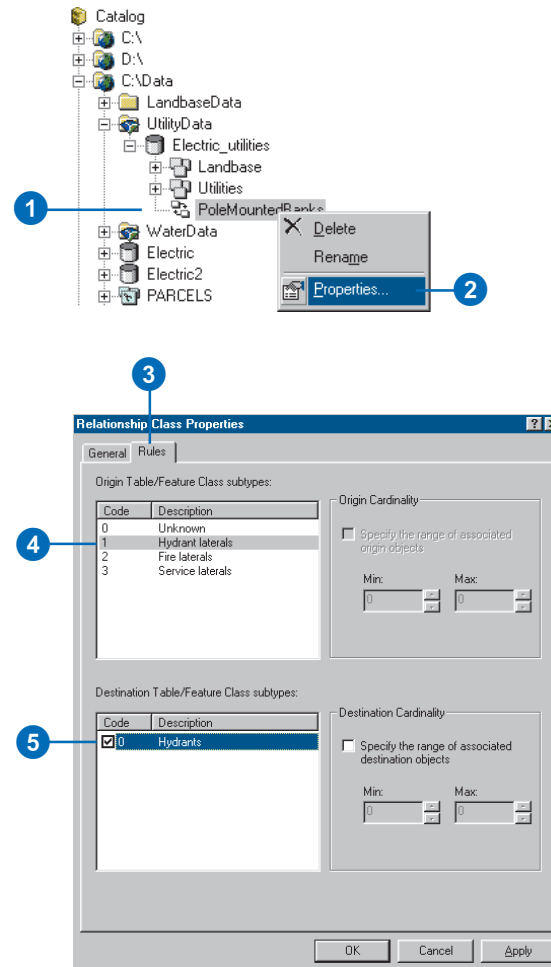
# Creating relationship rules

Relationship rules let you restrict the type of objects in the origin feature class or table that can be related to a certain kind of object in the destination feature class or table.

Relationship classes are created with general cardinalities such as one to many and many to many. In a real system, however, relationship cardinalities are more specific.

In this task, a relationship rule is being created between the hydrant laterals subtype on the water laterals feature class and the hydrants feature class. This rule says that it is valid for hydrants to be fed by hydrant laterals. Using the cardinality properties, you can specify exactly how many hydrants can be related to each hydrant lateral. In this example, it is invalid for a hydrant lateral not to feed a hydrant, and it is also invalid for a hydrant lateral to feed more than one hydrant. Therefore, the minimum and maximum cardinality will be 1.

1. Right-click the relationship class in the Catalog tree.
2. Click Properties.
3. Click the Rules tab.
4. Click the subtype that you want to associate with a relationship rule if your origin class has subtypes. If the origin class has no subtypes, the relationship rule will apply to all features.
5. Check the subtype that you want to make relatable to the selected subtype in the origin class if the destination class has subtypes. If the destination class has no subtypes, the relationship rule will apply to all features. ▶

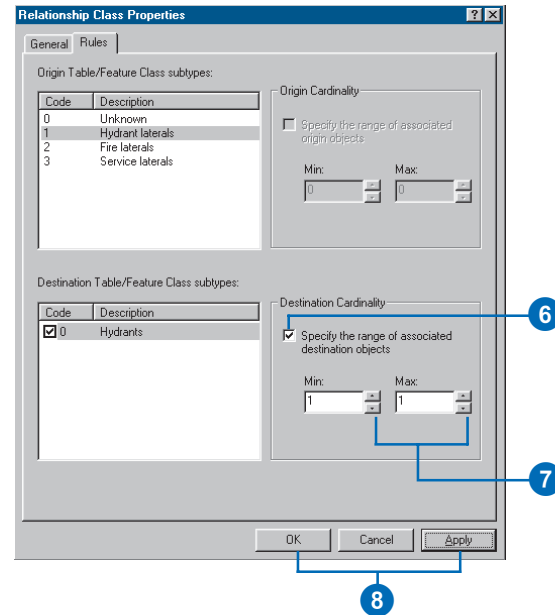


## Tip

### Relationship rules

Once a relationship rule is added to a relationship class, that rule becomes the only valid relationship that can exist. To make other relationship combinations and cardinalities valid, you must add additional relationship rules.

- If one or both sides of the relationship class is a many, you can limit the specific range of cardinality. In this example, the origin side of the relationship is a 1, so you cannot modify its range. However, the destination side is a many, so you can modify its range.
6. Check the check box to specify the range of destination objects per related origin objects.
  7. Click the up and down arrows to increase or decrease the minimum and maximum number of related destination objects.
  8. Repeat steps 4 through 7 until you have specified all of the relationship rules for this relationship class. Click OK or Apply to create the rules in the database.





# Managing relationship classes

Once created, a relationship class cannot be modified. However, you can add, delete, and modify its rules.

Relationship classes can be deleted and renamed using ArcCatalog. Relationship classes are deleted and renamed in the same manner as any other object in the database.

## Tip

### Deleting the origin or destination relationship class

*When you delete a feature class or table in ArcCatalog, if that feature class or table participates in a relationship class, the relationship class is also deleted.*

## Tip

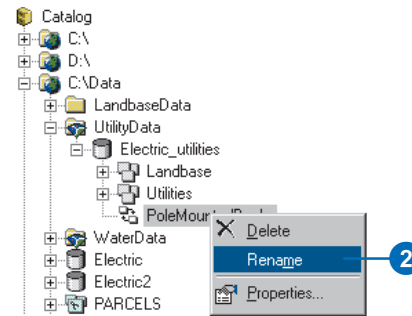
### Registering as versioned

*If you register either the origin or destination class as versioned in ArcCatalog, then both the relationship class and the class that it is related to are also registered as versioned.*

*To learn more about versioning, see the chapter 'Working with a versioned geodatabase' in this book.*

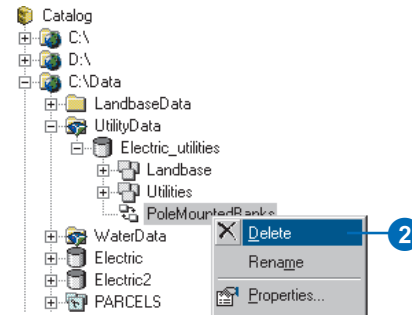
## Renaming a relationship class

1. Right-click the relationship class that you want to rename.
2. Click Rename.
3. Type the new name and press Enter.



## Deleting a relationship class

1. Right-click the relationship class you want to delete.
2. Click Delete.



# Exploring related objects in ArcMap

In ArcMap, you can explore what objects are related to any particular object in your geodatabase. When identifying features, the Identify Results dialog box allows you to navigate to a feature's related objects. When working with tables, you can navigate to a table of related objects.

## Tip

### Stacked relationships

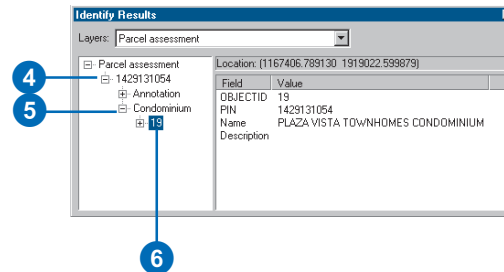
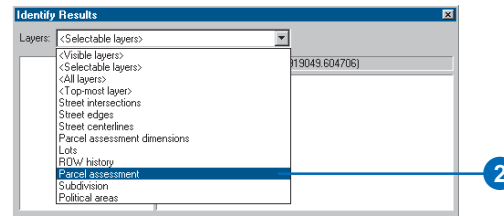
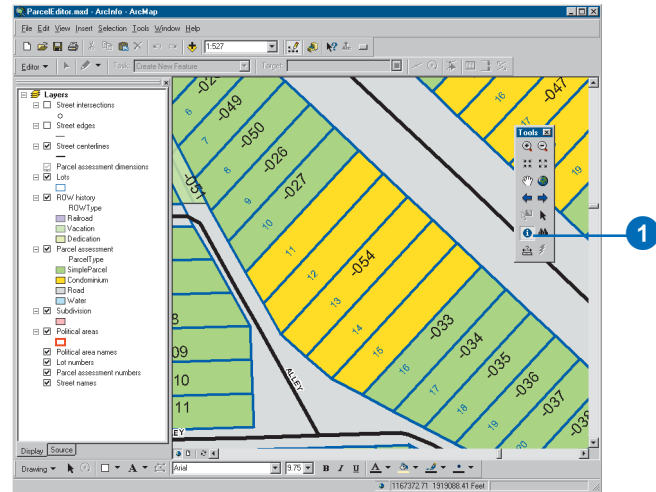
*If the related object that you navigate to in the Identify Results dialog box has objects related to it through other relationships, you can continue to navigate to those related objects.*

## Exploring the related objects of a feature

1. Click the Identify tool in ArcMap.
2. Click the Layers dropdown arrow and click the layer in your map whose features you want to identify in the Identify Results dialog box.
3. Click the feature on the map.
4. Double-click the feature in the left panel of the Identify Results dialog box.
5. Double-click the relationship path label.

The related objects are listed below the path label.

6. Click the related object whose properties you want to explore.



## See Also

If you are not already familiar with how to add data to your map, please refer to Using ArcMap.

## See Also

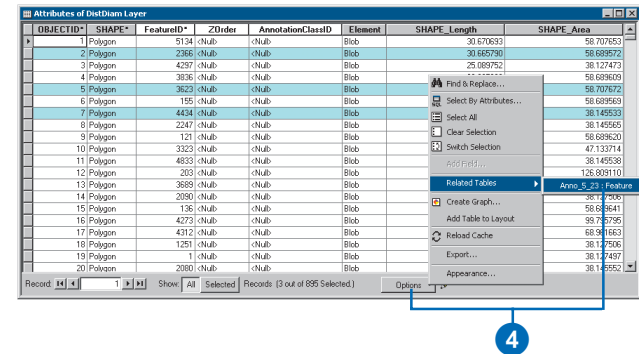
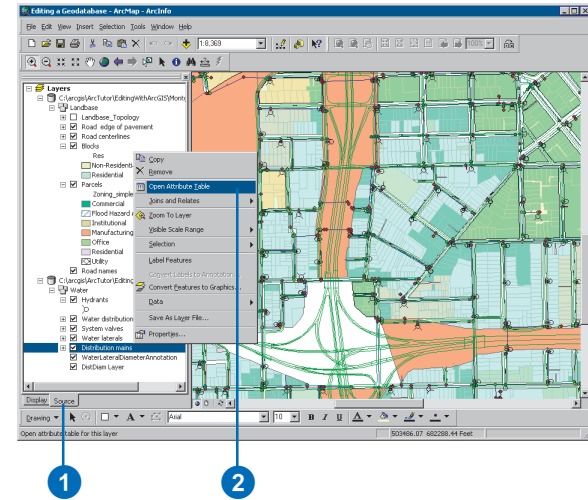
For more information on how to select records in a table, see Using ArcMap.

## Exploring the related objects of an object in a table

1. Click the Source tab on the ArcMap table of contents.
2. Right-click the name of the object of interest and click Open Attribute Table.

The table that contains the objects whose related objects you want to explore will open.

3. Select the objects whose related objects you want to explore.
4. Click Options, point to Related Tables, and click the path label for the relationship.



- A new table dialog box opens for the related table.
5. Click Show Selected to display only those objects related to the selected objects in the first table.

FID*	Shape*	Facility identifier	LOCATION	Installatio
3323	Polyline	24111		
1294	Polyline	19740		
1519	Polyline	3100		
3941	Polyline	6228		
155	Polyline	25404		
4320	Polyline	20281		
1100	Polyline	20936		
3359	Polyline	4911		
1062	Polyline	3715		
136	Polyline	3521		
3126	Polyline	21663		
203	Polyline	25304		
2366	Polyline	21076		
1330	Polyline	4661		
1	Polyline	2840		
3836	Polyline	21666		
3571	Polyline	24537		
1251	Polyline	24239		
3867	Polyline	20265		

5

# Using related fields in ArcMap

In order for fields from a related object to be used for symbolizing and labeling, you must create a join between the feature class and its related feature class or table. Once you have created this join, the fields from the related feature class or table are added to your feature layer. You can use these fields for labeling, symbolizing, and querying your features.

Using related fields in ArcMap is only possible with 1–1 relationship classes or 1–M relationship classes where you’re joining the origin attributes to the destination table.

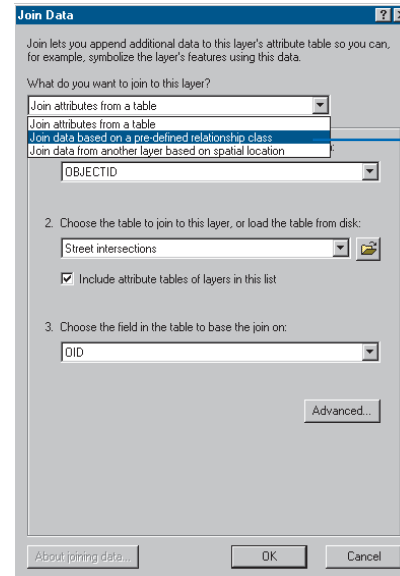
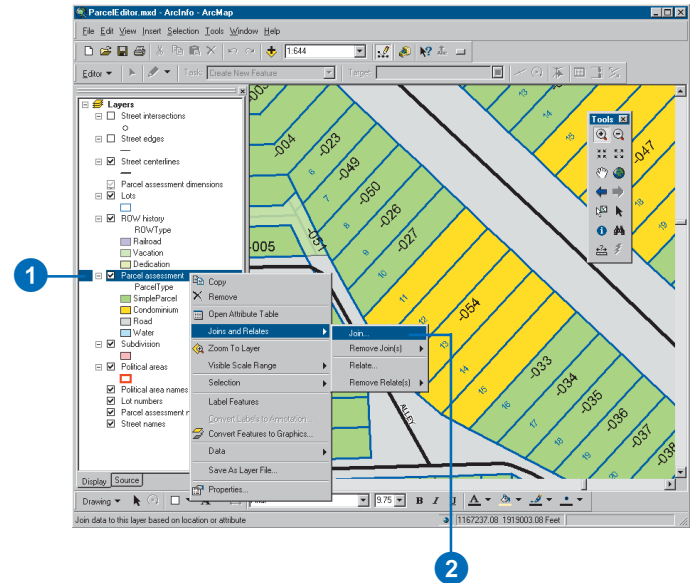
## See Also

*ArcMap has tools for editing relationships. To read more about relationships and editing in ArcMap, see Editing in ArcMap.*

## See Also

*If you are not already familiar with how to add data to your map, please refer to Using ArcMap.*

1. Right-click the feature layer in the ArcMap table of contents.
2. Point to Joins and Relates and click Join.
3. Click the Join options dropdown arrow and click Join data based on a predefined relationship class. ▶



## Tip

### 1–M and N–M relationships

*If the relationship class is 1–M or N–M, each feature can have multiple, related objects. In this case, the attributes of the first related object are joined to the feature.*

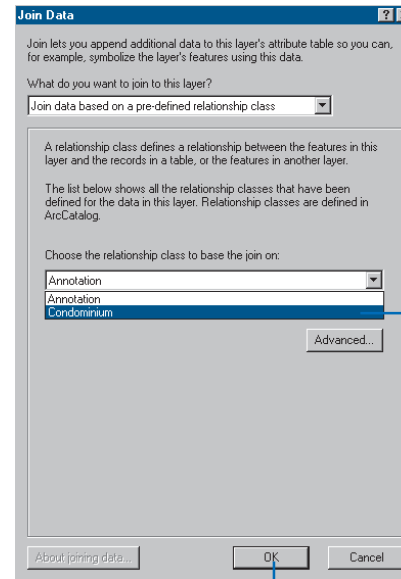
## See Also

*For more information about joins and using joined data in ArcMap, see Using ArcMap.*

4. Click the dropdown arrow to get a list of relationship classes, then click the relationship class.

5. Click OK.

You can now use the related fields for labeling, symbolizing, and querying your features.



# Geometric networks

# 7

## IN THIS CHAPTER

- **What is a geometric network?**
- **Geometric networks and ArcCatalog**
- **Creating geometric networks**
- **Creating a new geometric network**
- **Building a geometric network from existing simple feature classes**
- **Adding new feature classes to your geometric network**
- **Network connectivity: defining the rules**
- **Establishing connectivity rules**
- **Managing a geometric network**

When you model automated mapping/facilities management (AM/FM) or transportation networks, features have connectivity relationships with other features around them. This connectivity is maintained in the geodatabase through an association called a geometric network.

Geometric networks are created and managed using ArcCatalog. This chapter highlights the key tasks for creating and managing geometric networks. Another type of topological association is discussed in the ‘Topology’ chapter in this book.

ArcView can view feature classes that use advanced geodatabase functionality. To create or edit feature classes that take advantage of advanced geodatabase functionality, you need an ArcEditor or ArcInfo license.



# What is a geometric network?

The movement of people, the transportation and distribution of goods and services, the delivery of resources and energy, and the communication of information all occur through definable network systems. In the geodatabase, networks are modeled as a one-dimensional nonplanar graph, or geometric network, that is composed of features. These features are constrained to exist within the network and can, therefore, be considered network features. The geodatabase automatically maintains the explicit topological relationships between network features in a geometric network. Network connectivity is based on geometric coincidence, hence the name geometric network.

A geometric network has a corresponding logical network. The geometric network is the actual set of feature classes that make up the network. The *logical network* is the physical representation of the network connectivity. Each element in the logical network is associated with a feature in the geometric network.

Once a geometric network is in place, ArcMap and ArcCatalog have tools that treat the network features in a special way. Editing and *tracing* on the network, as well as managing the feature classes participating in the network, are all handled automatically by ArcGIS.

## Network feature types

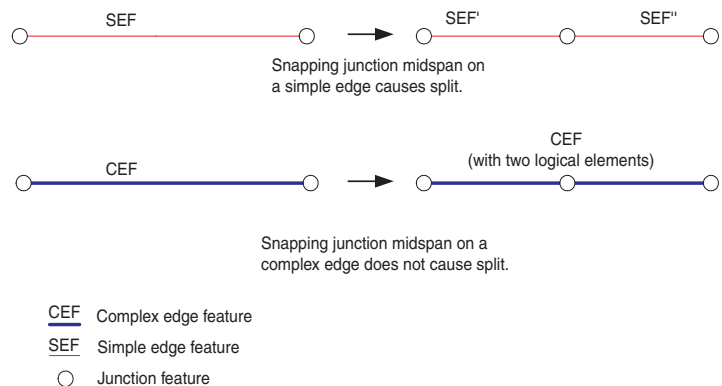
Geometric networks consist of edge network features and junction network features. An example of an edge feature is a water main, while a junction feature might be a valve. Edges must be connected to other edges through junctions. Edge features are related to *edge elements* in the network, and junction features are related to *junction elements* in the logical network.

There are two broad categories of network feature types: simple and complex. Simple network features correspond to a single network element in the logical network. A complex network feature

corresponds to more than one network element in the logical network.

A simple edge feature corresponds to a single network edge element in the logical network. Simple edges are always connected to exactly two junction features, one at each end. If a new junction feature is snapped midspan on a simple edge (thereby establishing connectivity), then that simple edge feature is physically split into two new features.

Complex edges correspond to one or more edge elements in the logical network. Complex edges are always connected to at least two junction features at their endpoints but can be connected to additional junction features along their length. If a new junction feature is snapped midspan on a complex edge, that complex edge remains a single feature. Snapping the junction does cause the complex edge to be split logically—for example, if it corresponded to one edge element in the logical network before the junction was connected, it now corresponds to two edge elements.



*Simple edge features are connected to exactly two junction features. Complex edges can be connected to two or more junction features.*



## Orphan junction feature class

When the first feature class is added to the geometric network, a simple junction feature class is created, called the orphan junction feature class. The name of the orphan junction feature class corresponds to the name of the geometric network appended with “\_Junction”. The orphan junction feature class is used by the geometric network to maintain network integrity. Every edge in the geometric network must have a junction connected to its endpoints. During network building, an orphan junction will be inserted at the endpoint of any edge at which a geometrically coincident junction does not already exist. Orphan junction features can be removed from the geometric network by subsuming them with other junction features. See the ArcGIS online Help system for more information.

## Sources and sinks

Networks are often used to model real-world systems in which the direction of movement through the network is well defined. For example, the flow of electricity in an electrical network is from the power generation station to the customers. In a water network, the flow direction may not be as well defined as in an electric network, but the flow of water may be from a pump station to a customer or from customers to a treatment plant.

Flow direction in a network is calculated from a set of sources and sinks. In the above examples, electricity and water flow are driven by sources and sinks. Flow is away from sources, such as the power generation station or a pump station, and toward sinks such as a water treatment plant (in the case of a wastewater network).

Junction features in geometric networks can act as sources or sinks. When you create a new junction feature class in a network, you can specify whether the features stored in it can represent sources, sinks, or neither in the network. If you specify that these

features can be sources or sinks, a field called *AncillaryRole* is added to the feature class to record if the feature is a source, sink, or neither a source nor a sink. When you calculate the flow direction for a geometric network in ArcMap, the flow direction will be calculated based on the sources and sinks in the network.

For example, you may have a tank in your water network that is down for maintenance, so its role in the network will be changed (temporarily) from source to none. The flow for the network is recalculated by the system; any traces on the network will be affected by the change in flow direction caused by the state of the tank. For more information on calculating flow direction and using flow direction in network analysis, see *Using ArcMap*.

## Network weights

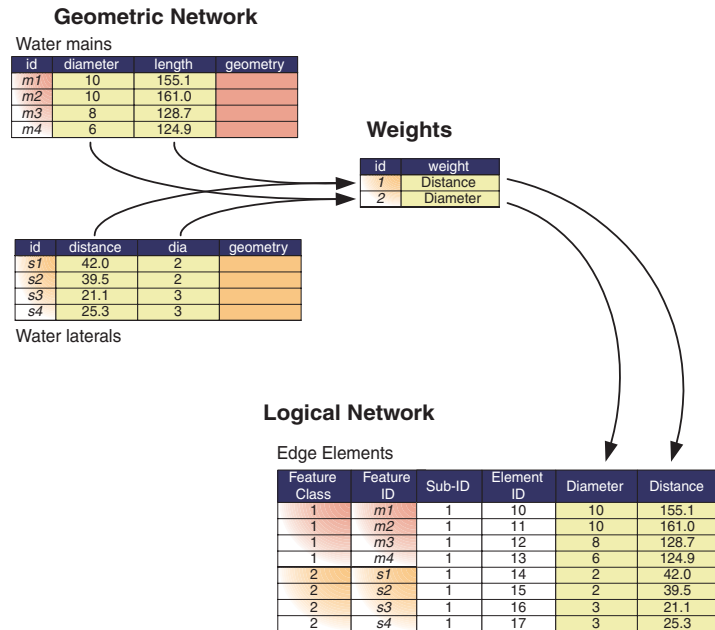
A network can have a set of weights associated with it. A weight can be used to represent the cost of traversing an element in the logical network. For example, in a water network, a certain amount of pressure is lost when traveling the length of a transmission main due to surface friction within the pipe.

Weights are calculated based on some attribute of each feature. In the transmission main example above, an attribute that affects the weight would be the length of the feature.

A network can have any number of weights associated with it. Each feature class in the network may have some, all, or none of these weights associated with its attributes. The weight for each feature is determined by some attribute for that feature. Each weight can be associated with one attribute in a feature but, at the same time, can be associated with multiple features. For example, a weight called *Diameter* can be associated with the attribute *Diameter* in water main features and can also be associated with the attribute *Dia* in water lateral features.

A network weight value of zero is reserved and is assigned to all orphan junction features. Also, if a weight is not associated with

any attributes of a feature class, then the weight values for all network elements corresponding to that feature class will be zero.



*A network can have any number of weights associated with it. Each feature class in the network may have some, all, or none of these weights associated with its attributes. The weight for each feature is determined by some attribute for that feature.*

## Enabled and disabled features

Any edge or junction feature in a geometric network may be enabled or disabled in the logical network. A feature that is disabled in the logical network acts as a barrier. When the network is traced, the trace will stop at any barriers it encounters in the network including disabled network features.

The enabled or disabled state of a network feature is a property maintained by an attribute field called Enabled. It can have one of two values: true or false. When building a geometric network from simple feature classes, this field is automatically added to the input feature classes. When you use ArcCatalog to create a network feature class, Enabled is a required field for the feature class.

For a discussion on required fields, see the chapter ‘Creating new items in a geodatabase’ in this book. When adding new features to a network, they are enabled by default. For more information on editing geometric networks, see *Editing in ArcMap*.

Values stored in the network weight, ancillary role, and enabled fields are the user’s view of the state of the feature in the logical network. When analysis—such as tracing and flow direction calculation—is performed against a network feature, the value of these fields within the feature is not directly referenced to determine the enabled, ancillary role state of the feature or its weight. Instead, these states of the feature are stored in the logical network, which is queried during these operations. This is done for performance reasons.

When you edit a network feature and change the value of the enabled, ancillary role or a weight field, the state of the feature in the internal topology tables is modified to remain in sync with the field values of the feature.

## Performance considerations

Geometric networks can be composed of a number of edge and junction feature classes. When editing geometric networks in ArcMap, connectivity between features is maintained while editing on the fly. The benefit of this model is that there is no need to perform a post editing process to build connectivity for the geometric network. The cost of continually maintaining

network connectivity imposes overhead on the time it takes to add or modify features in network feature classes.

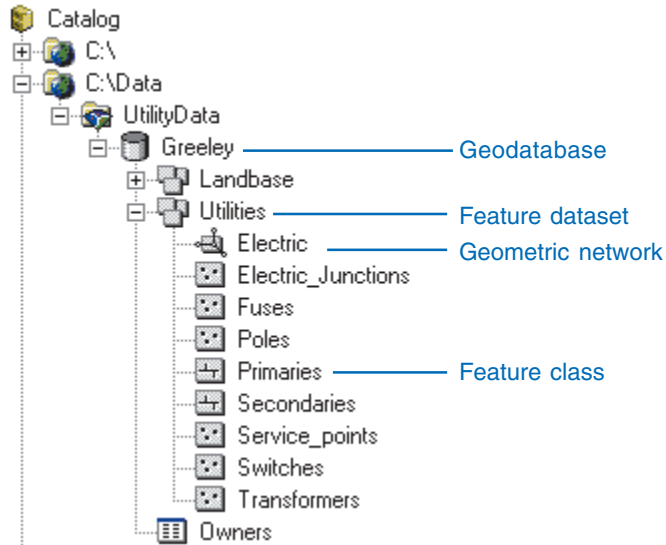
Connectivity in a network feature class is based on geometric coincidence. If a junction is added along an edge, the edge and junction will become connected to one another. When a new feature is added to a network feature class, this geometric coincidence must be discovered. So, each feature class in the network must be analyzed by performing a spatial query against it to determine if the new feature is coincident with network features. If coincidence is discovered, then network connectivity is established.

When discovering connectivity, a separate spatial query must be executed on the server for each feature class in the network. If you use the *map cache* while editing the network, these spatial queries do not need to go against the server and are much faster. You will not pay as much of a penalty in performance for having a large number of feature classes in your network if you use the map cache. Using the map cache when editing your network features will significantly improve performance when adding new features or connecting and moving existing features. For more information on editing geometric networks and using the cache, see *Editing in ArcMap*.

Try to reduce the number of feature classes you have in your geometric network by lumping feature classes together using subtypes. If your feature classes carry different attributes, you can use relationships to manage subtype-specific attributes in different tables in the database, or you can keep all the attributes in the same table, using nulls for those that don't apply to a particular subtype.

# Geometric networks and ArcCatalog

In ArcCatalog, you can view and manage geometric networks in geodatabases that you have access to. Because all geometric networks must be inside a *feature dataset*, they appear in the ArcCatalog tree under their feature datasets.



It is not immediately evident in the ArcCatalog tree which feature classes participate in the network, which participate in which network, and which participate in none. However, by examining the properties of both geometric networks and feature classes, the network feature classes can be determined.

ArcCatalog also contains various tools to create, delete, and manage both geometric networks and the feature classes that participate in geometric networks. These tools are discussed in more detail later in this chapter.

# Creating geometric networks

A geometric network is a connectivity relationship between a collection of feature classes in a feature dataset. Each feature has a role in the geometric network of either an edge or a junction. Multiple feature classes may have the same role in a single geometric network.

The basic methodology for creating a geometric network is to determine which feature classes will participate in the network and what role each will play. Optionally, a series of network weights can be specified, as can other more advanced parameters.

Two different methods are available for creating a network. These are creating a new, empty geometric network and building a geometric network from existing simple features.

## Creating a new, empty network

In ArcCatalog you can create, design, and build a geometric network from scratch. You can then use editing tools in ArcMap or custom Visual Basic® (VB), Visual Basic for Applications (VBA), or C++ code to add features to the geometric network.

The process of creating a network can be summarized in the following steps:

1. Use ArcCatalog to create the feature dataset that will contain the geometric network and its feature classes.
2. Use ArcCatalog to create an empty geometric network in the feature dataset.
3. Use ArcCatalog to create new feature classes in the feature dataset and assign each a role in the geometric network.
4. Use ArcCatalog to establish *connectivity rules* for elements of the geometric network.
5. Use custom scripts or ArcMap editing tools to add features to the network.

## Building a geometric network from existing data

You may already have data from which you want to create a geometric network in your geodatabase. ArcCatalog contains a wizard that creates a geometric network from existing data.

The process of building a geometric network from existing data can be summarized in the following steps, each performed in ArcCatalog:

1. Convert and load your data into a geodatabase.
2. Build a geometric network from existing simple feature classes.
3. Add any additional empty feature classes to the geometric network.
4. Establish connectivity rules for the geometric network.

## How networks are built

Building networks from existing features is a computationally intense operation that may take a considerable amount of time and system resources, depending on the number of input features. If those features require snapping, the network building operation will spend most of its time in the feature snapping phase. The network building process proceeds in the following sequence:

1. If snapping is specified, snap simple features.
2. If snapping is specified, snap complex features.
3. Create an empty logical network.
4. Create the network schema in the database.
5. Extract attributes from the input feature classes for weight calculations.
6. Create the topology.

7. Create orphan junctions as required, add input junction features to the logical network, and initialize the junction-enabled values.
8. Set weight values for the junction elements.
9. Add edges to the logical network.
10. Set weight values for the edge elements.
11. Create all necessary indexes in the database.

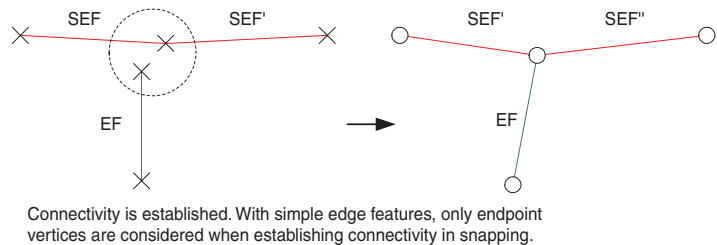
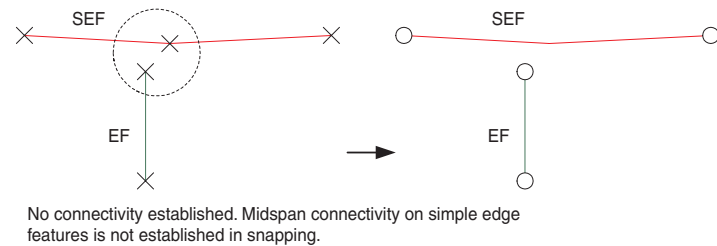
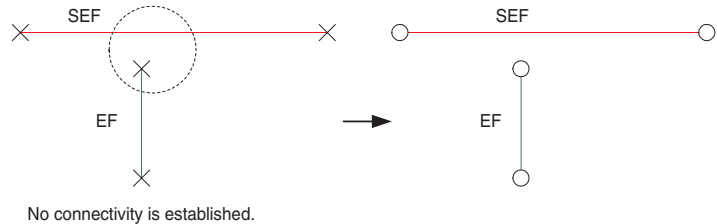
## Network snapping models

Ideally, your data should be clean before you build a network. Clean data means that all features that should be connected in the network are geometrically coincident—that is, no *overshoots* or *undershoots*. However, if this is not the case, the data may be snapped during the network building process.

It is important to understand how connectivity is established based on snapping during the network building process and how feature geometries are adjusted to establish that connectivity. The following is a series of examples of how connectivity is established in given scenarios. Use the key below to identify what types of features are illustrated in each scenario:

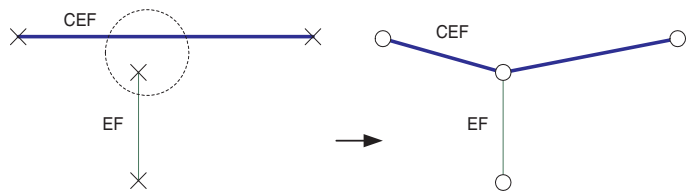
<u>EF</u>	Edge feature (simple or complex)
<u>CEF</u>	Complex edge feature
<u>SEF</u>	Simple edge feature
×	Vertex
● ○	Junction features
○	Snapping tolerance

Simple edges: Connectivity against simple edges is established only at the ends of edge features. Midspan connectivity will not be established, even if there is a vertex along the simple edge feature.

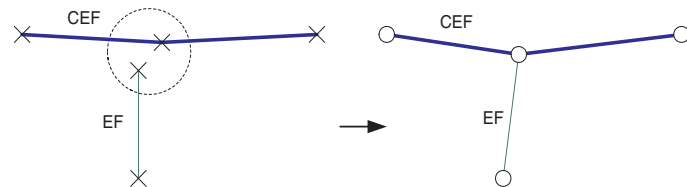


### Simple edge connectivity models

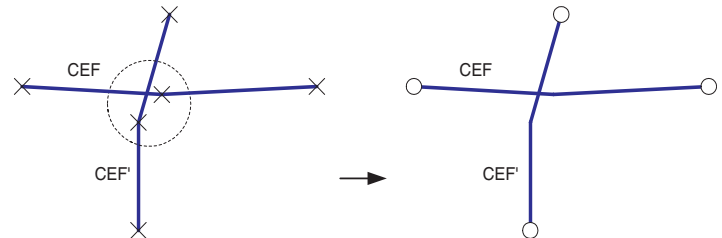
Complex edges: Connectivity against complex edges is established both at the ends of features and midspan. If there is no vertex along the complex edge where connectivity is established, a new vertex is created. When snapping complex edges, connectivity must be at the endpoint of at least one of the edges. Connectivity will not be established between the midspan of one edge and the midspan of another edge.



Connectivity established. Intersection detection is performed along complex edges, and new vertices are inserted as required.



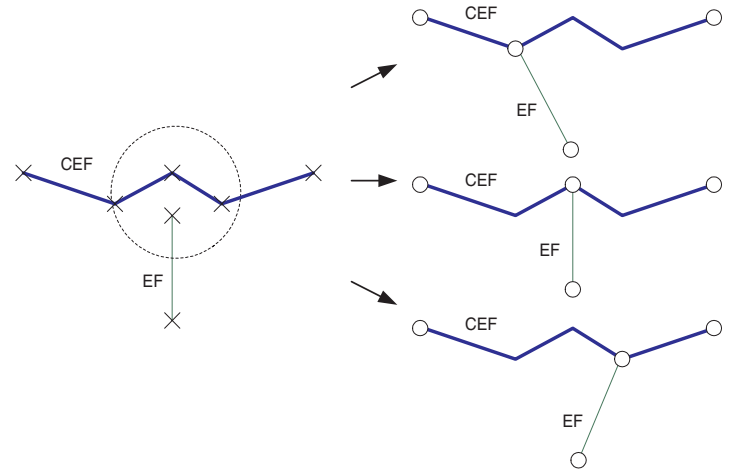
Connectivity established. Midspan connectivity on complex edge features is established in snapping.



No connectivity established. Connectivity must be at an endpoint of one of the two edge features.

*Complex edge connectivity models*

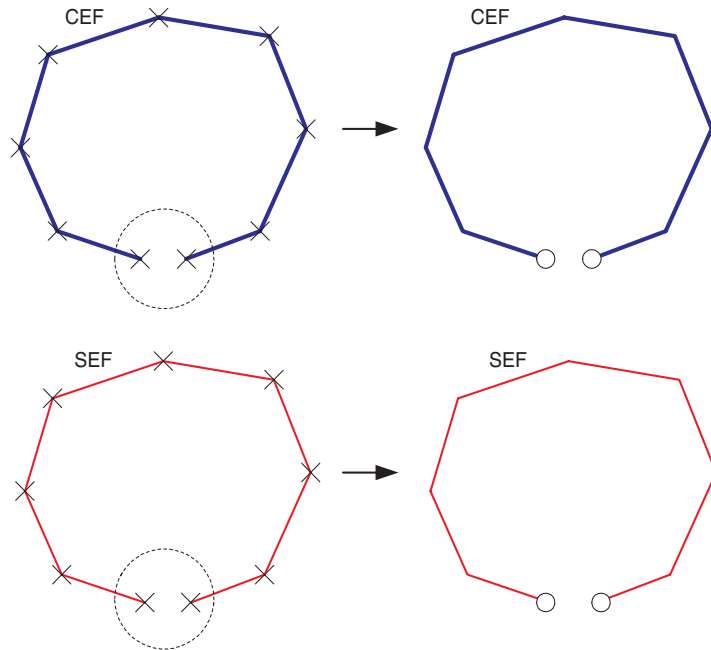
Vertex clustering: When snapping two features, if there is more than one vertex within the *snapping tolerance*, then those vertices are treated as a cluster. Snapping will occur to one of the vertices in the cluster, but not necessarily the closest.



Connectivity established. One of the vertices within the snap tolerance is snapped to, but not necessarily the closest.

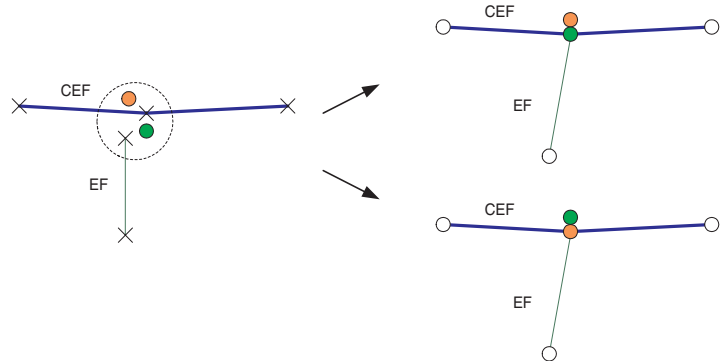
*Network snapping vertex clustering does not guarantee the closest vertex is snapped to—it may be any of the vertices.*

Connecting features to themselves: When the endpoints of a single edge feature come within the snapping tolerance of itself, the endpoint will not be snapped and no connectivity will be established. Connectivity is not established between a feature and itself.



No connectivity established. Connectivity is not created between features and themselves.

Coincident junctions: When the network building process encounters coincident junctions, or when the snapping process results in coincident junctions, the resulting connectivity will be nondeterministic. That is, connectivity will only be established to one of the coincident junctions.



*Connectivity is nondeterministic when coincident junctions are encountered.*

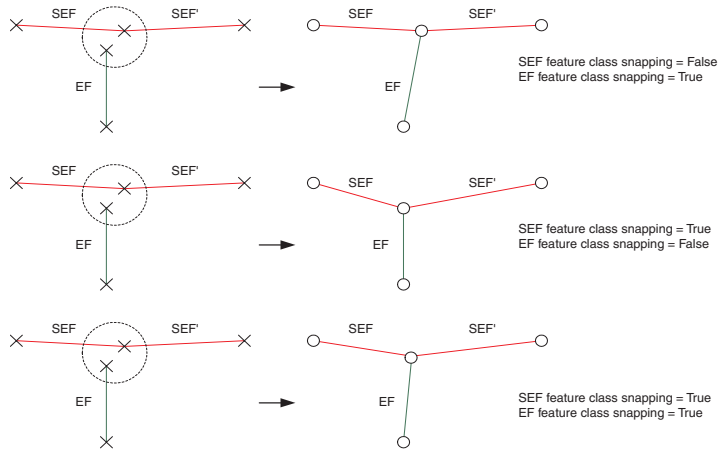
## Adjusting features

When snapping features during network building, it is important to understand how the geometry of features is adjusted when snapping. All or part of any feature in a feature class that was specified as being adjustable in the Build Geometric Network wizard can potentially be moved. Those features that are in feature classes that are not adjustable will remain fixed throughout the network building process.

All features in all feature classes have equal weights when being adjusted during snapping. This means that if the endpoints for two edges need to be snapped and both features can be adjusted, then they will move an equal distance to snap together. If one of



the features is not adjustable, then only the adjustable feature will move to snap to the static feature.



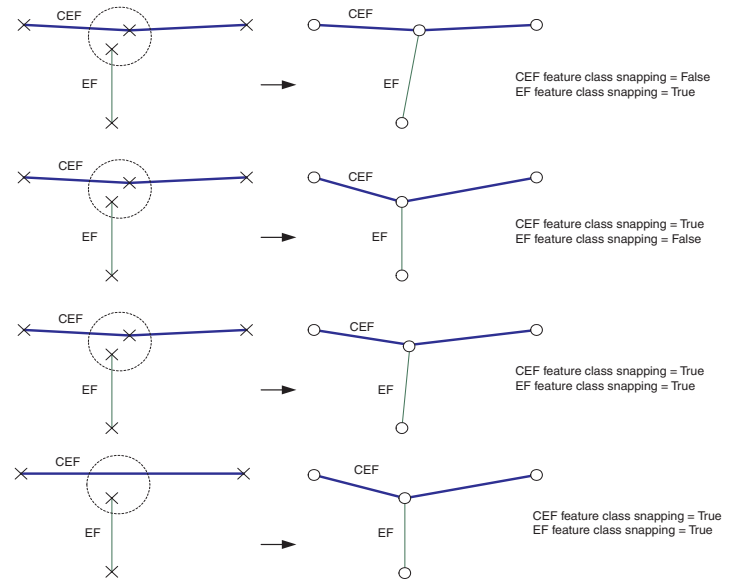
*How simple edge features are adjusted depends on whether the features they are snapping to can or cannot be adjusted.*

## Identifying network build errors

When building a geometric network, the feature classes that are selected to participate in the network may contain features whose geometries are invalid within the context of a geometric network. These geometries include:

- features that have empty geometry
- edge features that contain multiple parts
- edge features that form a closed loop
- edge features that have zero length

At the end of the build process, a summary of these errors displays in a dialog box. Features with invalid geometries are also identified and recorded in the network build errors table. The



*How complex edge features are adjusted depends on whether the features they are snapping to can or cannot be adjusted.*

table lists the Object ID, Class ID and the reason why a feature's geometry is invalid. The table is located at the workspace level and named as the geometric network's name appended with '\_BUILDERR'. For example, a network called 'MyNetwork' will have a network build errors table called 'MyNetwork\_BUILDERR'.

The table is used by the Network Build Errors command within ArcMap to identify the features with invalid geometries. The network build errors table does not get updated when you edit features, so you should update the table's contents as soon as possible after editing features. For information on how to repair network feature geometry, please see the ArcGIS online Help system.

## Schema locking

An exclusive lock is required on all of the input feature classes when building a geometric network. If any of the input feature classes have a shared lock, the network will not be built.

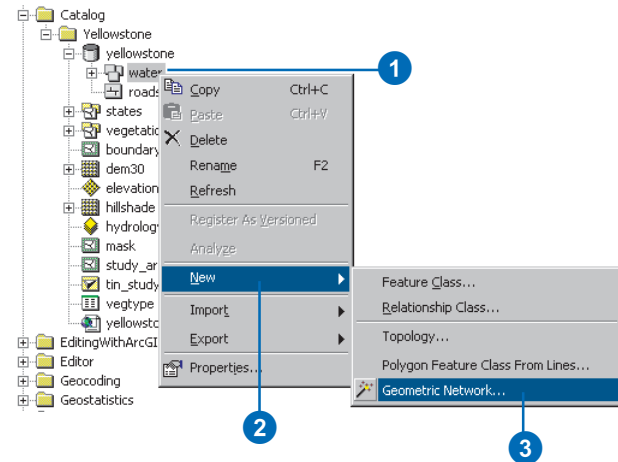
If any of the feature classes in a network have a shared or exclusive lock, that lock is propagated to all of the other feature classes in the network. For more information on exclusive locks and schema locking, see the chapter 'Creating new items in a geodatabase' in this book.

# Creating a new geometric network

Geometric networks are created inside feature datasets. Once a geometric network has been created, you must add feature classes to the feature dataset and assign them roles in the network.

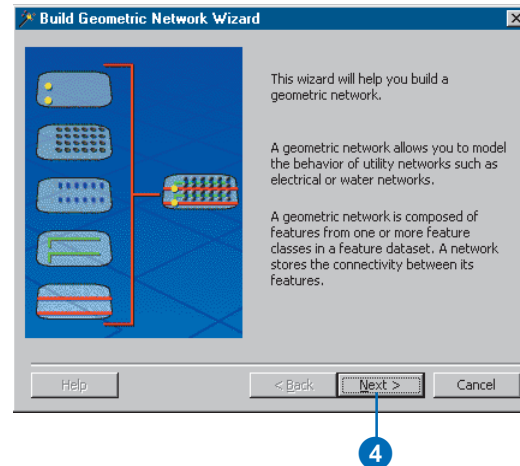
New feature classes can be added to a geometric network at any time.

1. Right-click the feature dataset that will contain the network.
2. Point to New.
3. Click Geometric Network.
4. Read the information on the first panel and click Next.

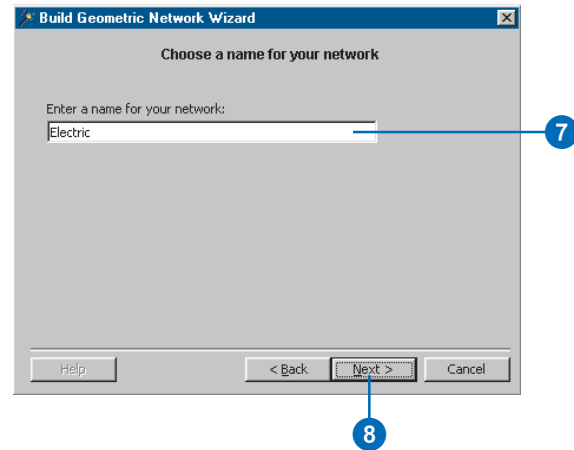
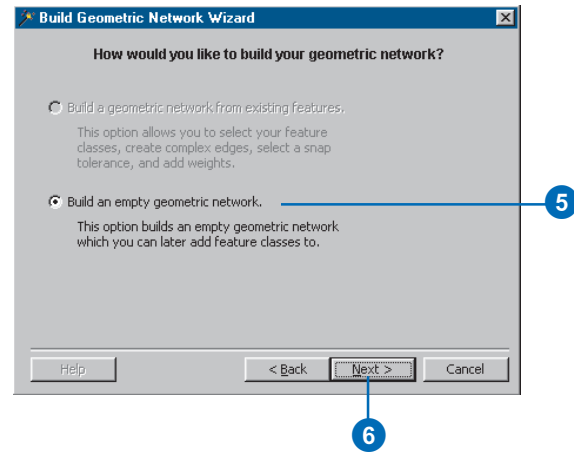


## See Also

*For more information on creating feature datasets and feature classes, see the chapter 'Creating new items in a geodatabase' in this book.*



5. Click the second option to build an empty geometric network.
6. Click Next.
7. Type a name for the new geometric network.
8. Click Next. ►



## Tip

### Network weights

*Network weights apply to all elements in the network. You can assign which weights are associated with which field on each feature class when you create the network feature class.*

*You can't remove or add weights once the geometric network is created.*

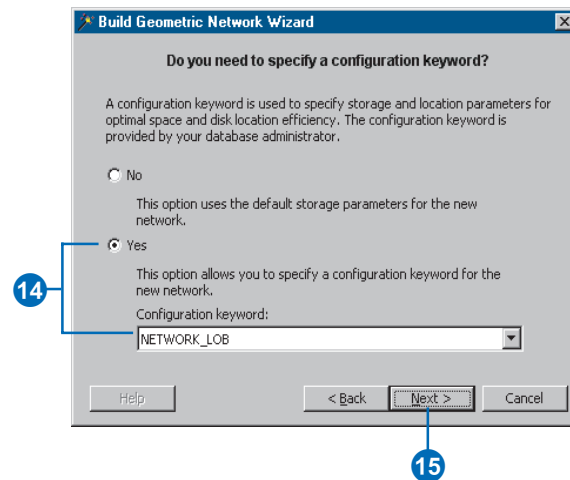
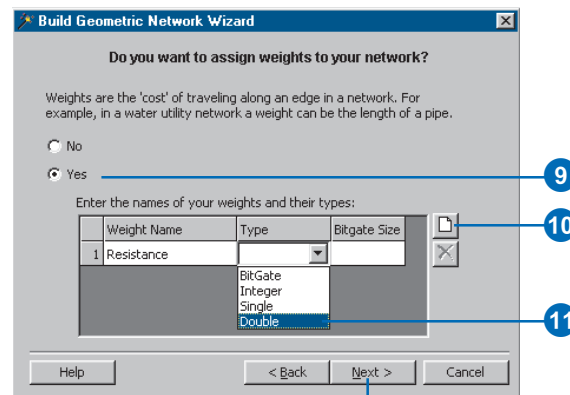
## See Also

*For more information on geometric networks and network weights, see Modeling Our World.*

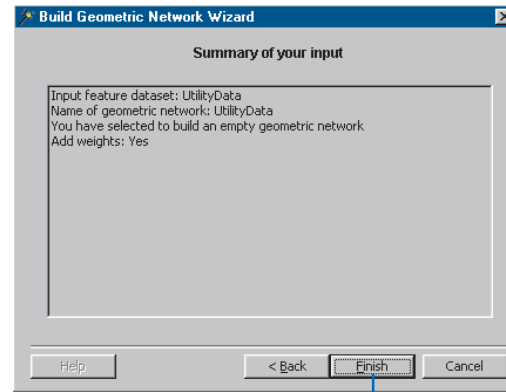
## See Also

*For details about using storage keywords with ArcSDE, see Managing ArcSDE Services.*

- Click Yes if you want to include weights in the network; otherwise, skip to step 13.
- Click the New button and type a name to add a new weight.
- Click the dropdown arrow and click the weight type.
- Repeat steps 10 and 11 until all of the network's weights have been defined.
- Click Next.
- Click Yes and type the name of the keyword if your geodatabase is stored in an ArcSDE database and you have a configuration keyword for the network storage. If not, skip to step 15.
- Click Next. ►



16. Review the options you specified for your new network. If you want to change something, you can go back through the wizard by clicking the Back button.
17. Click Finish to create the new geometric network.



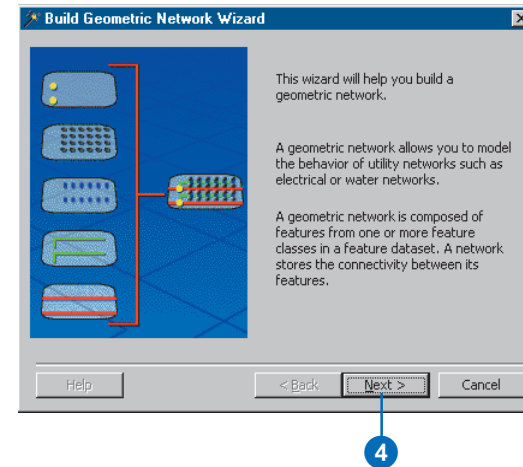
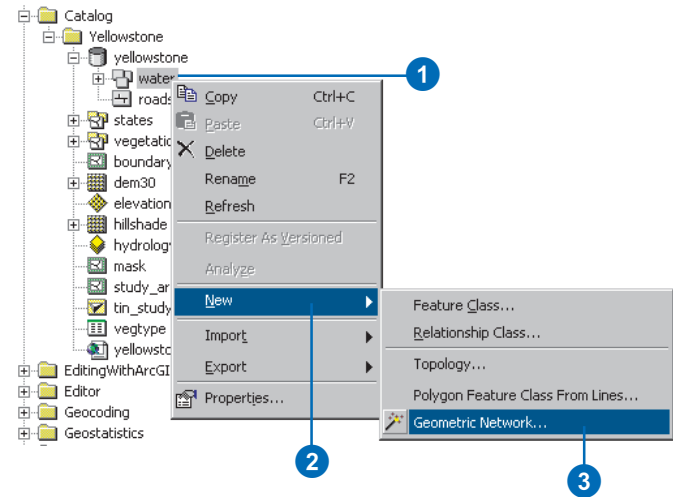
## Building a geometric network from existing simple feature classes

An alternative to creating and populating an empty geometric network is to build a geometric network from existing simple feature classes.

The Build Geometric Network wizard discovers the connectivity for a group of feature classes in a feature dataset and promotes them from simple feature types (lines and points) to network feature types (edges and junctions).

When you build a geometric network, the feature classes must already exist in the feature dataset. However, they can be empty. After the network has been built, you can add new empty network feature classes.

1. Right-click the feature dataset that will contain the network.
2. Point to New.
3. Click Geometric Network.
4. Read the information on the first panel and click Next. ▶



## Tip

### Versioned data

When building a geometric network from simple feature classes in an ArcSDE geodatabase, the input feature classes cannot be versioned.

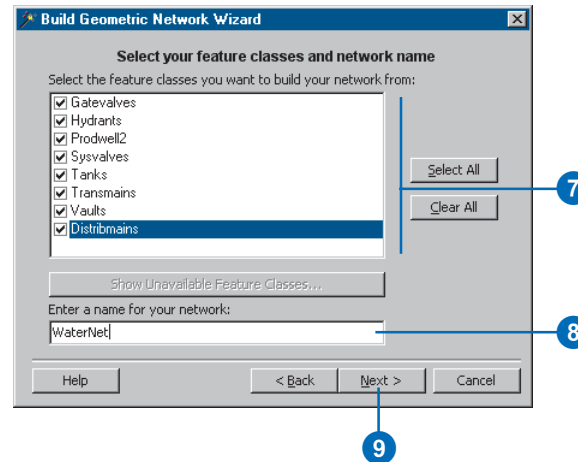
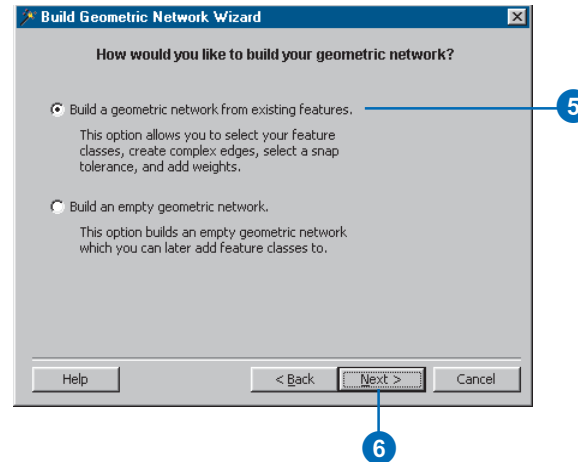
For more information on versions, see the chapter 'Working with a versioned geodatabase' in this book.

## Tip

### Input schema

All network feature classes require a short integer field named Enabled to record whether or not a feature is enabled or disabled in the logical network. The network building wizard will automatically add this field to your input feature classes.

5. Click the first option to build a geometric network from existing features.
6. Click Next.
7. Click the feature classes that you wish to include in this geometric network.
8. Type a name for the new geometric network.
9. Click Next. ▶





## Tip

### Complex edges

When you build a geometric network from existing simple feature classes, the line feature classes become simple edges by default. However, you can specify that you want some of the line feature classes to be complex edges in the geometric network.

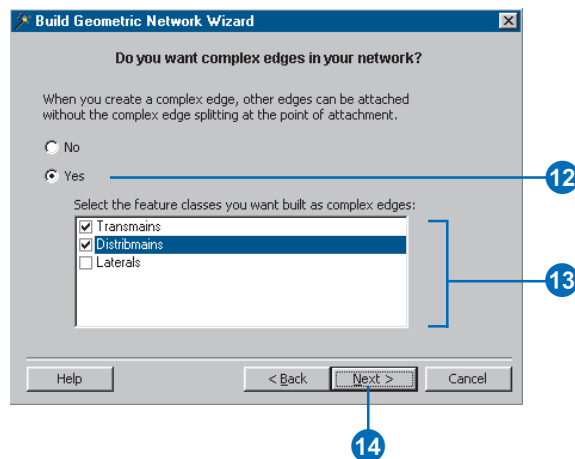
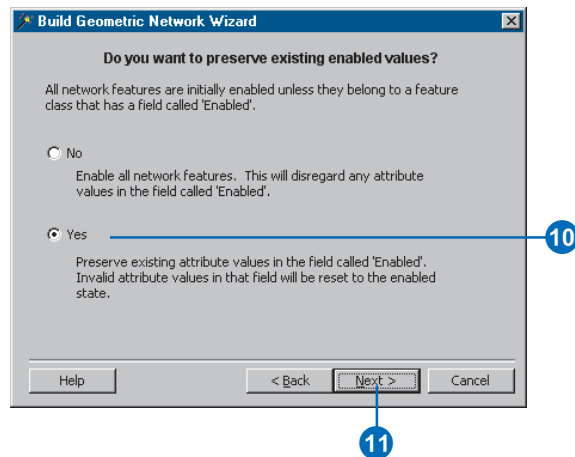
## Tip

### Snapping features

The geometric network builder can automatically adjust features in the input feature classes to correctly snap to connecting features. The default snapping tolerance is  $1.5 * 1/XY$  scale of the feature dataset's spatial reference.

If snapping, you cannot use a value smaller than the default. Large snapping tolerances may cause unanticipated results. For best results, examine your data and provide an appropriate tolerance. Snapping (geometry changes) cannot be undone.

10. If any of the feature classes used to be part of a network, you can choose to keep enabled values for the new network you are creating. If this panel doesn't display, skip to step 12.
11. Click Next.
12. Click Yes if you want some of the input line feature classes to become complex edges, otherwise skip to step 14.
13. Check the line feature classes that you want to become complex edges. Those that are not selected will become simple edges.
14. Click Next. ►



## Tip

### Sources and sinks

If you specify that you want to store sources and sinks in a junction feature class, a field called *AncillaryRole* will automatically be added to the feature class.

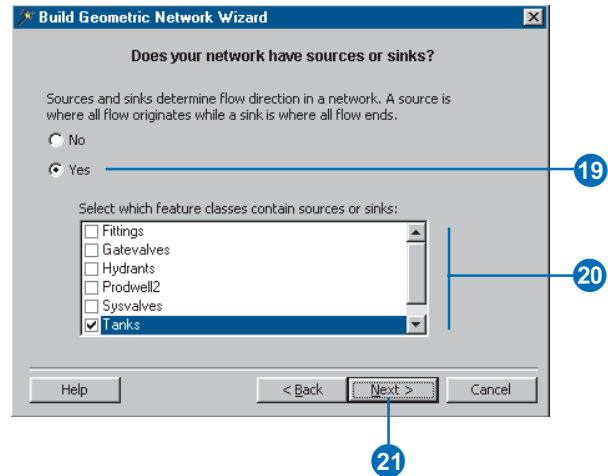
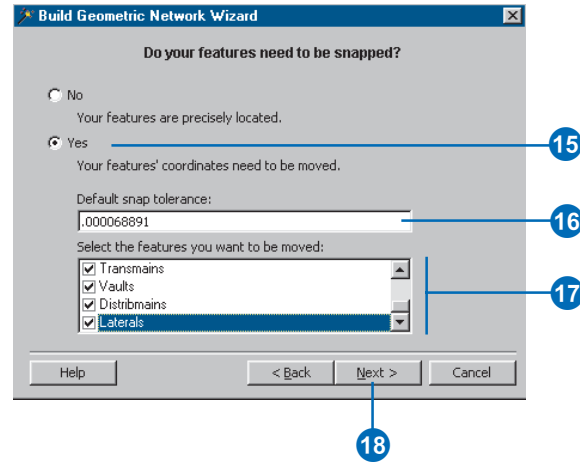
## Tip

### Network weights

Once the geometric network is built, no additional weights can be added to it. Also, the feature class field to which a weight is associated can't be altered.

When you add a new feature class to the geometric network, it can be associated with the existing network weights.

15. Click Yes if you want features in some of the input feature classes to be automatically adjusted and snapped during the network building process; otherwise, skip to step 18.
16. Type a *snapping tolerance* if you don't want to use the default tolerance.
17. Check the feature classes whose features you want automatically adjusted and snapped. Feature classes that are not selected are not adjusted.
18. Click Next.
19. Click Yes if you want features in some of your junction feature classes to be able to act as sources or sinks; otherwise, skip to step 21.
20. Check those junction feature classes that you want to store sources and sinks.
21. Click Next. ►



## Tip

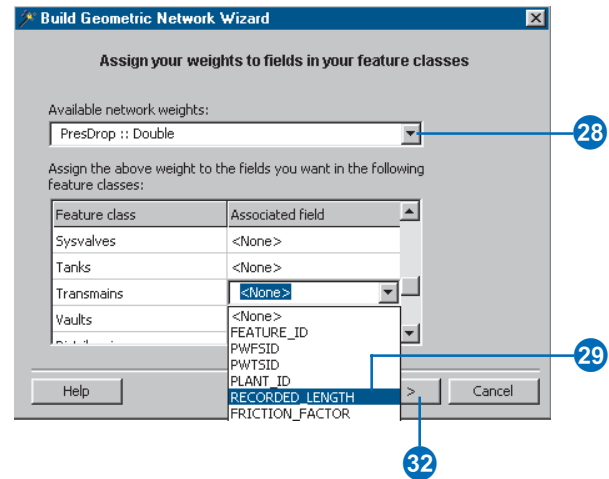
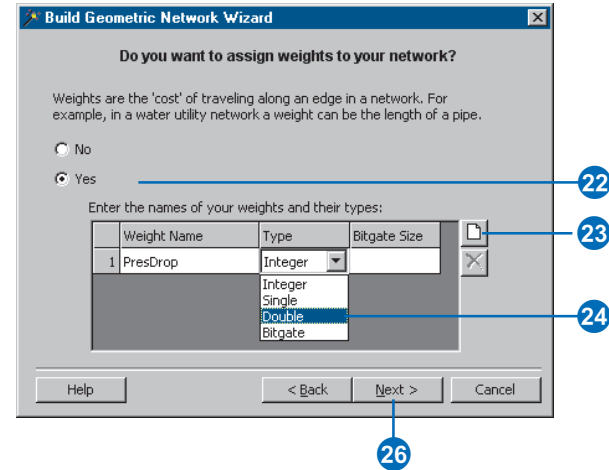
### Building progress

You will be notified of the building progress with a series of progress bars indicating the progress of each step along the way.

## See Also

For details about using storage keywords with ArcSDE, see Managing ArcSDE Services.

22. Click Yes if you want to add weights to your network. Otherwise, skip to step 26, then skip over steps 27 through 31.
23. Click the New button to add a new weight.
24. Type a name for the new weight, click the dropdown arrow, and click a weight type.
25. Repeat steps 23 and 24 until all of the network's weights have been defined.
26. Click Next.
27. Assign the weights, if you added any, to specific fields in each feature class.
28. Click the dropdown arrow and click the network weight to which you will assign an attribute.
29. Click the dropdown arrow and click the field you want associated with this weight.
30. Repeat step 29 for each feature class that you want to associate with this weight.
31. Repeat steps 28 through 30 until you are finished associating network weights with feature class attributes.
32. Click Next. ►



## Tip

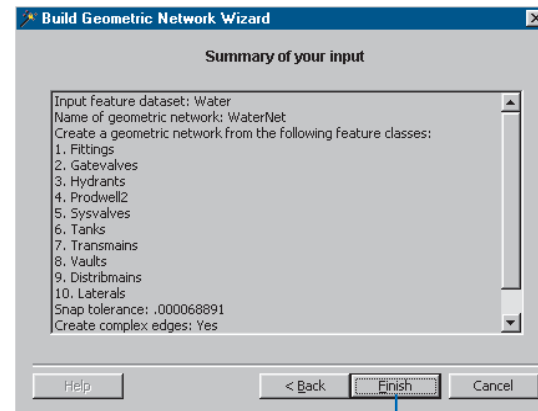
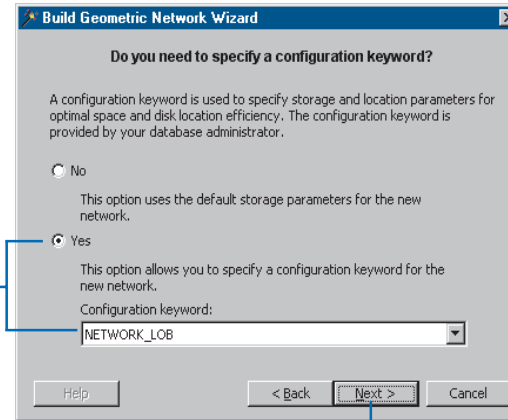
### Aborting

At any time during the building process, you can abort by clicking *Abort* on the *Progress* dialog box.

When you abort the build, the system deletes any network tables created and sets the database to the state it was before the build started.

If snapping was already complete, that change is permanent and will not be restored.

33. Click Yes and type the name of the keyword if your geodatabase is stored in an ArcSDE database and you have a configuration keyword for the network storage. If not, skip to step 36.
34. Click Next.
35. Review the options you specified for your new network. If you want to change something, you can go back through the wizard by clicking the Back button.
36. Click Finish to create the new geometric network.



## Adding new feature classes to your geometric network

At any time in a geometric network's life, you can add new edge and junction feature classes. These new feature classes are empty—you cannot add populated feature classes to an existing geometric network. Adding a new feature class to a geometric network is similar to the task of creating a new feature class to store *custom features*. See the chapter 'Creating new items in a geodatabase' in this book.

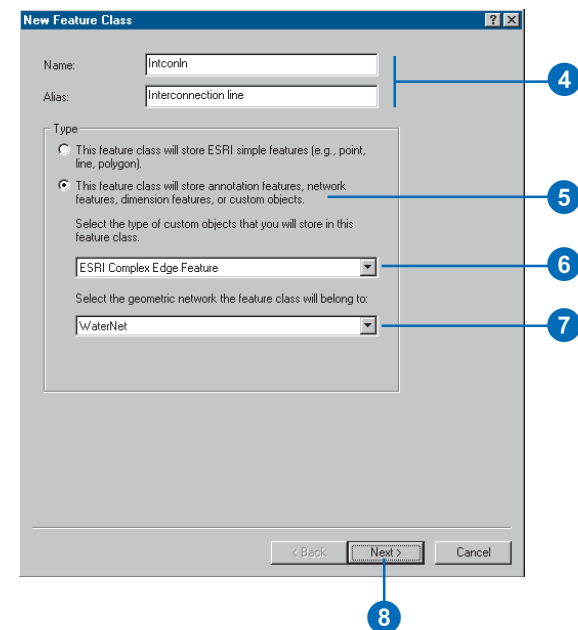
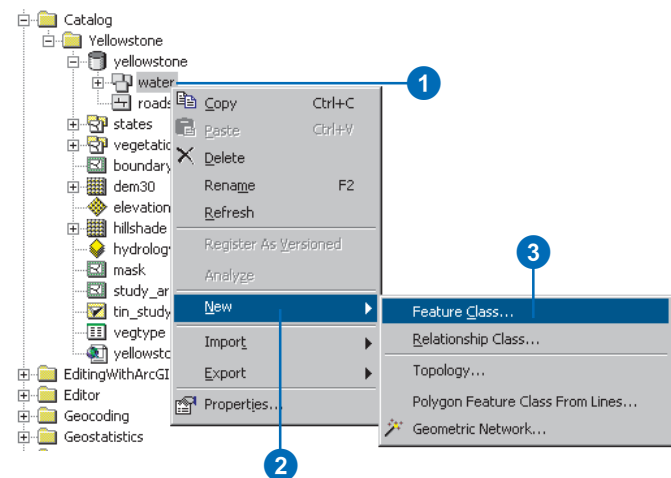
When creating a new network feature class, you must specify a feature type other than simple as well as specify the geometric network in which that feature class will participate. The new feature class must be created in the same feature dataset as the geometric network.

If you create a new junction feature class, you can specify if you want its features to be able to act as sources or sinks.

All network feature classes have the same required fields as simple feature classes—OBJECTID and Shape. In addition to this, network ►

## Creating a new network edge feature class

1. Right-click the feature dataset that contains the network.
2. Point to New.
3. Click Feature Class.
4. Type a name and an *alias* for the new feature class.
5. Click the second option to store network objects in the feature class.
6. Click the dropdown arrow and click ESRI Simple Edge Feature to create a feature class that stores simple edges. Click ESRI Complex Edge Feature to create a feature class that stores complex edges.
7. Click the dropdown arrow and click the geometric network in which this feature class will participate.
8. Click Next. ►





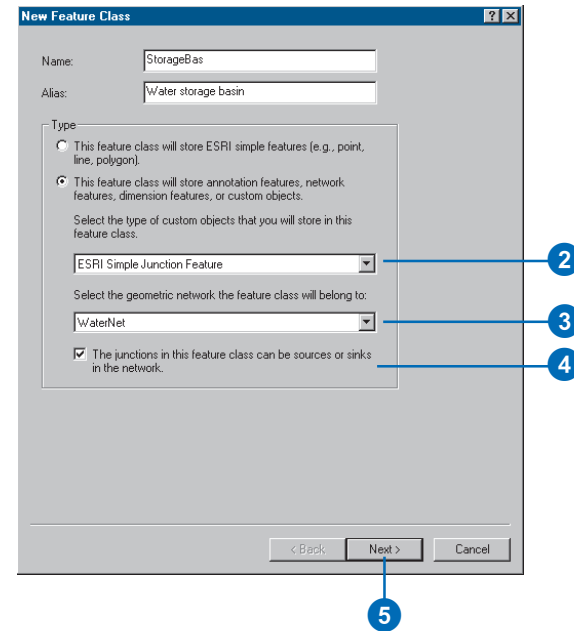
## Tip

### Sources and sinks

If you specify that you want to store sources and sinks in a junction feature class, a field called *AncillaryRole* will automatically be added to it. If not, then the *AncillaryRole* field will not be included in the feature class.

## Creating a new network junction feature class

1. Follow steps 1 through 5 for 'Creating a new network edge feature class'.
2. Click the dropdown arrow and click ESRI Simple Junction Feature to create a feature class that stores network junctions.
3. Click the dropdown arrow and click the geometric network in which this feature class will participate.
4. Check the box to allow the junctions in this feature class to be able to act as sources or sinks in the network.
5. Click Next.
6. Follow steps 9 through 13 for 'Creating a new network edge feature class'.



## Network connectivity: defining the rules

In most networks, you do not want all edge types to be able to logically connect to all junction types. Similarly, not all edge types can logically connect to all other edge types through all junction types. For example, in a water network, a hydrant can connect to a hydrant lateral but not to a service lateral. Similarly, in the same water network, a 10-inch transmission main can only connect to an 8-inch transmission main through a reducer.

Network connectivity rules constrain the type of network features that may be connected to one another and the number of features of any particular type that can be connected to features of another type. By establishing these rules, along with other rules such as attribute domains, you can maintain the integrity of the network data in the database. At any time, you can selectively validate features in the database and generate reports as to which features in the network are invalid—that is, are violating one of the connectivity or other rules.

There are two types of connectivity rules: edge–junction and edge–edge rules. An *edge–junction rule* is a connectivity rule that establishes that an edge of one type may connect to a junction of another type. An *edge–edge rule* is a connectivity rule that establishes that an edge of one type may connect to an edge of another type through a set of junction types. Edge–edge rules always involve a set of junctions.

You can establish and modify the connectivity rules for a network from within ArcCatalog by modifying the geometric network properties. You can establish connectivity rules between two feature classes, a feature class and the *subtype* of another feature class, or a subtype of one feature class and a subtype of another. In the water network example above, a connectivity rule would be established between two subtypes of the same edge feature class and a subtype of a third junction feature class (10-inch and 8-inch transmission mains and reducer valves).

### Default junctions

Both edge–edge and edge–junction connectivity rules can have default junctions associated with them. Default junctions are automatically inserted by ArcMap when creating connectivity in a network.

When an edge pair has an edge–edge connectivity rule defined in the database and you create a new edge that connects to an existing edge, the default junction is automatically inserted. For an edge–junction connectivity rule, ArcMap automatically inserts the default junction at the free end of new edges that are created in the network.



# Establishing connectivity rules

Connectivity rules are established and modified using the geometric network's Properties dialog box in ArcCatalog.

The two examples given here describe how to establish an edge–junction rule and an edge–edge rule. For simplicity, each is done separately in the examples, but any number of rules can be established or modified for the network at a single time.

## Tip

### Junction rules

*If an edge–junction rule does not yet exist between one of the edge subtypes or feature classes and one of the junction subtypes or feature classes, a rule is automatically created.*

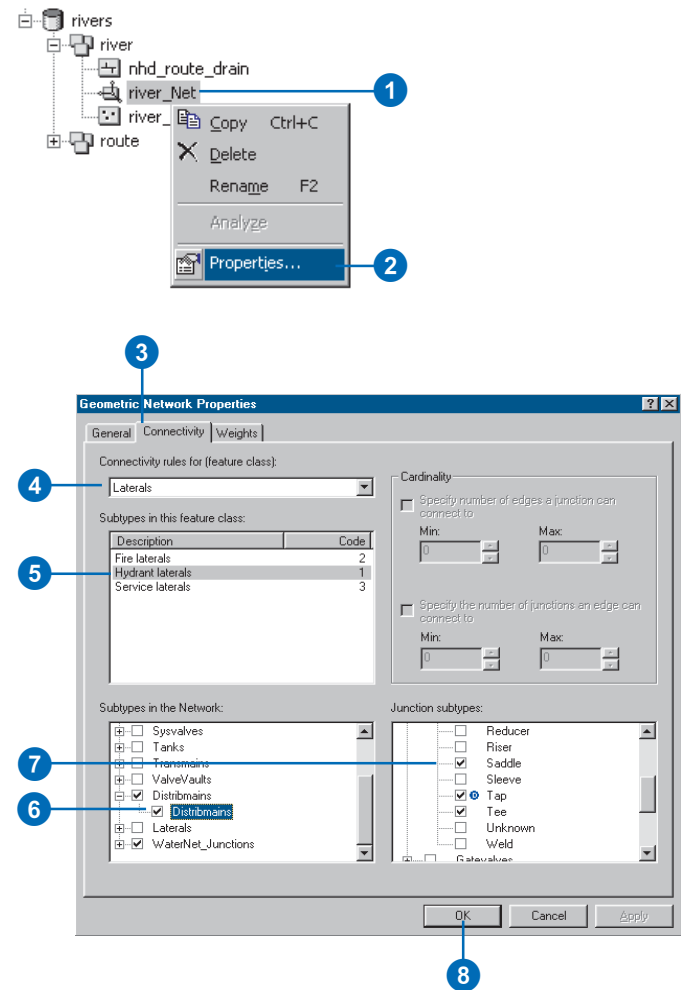
## Tip

### Default junction type

*To set a default junction type, right-click the junction subtype or feature class in the Junction subtypes list, then click Set as default.*

## Adding an edge–edge rule

1. Right-click the geometric network.
2. Click Properties.
3. Click the Connectivity tab.
4. Click the dropdown arrow and click the feature class for which you want to create a rule.
5. Click the subtype of the feature class if your feature class has subtypes.
6. Navigate to and check the edge feature class or subtype you want to make connectable to this edge subtype or feature class.
7. Browse to and check the junction feature classes and subtypes through which these edge feature classes or subtypes will be permitted to connect.
8. Click OK to create the rule in the database.



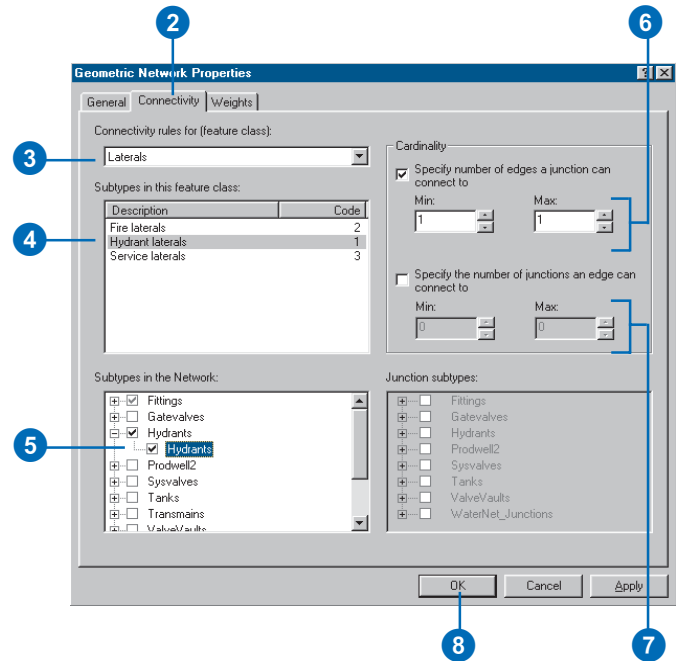
## Tip

### Default junction type

To set a default junction type, right-click the junction subtype or feature class in the Junction subtypes list, then click Set as default.

## Adding an edge–junction rule

1. Follow steps 1 and 2 for 'Adding an edge–edge rule'.
2. Click the Connectivity tab.
3. Click the dropdown arrow and click the feature class for which you want to create a rule.
4. Click the subtype of the feature class if your feature class has subtypes.
5. Navigate and check the junction feature class or subtype you want to make connectable to this edge feature class or subtype.
6. Click the check box and enter the minimum and maximum number of permissible edges if you want to restrict the number of edges of this type that can connect to a single junction of this type.
7. Check the check box and type the minimum and maximum number of permissible junctions if you want to restrict the number of junctions of this type that can connect to a single edge of this type.
8. Click OK to create the rule in the database.



# Managing a geometric network

You can manage geometric networks using ArcCatalog. Unlike most items that appear in ArcCatalog, the geometric network does not represent a single entity, such as a table, *shapefile*, or feature class. A geometric network is actually an association among several feature classes and is represented by several tables in the database. Managing a geometric network is different from managing other items in ArcCatalog.

## Managing the geometric network itself

Some of the standard operations on the geometric network work the same way as with other items in ArcCatalog. A geometric network can be copied or deleted. Copying a geometric network persists the network connectivity and feature classes. Deleting a geometric network also deletes the network schema and causes the network feature classes to revert to simple feature classes.

Geometric networks can be copied in two ways: either by copying the feature dataset containing the geometric network or copying the geometric network itself. If the geometric network is copied, the target feature dataset must have the same spatial reference and extent as the source feature dataset.

Geometric networks can be deleted by deleting the feature dataset that contains it. This will remove the geometric network and all the feature classes participating in it. You can also delete the geometric network to leave the feature dataset and feature classes intact. When you delete the network this way, all of the feature classes in the network are demoted to simple feature types. Edge feature classes become line feature classes, and junction feature classes become point feature classes. Deleting the network will also delete all of the related network tables and the orphan junction feature class from the geodatabase. However, the enabled and ancillary role fields will not be removed from their

respective feature classes and any snapping performed on the network during building will not be undone.

## Managing network feature classes

Managing network feature classes is more restrictive than managing simple feature classes. While a network feature class's alias can be changed, its name cannot. Deleting a network feature class is also more difficult than deleting simple feature classes. To delete the network feature class, you must first delete the geometric network; this action converts the network feature class to a simple feature class that can then be deleted. The alternative is to delete the entire feature dataset, which deletes the network and all of the feature classes.

## Schema locking

An exclusive lock is required to modify a geometric network's connectivity rules, add a feature class to the geometric network, or delete a geometric network. An exclusive lock can only be acquired for a geometric network if the feature classes that participate in the network can also be locked. Therefore, if a user has an exclusive or shared lock on any of the feature classes in a geometric network, then the properties of the geometric network can't be edited. For more information on exclusive locks and schema locking, see the chapter 'Creating new items in a geodatabase' in this book.



# Managing annotation

# 8

## IN THIS CHAPTER

- **Annotation in the geodatabase**
- **Annotation and ArcCatalog**
- **Creating annotation feature classes**
- **Converting labels to annotation**
- **Importing coverage annotation**

In addition to geometry and location, geographic features may also have some descriptive text associated with them. For example, a feature class that contains streets may have text with the street names associated with it. Annotation may also be a geographically located piece of text that exists independently of any other feature, such as the name of a mountain range on a physical map.

Annotation refers either to the process of automating text placement or to the text itself. This chapter describes how to create annotation for your feature classes and how to convert annotation that you have in coverages to geodatabase annotation.

You can create and store annotation in a personal geodatabase with ArcView. To take advantage of advanced annotation functionality, you need an ArcEditor or ArcInfo license.

# Annotation in the geodatabase

Annotation in the geodatabase is stored in annotation feature classes. Like other feature classes, all features in an annotation feature class have a geographic location and attributes and can either be a standalone feature class or inside a feature dataset. Each annotation feature that is text has symbology including font, size, color, and any other text symbol property. Annotation is typically text, but it can also include graphic shapes—for instance, boxes or arrows that require other types of symbology.

## Standard and feature-linked annotation

There are two kinds of annotation in the geodatabase, standard and feature-linked. Standard annotation is not formally associated with features in the geodatabase. An example of standard annotation is the text on a map for a mountain range. No specific feature represents the mountain range, but it is an area you want to mark.

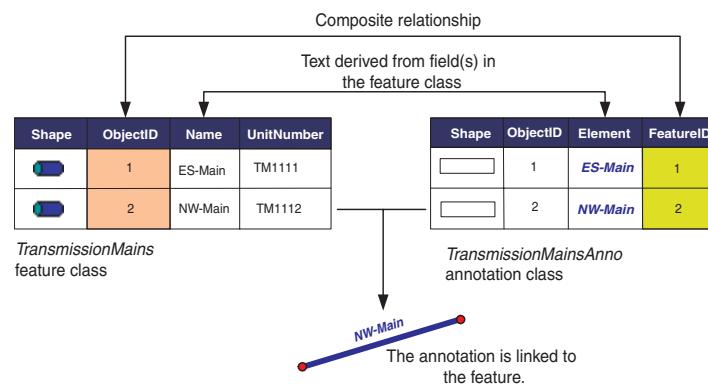
Feature-linked annotation is associated with a specific feature in another feature class in the geodatabase. The text in feature-linked annotation reflects the value of a field or fields from the feature it's linked to. For example, the water transmission mains in a water network can be annotated with their names, which are stored in a field in the transmission mains feature class.

Annotation links to features through a composite relationship with messaging. The feature class being annotated is the origin class in the relationship, while the annotation feature class is the destination class. As with other composite relationships, the origin feature controls the destination feature. So if an attribute value for the origin feature changes, the linked annotation that is based on this attribute will automatically update to reflect the change. When the origin feature is moved or rotated, the linked annotation also moves or rotates with it. And when an origin feature is deleted from the geodatabase, the linked annotation feature is also deleted.

In the water network example, a hydrant may be too close to a busy intersection and may need to be moved by 50 feet. When the hydrant is moved, its linked annotation moves with it. In the same network, the name of a transmission main may change. When the value in its name field is modified, the text stored in its linked annotation feature is automatically updated with the new name.

A feature-linked annotation feature class inside a feature dataset should link to a feature class within the same dataset. Similarly, standalone feature-linked annotation feature classes should link to standalone feature classes in the same geodatabase.

An annotation feature class can link to only one feature class, but a feature class can have any number of linked annotation feature classes.



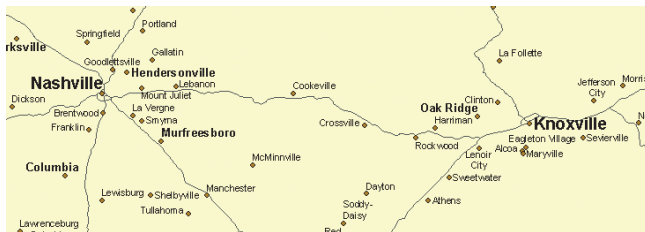
*The link between a feature and its annotation is maintained through a composite relationship. Special behaviors in the ESRI Annotation Features allow you to derive an annotation feature's text from a field or combination of fields in the feature class.*

## Annotation classes

A standard or feature-linked geodatabase annotation feature class contains one or more annotation classes. Each annotation class contains properties that determine how a subset of annotation in the feature class displays. For standard annotation, these include default symbology applied when creating new annotation and the scale range at which the annotation displays. For feature-linked annotation, these also include:

- How the annotation text strings will be defined based on attributes in the linked feature class
- Which features in the linked feature class will be annotated by the annotation class
- How to place new annotation

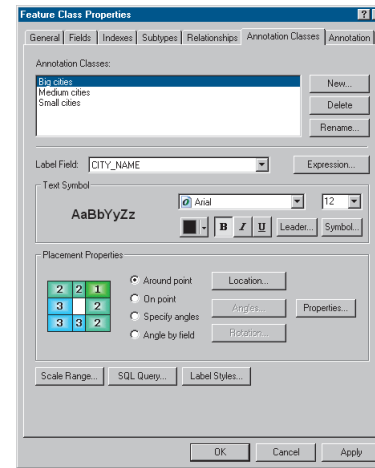
For example, if you have an annotation feature class for cities, you could have annotation classes of varying text sizes and scale ranges for small, medium, and large cities—all managed within a single annotation feature class. Annotation classes save you from having to define and maintain multiple annotation feature classes.



If you have ArcInfo or ArcEditor, you can create and modify annotation classes. You create annotation classes when you create an empty annotation feature class with ArcCatalog or when you convert labels to annotation. When you convert labels, each label class converts to an annotation class. Once you've created

an annotation feature class, you can create and modify annotation classes with the Feature Class Properties dialog box.

To maximize ArcMap display and query performance, always define a visible scale range for each annotation class so annotation features only draw when you're zoomed in enough to read their text.



*You can create and manage annotation classes with the Feature Class Properties dialog box.*

If you have more than one annotation class, the annotation classes are implemented as subtypes in the annotation feature class. To create new annotation for an annotation class, choose the class as the target in the Editing toolbar.

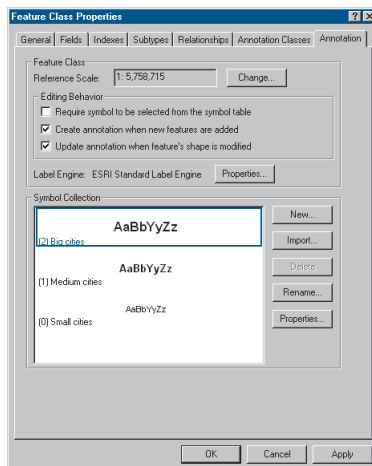
You can add and modify annotation classes with any license, including ArcView.



## Text symbols

An annotation feature class contains a collection of one or more text symbols you define. Every time you create a new annotation feature, you assign it one of these predefined symbols. The symbol contains properties that describe how the annotation feature is drawn, including its font, size, and color. For example, if you have annotation for small, medium, and large cities, you could have three text symbols of varying font sizes you assign to the annotation. Because each annotation feature doesn't store its own symbol properties, ArcGIS is able to reduce storage requirements and maximize display and query performance. Committing to a limited list of symbols can also help you promote standards for any new annotation features you create.

Whenever you create an annotation class, you create a default text symbol for it. You can also create additional text symbols for a feature class at any time with the Feature Class Properties



*You can create and manage text symbols with the Feature Class Properties dialog box.*

dialog box. Once you've created the symbols, you assign them to annotation features when editing in ArcMap.

When assigning symbols to annotation features in ArcMap, you may discover that the text symbols you created do not contain the properties you need for one or more annotation features. For example, you may require a smaller font size in order to fit annotation into a congested area. One approach is to create a new text symbol with the new properties in ArcCatalog, then assign the new text symbol to the annotation features in an ArcMap edit session.

Creating a new symbol for every unique set of properties you require could result in a long list of symbols that is difficult to work with. So ArcMap also lets you modify symbol properties on a feature-by-feature basis. When editing in ArcMap, you can select annotation and change any symbol property for that annotation.

Editing certain symbol properties on a feature-by-feature basis, though, causes annotation to lose its reference to its associated text symbol in the collection. When this happens, the annotation feature stores all of its symbol properties. This increases the storage requirement and reduces display and query performance for the annotation.

The properties you are able to edit on a feature-by-feature basis while continuing to maintain reference to a text symbol in the collection are:

- X and y offset
- Horizontal and vertical alignment
- Flip angle
- Font name, size, color, character spacing, character width, and background symbol
- Font style—bold, italic, and underline



- Word spacing
- Leading property

For example, if you reduce an annotation feature's font size from 12 to 8, the annotation feature still references a predefined symbol; however, it stores its own font size of 8. If you change a property not listed above, such as giving the annotation feature a halo, the annotation feature loses reference to its associated text symbol and stores all of the symbol properties.

You can always use text formatting tags to modify the format of a portion of a piece of annotation. This lets you create mixed-format annotation where, for example, one word in a sentence is underlined. Formatting tags always take precedence over an annotation feature's symbol properties. For more information on formatting tags, see the ArcGIS online Help system.

When deciding what symbols to store in your annotation feature class, choose a default symbol for each annotation class when you create it, then add any additional symbols containing the properties you'll commonly need. As for properties that are seldom needed, you can apply them on a feature-by-feature basis. A limited number of features storing their own symbol properties will have little impact on the storage requirements and performance of a feature class.

## Creating annotation

ArcCatalog and ArcMap contain tools that create standard and feature-linked annotation. Creating annotation requires both creating a feature class to store the annotation and populating it with annotation features. You can follow one of three approaches:

- If you've already invested time and effort creating coverage, SDE 3, CAD, or vector product format (VPF) annotation, you can display these formats in ArcMap and import them into a new standard or feature-linked annotation feature class.

Importing from these formats into a feature-linked annotation feature class does not link annotation to features. If the annotation feature class you imported has a field that relates to a field in the feature class you want to link to, you can use SQL to link the annotation to their features. See <http://support.esri.com> for more information. If you don't have these fields, you can manually link the imported annotation to features with the editing tools in ArcMap.

- If the text you want to display is stored in one or more attributes in a feature class, you can display the text as labels in ArcMap and convert them to annotation. This creates a new standard or feature-linked annotation feature class.
- If you don't have any data or want to design your annotation feature class schema from scratch, you can create an empty standard or feature-linked annotation feature class in ArcCatalog, then populate it with annotation.

To populate an empty standard annotation feature class, you can display and convert labels to annotation in ArcMap or manually add annotation using the tools on the Annotation toolbar.

To populate an empty feature-linked annotation feature class, you can select features in ArcMap and use the Annotate Selected Features command to generate annotation. Or you can have ArcMap create new annotation for you as you add new features.

When creating annotation in an ArcSDE geodatabase by converting labels or importing from another format, you should always try to create the annotation before registering your data as versioned. If the data is not versioned, all of the data will be loaded directly into the base tables and a database compress will not be required. For more information about data loading strategies and their versioning impacts, see the chapter 'Importing data' in this book.

If you are creating feature-linked annotation that is linked to a network feature class, you should create the annotation after building the geometric network. When features are snapped in the network building process, their geometry is modified at a level at which messages are not sent to linked annotation features to update themselves. So after building the network, if a feature's geometry was changed in the snapping process, the feature and its linked annotation may no longer correspond correctly. To learn more about geometric networks, see the chapter 'Geometric networks' in this book.

### **Managing annotation feature classes**

Annotation feature classes can be managed the same way as other feature classes and tables. They can be renamed, copied, and deleted using ArcCatalog. You can also use the Feature Class Properties dialog box to set the alias and spatial and attribute indexes as well as manage properties unique to annotation feature classes—annotation classes, symbols, and settings that determine how the annotation features can be edited.

To save you from having to maintain several annotation feature classes, you may want to combine them into a single annotation feature class. You might do this, for example, if they annotate features in the same feature class or if they are standard annotation feature classes that annotate similar features in more than one feature class. When you combine annotation feature classes, the annotation from each feature class becomes an annotation class in the new annotation feature class. You combine annotation feature classes with the Append Annotation Feature Classes tool. For more information, see the ArcGIS online Help system.

The relationship class that links feature-linked annotation is managed like any other relationship class. If the relationship class is deleted, the annotation will no longer be feature-linked, but will

become standard annotation. To learn how to re-create this relationship class, see the chapter 'Defining relationship classes' in this book.

### **Upgrading ArcGIS 8 annotation feature classes**

ArcGIS 9 adds functionality to annotation feature classes that was not available in previous versions. If you have annotation feature classes created in ArcGIS 8, you can display and query them. However, to edit them or modify their properties, you must upgrade them. Upgrading annotation feature classes also adds text symbol fields that are useful for querying and editing. For more information on upgrading annotation feature classes, see the ArcGIS online Help system.

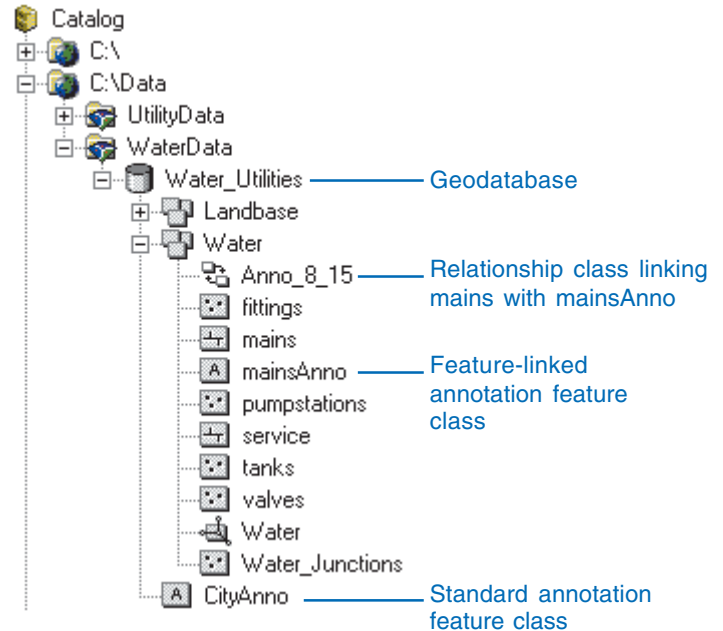
# Annotation and ArcCatalog

You can create and manage annotation feature classes in ArcCatalog.

Annotation classes can exist both inside feature datasets and at the root level of the geodatabase. Feature-linked annotation classes inside a feature dataset should link to another feature class within the same dataset. Similarly, standalone feature-linked annotation feature classes should link to standalone feature classes in the same geodatabase.

A feature-linked annotation class can only be linked to a single feature class. However, a feature class can have any number of linked annotation feature classes.

When you look at annotation feature classes in the ArcCatalog tree, it's not immediately evident which ones are standard and which ones are feature-linked. Moreover, simply looking at the Catalog tree won't tell you which feature class feature-linked annotation links to. Examining the properties of an annotation feature class can tell you if it is the destination class in a composite relationship. If it is, this indicates it is a feature-linked annotation class.



*Annotation feature classes appear in ArcCatalog at either the database or feature dataset level. Feature-linked annotation feature classes also have a relationship class linking them to another feature class.*

# Creating annotation feature classes

In ArcCatalog, you can create an empty standard or feature-linked annotation feature class and create annotation for it later.

Using the tools on the Annotation toolbar, you can manually add annotation to a standard annotation feature class. For details on how to create and edit individual annotation features, see *Editing in ArcMap*. You can also display and convert labels to annotation in ArcMap. This process is described later in this chapter.

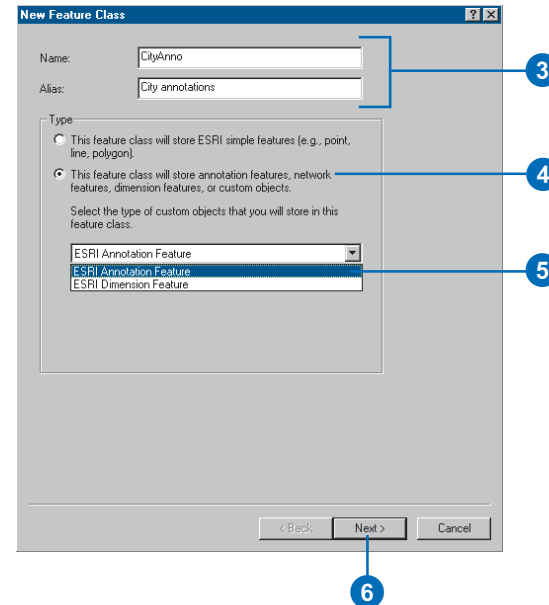
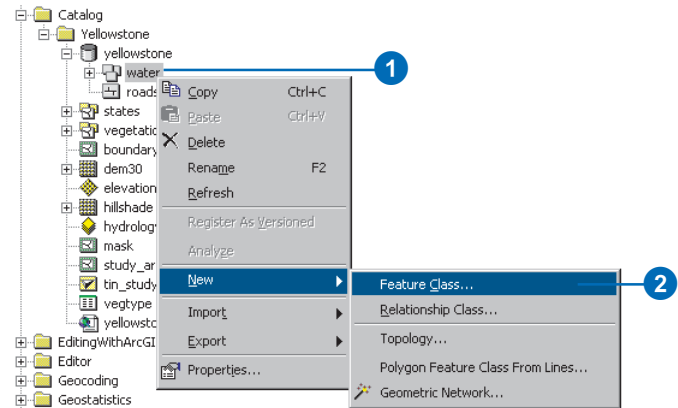
To create annotation for a feature-linked annotation feature class, you can select features in ArcMap and use Annotate Selected Features to generate annotation. Generating annotation creates and places annotation based on the linked features. This process is described later in this chapter. You can also have ArcMap create new annotation as you add new features.

## Creating a standard annotation feature class

1. Right-click the geodatabase or feature dataset in which you want to create the new annotation feature class.
2. Point to New and click Feature Class.
3. Type the name. To create an alias for this annotation feature class, type the alias.
4. In the Type category, click the second option to store custom objects.
5. Click the dropdown arrow and choose ESRI Annotation Feature.
6. Click Next.
7. Type the scale at which you want the font size displayed.
8. If you're creating a standalone feature class, click the Map Units dropdown arrow and choose the units for your data.

If you're creating the feature class inside a feature dataset, the map units are automatically set for you.

9. Check the Require symbol to be selected from the symbol table check box. This requires edited annotation features to maintain a reference to a text symbol in the feature class symbol collection. ►



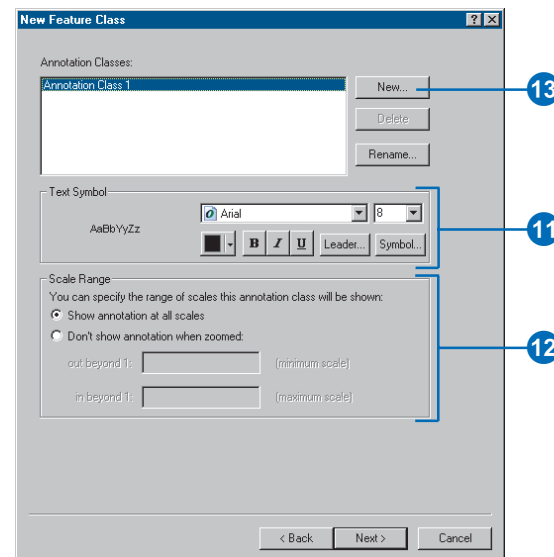
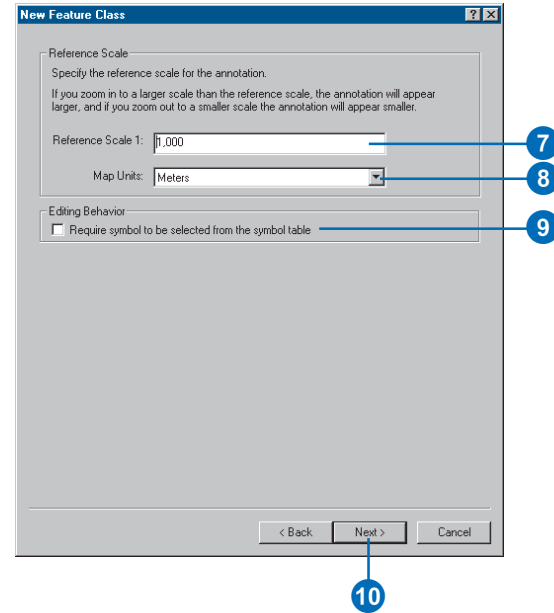
## Tip

### Reference scale

*The reference scale is the scale at which the annotation text is displayed at the font size specified. As you zoom out, the text will get smaller, and as you zoom in, the text will get larger.*

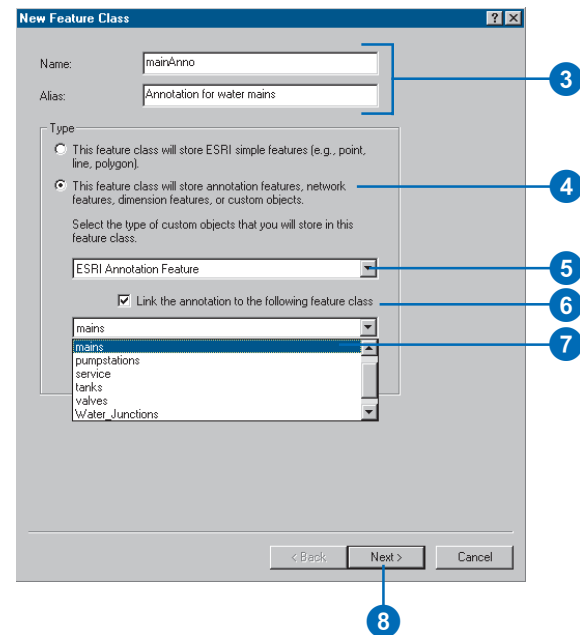
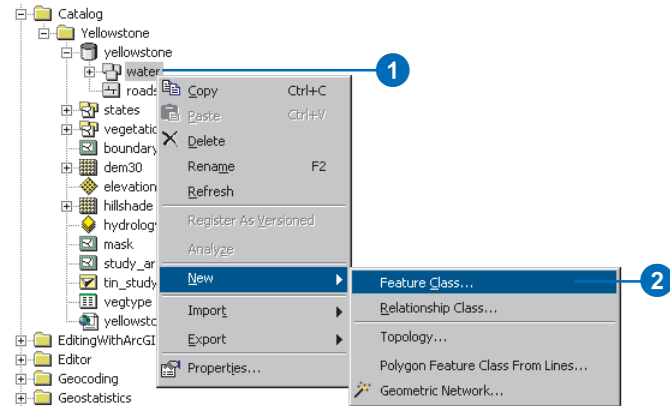
*The reference scale should always be in the same units as the spatial reference of the annotation class.*

10. Click Next.
11. For Text Symbol, set the default text symbol properties for the first annotation class.
12. Specify the visible scale range for annotation in this class.
13. If you want to add an additional annotation class, click New and specify the name of the new annotation class. Repeat steps 11 and 12 to set its properties.
14. Repeat step 13 until you have specified all the annotation classes and their properties.
15. Click Next.
16. If you're creating an annotation feature class inside a feature dataset, follow steps 6 through 15 for 'Creating a feature class in a feature dataset' as described in the chapter 'Creating new items in a geodatabase'.  
  
If you're creating a standalone annotation feature class, follow steps 2 through 12 for 'Creating a standalone feature class' as described in the chapter 'Creating new items in a geodatabase'.



## Creating a feature-linked annotation feature class

1. Right-click the geodatabase or feature dataset in which you want to create the new annotation feature class.
2. Point to New and click Feature Class.
3. Type the name. To create an alias for this annotation feature class, type the alias.
4. Click the second option to store custom objects.
5. Click the dropdown arrow and choose ESRI Annotation Feature.
6. Check the check box to link the annotation to a feature class.
7. Click the dropdown arrow and click the feature class to which you want to link this annotation feature class.
8. Click Next.
9. Type the scale at which you want the font size displayed.
10. Click the Map Units dropdown arrow and click the units for your data.
11. Set the editing behavior for the new annotation feature class. Also, if Maplex is installed, choose a label engine and specify its properties. ▶



## Tip

### Relationship class

A relationship class will automatically be created to link the annotation class with the feature class it is annotating. The name of the relationship class will be `Anno_<feature class ID>_<annotation class ID>`.

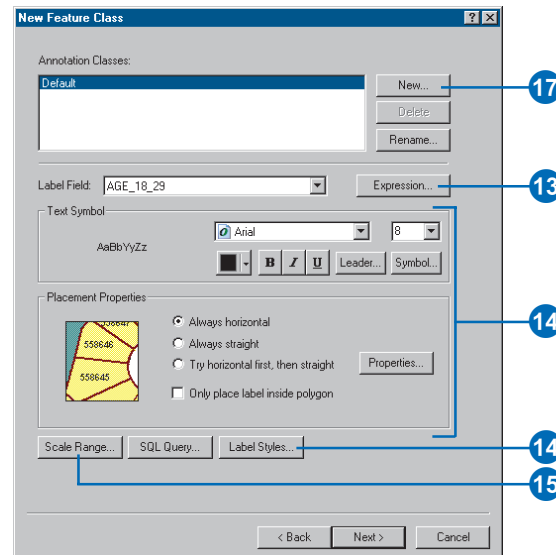
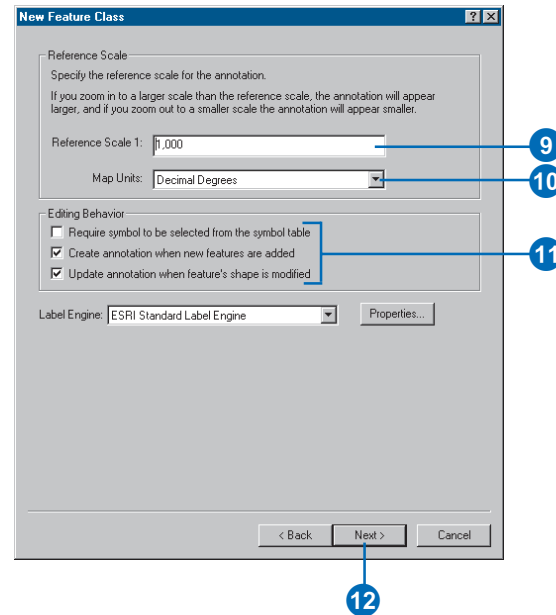
## See Also

For a detailed discussion of labeling including specifying label expressions and placement properties, see *Using ArcMap*.

12. Click Next.
13. For the first annotation class, specify the label field in the linked feature class that contains the annotation text. Click Expression to specify more than one field.
14. Set the default text symbol and placement properties.

You can either set these properties individually or click the Label Styles button to load a label style.

15. Click Scale Range to specify the visible scale range for annotation in the class.
16. Click SQL Query to specify which features in the linked feature class will be annotated by the annotation class.
17. If you want to add an additional annotation class, click New and specify the name of the annotation class. Repeat steps 13 through 16 to set its properties.
18. Repeat step 17 until you have specified all the annotation classes and their properties.
19. Click Next.
20. Follow steps 6 through 15 for 'Creating a feature class in a feature dataset' as described in the chapter 'Creating new items in a geodatabase'.



## Tip

### Versioning

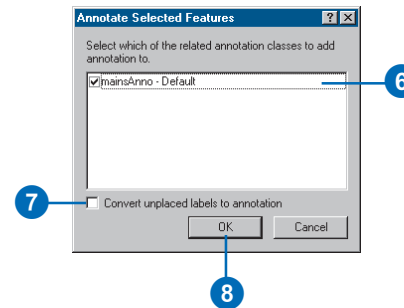
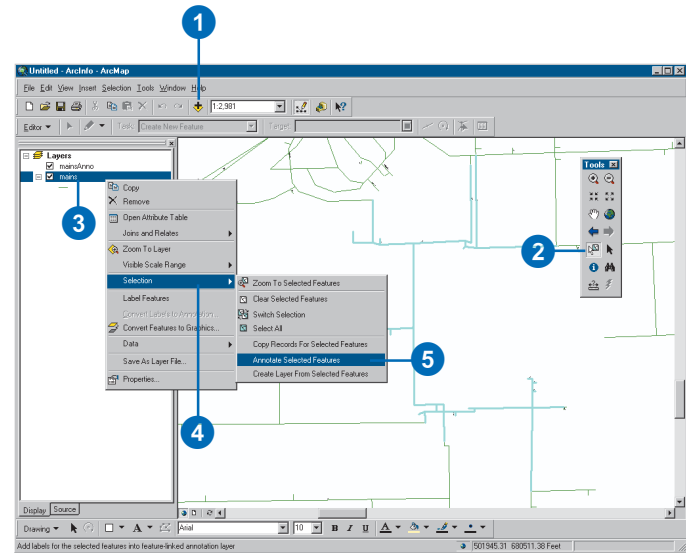
When possible, generate your annotation before you version your data.

## See Also

For more information on how to use ArcMap to add feature classes to maps and how to select features, see Using ArcMap.

## Generating feature-linked annotation

1. Click the Add Data button in ArcMap to add a feature class and its linked annotation class to your map.
2. Use the Select Features tool to select the features for which you want to create annotation. To create annotation for all of the features, select all of the features.
3. Right-click the layer in the table of contents.
4. Point to Selection.
5. Click Annotate Selected Features.
6. Check the related annotation classes in which you want to store the annotation.
7. Check the check box to convert unplaced labels.
8. Click OK.





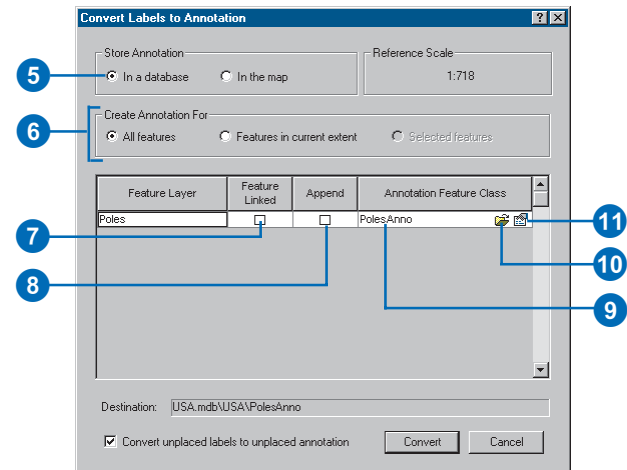
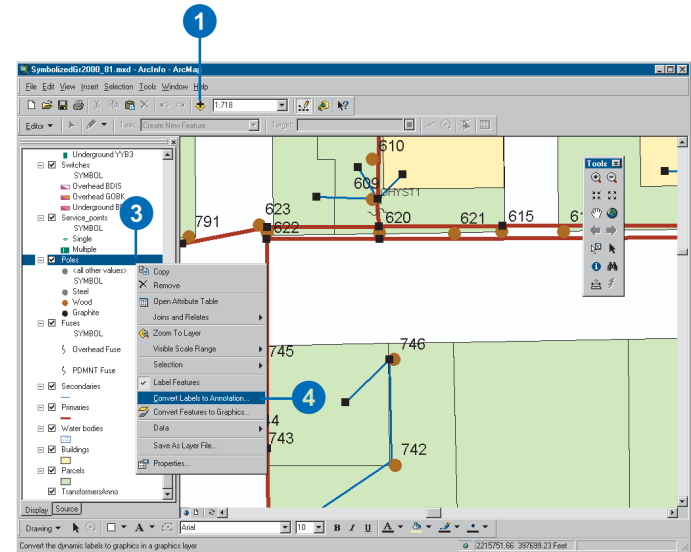
# Converting labels to annotation

In ArcMap you can convert dynamic labels to standard or feature-linked annotation features.

You can convert labels to standard annotation features from any layer with a geodatabase, coverage, shapefile, or CAD feature class data source. You can create a new standard annotation feature class or add the new annotation to an existing one.

To convert to feature-linked annotation, the layer being labeled must have a geodatabase feature class data source. Unlike when you convert to standard annotation, you always create a new feature-linked annotation feature class in the process—you cannot add the new annotation to an existing feature-linked annotation feature class. The new annotation feature class is created in the same feature dataset as the geodatabase feature class you are labeling or at the root level of the geodatabase if the feature class you are labeling is at the root level of a geodatabase. ►

1. Click the Add Data button in ArcMap to add the feature class for which you want to create annotation to your map.
2. Label the features in your map as described in *Using ArcMap*.
3. Right-click the layer in the table of contents. To convert labels from more than one layer, right-click the data frame.
4. Click Convert Labels to Annotation.
5. For Store Annotation, click In a database.
6. Specify the features you want to create annotation for.
7. To create feature-linked annotation, check the Feature Linked box.
8. If you're creating standard annotation and want to add the annotation to an existing standard annotation feature class, check the Append box.
9. If you're creating feature-linked annotation, click the name of the new annotation feature class to change it. ►



Before you convert labels, set the data frame reference scale and label properties carefully as they determine the size, position, and appearance of the new annotation. For more information on how to prepare labels for conversion, see the ArcGIS online Help system.

### Tip

#### New annotation class

Once labels have been converted to annotation, the new annotation class is automatically added to the map.

### Tip

#### Versioning

When appending to an existing feature class, if possible, convert your labels before you version your data.

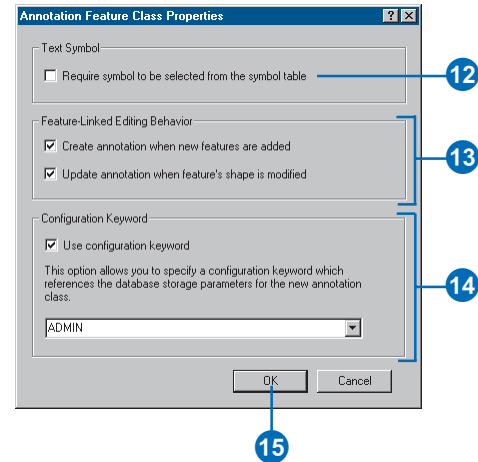
### See Also

For a detailed discussion about labeling maps and the different advanced labeling methods you can use, see Using ArcMap.

10. If you're creating standard annotation, click the open folder icon and specify the path and name of the new annotation feature class you will create, or if you're appending, the existing standard annotation feature class you're appending to.

If you're appending to an existing feature class, skip to step 17.

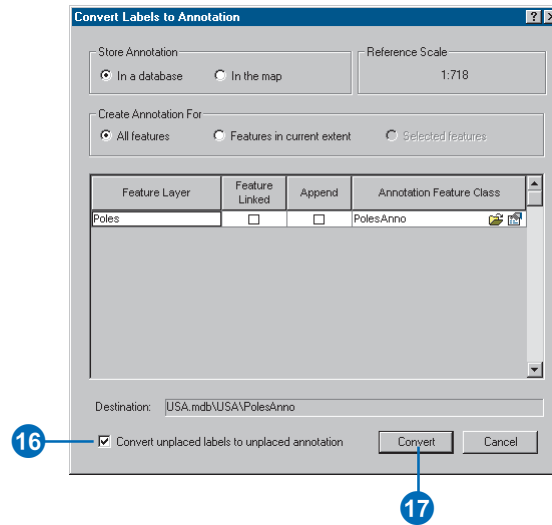
11. Click the properties icon.
12. Check this check box to require edited annotation features to maintain reference to their associated text symbols stored in the feature class.
13. Specify additional editing behavior for the new annotation feature class.
14. If you are creating the new annotation feature class in an ArcSDE geodatabase and want to use a custom storage keyword, click Use configuration keyword, then choose the keyword you want to use (ArcInfo and ArcEditor only).
15. Click OK. ►



16. Some labels may not currently display on the map because there is no room for them.

To convert these labels, check the Convert unplaced labels box. This saves the unplaced labels in the annotation feature class, allowing you to later position them one at a time in an ArcMap edit session.

17. Click Convert.



# Importing coverage annotation

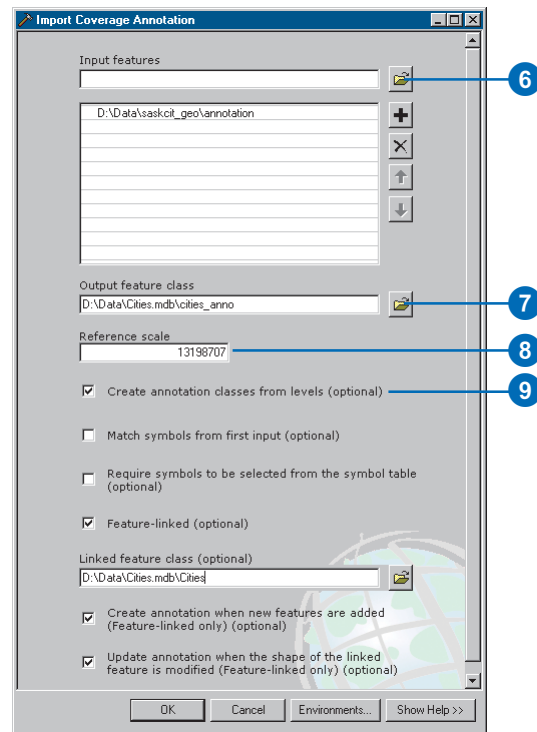
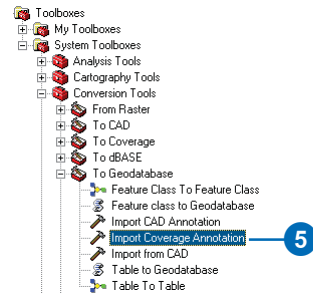
You import one or more coverage annotation feature classes with the Import Coverage Annotation tool. It imports into a new standard or feature-linked annotation feature class it creates in the process.

If the coverage annotation feature classes contain attributes, they will automatically import as well. Coverage annotation features store display characteristics in attribute fields called pseudo items. Pseudo items do not import as attribute fields but translate to ArcGIS text symbol properties at import.

Before importing, add the annotation as a layer to ArcMap and zoom to the scale at which you'll normally display the new annotation. Symbolize the annotation the way you want it to appear once imported. If you're importing several tiled annotation feature classes at the same time, you need only set up symbology for the first annotation feature class you list in the Import Coverage Annotation tool. The tool allows you to apply the same symbology to all the other feature classes you import. ►

1. In ArcMap, add the coverage annotation feature class you want to import.
2. Zoom to the scale at which you'll normally view the imported annotation.
3. Symbolize the coverage annotation as you want it to appear once converted.
4. Open the ArcToolbox window, navigate to Toolboxes/System Toolboxes/Conversion Tools, and expand To Geodatabase.
5. Double-click Import Coverage Annotation.
6. Add the annotation feature classes you want to import.
7. Navigate to the location of the new annotation feature class and specify its name.
8. Specify the reference scale.
9. If you want to create an annotation class for each level defined by the \$Level field, check the check box. ►

If you're importing into a new feature-linked annotation feature class, you must specify a feature class in the same feature dataset as the linked feature class or at the root level of the geodatabase if the feature class you're linking to is at the root level of a geodatabase.



The values in the pseudo item \$Size determine how the text size is set for the imported annotation. If the value is zero, the ArcMap text symbol size is used. This means you must display the annotation in the correct text symbol size before importing. The size you choose is the size in which the new annotation will display at the scale you're zoomed to.

If the \$Size value is not zero, you cannot change the size of the annotation in ArcMap. In this scenario, the \$Size value is used as the size instead of the ArcMap text symbol size.

## Tip

### Importing from ArcMap versus ArcCatalog

You can run the Import Coverage Annotation tool from ArcCatalog as well. However, all imported annotation is assigned the default font Arial—and if the \$Size value is zero—the default size 14 pt.

If you want to combine levels into a single annotation class in the output feature class, leave this box unchecked.

- If you're importing a set of annotation feature classes that all use the same symbols and the same \$Symbol value to refer to each symbol, check the check box.

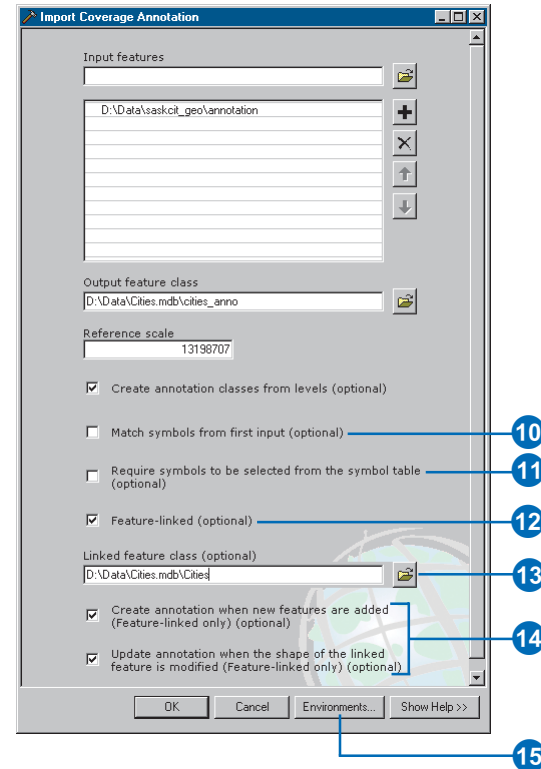
This prevents a symbol from being added to the new feature class's symbol collection more than once.

- Check this box to require edited annotation features to maintain a reference to a text symbol in the feature class symbol collection.

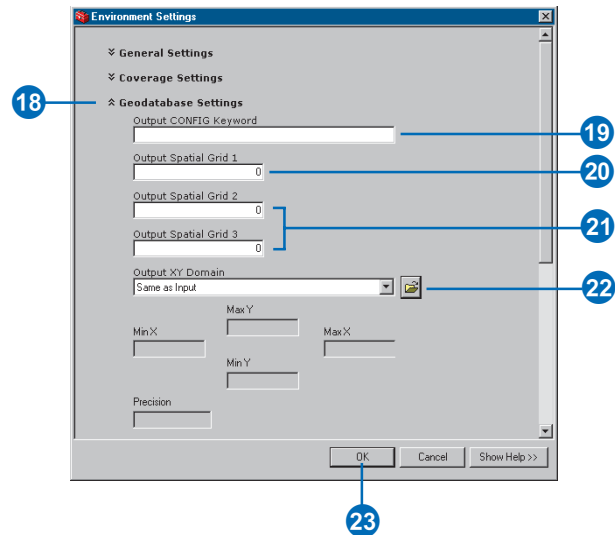
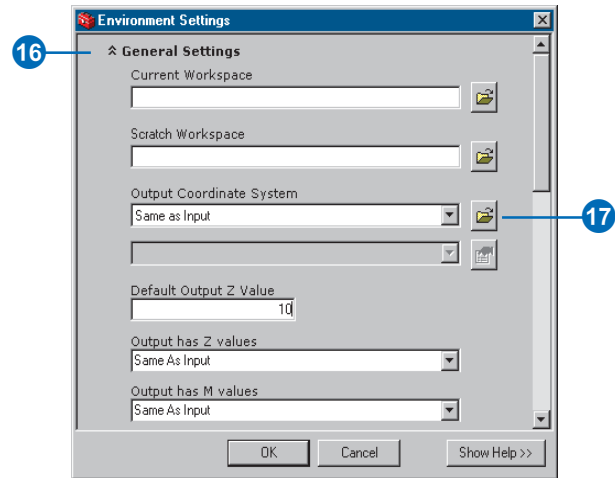
- If you want to import to a feature-linked annotation feature class, check the check box.

If you don't want to import to a feature-linked annotation feature class, skip to step 15.

- Navigate to the existing geodatabase feature class that the new annotation feature class will link to.
- Specify additional editing behavior for the new annotation feature class.
- Click Environments. ►



16. Expand General Settings.
17. Set the Output Coordinate System.
18. Expand Geodatabase Settings.
19. If you're importing to ArcSDE and you want to create the feature classes using a custom storage keyword, type the keyword.
20. If you know an optimal spatial index grid size for your data, specify it in map units.
21. If you're importing to an ArcSDE geodatabase and have additional grid sizes, type them in.
22. Set the x,y domain.
23. Click OK.
24. Click OK to import the annotation feature classes.



# Dimensioning

# 9

## IN THIS CHAPTER

- **Dimensions in the geodatabase**
- **Dimensions and ArcCatalog**
- **Creating dimension feature classes**
- **Creating and managing dimension styles**

For many applications, plotting a map that shows a feature's shape and some descriptive annotation isn't sufficient for showing precise measurements or distances. Dimensions are a special kind of map annotation for showing just that—specific lengths or distances. A dimension may indicate the length of a side of a building or land parcel, or it may indicate the distance between two features—for example, a fire hydrant and the corner of a building.

ArcGIS provides tools that allow you to store dimensions in your geodatabase. You can also create dimension styles to apply to your *dimension features* so they are consistent with your application standards. ArcGIS supports a number of dimension types and methods for creating dimensions. Dimensions can be created automatically from existing features, or you can use the rich set of construction tools available in ArcMap.

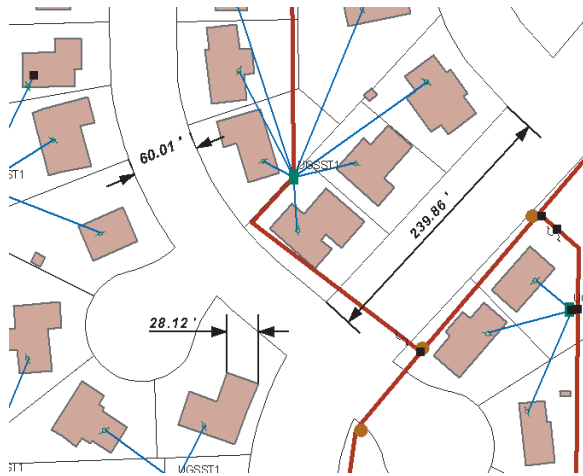
This chapter describes how to create *dimension feature classes* and *dimension styles*. *Editing in ArcMap* contains a chapter dedicated to using the editing capabilities of ArcMap to create and modify dimension features.

You can view dimension feature classes in ArcView. To take advantage of other dimensioning functionality, you need an ArcEditor or ArcInfo license.



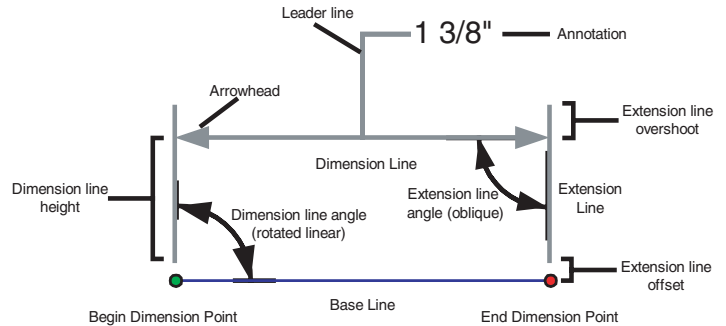
# Dimensions in the geodatabase

Dimensions are a special kind of map annotation that show lengths or distances. A dimension may indicate the length of a side of a building or land parcel or it may indicate the distance between two features such as a fire hydrant and the corner of a building. Dimensions can be as simple as a piece of text with a leader line or as elaborate as the diagram shown opposite.



*Dimensions show specific lengths or distances on a map. For example, a dimension may indicate the length of a side of a building, the width of a street, or the length of a parcel.*

A dimension feature is composed of several parts that may or may not be displayed, depending on the application. The following is an illustration of the anatomy of a dimension feature. (Throughout this chapter, these parts will be referred to by the names in this diagram.)

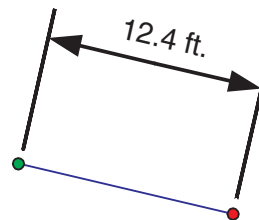


*A dimension feature is composed of many parts. Each dimension feature can represent each part differently by using different symbology and placement rules. Dimensions may also display a subset of these parts.*

## Dimension types

ArcGIS supports two types of dimensions: aligned and linear. *Aligned dimensions* run parallel to the baseline and represent the true distance between the begin and end dimension points.

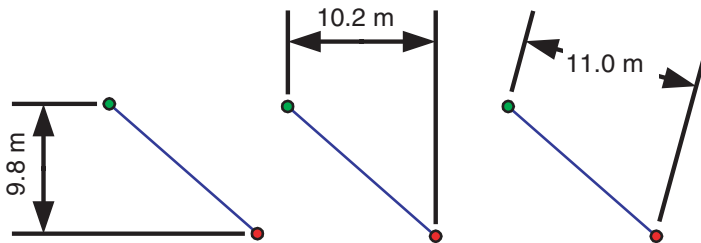
Unlike aligned dimensions, *linear dimensions* don't represent the true distance between the begin and end dimension points. Linear dimensions can be vertical, horizontal, or rotated. A vertical dimension's line represents the vertical distance between the begin and end dimension points. A horizontal linear dimension's



*An aligned dimension has its dimension line parallel to the baseline, and its length represents the true distance between the begin and end dimension points.*



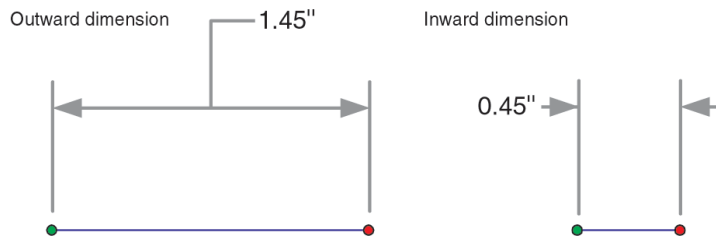
line represents the horizontal distance between the begin and end dimension points. A rotated linear dimension is a dimension whose line is at some angle to the baseline and whose length represents the length of the dimension line itself, not the baseline.



*Linear dimensions may be vertical, horizontal, or rotated. In each case the dimension's length represents something other than the true distance between the begin and end dimension points.*

All dimensions can be oriented either outward or inward. Outward dimensions have dimension lines pointing to the outside of the feature and represent the distance being measured. Inward dimensions have arrows pointing in from the outside of the feature and measure the distance between these two arrows. Whether a dimension is outward or inward is determined by the distance that the dimension represents and whether that distance on the map is sufficient to display all of the elements of the dimension between the extension lines.

To learn more about the tools and construction methods used to create all of these types of dimensions while editing in ArcMap, see the chapter 'Editing dimension features' in *Editing in ArcMap*.



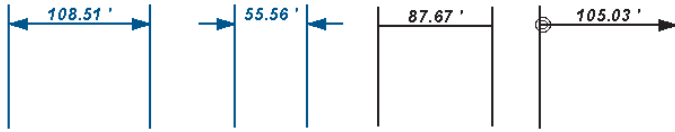
*A dimension feature can be either outward or inward. Whether a dimension is inward or outward usually depends on its length and symbology.*

## Dimension feature classes

In the geodatabase, dimensions are stored in dimension feature classes. Like other feature classes in the geodatabase, all features in a dimension feature class have a geographic location and attributes and can either be inside or outside a feature dataset. Like annotation features, dimension features are graphic features and their symbology is stored in the geodatabase.

## Dimension styles

A collection of dimension styles is associated with a dimension feature class. A dimension feature's style describes its symbology, what parts of it are drawn, and how it is drawn. Every time you create a new dimension feature, it is assigned a particular style. All dimension features of a particular style share certain characteristics, some of which can be changed on a feature-by-feature basis. Styles for a dimension feature class are created, copied, and managed using ArcCatalog. Using the editing capabilities in ArcMap, styles are then assigned to individual dimension features.



*Dimension style examples. All dimension features are associated with a style. The style describes how the dimension feature is symbolized and the content of the dimension's text.*

The following are the properties of a dimension feature that can be assigned based on a style:

- Dimension line symbol: the symbol used for the dimension line.
- Begin symbol: the symbol used for the arrow at the end of the begin dimension line.
- End symbol: the symbol used for the arrow at the end of the end dimension line.
- Dimension line display: indicates which of the dimension lines should be displayed—both, begin only, end only, or neither.
- Arrow display: indicates which dimension line arrows should be displayed—both, begin only, end only, or neither.
- Extension line symbol: the symbol used for the extension lines.
- Extension line display: indicates which of the extension lines should be displayed—both, begin only, end only, or neither.
- Offset and overshoot: the distance that the extension lines are drawn from the dimension points and how far they extend beyond the dimension line, respectively.
- The value to display for the text: The actual string that is displayed for the dimension text. This may be derived from the feature itself or from a user-specified value or string. The value may be in map units or converted and displayed in other units.

- Arrow fit and text fit: The adjustment in the display of the arrows and text when the dimension's length is too short to show the arrows and text between the extension lines.

ArcGIS provides tools that allow you to store dimensions in your geodatabase. You can also create dimension styles to apply to your dimension features so they are consistent with your application standards. ArcGIS supports a number of dimension types and methods for creating dimensions. Dimensions can be created automatically from existing features, or you can use the rich set of construction tools available in the ArcMap editing environment.

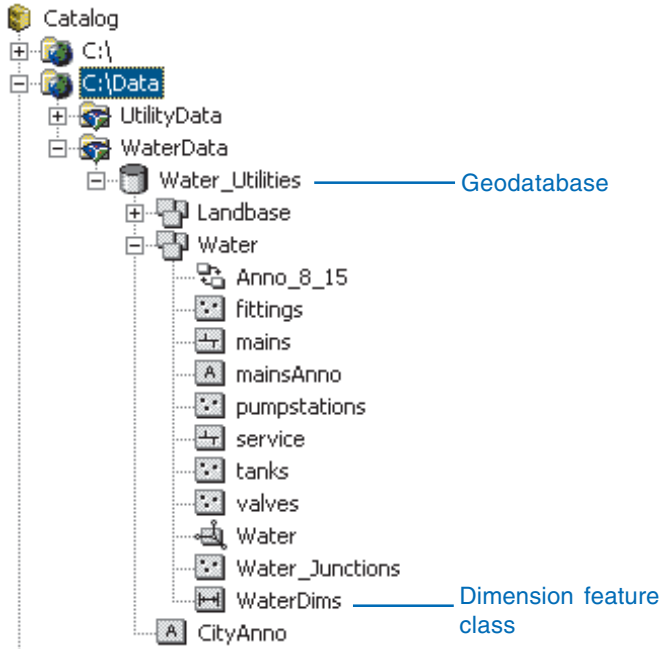
Once a style is created in a dimension feature class, it cannot be modified. If you want to modify the properties of an existing dimension style, you must create a new style with the new properties. You can create new styles based on the properties of an existing style, or you can import styles from dimension feature classes in other geodatabases. For more information on ArcMap editing capabilities, see *Editing in ArcMap*.

## Performance considerations

Dimensions are a type of map annotation. Like regular annotation, the information that dimension features convey is not useful unless you have displayed your map at a scale in which you can visualize the dimension features clearly. Like annotation features, dimension features are costly to retrieve from the database and draw on the map display. When working with dimension feature classes in ArcMap, you should always apply scale suppression so the dimension features only draw at scales in which they can be visualized. For more information on layers and scale suppression, see *Using ArcMap*.

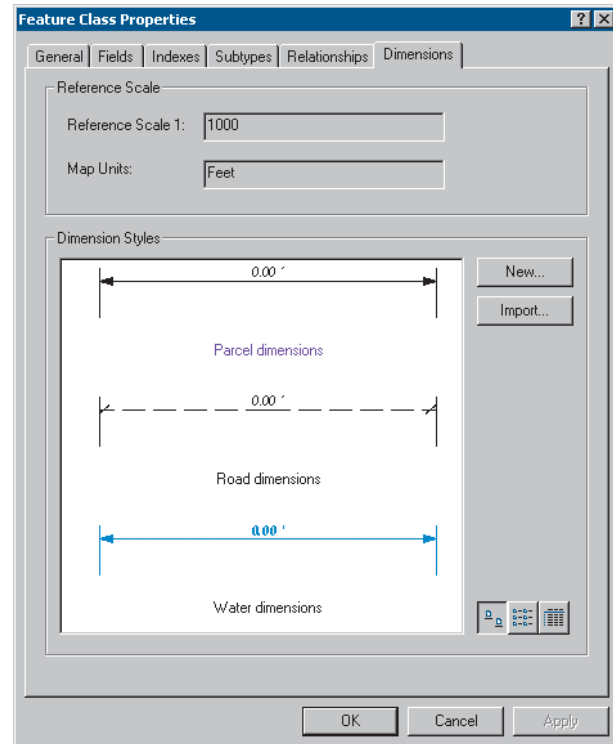
# Dimensions and ArcCatalog

In ArcCatalog, you can work with a dimension class in any accessible geodatabase. Dimension feature classes can exist both inside a feature dataset and at the root level of the geodatabase.



*Dimension feature classes appear in ArcCatalog at either the database or feature dataset level.*

You can use ArcCatalog to create and manage dimension feature classes. The Feature Class Properties dialog box displays special information about the dimension feature's styles and at what scale those styles are displayed with their specific symbol sizes. You can use the Feature Class Properties dialog box to create new styles, delete styles, and import dimension styles from other dimension feature classes.



*The Dimensions tab on the Feature Class Properties dialog box displays dimension information, such as the default style and the reference scale, for the dimension features.*

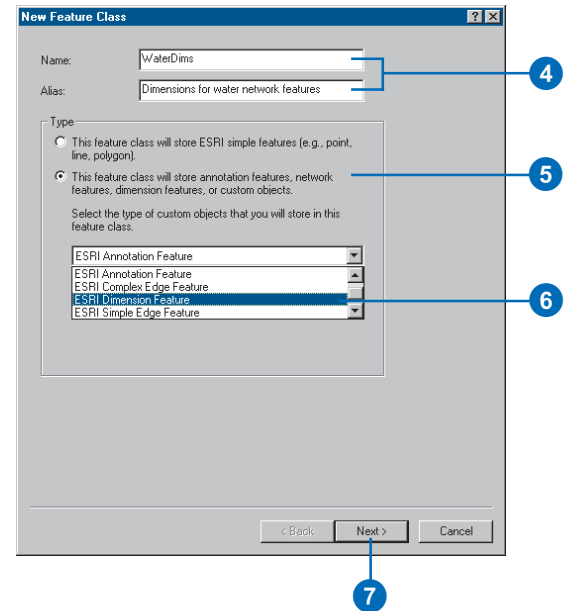
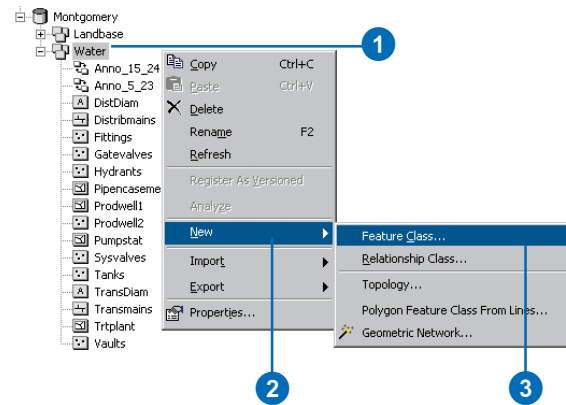
# Creating dimension feature classes

Dimension features are stored in dimension feature classes. When creating a dimension feature class, you must create at least one style for the dimension features you will create. You can specify the properties of the style yourself, import the style from another dimension feature class, or let the wizard create a default style for you.

Once you have created a dimension feature class, you can use ArcCatalog to create and import additional styles.

## Creating a dimension feature class with the default style

1. Right-click the geodatabase or feature dataset in which you want to create the new dimension class.
2. Point to New.
3. Click Feature Class.
4. Type the name for the new dimension feature class. To create an alias for this feature class, type the alias.
5. Click the second option to store custom objects in the feature class.
6. Click the dropdown arrow and click ESRI Dimension Feature.
7. Click Next. ►

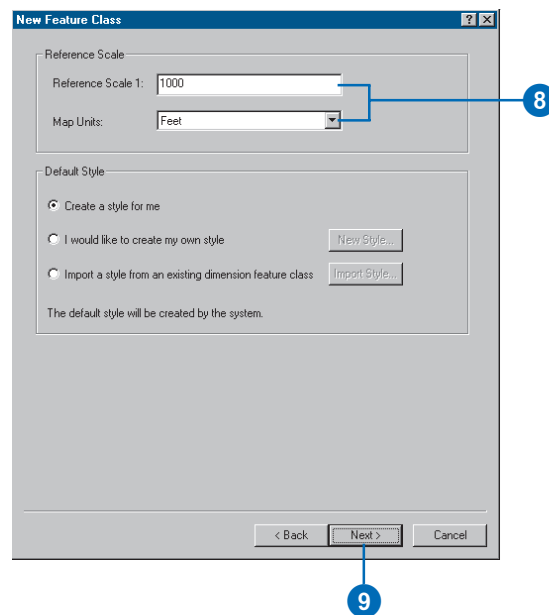


## Tip

### Reference scale

The reference scale describes the scale at which the symbology of the dimension feature is the same size as described in the style. For example, if your dimension text is Arial 12 pt. and your reference scale is 1:1,000, then your text will be 12 pt. at 1:1,000. As you zoom out from this scale, the text becomes smaller, and as you zoom in from this scale, the text becomes larger.

8. Type a reference scale. The reference scale units will automatically match the spatial reference's units if the dimension class is being created in a feature dataset. If this is a standalone dimension class, then you should pick the units for your spatial reference, which you will specify later in the wizard.
9. Click Next.
10. Follow steps 6 through 14 for 'Creating a feature class in a feature dataset' in the chapter 'Creating new items in a geodatabase' in this book if you are creating this dimension feature class inside a feature dataset. Follow steps 2 through 12 for 'Creating a standalone feature class' in the chapter 'Creating new items in a geodatabase' in this book if you are creating a standalone dimension feature class.

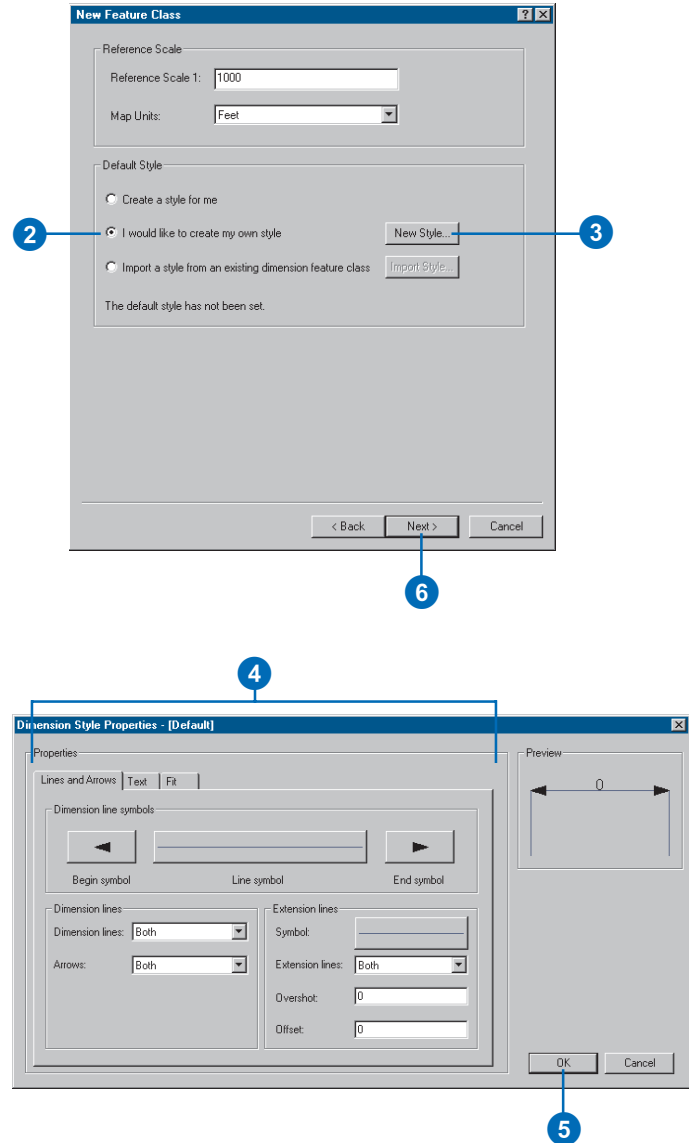


## See Also

For more information on what each style element is and how to create styles, see 'Creating and managing dimension styles' later in this chapter.

## Creating a dimension feature class with a custom style

1. Follow steps 1 through 8 for 'Creating a dimension feature class with the default style'.
2. Click the second option to create your own style.
3. Click New Style to open the style properties dialog box.
4. Use the Dimension Style Properties dialog box to set the characteristics of your dimension style.
5. Click OK.
6. Click Next.
7. Follow steps 6 through 14 for 'Creating a feature class in a feature dataset' in the chapter 'Creating new items in a geodatabase' in this book if you are creating this dimension feature class inside a feature dataset. Follow steps 2 through 12 for 'Creating a standalone feature class' in the chapter 'Creating new items in a geodatabase' in this book if you are creating a standalone dimension feature class.



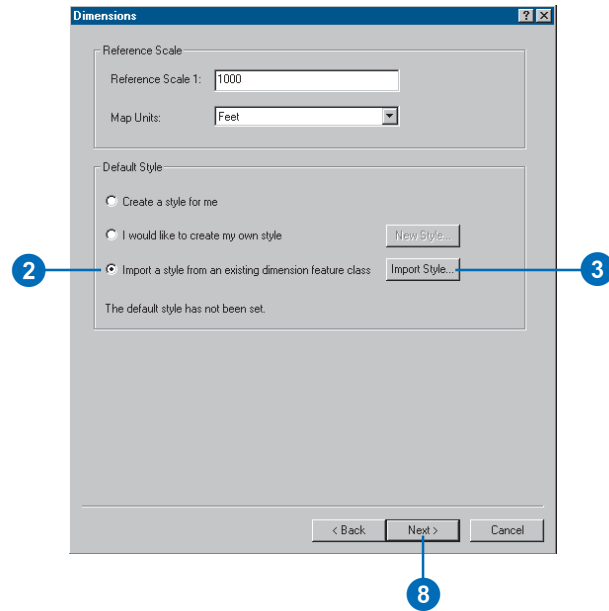
## Tip

### Browsing styles

You can browse styles by looking at an example dimension, or you can also click the *View Options* button at the bottom of the *Import* dialog box to switch the view to a list of the style names or the style names and style IDs.

## Creating a dimension feature class by importing a style

1. Follow steps 1 through 8 for 'Creating a dimension feature class with the default style'.
2. Click the third option to import a style from an existing dimension feature class.
3. Click *Import Style* to browse for the dimension feature class from which you want to import a style. ►

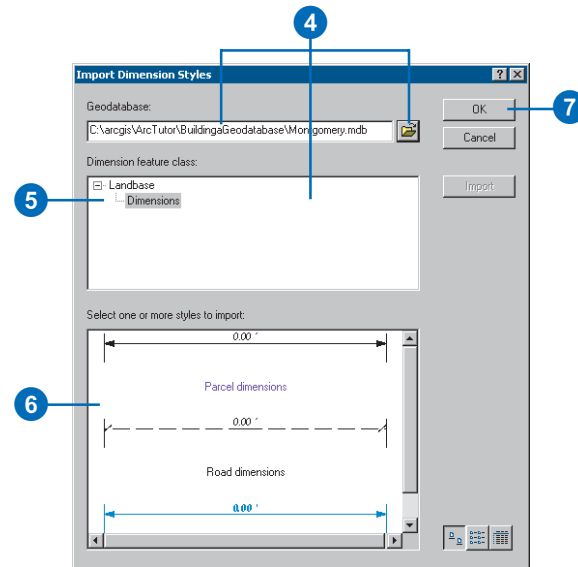


## Tip

### Importing more than one style

You can import multiple styles from multiple dimension classes by opening the property page for an existing dimension class. See 'Creating and managing dimension styles' later in this chapter.

- Click the Browse button to browse for a geodatabase. Once a geodatabase is selected, the dimension feature classes and feature datasets containing dimension feature classes are listed in the tree view.
- Click the dimension feature class that contains the style you want to copy.
- Click the dimension style that you want to copy.
- Click OK.
- Click Next.
- Follow steps 6 through 14 for 'Creating a feature class in a feature dataset' in the chapter 'Creating new items in a geodatabase' in this book if you are creating this dimension feature class inside a feature dataset. Follow steps 2 through 12 for 'Creating a standalone feature class' in the chapter 'Creating new items in a geodatabase' in this book if you are creating a standalone dimension feature class.

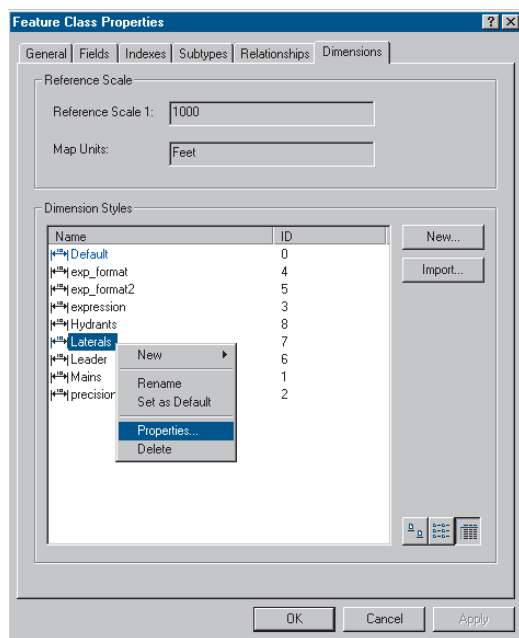




# Creating and managing dimension styles

A dimension style describes how a dimension feature is displayed including its symbology, label font, and label text. Each dimension feature class has at least one style. Dimension features within the dimension feature class are associated with a particular style. All dimension features of a particular style have certain characteristics that are the same, while other characteristics can be overridden on a feature-by-feature basis.

Dimension styles are created and managed in ArcCatalog in the Feature Class Properties dialog box. You can create, delete, rename, and import dimension styles, and you can specify the default style for a dimension feature class.



*Dimension styles are created and managed in ArcCatalog in the Feature Class Properties dialog box.*

The dimension styles' properties are defined in the Dimension Style Properties dialog box. This dialog box has three tabs: Lines and Arrows, Text, and Fit. The Lines and Arrows tab allows you to set the properties for the dimension lines, line arrows, and extension lines. The Text tab allows you to control the content of the dimension text as well as its symbology. The Fit tab allows you to define how the dimension and dimension text adjust when the dimension's length is too short to display the arrows and text between the extension lines. Each tab and property on the Dimension Style Properties dialog box is discussed in the following pages.

When creating dimension features and assigning them a style, some properties can be overridden on a feature-by-feature basis. The properties that you can override for each feature are:

- Dimension line display
- Dimension line arrow symbol display
- Dimension text value
- Extension line display

For more information on editing dimension features, see *Editing in ArcMap*.

## Schema locking

An exclusive lock is required when creating, renaming, or deleting styles in a dimension feature class. For more information on schema locking, see the chapter 'Creating new items in a geodatabase' in this book.

## Lines and Arrows tab

The image shows the 'Dimension Style Properties - [Style1]' dialog box, specifically the 'Lines and Arrows' tab. The dialog is divided into several sections:

- Properties:** Includes sub-tabs for 'Lines and Arrows', 'Text', and 'Fit'.
- Dimension line symbols:** Contains three preview boxes: 'Begin symbol' (a dashed box with a diagonal line), 'Line symbol' (a solid orange line), and 'End symbol' (a black arrowhead).
- Dimension lines:** Includes a 'Dimension lines' dropdown set to 'Both', an 'Arrows' dropdown set to 'Both', and a 'Baseline Height' text box containing '0'.
- Extension lines:** Includes a 'Symbol' dropdown (showing a blue line), an 'Extension lines' dropdown set to 'Both', an 'Overshot' text box containing '15', and an 'Offset' text box containing '5'.
- Preview:** A window showing a dimension line with a value of '0.00 ft' and arrows at both ends.
- Buttons:** 'OK' and 'Cancel' buttons at the bottom right.

Callouts provide the following explanations:

- Symbology for the dimension line and the endpoints of the dimension lines.** Points to the 'Dimension line symbols' section.
- Symbology for the extension lines.** Points to the 'Symbol' dropdown in the 'Extension lines' section.
- Setting for which dimension lines for dimension features with this style are displayed.** Points to the 'Dimension lines' dropdown.
- Setting for which extension lines are displayed.** Points to the 'Extension lines' dropdown.
- The extension line overshoot in map units.** Points to the 'Overshot' text box.
- The height (in map units) of the dimension line that each dimension feature created by the Baseline Dimension command will be above the preceding dimension feature's dimension line.** Points to the 'Baseline Height' text box.
- Setting for which dimension line end arrows are displayed.** Points to the 'Arrows' dropdown.
- The extension line offset in map units.** Points to the 'Offset' text box.
- Preview of the dimension style. The preview updates as you change elements of the style.** Points to the 'Preview' window.

## Text tab

The actual string that is displayed for the dimension text. The text label can be turned off.

Setting for whether to align the text with the dimension line. If this is checked, the text will always be parallel to the dimension line. If this is unchecked, the text will be oriented as described by the properties of its symbol.

Setting for the precision of the value displayed by the dimension text. The value will be rounded off to the precision selected.

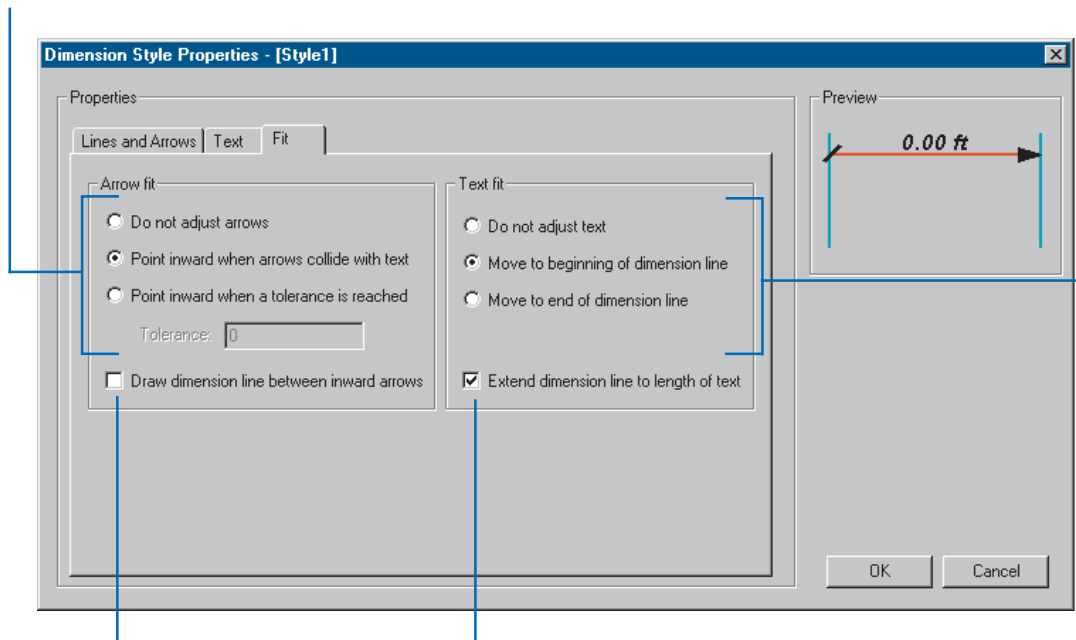
Symbology for the dimension text.

The units of the dimension text. If set to Map units, the dimension text value will be in the units described by the dimension feature class's coordinate system. If a value other than map units is specified, then the dimension value will automatically be converted to those units at display time.

## Fit tab

The behavior of the dimension when its length is too short for the dimension line arrows and the text to fit between the extension lines

Setting for how to resolve the case where the dimension is too short for the dimension text to fit between the extension lines



Setting for whether to draw the dimension line between the arrows for inward dimensions

Setting for whether to extend the dimension line to underline the dimension text when it is moved to the outside of the dimension

## Tip

### New dimension styles

Newly created dimension styles' names are colored red, indicating that they have not yet been written to the database. Once you click Apply or OK, these dimension styles' names are colored black to indicate they are in the database.

## Tip

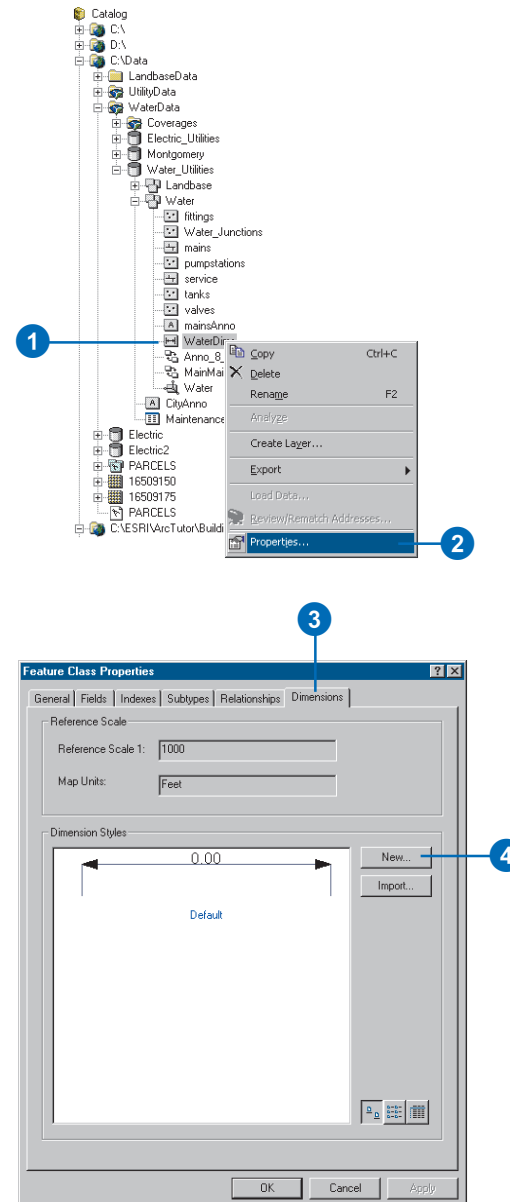
### Style IDs

Each dimension style has an ID. A dimension feature stores this ID in a field called *STYLEID*. You can create and apply default values and domains to this field at the feature class level or the subtype level. You can do this as an alternative to using the Dimensioning toolbar in ArcMap to assign styles to your dimension features.

For more information on default values, domains, and subtypes, see the chapter 'Subtypes and attribute domains' in this book; for more information on the Dimensioning toolbar and editing dimension features, see Editing in ArcMap.

## Creating a new dimension style

1. Right-click the dimension feature class.
2. Click Properties.
3. Click the Dimensions tab.
4. Click New. ▶



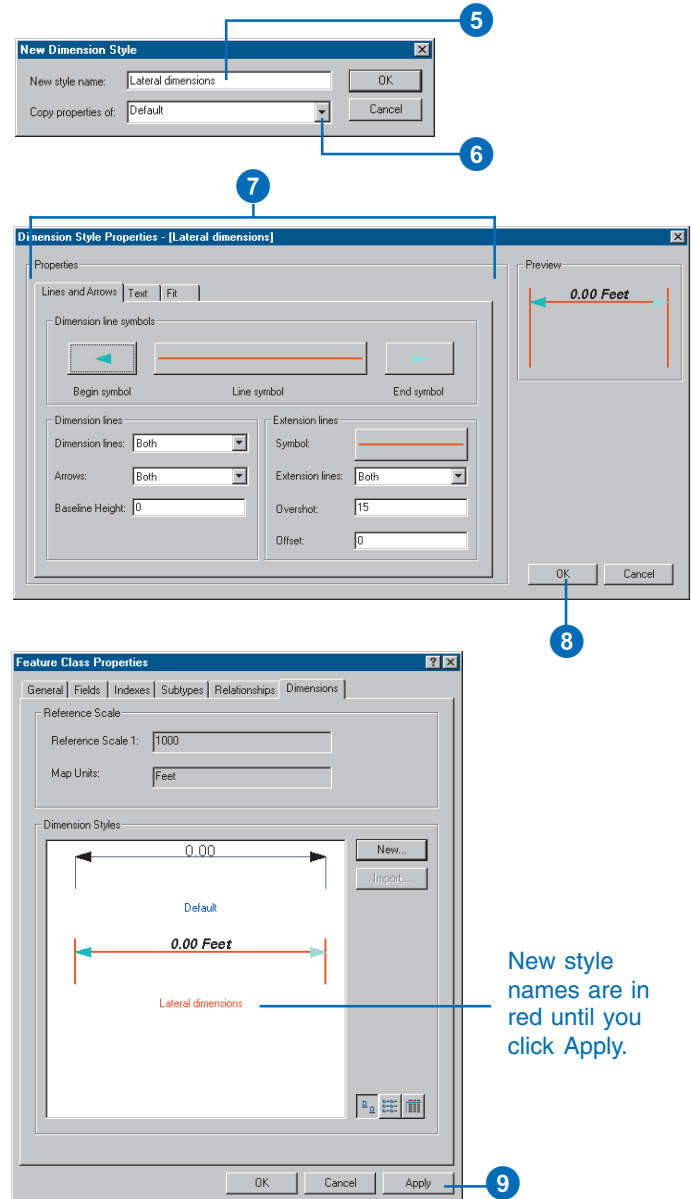
## Tip

### The template style

You can choose to copy the characteristics of an existing style to your new style, then use the Dimension Style Properties dialog box to modify the style elements.

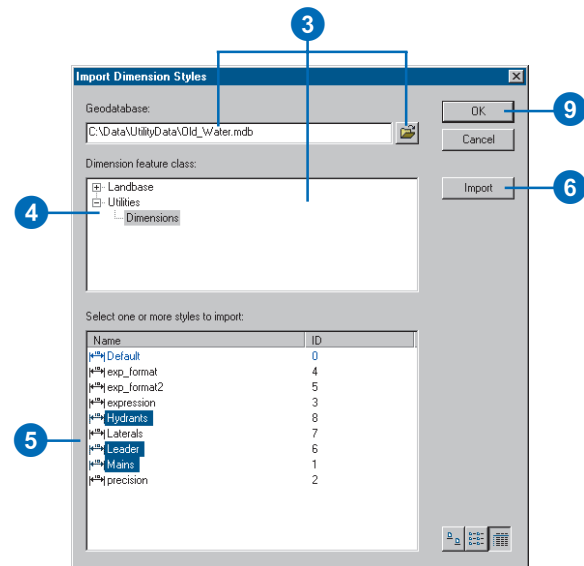
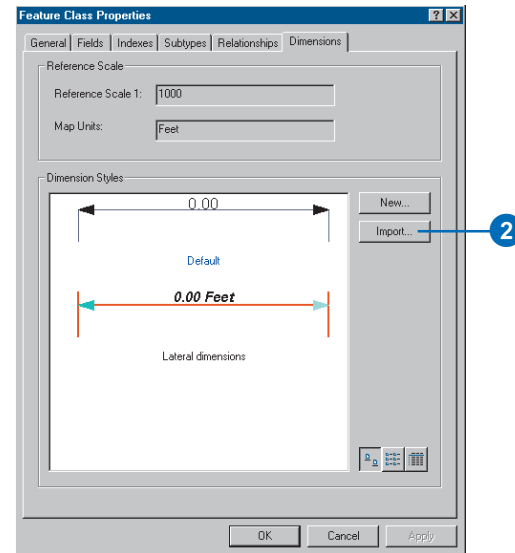
You cannot use styles that have not yet been stored in the database as a template for your new style. Any styles whose names are in red cannot be used as templates.

5. Type a name for the new style.
6. Click the dropdown arrow and click the style in the dimension feature class whose properties you want to copy into the new style.
7. Modify those elements of the dimension style you wish to change in the Dimension Style Properties dialog box.
8. Click OK.
9. Click Apply.



## Importing dimension styles

1. Follow steps 1 through 3 for 'Creating a new dimension style'.
2. Click Import.
3. Click the Browse button to browse for a geodatabase. Once a geodatabase is selected, the dimension feature classes and feature datasets containing dimension feature classes are listed in the tree view.
4. Click the dimension feature class that contains the styles you want to import.
5. Hold down the Ctrl key and click the dimension styles that you want to import.
6. Click Import.
7. Click OK.
8. Repeat steps 4 through 8 to import more styles from the same geodatabase or steps 3 through 8 to import styles from a different geodatabase.
9. Click OK.



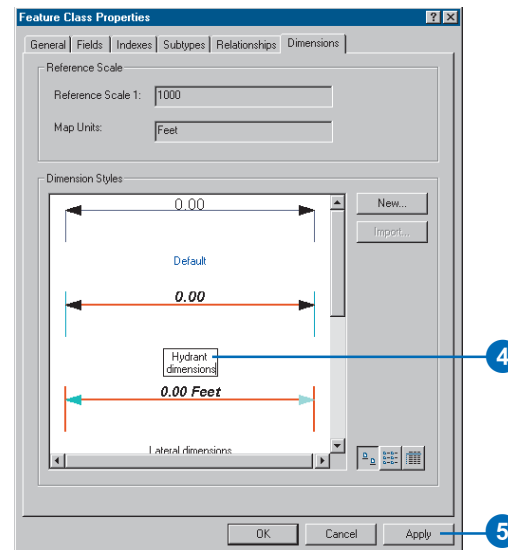
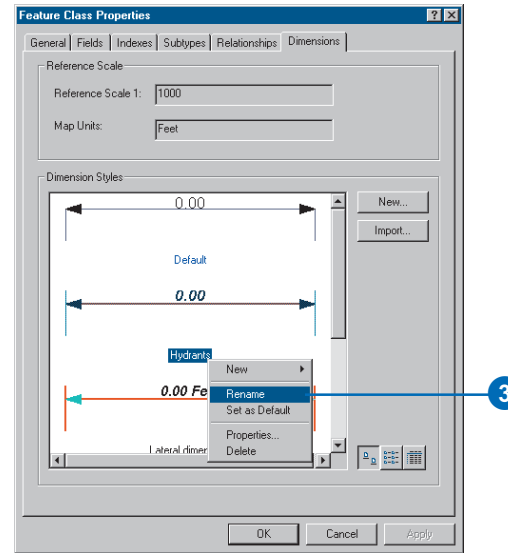
## Tip

### Dimension features

If dimension features that reference the style you want to rename already exist in your dimension feature class, they will not be affected. Those features will still reference the same style after it has been renamed.

## Renaming a dimension style

1. Follow steps 1 through 3 for 'Creating a new dimension style'.
2. Right-click the dimension style you want to rename.
3. Click Rename.
4. Type the new name for the style and press Enter.
5. Click Apply.





## Tip

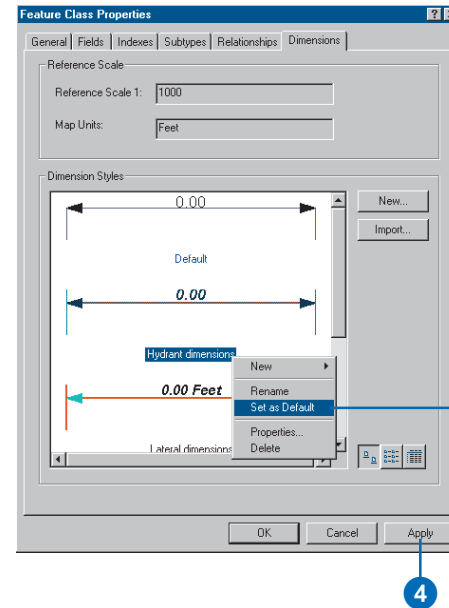
### The default style

Any style can be the default style. The default style is the style that is to be assigned to all new dimensions when they are created in ArcMap, unless another style has been selected in the Dimensioning toolbar.

For more information on creating dimension features and the Dimensioning toolbar, see Editing in ArcMap.

## Setting the default dimension style

1. Follow steps 1 through 3 for 'Creating a new dimension style'.
2. Right-click the dimension style you want to set as the default.
3. Click Set as Default.
4. Click Apply.



## Tip

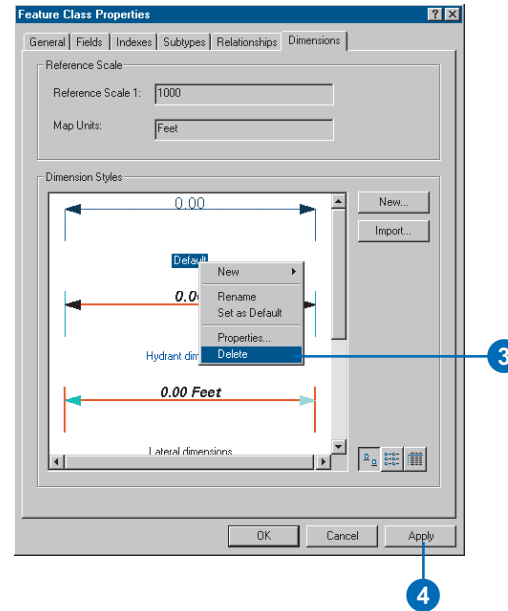
### Deleting styles

All features that reference a style that has been deleted are symbolized as raw lines with a box for the text. You can use ArcMap to assign a different style to those dimension features.

For more information on editing dimension features, see Editing in ArcMap.

## Deleting a dimension style

1. Follow steps 1 through 3 for 'Creating a new dimension style'.
2. Right-click the dimension style you want to delete.
3. Click Delete.
4. Click Apply.



# Working with a versioned geodatabase

# 10

## IN THIS CHAPTER

- **Integrating versioning with your organization's work flow**
- **Registering data as versioned**
- **Creating and administering versions in ArcCatalog**
- **Working with versions in ArcMap**
- **Editing and conflict resolution**
- **Editing a version**
- **Versioning scenarios**

With ArcGIS, multiple users can access geographic data in a geodatabase through *versioning*. Versioning lets users simultaneously create multiple, persistent representations of the database without data replication. Users can edit the same features or rows without explicitly applying locks to prohibit other users from modifying the same data.

An organization can use versioning to manage alternative engineering designs, solve complex what-if scenarios without impacting the corporate database, and create point-in-time representations of the database.

Primarily, versioning simplifies the editing experience. Multiple users can directly modify the database without having to extract data or lock features and rows before editing. If, by chance, the same features are modified, a *conflict* resolution dialog box guides the user through the process of determining the feature's correct representation and attributes.

Versioned databases may contain topologies. For more information on how versioning affects topologies, see the chapter 'Topology' in this book.

Versioned databases may also be the check-out databases for disconnected editors. See the chapter 'Disconnected editing' in this book for more information on using versioned databases for disconnected editing.

You can view versioned geodatabases in ArcView. To take advantage of other versioning functionality, you need an ArcEditor or ArcInfo license.

# Integrating versioning with your organization's work flow

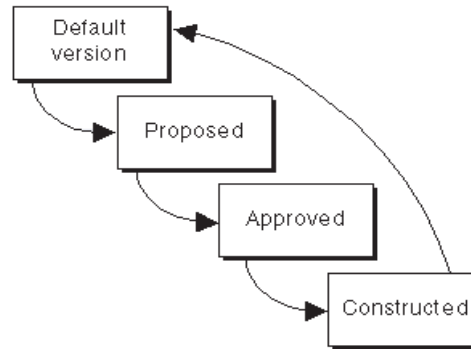
The geodatabase and versioning provide organizations with advanced data storage techniques that revolutionize the *work flow* process in many applications where spatial information is used. Engineers can generate design alternatives using the entire database. Spatial analysts can perform complex “what if” scenarios without affecting the current representation of the database. Database administrators can create “historical” snapshots of the database for archiving or database recovery.

In the long run, an organization benefits from implementing a versioned database. The data is centrally located in one corporate database. There is never a need to extract units of the database to update or lock map sheets or individual features. These factors simplify the administration process.

## The work flow process

The evolution of the work flow process—how projects or *work orders* transpire over time—varies greatly from organization to organization and throughout each sector of the business community. Therefore, the geodatabase's versioning process has been designed to be flexible enough to accommodate the most basic of work flow processes as well as the most complex and to be sufficiently restrictive with or without additional application customization.

Common work flow processes usually progress in discrete stages. At each stage, different requirements or business rules may be enforced. Typically, during each stage of the process, the project or work order is associated with a named stage. For example, within the utility domain, common stages include “working”, “proposed”, “accepted”, “under construction”, and “as built”. The process is essentially cyclical. The work order is initially generated and assigned to an engineer, then modified over time as it progresses from stage to stage, and finally the changes are “posted” or applied back to the corporate database.



*A common work flow process evolving through each stage of a project*

This is one example of how versioning can help simplify the work flow process. Because the work flow process may span days, months, and even years, the corporate database requires continuous availability for daily operations. If a work order applied restrictive locks to the data involved in the process, other database users might not be able to perform their daily work assignments.

To implement your work flow in the geodatabase, versions can be created to correspond with each stage of the work flow process. Alternatively, you may want to create one version for each work order and modify the version's name to represent the current stage as the process proceeds through each step.

The current structure of your organization's work flow will significantly influence how you implement the geodatabase versioning process to manage your spatial transactions. The flexibility and openness of the system will allow you to determine the best solution to meet the requirements of your business processes.

The remaining sections of this chapter will help illustrate how to use ArcCatalog and ArcMap to perform various versioning tasks. In particular, the last section provides examples of how an organization can implement work flow processes using the geodatabase's versioning capabilities. For additional details on managing your organization's work flow with versions, read *Modeling Our World*.

## Registering data as versioned

Before editing feature datasets, feature classes, and tables in a multiuser geodatabase, you must first register the data as versioned in ArcCatalog.

Making a feature class or table multiversioned requires a unique integer field. This is typically the OBJECTID field. Only the owner of the data may register or unregister the object as versioned.

When unregistering a dataset or feature class as versioned in ArcCatalog, a warning dialog box may appear informing you that outstanding edits still remain in existing versions. Unregistering the class as versioned will remove all the edits. To preserve the edits, you must compress the database before unregistering the versioned data.

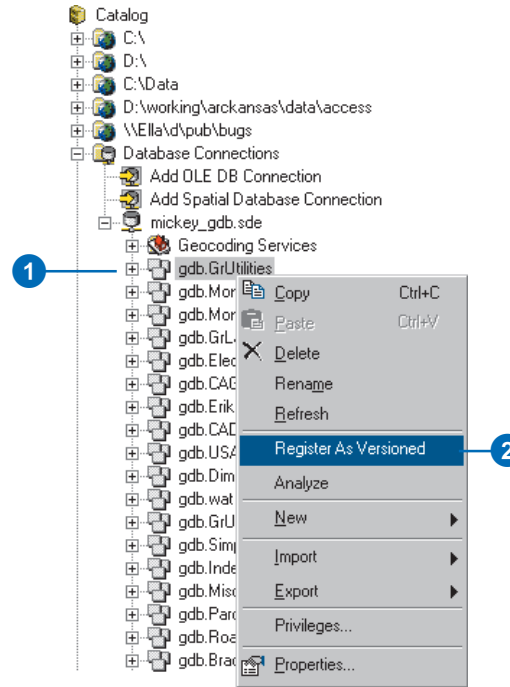
Personal geodatabases do not support versioning.

### Tip

#### Registering data as versioned

*Registering a feature dataset as versioned registers all feature classes within the feature dataset as versioned.*

1. Right-click the feature dataset, feature class, or table you want to register as versioned in the ArcCatalog tree.
2. Click Register As Versioned.



# Creating and administering versions in ArcCatalog

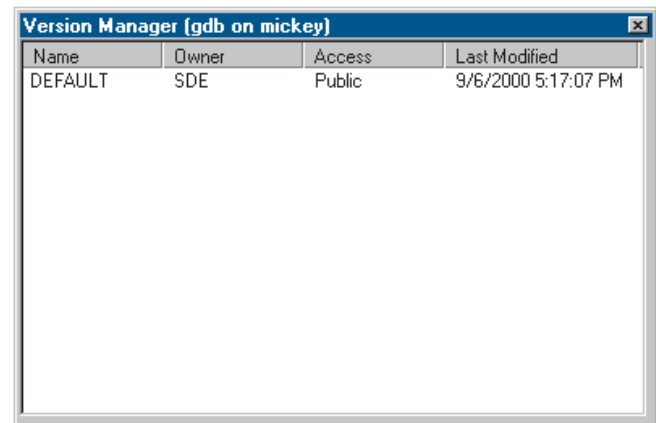
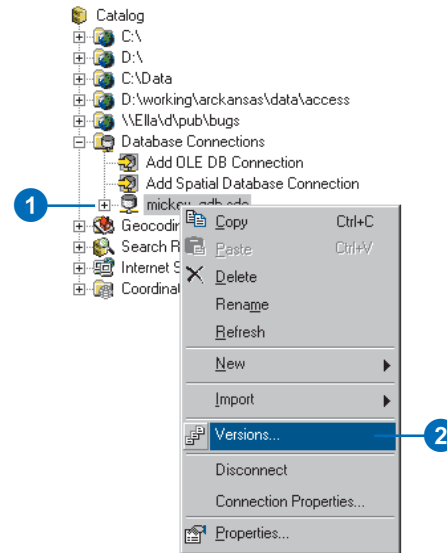
ArcCatalog lets you create new versions, rename existing versions, delete versions, and modify version properties. These administrative tasks are accomplished using the Version Manager dialog box.

Initially, the database consists of one version named “DEFAULT”, owned by the ArcSDE administrative user. The new versions that are created are always based on an existing version. When the new version is created, it is identical to the version from which it was derived. Over time, the versions will diverge as changes are made to the parent version and to the new version.

Each version has several properties: an alphanumeric name, an owner, an optional description, the creation date, the last modified date, the parent version, and the version’s permission. ▶

## Creating a new version

1. Create a new connection to the database in ArcCatalog with the Add SDE Connection dialog boxes described in the ‘Introduction’ chapter of this book.
2. Right-click your database connection in the Catalog tree and click Versions. ▶



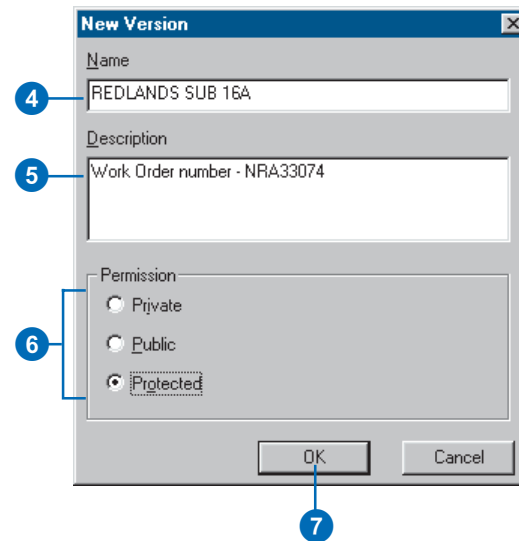
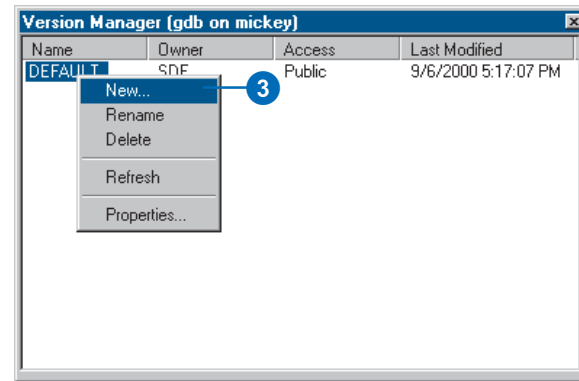
A version's permission can only be changed by its owner. The available permission settings are:

- Private—only the owner may view the version and modify available feature classes.
- Protected—any user may view the version, but only the owner may modify available feature classes.
- Public—any user may view the version and modify available feature classes.

Only the version's owner can rename, delete, or alter the version. A parent version cannot be deleted until all dependent child versions are first deleted.

To improve database performance, the database should be compressed periodically. Compressing the database removes all unreferenced database states and redundant rows. Only the ArcSDE administrator can perform this task. For additional details, see the 'Versioning scenarios' section at the end of this chapter. ►

3. Right-click a version and click New.
4. Type the new version's name.
5. Type a description.
6. Click the appropriate permission type; the default is Private.
7. Click OK.





Finally, after compressing the database or editing the data, the Analyze command should be executed to update the database statistics for each dataset or feature class. This will help improve display and query performance.

### Tip

#### Descriptions

You can use the version description to provide additional information regarding the version's purpose.

### Tip

#### Sorting versions

In the Version Manager dialog box, you can sort versions by clicking on a column heading.

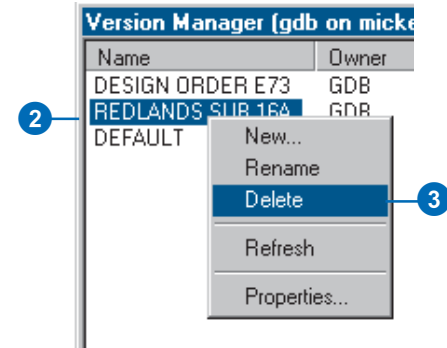
## Renaming a version

1. Right-click your database connection and click Versions.
2. Right-click the version you want to rename and click Rename.
3. Type a new name and press Enter.



## Deleting a version

1. Right-click your database connection and click Versions.
2. Right-click the version you want to delete.
3. Click Delete or press Delete on your keyboard.



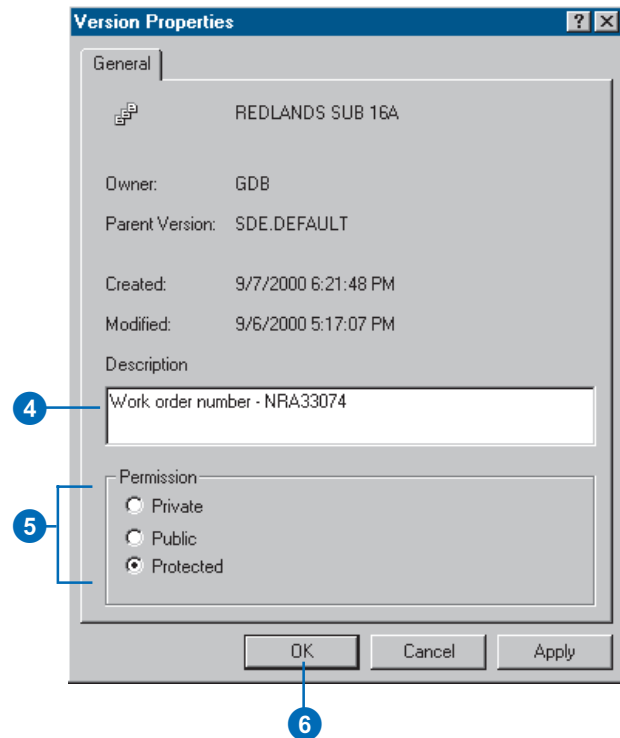
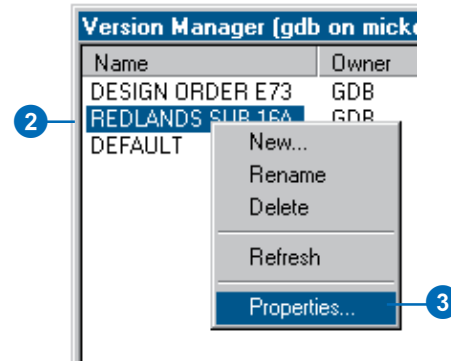
## Tip

### Refresh

Use the Refresh command to update the properties of each version with its current values.

## Changing a version's properties

1. Right-click your database connection and click Versions.
2. Right-click a version.
3. Click Properties.
4. Type the new description.
5. Click the new permission type.
6. Click OK.

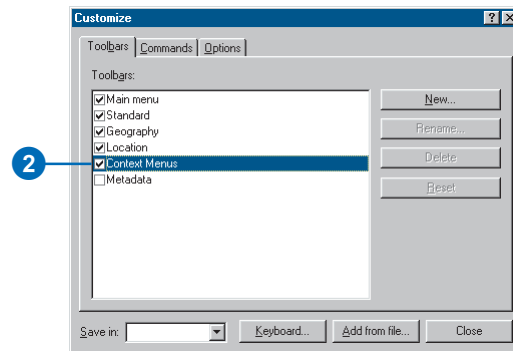
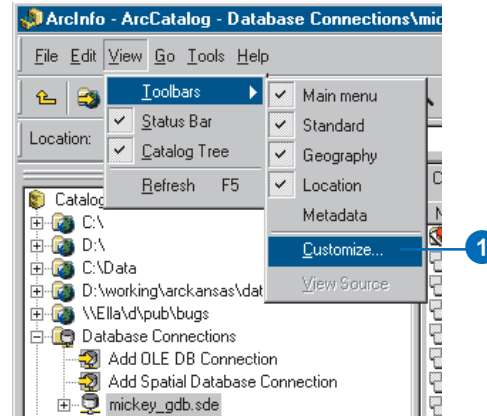


## See Also

For more information on how to customize ArcCatalog, see *Using ArcCatalog* and *Exploring ArcObjects*.

## Adding the Compress command to ArcCatalog

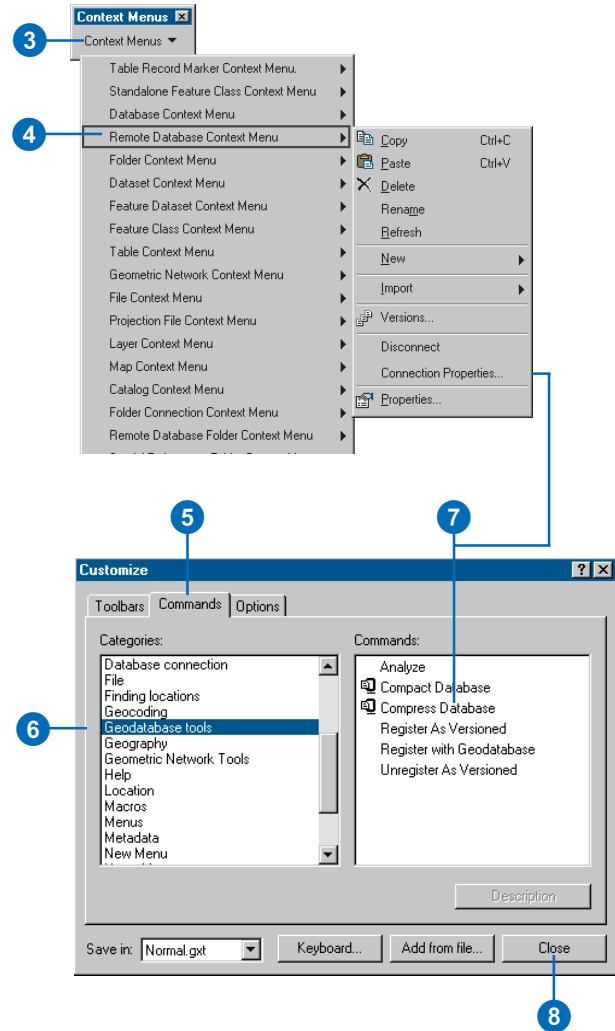
1. Click View, click Toolbars, and click Customize in ArcCatalog.
2. Check Context Menus in the list of toolbars. ▶



3. Click the Context Menus menu.
4. Click the arrow next to the Remote Database Context Menu.
5. Click the Commands tab in the Customize dialog box.
6. Click Geodatabase tools.
7. Click and drag the Compress Database command from the Commands list and drop it on the context menu.

The command appears in the context menu.

8. Click Close on the Customize dialog box.



## Tip

### Analyze

After compressing a database, always analyze your data to update the database statistics. Use the same method to add the Analyze command to ArcCatalog as you use to add the Compress command.

## See Also

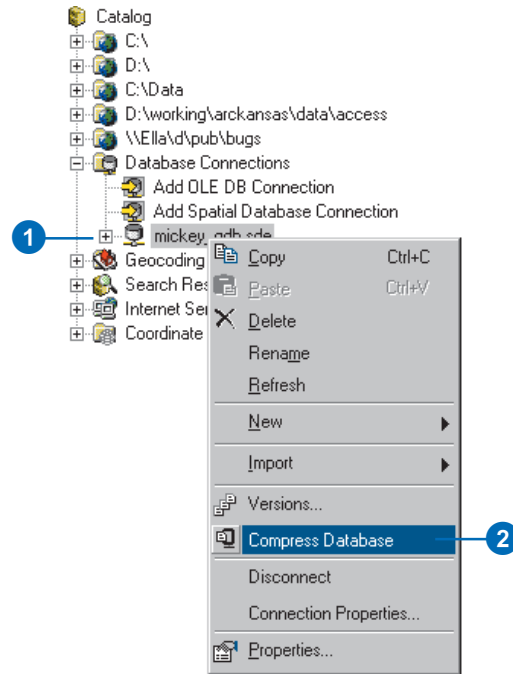
For more information on how to administer an ArcSDE database, see Managing ArcSDE Services.

## See Also

For more information on how to create connections to databases in ArcCatalog, see the 'Introduction' chapter in this book.

## Compressing the database

1. Create a new database connection in ArcCatalog as the ArcSDE administrative user.
2. Right-click the new database connection and click Compress Database.



## Working with versions in ArcMap

In ArcMap, you can view and work with multiple versions simultaneously, create new versions, and change the feature classes or tables from one version to another version. You can also use the version manager, refresh a version's *workspace* connection, and modify available feature classes in ArcMap.

To create a new version, at least one version must be present in the map. If multiple versions are present, you will need to specify the parent version. The newly created version will then be identical to the parent version.

Changing versions allows you to quickly navigate between two versions by changing the feature classes currently in the map. This simplifies the process of viewing the differences between feature classes or performing an analysis with two versions. ►

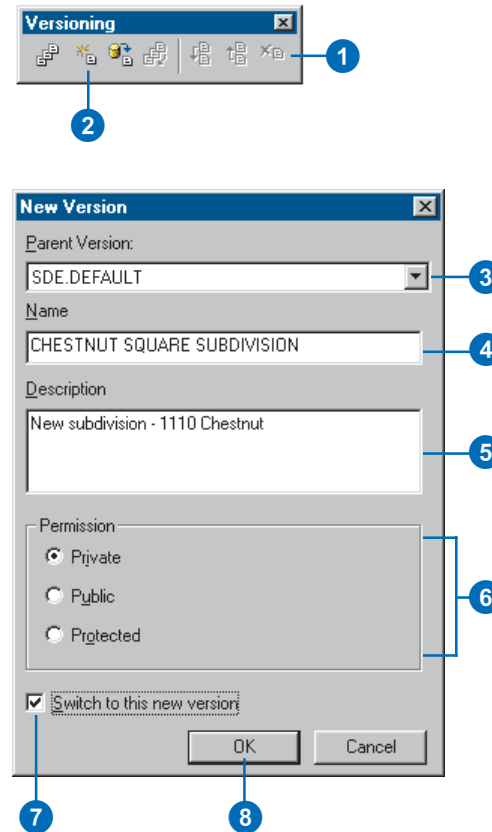
### Tip

#### Creating backup versions

You can create alternative versions as online backups to the original version.

## Creating a new version in ArcMap

1. Add the Versioning toolbar to the map.
2. Click the Create New Version button. At least one version must be in ArcMap before the button is enabled.
3. Click the Parent Version dropdown arrow and click the parent version from which you want to create the new version.
4. Type the new version's name.
5. Optionally, type a description.
6. Click the appropriate permission type.
7. Optionally, if you are not currently editing, check the check box to switch the parent version to the new version.
8. Click OK.



When a version workspace is changed to a different version, all feature classes present in the workspace will represent the new “target” version.

Two methods are available in ArcMap for changing versions. You can change versions from the Versioning toolbar or in the table of contents.

When you work in a multiuser environment, the database may be modified by another user at the same time you’re viewing the database. Therefore, the feature classes present in ArcMap may become outdated.

To update feature classes in ArcMap you can refresh one or all of the version workspaces present by clicking the Refresh button on the Versioning toolbar. While you are editing, the Refresh button is unavailable.

You can have as many versions in the map as needed, but you can only edit one version per *edit session*.

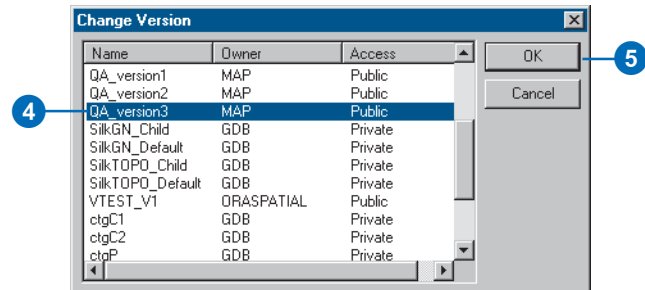
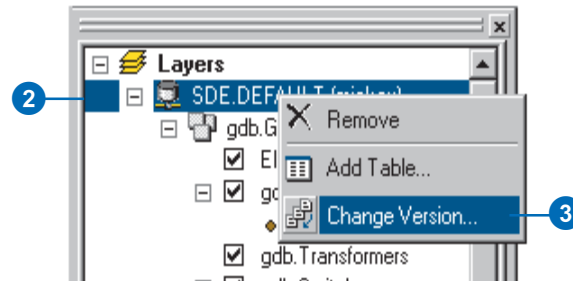
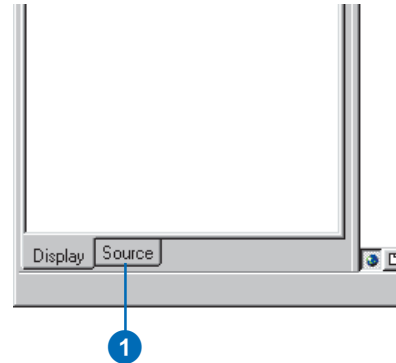
## Tip

### The Change Version command

Use the *Change Version* command instead of adding multiple version workspaces to your map document.

## Changing versions

1. Click the Source tab at the bottom of the ArcMap table of contents to list the workspaces in your map.
2. Right-click a version workspace.
3. Click Change Version.
4. Click the version to which you want to change.
5. Click OK.



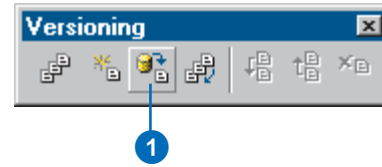
## Tip

### Preserving a version

*If you need to preserve a current representation of the database, create a new version before refreshing.*

## Refreshing a workspace

1. Click the Refresh button on the Versioning toolbar.





## Editing and conflict resolution

The geodatabase is designed to efficiently manage and support long *transactions* using versions. The geodatabase also allows multiple users to edit the same version at the same time. Each edit session in ArcMap is its own representation of the version until it is saved. Saving the edit session applies your modifications to the version, making these changes immediately accessible in the database.

When multiple users simultaneously edit a version or *reconcile* two versions, *conflicts* can occur. Reconciling is the process of merging two versions. Conflicts occur when the same feature or topologically related features are edited by two or more users, and the database is unclear about which representation is valid. Conflicts are rare but can occur when overlapping geographic areas in the database are edited. To ensure database integrity, the geodatabase detects when a feature has been edited in two versions and reports it as a conflict. ArcMap provides the necessary tools for you to investigate conflicts, though you make the final decision as to the feature's correct representation.

ArcMap provides tools to resolve conflicts and reconcile and *post* versions. The next sections explain these capabilities in more detail.

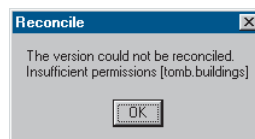
### Reconcile

The Reconcile button in ArcMap merges all modifications between the current edit session and a target version you select. Any differences between the features in the target version and the features in the edit session are applied to the edit session. Differences can consist of newly inserted, deleted, or updated features. The reconcile process detects these differences and discovers any conflicts. If conflicts exist, a message is displayed, followed by the conflict resolution dialog box. You must reconcile edits with a target version before posting them to that version. A

target version is any version in the direct ancestry of the version such as the parent version or the DEFAULT version.

In addition, the reconcile process requires that you are the only user currently editing the version and that you are the only user able to edit the version throughout the reconcile process until you save or post. If another user is simultaneously editing the version or attempts to start editing since you have reconciled, an error message will inform you that the version is currently in use.

The reconcile process requires that you have full permissions to all the feature classes that have been modified in the version being edited. If a feature class is modified in the version for which you do not have update privileges, an error message appears. You will not be able to reconcile the versions; a user with adequate permissions to perform the reconcile must do this for you.



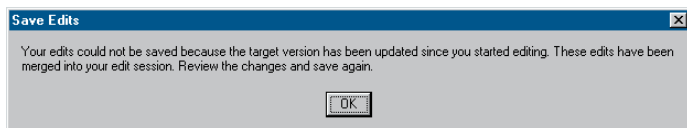
*An error message appears when you do not have permission to a feature class to reconcile versions.*

For example, suppose you have completed your changes in a version and need to post the version to the DEFAULT version. You must first reconcile the version with a target version you select, resolve any conflicts if necessary, then post.

### Autoreconciliation

Suppose that since you started editing a version, another user has saved edits to the same version. Enabling or disabling

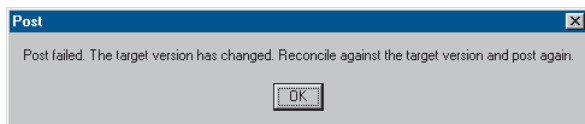
autoreconciliation affects whether you are notified of the other user's edits when you save yours. If you want to be notified so you can review the results of the merge before saving your edits, disable autoreconciliation. If you do not want to be notified and want to save without reviewing the results of the merge, enable autoreconciliation. Regardless of how you set autoreconciliation, ArcMap will always notify you if there are conflicts when you save.



*This message displays if you disable autoreconciliation and attempt to save after another user has saved.*

## Post

You can post a version after you have performed a reconcile. Once the edit session has reconciled with a target version, clicking the Post button synchronizes the version with the reconciled version and saves the data. Posting cannot be undone, as you are applying changes to a version that you are not currently editing. If the reconciled version is modified between reconciling and posting, you will be notified to reconcile again before posting.



*This message indicates that the target version has been modified since the reconciliation; reconcile again before posting.*

## Conflicts

Conflicts occur when the same feature (topologically related features or relationship classes) is modified in two versions—the current version being edited and a target version. Conflict detection only occurs during the reconciliation process. If conflicts are detected, a message appears, followed by the conflict resolution dialog box.

There are three types of conflicts that can occur when an edit version is reconciled with a target version; update–update, update–delete and delete–update. An update–update conflict occurs when the same feature has been updated in each version. An update–delete conflict occurs when a feature has been updated in the edit version and deleted in the reconcile version. A delete–update conflict occurs when a feature has been deleted in the edit version and updated in the reconcile version.

When conflicts are detected, the parent version's feature representation takes precedence over the edit session's representation. Therefore, all conflicting features in the current edit session are replaced by their representation in the parent version. If multiple users are editing the same version and conflicts are detected, the feature that was first saved, the current version's representation, is preserved by replacing the edit session's feature representation. ArcMap ensures database integrity by forcing you to interactively inspect each conflict and resolve the conflict by replacing the feature in the current version with your edit session's representation.

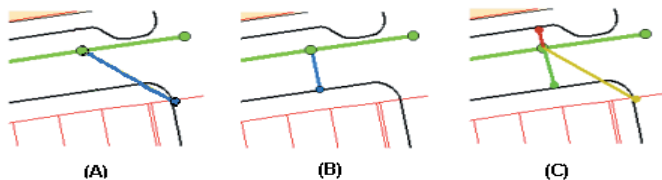
## Conflict resolution

Once conflicts are detected, a conflict resolution dialog box appears containing all the conflict classes and their features or rows in conflict. The conflict resolution dialog box allows you to interactively resolve conflicts at the level of the feature class or individual feature. Resolving the conflict implies that you will

make a decision as to the feature's correct representation; this could mean doing nothing at all if you are satisfied with the current feature's representation.

You can choose from three representations of the conflicting feature or row to resolve the conflict. The preedit version is the feature's representation when you initially started editing, before making any changes. The edit session version represents the feature as it existed before you performed the reconcile. The last representation is the conflict version, the feature's representation in the conflicting version.

Selecting a feature class or individual feature displays any of the three representations of the feature in the map. The preedit's version is displayed in yellow, the edit session's version is displayed in green, and the conflict's version is displayed in red. You can also optionally enable or disable the display settings for each version (preedit, edit session, and conflict) by clicking the Display command on the context menu and checking or unchecking the corresponding version.



The lateral in blue as it existed prior to editing (A), the lateral after being modified (B), and the three representations during conflict resolution (C).

When you select a feature in the conflict resolution dialog box, each version's representation of the feature's or row's attributes is listed in the bottom half of the box. A red dot to the left of the field name identifies why the feature is a conflict. For example, if the feature's geometry was edited in each version, a red dot appears next to the shape field. The same principle holds true for

attribute conflicts. If a feature has been deleted in either version, “<deleted>” appears for that version's attribute value. Therefore, a red dot marks each column, signifying that each column is an update–delete or a delete–update conflict.

Property	Conflict	Edit	Pre-Edit
FID	1	1	1
• SHAPE			
SEG_ID	33211070	33211070	33211070
• SYMBOL	14	6	14
PIPE_SIZE	8	8	8
ACC_NO	16-A	16-A	16-A
SEW_NO	38952	38952	38952
MATERIAL	STEEL	STEEL	STEEL
SEW_SHAPE	CIRC	CIRC	CIRC
HEIGHT	0	0	0
WIDTH	0	0	0
INST_YEAR	1982	1982	1982
DRAIN_AREA	S.BRANCH	S.BRANCH	S.BRANCH
SEP_COMB	SEPTIC	SEPTIC	SEPTIC
PUB_PRI	PUBLIC	PUBLIC	PUBLIC

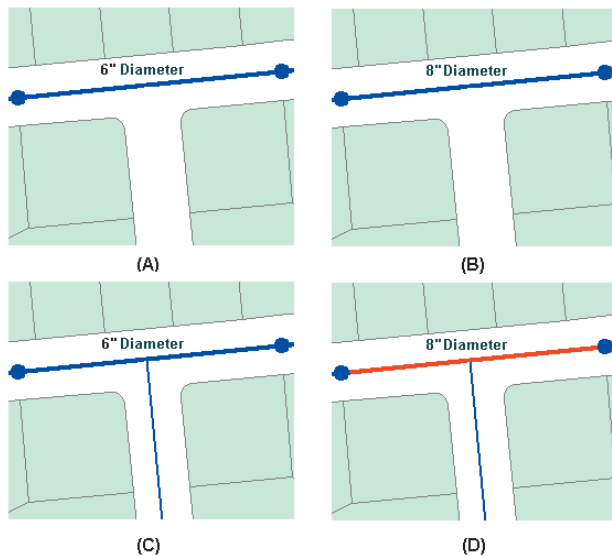
This conflict resolution dialog box shows two feature classes with conflicts and a feature with each of its version's attributes.

Resolving a conflict implies that you made a conscious decision about the feature's correct representation. You can select the feature in the conflict resolution dialog box and replace the current feature in the map with any of the three representations of the feature. This allows you to quickly update and replace conflicting features. If further modifications are required, you can simply use any of the ArcMap editing tools to update the feature.

## Conflicts with geometric networks, feature-linked annotation, and relationships

Resolving conflicts with features that are related to other features through *geometric networks*, *feature-linked annotation*, and *relationship classes* is different from resolving conflicts with simple feature classes. Because each of these feature classes has specific geodatabase behaviors that can impact other feature classes, resolving a feature conflict may impact related features.

When you edit network features, changes to the geometric network and to the logical network may create conflicts.



*The original water main (A), the water main changed to an 8-inch diameter in the first edit session (B), a new service inserted in the second edit session (C), and the water main in red shown as a conflict (D).*

For example, when you add a service to a main, the main will not be physically split in the geometric network but will be split in the *logical network*. Therefore, while you have not directly edited the main's geometry, it has been edited logically. If the target version you are reconciling has also modified the main, then the new service you inserted will create a conflict with the main.

Resolving a conflict involving geometric network feature classes requires understanding how the Replace With command in the conflict resolution dialog box will update the existing network topology present in the edit session.

In the previous example, two users modified the water main—one by changing an attribute and the other by connecting a new service. Resolving the conflict would merely require investigating the differences and seeing that the conflict is valid and no further resolution is required. Since the main contains the correct attribute for the diameter, the new service is correctly connected to the main. But there are cases when resolving conflicts involving a junction feature class will also update the connected network edge.

Working with feature-linked annotation requires that you remember one rule: when replacing a feature that has feature-linked annotation, both the feature and the annotation are replaced with the new feature and annotation. You may have to further edit the new annotation. For example, you may encounter a conflict in which you have moved a feature and repositioned its annotation. The conflict version has performed the same edit, moving the feature and rotating the annotation. Your decision is to replace the feature with the conflict version's feature. This action deletes the existing feature-linked annotation, inserts the conflict feature, and creates a new annotation. You will then need to further edit the new annotation by moving and rotating it as necessary.

*Relationships* have similar dependencies to feature-linked annotation. Deleting a feature from an origin relationship class may trigger a message to delete a feature from the destination relationship class. Therefore, be aware of the ramifications of simply replacing conflicts involving feature classes that participate in relationship classes.

An example of when a conflict can arise between relationship classes is if you were to update the origin class primary field, breaking the relationship in version A. At the same time, in version B, the destination class-related feature is also updated. When you reconcile the versions, since the destination class is dependent on the origin class, a conflict is detected. A similar example is if you were to delete a pole that has a relationship to a transformer, the transformer is also deleted. But in the conflict version, the transformer's attributes are edited. An update–delete conflict would be detected when reconciled.

## Conflicts with topologies

Because features in feature classes that participate in a topology can share geometry with other features, the process of resolving conflicts between versions of topological feature classes is different from resolving conflicts with simple feature classes. It is also different from that used to resolve conflicts with geometric networks, relationship classes, and feature-linked annotation.

When a feature class that participates in a topology is edited, other topologically related features may be simultaneously changed. The changed features may belong to the same feature class or to one or more other feature classes. To manage the process of detecting new topology errors that may have been introduced by edits, topologies record where edits have been made as dirty areas. Editing features in a topology creates dirty areas in the topology.

New topology errors may occur when edited parent and child versions are reconciled, even when the dirty areas within each version have been validated and are free of errors. In order to detect such topology errors, the dirty areas in a child version are all returned to dirty status after changes from the parent version are brought into it during a reconcile. After the reconcile these areas can be revalidated and any errors will be detected.

Reconciling two versions that do not contain active dirty areas may still result in dirty areas. Any dirty area that has been present in the child version, whether it has been validated or not, will be a dirty area after the versions are reconciled. In general, when reconciling a version:

- Any dirty area the child version inherited from the parent version, whether it is validated in the child version or not, will be a dirty area after the reconcile.
- Any dirty area that was created for any feature that was created, updated, or deleted in the child version, whether it is validated or not, will be a dirty area after the reconcile.

For more information on reconciling versions with topologies, see the section ‘Topology and versioning’ in the ‘Topology’ chapter of this book or in the chapter ‘Editing topology’ in *Editing in ArcMap*.

## Editing a version

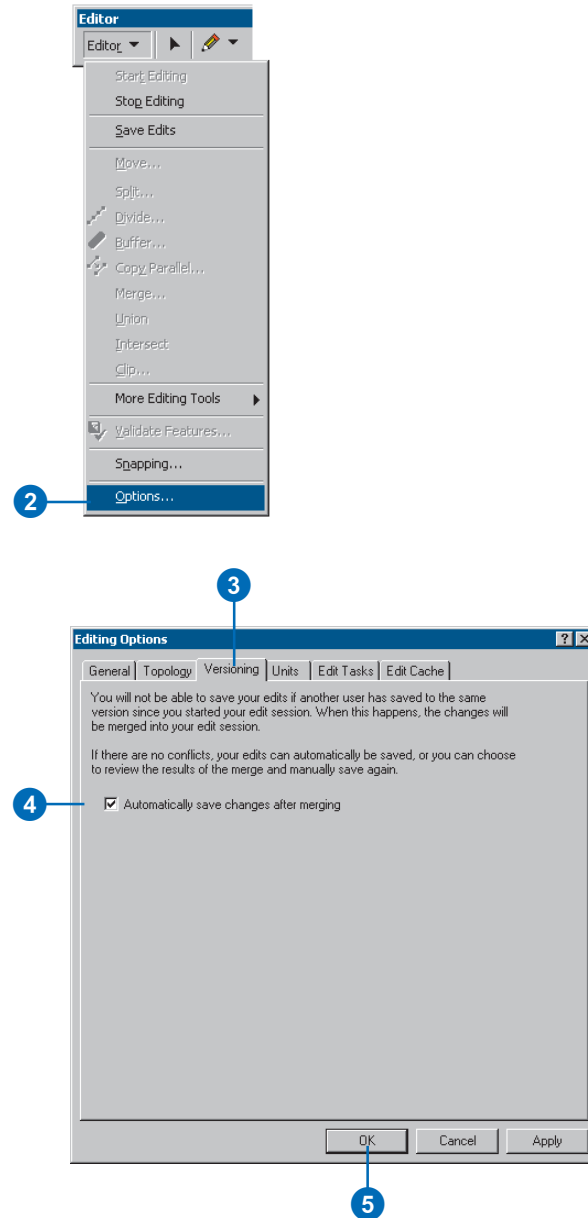
You use the versioning toolbar in ArcMap to reconcile versions, resolve conflicts, and post versions.

When you start editing, if multiple versions are present in the map, you will have to select one version. Starting an edit session on a version creates a new, unnamed, temporary version that exists until you save or end the edit session. You are the only user who can see your changes until you explicitly save.

When saving an edit session, you have an option to enable or disable autoreconciliation. If enabled, autoreconciliation will automatically reconcile your edit session with the version's current database state and save, making your changes available to others using the database. If autoreconciliation is not enabled, then when you save, your edit session will be reconciled with the version's current database state. A message will inform you that the edit session has been reconciled but has not been saved. This will only occur if a second user has also edited the version and saved since you ►

## Enabling and disabling autoreconciliation

1. Click Editor and click Start Editing.
2. Click Editor and click Options.
3. Click the Versioning tab.
4. Check the check box to enable autoreconciliation or uncheck the check box to disable autoreconciliation.
5. Click OK.



started editing. You will need to save again to make your changes available to others using the database.

Based on your organization's work flow, you may eventually need to reconcile two versions. Reconciliation is the process of merging features from a target version into the current edit session. Reconciliation must be done before posting changes to another version.

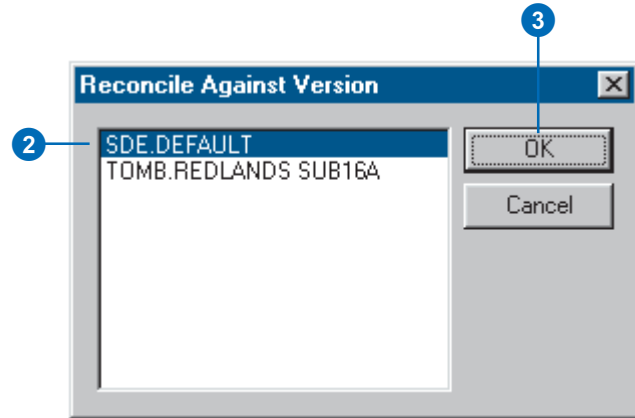
During reconciliation, conflicts may be discovered. Conflicts arise when the same feature is updated in each version or updated in one and deleted in the other.

When conflicts arise, an interactive conflict resolution dialog box will provide the tools necessary to resolve the conflicts. For each conflict, you can choose whether to replace the feature in your edit session with the conflict version, the version from your edit session, or the version as it existed at the beginning of your edit session.

Once you have successfully completed the reconciliation, you can post the version. The post operation synchronizes your edit session with the target version. They are then identical.

## Reconciling

1. Click the Reconcile button on the Versioning toolbar.
2. Click the target version.
3. Click OK.



---

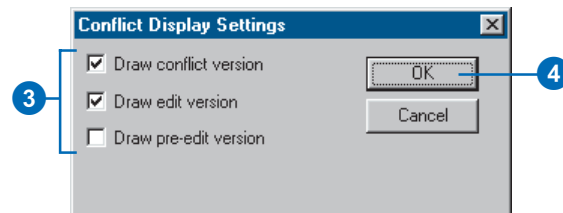
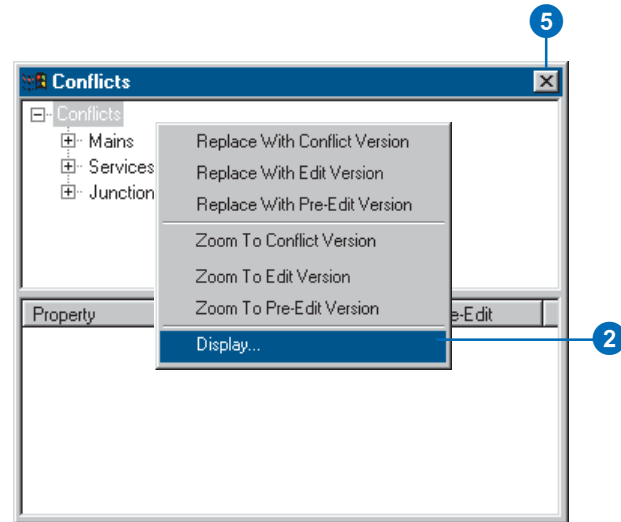
## Posting

1. Click the Post button on the Versioning toolbar.



## Displaying conflicts

1. Click the Conflicts button on the Versioning toolbar.
2. Right-click Conflicts and click Display.
3. Click a check box to display a conflict category.
4. Click OK.
5. Click the Close button to close the Conflicts dialog box.





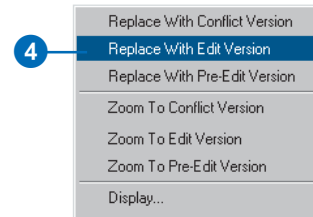
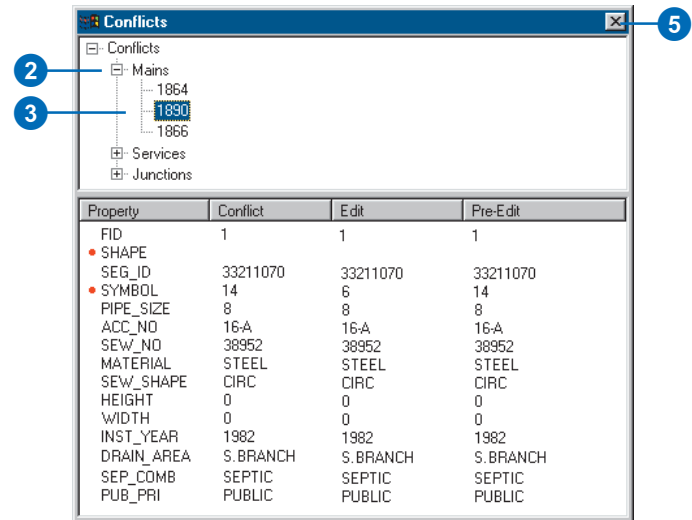
## Tip

### Topology errors

While editing a versioned geodatabase, the topology rules might have been violated. To validate these errors, use the validation techniques described in the 'Topology' chapter in this book or in the chapter 'Editing topology' in Editing in ArcMap.

## Resolving conflicts

1. Click the Conflicts button on the Versioning toolbar.
2. Click a feature class.
3. Click a feature and right-click to display the context menu.
4. Click the appropriate Replace With command to resolve the conflict.
5. Click the Close button to close the Conflicts dialog box.



# Versioning scenarios

The following scenarios show how an organization can implement its work flow process using a versioned database. These examples demonstrate several techniques available for performing long transactions in a multiuser environment. It is likely that organizations will, in some manner, use each of these techniques depending on the task.

## Scenario 1: Simple database modifications

**Task:** Multiple users are concurrently editing the database, performing common map sheet changes such as inserting new features, updating attributes, and removing out-of-date facilities.

**Solution:** Users can simply connect to the DEFAULT version (simultaneously), start editing, and save their changes when their work is complete. Users do not have to create new versions to modify the database. If another user has edited the DEFAULT version since the current user has started editing, the user saving is notified that the version has been changed and, therefore, the version will need to be saved again. Users may bypass this warning message by enabling autoreconciliation in the ArcMap Options dialog box. Also, if two users modify the same feature during their edit sessions, the second user to save encounters a conflict. The user then has to decide what the feature's correct representation is and save the edit session.

## Scenario 2: Transactions spanning multiple days

**Task:** Update the database to incorporate new and updated facilities in the field, which will likely require multiple edit sessions and a couple of days to complete.

**Solution:** A user creates and switches to a new version derived from the DEFAULT version. The user starts editing the new version and begins modifying features and saving as required. The user can resume the edit session, as appropriate, the following day or possibly the following week. When the changes

are complete and ready to be posted to the DEFAULT version, the user must first click the Reconcile button on the Versioning toolbar. If conflicts are detected, the user can resolve the differences and complete the transaction by clicking the Post button. The posting process applies all the changes in the user's version to the DEFAULT version. The user can then delete the version.

## Scenario 3: A work flow process

**Task:** Create individual versions for each step or stage of the work order and work flow process and post the work order to the database.

**Solution:** A user or supervisor creates a new version derived from the DEFAULT version. The user starts editing the new version and begins modifying features or creating a new design. When the user has completed the design or proposed modifications, the work order can be submitted to a supervisor for review. At this time, a new version can be created to ensure the preservation of the initial design. The new version can then be further modified or adjusted as required. Once the work order has been approved for construction, another version can be created. The purpose of this version is to reflect any changes that may occur while the work order is being constructed in the field. Finally, as the construction is completed and the new facilities are in service, the work order must be posted to the database. A user can then start editing the work order, perform a reconcile with the DEFAULT version, resolve any conflicts if necessary, and post.

The solution allows the organization to create new versions of the work order for each step of the project—the initial design or proposed version, a working or accepted version, and a version for the construction phase. Each version is preserved and available to review for historical purposes. The final step is to post the constructed version to the database. The project

completes a full circle from start to finish, creating individual versions at each step.

### **Scenario 4: Restricting permissions to the database**

**Task:** The organization’s supervisor has restricted write access to the DEFAULT version, requiring managerial review of each user’s edits prior to posting the changes to the database.

**Solution:** To restrict write permissions to the database (the DEFAULT version), the ArcSDE administrative user can set the permission of the DEFAULT version to “protected” using the version manager. This allows users to continue to view the DEFAULT version but does not allow users to start editing the version. Therefore, users will need to create new versions for editing the database, similar to Scenario 2. When a user has completed and saved the edit session, the ArcSDE administrator can reconcile the version with the DEFAULT version. To accomplish this task, the manager who connects to the database as the ArcSDE administrator starts editing the user’s version and clicks the Reconcile button. The process will merge all the changes in the user’s version and the DEFAULT version. If conflicts are detected, the manager can resolve the conflicts and save the edit session. Once the edits are acceptable to the manager, the version is ready to be posted to the DEFAULT version. The ArcSDE administrative user can then start editing the version, perform a reconcile, and post the version. The user’s version can then be deleted.

### **Scenario 5: Compressing the database**

**Task:** The geodatabase has been edited for an extended time, and the number of database states and rows in each feature class’s delta tables has significantly increased. How do you improve performance?

**Solution:** The Compress command will remove all database states that are no longer referenced by a version and move all the rows in the delta tables, which are common to all versions, to the base table. To achieve the maximum benefit when running the Compress command, you can optionally first reconcile, post, and delete each version with the DEFAULT version. Sometimes this may not be a reasonable option based on your organization’s work flow. At minimum, to improve performance, simply reconcile each version with the DEFAULT version and save, then perform the compress. This will ensure that all the edits in the DEFAULT version will be compressed from the delta tables to the business table. Remember, the Compress command can still be executed without first reconciling, posting, and deleting each version, but the benefits may not be as noticeable.



# Disconnected editing

## IN THIS CHAPTER

- **Disconnected editing**
- **Checking out data from a geodatabase**
- **Customizing a check-out**
- **Checking in data to a geodatabase**
- **Managing check-outs**

Exchanging and maintaining data between geographically dispersed users has become an integral part of the work flow process. Some of these remote users may be connected via a wide area network (WAN) or local area network (LAN) to a central office infrastructure; others will be more mobile, requiring only to download and upload data to and from central office databases on an infrequent basis.

These organizational requirements are supported in the geodatabase.

ArcInfo or ArcEditor is required to check data in and out, and to edit an ArcSDE check-out geodatabase. You can edit a personal check-out geodatabase with ArcView.

# Disconnected editing

Many organizations have users who work at remote locations, such as a regional suboffice, but who still operate within the same infrastructure as the central office via a WAN or LAN. For those remote users, the overhead of maintaining an active connection to a remote database server may be a consideration. One solution would be to transfer data from the central database to the remote location, disconnect from that central database, and continue to work offline. This avoids the overhead of the remote database connection, but allows users to maintain connections to other network-based services such as e-mail.

More mobile users, such as a field survey crew, require similar functionality. They need to disconnect completely from an organization's infrastructure, often for a prolonged period of time. When preparing for a particular work order or project, the relevant data would be transferred to a portable device, such as a laptop. This device would then be disconnected from the network, enabling the user to operate independently.

Remote or mobile users may continue to work with and modify the data even though they are disconnected from the network. When a connection to the network is reestablished, any changes made to the data will be transferred to, and integrated with, data maintained in the central database.

Geodatabase disconnected editing allows organizations to disseminate their spatial data to other departments, associated agencies, or mobile workers and maintain the integrity and currency of that data.

## How does disconnected editing work?

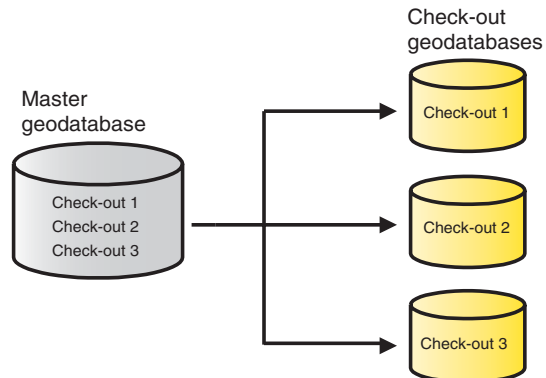
Disconnected editing is supported by two principal mechanisms—checking out and checking in data.

*Checking out* data from a geodatabase involves taking a copy of some data from one geodatabase, *the master geodatabase*, and transferring it to another geodatabase, the *check-out geodatabase*. This copy is undertaken within a framework that allows the data to be reintegrated with the master geodatabase at a later time.

*Checking in* the data from a check-out geodatabase involves establishing whether any changes have been made to the data in the check-out geodatabase, and transferring those changes back to the master geodatabase.

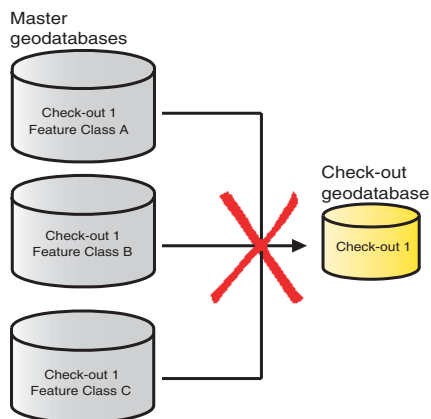
## The master geodatabase

The master geodatabase must be an ArcSDE geodatabase. Data cannot be checked out from a personal geodatabase. Data in a master geodatabase must be registered as versioned before it can be checked out. A master geodatabase may also host multiple check-outs to multiple check-out geodatabases.



*A master geodatabase hosting multiple check-outs*

The creation of each check-out is performed on a per geodatabase connection basis, which means data from only one ArcSDE connection may be checked out to one check-out geodatabase; it is not possible to check out data from multiple master geodatabases to one check-out geodatabase at the same time.



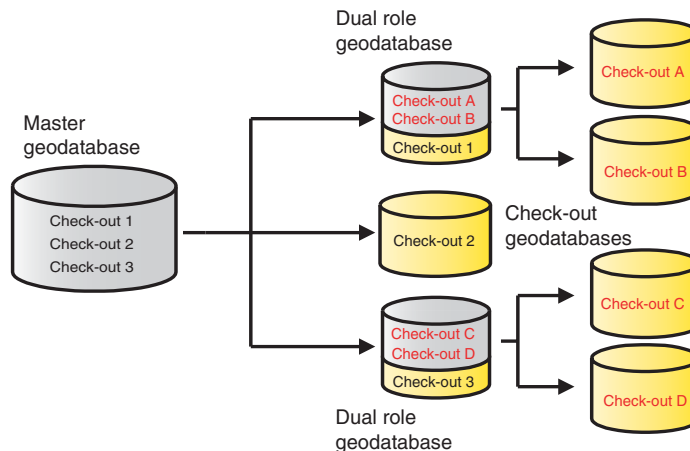
*Checking out data from multiple geodatabases to one check-out geodatabase is not supported.*

## The check-out geodatabase

The check-out geodatabase may be either a personal geodatabase or another ArcSDE geodatabase. ArcSDE, on all the currently supported DBMS platforms, is supported as a check-out geodatabase.

Once a geodatabase contains data checked out from a master geodatabase, it cannot be used again for checking out until the changes have been checked in. Each *check-out* geodatabase will support only one check-out from a master geodatabase.

However, any ArcSDE geodatabase may also adopt a dual role by acting as both a check-out and a master geodatabase from which further check-outs may be made. This database may contain one check-out from another master geodatabase and could also host multiple check-outs to other check-out geodatabases.



*Dual role master and check-out geodatabase*



## Preparing a check-out

The first step in preparing a check-out is to identify what data is required. It is important that the data model and nature of edits to be made in the check-out geodatabase are understood and that the appropriate amount of data is included in the check-out. As a general rule, check out more data than is required for editing purposes. Edits made to features close to, or on the boundary of, the check-out area may affect other features not included in the check-out. Including more data in the check-out than you need will avoid potential problems or unexpected behavior when the data is checked in.

ArcMap provides the framework for checking out data in ArcGIS. An ArcMap document should be prepared to support each check-out operation. For example, if you wish to limit the extent of data to check out, zoom in to the area of interest before you start the check-out process. If you wish to check out certain features only, apply the selection or define the query on the relevant layer or table before you start.

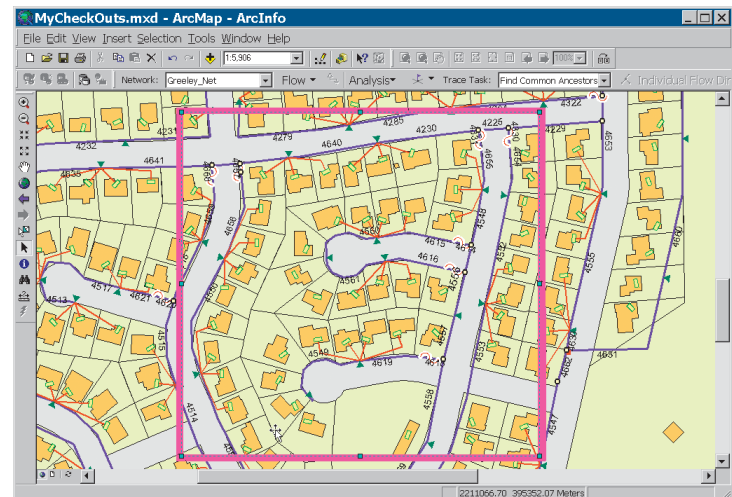
Once the check-out is initiated, the following default behavior will determine what data is checked out:

- All layers and tables present in the ArcMap document for the selected ArcSDE geodatabase will be included in the check-out.
- The following data filters will be applied:
  - The spatial extent of the data to check out will be determined by the current view extent of the ArcMap document or the boundary of a currently selected graphic.
  - Any selections that already exist on those layers or tables will be maintained.
  - Any definition queries already applied to those layers or tables will be maintained.Note: The intersection of these data filters will determine what data will be checked out.

- To maintain data integrity, any related objects will automatically be included in the check-out, regardless of whether or not they are currently present in the ArcMap document.
- The list of data to check out will automatically be expanded to include dependent datasets. For example, all feature classes in a geometric network, topology, or feature dataset will be included if just one feature class in the network, topology, or feature dataset is selected for check-out.

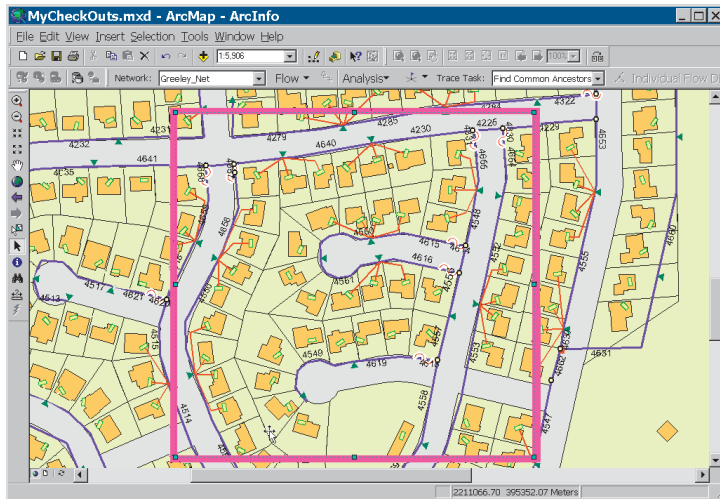
The following electric utilities' maintenance work orders will help illustrate some of the default behavior for checking out data.

A maintenance crew is preparing to inspect some of the electric utilities in a residential area. In order to do some field editing, they need to check out that part of the electric network that covers this residential area. To start the check-out process, the spatial extent of the inspection area is identified using a spatial filter (in this case, the extent is determined by a selected graphic).



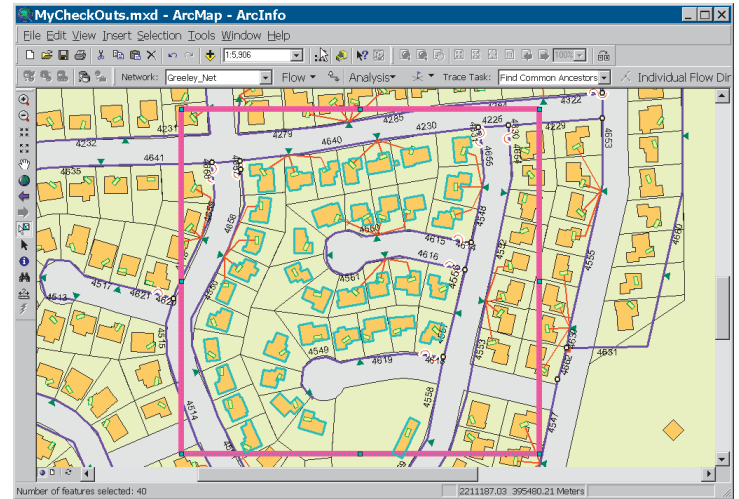


The crew is to concentrate on cables that have been insulated with a particular material. To identify these cables, a query is applied to the relevant dataset.



*The results of a definition query—cables (in red) insulated with a particular material*

Finally, as each maintenance crew can expect to visit only a certain number of properties in a day, the homes in one residential block are identified by a selection based on property numbers.

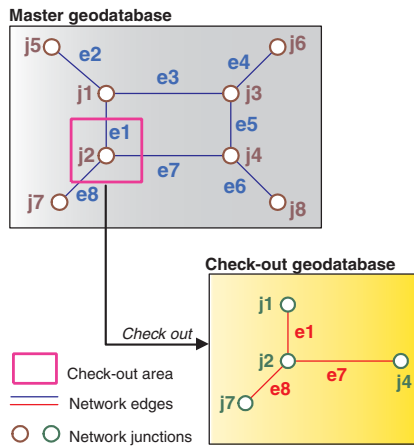


*Attribute selection to identify certain properties*



## Checking out geometric networks

All the feature classes that participate in a geometric network are checked out together to maintain the connectivity of the network in the check-out geodatabase. Individual network classes cannot be excluded from the check-out. Where the check-out boundary intersects a network edge, the next junction in the network will automatically be included in the check-out.



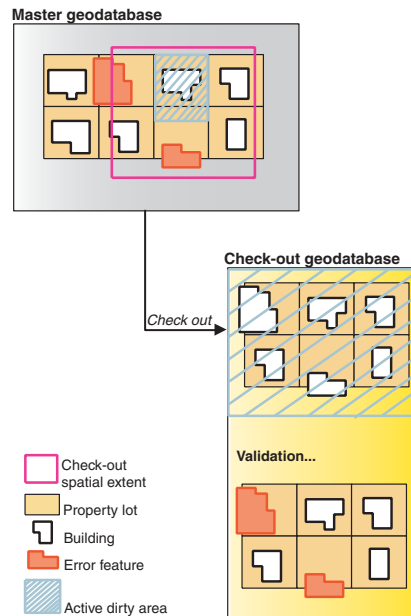
### Checking out network features

In this example, all the network elements that intersect with the check-out extent are checked out from the master geodatabase—that is, junction *j2* and edges *e1*, *e7*, and *e8*. Junctions *j1*, *j7*, and *j4* will also automatically be included in the check-out to maintain connectivity in the check-out geodatabase. Again, it is important that the extent of the check-out is large enough to accommodate any modifications to features that may be close to or on the boundary of the check-out area. The implications of not including sufficient network features in the check-out will be discussed later in this chapter.

## Checking out topologies

As with geometric networks, all the feature classes that participate in a topology are checked out together; individual topology feature classes cannot be excluded from a check-out.

When topology feature classes are checked out, the whole extent of the topology in the check-out geodatabase will be marked as dirty. For more information on dirty areas and topologies in general, see the ‘Topology’ chapter in this book. To discover existing errors, the topology must first be validated. The topology will behave as it would in the master geodatabase: edits create dirty areas, and validation creates or deletes errors.

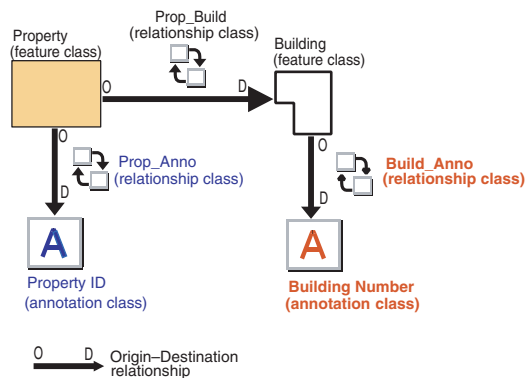


*Checking out topological data. In this example, one topology rule is in effect—buildings must be contained by the parcels.*

## Checking out related data

To maintain data integrity and avoid unexpected behavior, related objects should always be checked out. The check-out process will automatically include related objects that may or may not be present in the check-out ArcMap document. For every geodatabase object that participates as an origin in a relationship class, the check-out will include directly related objects, one step away in the relationship, in the associated destination object. This destination object may itself act as an origin in some other relationship with a different geodatabase object; for performance reasons, the check-out process will not recursively include other objects that may be indirectly related to the related objects already included in the check-out.

The following examples illustrate the default check-out behavior with respect to related objects. The data model used in these examples is a simple origin–destination relationship between properties, buildings, and their related annotation.

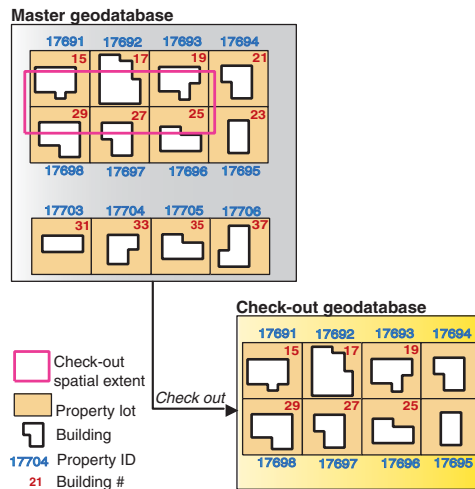


Simple data model for property and buildings

In this model, each property has related annotation, represented by the relationship class Prop\_Anno. The annotation’s objects

are one step away in the relationship with the properties. Each property also has a building—the relationship here is represented by Prop\_Build. The buildings are, again, one step away from the properties in the relationship. Each building, in turn, has some related annotation, represented by Build\_Anno. This building annotation is indirectly related to the properties—it is two steps away from the property in the relationship.

Some properties are identified for check-out using a spatial extent filter.

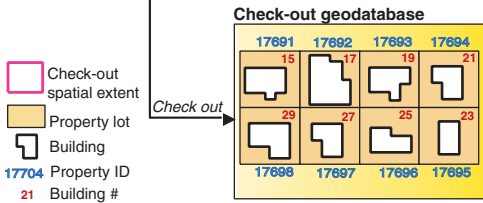
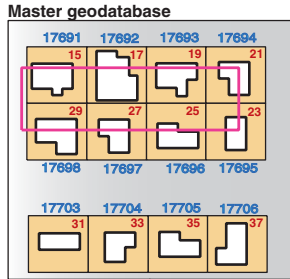


The features and related annotation checked out by default

Although the annotation related to the properties falls outside the spatial extent of the check-out, they are automatically included in the check-out—all features directly related to, or one step away from, the features in the check-out are included by default.

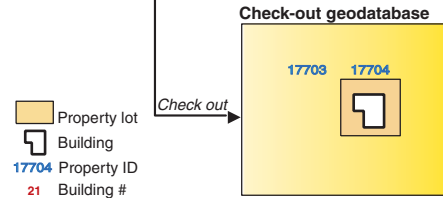
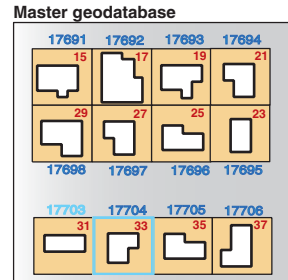
The check-out also includes two building features related to properties 17694 and 17695. Applying the one-step related objects

rule, these features are included in the check-out, but the related building annotation is not. To avoid this, include more objects in the check-out than required for editing purposes. In this example, extending the spatial extent of the check-out will ensure that all related features and annotation are included.

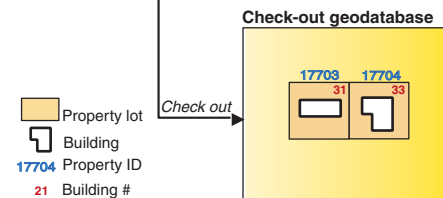
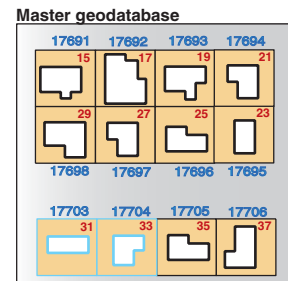


*Extend the spatial extent of the check-out to include all related annotation*

In the next example, two properties and related buildings are required in the check-out. The property 17704 is identified through attribute selection, along with the annotation for property 17703. In this case, while property 17704, its related annotation, and the related building feature are checked out, the annotation related to the building is excluded. This annotation feature is two steps away in the relationship to the property in the check-out. Because it was selected, the annotation relating to property 17703 is included in the check-out but not the property itself. When checking out data, relationships are traversed in one direction only, from the origin to the destination. As the annotation for the property 17703 represents the destination in relationship to the property, no related property is included.

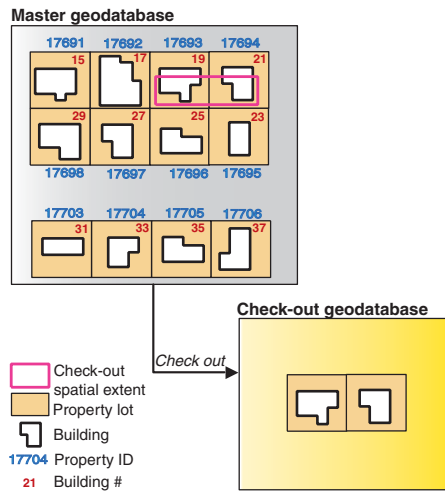


To include the features and the related annotation, redefine the selection to include both the lot and the related building features.



*Including all related features in the check-out*

In the previous examples, each check-out was made using the default behavior of including related objects. It is possible to override this behavior either at the global or local level to customize each check-out. At the global level, each check-out may be configured to not include any related objects associated with features identified for check-out.



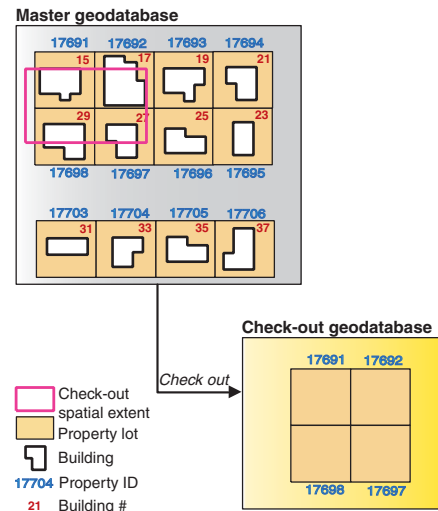
*No related objects were required in this check-out—no annotation features are included.*

Alternatively, the choice to include or exclude specific classes may also be controlled at an individual class level. The following general rules will apply in this case:

- If either an origin class—for example, lots—or destination class—for example, buildings—is excluded from a check-out, the associated relationship class and, therefore, the related objects will not be included.

- If a relationship class is excluded from a check-out, the one-step related objects rule does not apply, and no related objects will be included in the check-out.
- Nonattributed relationship classes cannot be checked out without the corresponding origin and destination classes; attributed relationship classes may be checked out on their own.

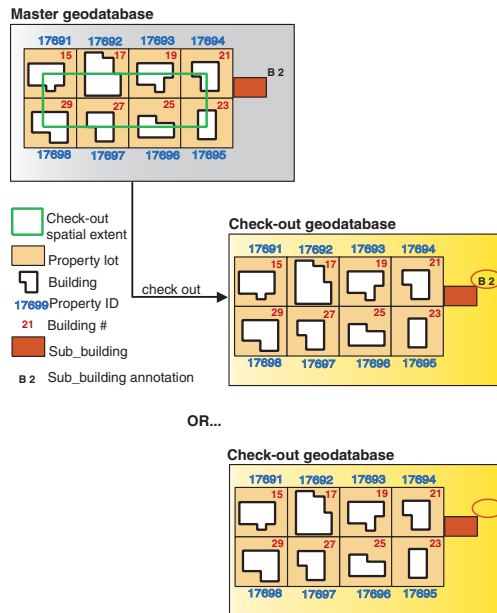
In the next example, although the spatial extent of the check-out includes four properties (17691, 17692, 17698, 17697) that have related buildings, all buildings have been explicitly excluded from the check-out. As the global default behavior to always include related objects is still in effect for the other feature classes, the property annotation will again be included in the check-out.



*The features checked out to the check-out geodatabase after buildings have been excluded*

In some cases it appears that you are getting more in the check-out than you had anticipated given the model for checking out related data. The order in which the data to be checked out is processed can determine which related data will be included in the check-out. In this next example, one of the properties, property lot 21 (an origin) also has an additional related sub-building feature. This sub-building feature itself is both a destination and an origin as it also has some related annotation (a destination). This related sub-building is included in the list of features to check out by virtue of being a destination object in relation to the property lot feature identified by the check-out extent.

If the property feature class was first in the list of datasets to check out, the check-out process would evaluate that feature class first, see what was related to the features selected, evaluate the related feature class next, and see what was related to that dataset. In this case, the property, its sub-building, and the sub-building's annotation would be checked out. If the sub-building was processed first, as no rows in that feature class had been explicitly selected for checking out (i.e., it's outside the check-out area), no data related to that sub-building (in this case, annotation) would be included in the check-out. With default behavior automatically including related data, you are always guaranteed to get the data related to the features you have identified for check-out via your check-out extent, selections, or definition queries, but in some cases, depending on the order of processing, you may get additional data related to those related objects as well.

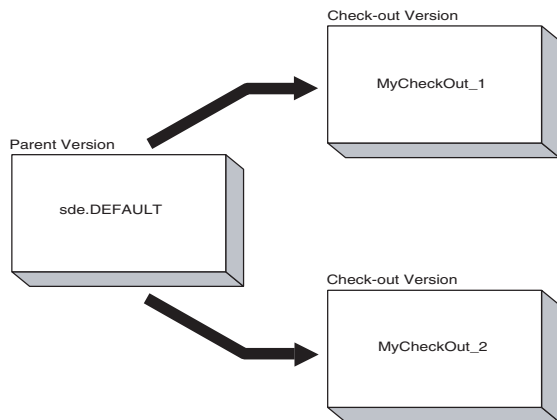


*The features checked out to the check-out geodatabase after buildings have been excluded*

## The check-out process

Having identified the data to check out, the check-out process itself involves a number of steps. The selected data, along with any related objects, is copied to the designated check-out geodatabase. A new version of the data, the *master check-out version*, will be created in the master geodatabase as a child of the version to which you are currently connected. This will be created as a public version accessible to all users in the master geodatabase. This version may be set to private, or hidden from general access, if required.

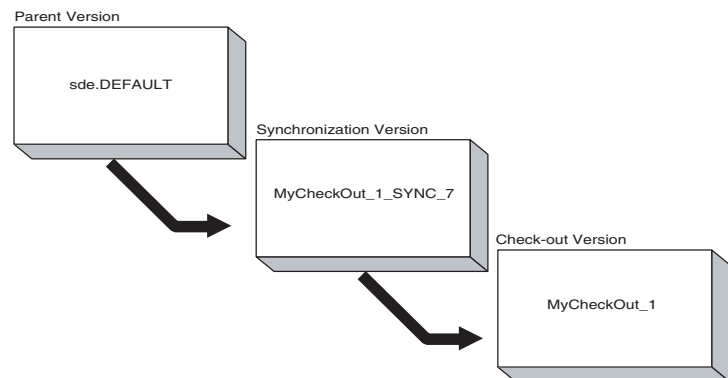
For example, if you are connected to the DEFAULT version, a new version with the same name as the check-out will be created as a child of DEFAULT. This version will receive the changes from a check-out geodatabase when the data is checked in.



*The check-out version tree in a master geodatabase*

If the check-out geodatabase is an ArcSDE geodatabase, then two new versions are created in the check-out geodatabase. The first version, the *synchronization version*, will be created as a child of the DEFAULT version. This synchronization version will reflect the state of the data at the time the check-out was created and should not be edited.

The second version, the *check-out version*, will be created as a child of the synchronization version. Only the edits made to this version can be checked back into the master geodatabase. Once data has been checked out to an ArcSDE geodatabase, it is automatically registered as versioned.



*The check-out version tree in a check-out ArcSDE geodatabase*

If the check-out geodatabase is a personal geodatabase that does not support versioning, changes to the data are maintained in a separate DBMS table.

The check-out process concludes by adding information to both the master and check-out geodatabases that describes each check-out. This information includes the name of the user who created the check-out, the data that has been checked out, the time the check-out was created, and the name of the new check-out version in the master geodatabase.



Details about the source of the data, the master geodatabase, are also recorded. This involves storing partial ArcSDE connection information, which will be used during the check-in process. Usernames and passwords are not stored in the check-out geodatabase.

## Check-out restrictions

There are some important restrictions to remember when checking out data. First, the version created in the master geodatabase should not be edited while the data it represents is still checked out. As this check-out version in the master geodatabase is public and, therefore, editable when created, any changes made to this version may be overwritten during the check-in process, as no conflict reconciliation is undertaken at this point.

Second, it is not possible to append data to or refresh an existing check-out. You cannot check out an updated version of the same data to a check-out geodatabase while it contains a check-out. You must first check in the original check-out, then make a second check-out. Once data has been checked back in, the check-out geodatabase no longer participates in any check-out/check-in relationship with a master geodatabase. However, it is possible to reuse a geodatabase for a new check-out. You can reuse the existing schema in a geodatabase to form a template for a second check-out. If the data to be checked out is complex, reusing the schema in a geodatabase will afford some performance improvements. When creating the new check-out, if the geodatabase to be reused contains an older copy of the data, this will be deleted in preparation for receiving the latest copy of the data from the master geodatabase. If required, additional objects may be included to augment the check-out.

Finally, the check-out model does not support schema modifications to data that have been checked out. If the schema is altered in any way, either in the master or check-out

geodatabase, for example, by adding a field to a table or feature class, the check-out is rendered invalid and any attempts to check in the modified schema and data will fail. If a new table has been created in the check-out geodatabase, it will be ignored when the rest of the data is checked in.

## Managing check-outs

Once a check-out has been created, any subsequent modifications to the properties of the check-out are made *asynchronously*. This means that changes made to the properties of a check-out in a master geodatabase are independent of the associated check-out in the check-out geodatabase and vice versa. For example, if a check-out is unregistered in the master geodatabase, this will not unregister the associated check-out in the check-out geodatabase. Similarly, if a check-out is renamed in the master geodatabase, that operation does not rename the check-out in the check-out geodatabase. The name of a check-out in a master geodatabase does not have to match the name of the check-out in the corresponding check-out geodatabase. If a check-out is renamed in the master but not the check-out geodatabase, the data can still be checked in.

## Checking in data to a master geodatabase

Once a connection to the master geodatabase has been established, such as reconnecting a laptop to a network, data may be checked back in to the master geodatabase.

Two check-in models are supported; the *pull* model, in which the check-in is initiated from the master geodatabase, and the *push* model, in which the check-in is initiated from the check-out geodatabase. The choice of which approach to take is a matter of user preference. A data administrator checking in many check-outs to one master geodatabase might use the pull check-in model; open one connection to the master geodatabase and locate each check-out geodatabase in turn. Alternately, once an editor has completed editing the data in a check-out geodatabase, the push model would be the most convenient option for checking in the changes; checking out, editing, and checking in can all be accomplished within one ArcMap session without having to reopen a separate connection to the master geodatabase.

The master geodatabase does not record the location and type of the check-out geodatabase, so for a *pull check-in*, the check-out geodatabase has to be located as part of the check-in process. Changes are then pulled in from the check-out geodatabase to the master geodatabase.

For a *push check-in* from the check-out geodatabase to the master geodatabase, most of the connection information required to check in the data is already stored in the check-out geodatabase. All that is required is a username and password to complete the connection to the master geodatabase. Once this information has been supplied, the check-out geodatabase pushes the changes to the master geodatabase.

For both check-in models, any user who has write permissions to the data in the master geodatabase may check in the data.

However, only the user who created the check-out, or the SDE user, can check data back in to a master geodatabase with reconcile and post as part of the same operation. Once the reconcile and post have successfully been completed, the check-out version in the master geodatabase is deleted—an operation that requires version owner or SDE permissions.

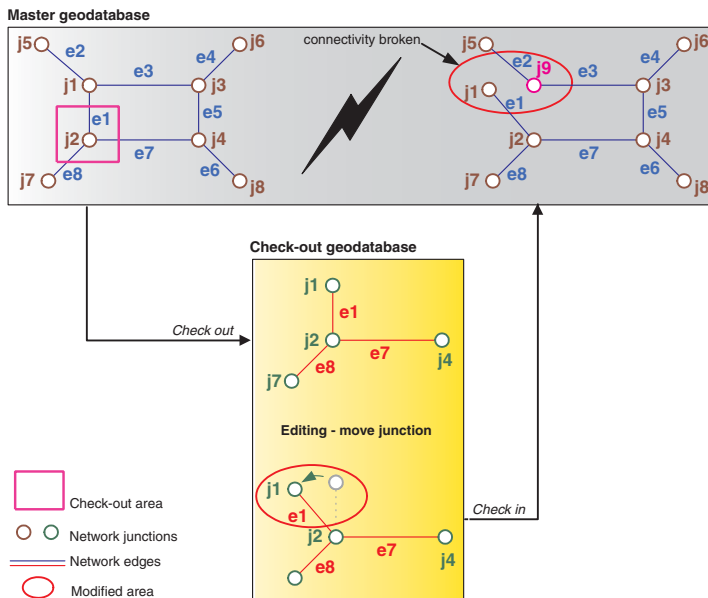
## Delta databases and XML files

If you connect to the master geodatabase via a WAN with less bandwidth for data transfer or unreliable connections, the check-in can take a long time or get interrupted. Instead of checking in directly from the check-out geodatabase, you can export the changes only from the check-out geodatabase to a delta database or delta XML file. As delta databases and XML files are smaller than the original check-out geodatabase, they can easily be transferred independently of the check-out/check-in process to a machine on the same LAN as the server. The changes can then be checked in from that machine.

The changes in the delta database or XML file may be checked in as a pull check-in from the master geodatabase. As with a check-in from a check-out geodatabase, when the check-in from the delta database or XML file succeeds, the check-out in the master geodatabase will be unregistered. Checking in changes from a delta database or XML file does not automatically unregister the check-out in the associated check-out geodatabase; this must be done manually.

## Checking in geometric networks

Checking in geometric networks involves transferring any changes to the network data to the master geodatabase and rebuilding connectivity in any part of the network that has been modified. Areas of the network that are unaffected by changes checked in are not rebuilt. If a junction has been moved in the check-out geodatabase, and that move results in two edges without a junction, to maintain the connectivity of the network in the master geodatabase a new junction will automatically be added at the old location.

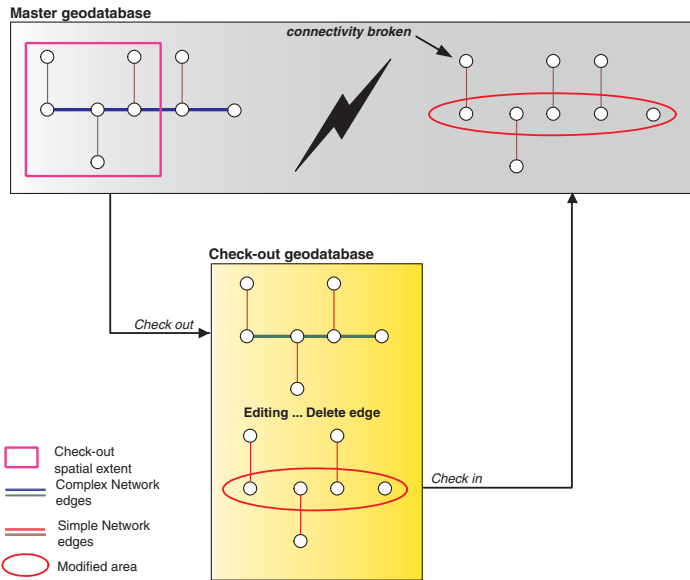


Checking in modified network junctions

In the example to the left, junction  $j1$  is moved in the check-out geodatabase. As junction  $j1$  was checked out as a result of edge  $e1$  being checked out to maintain valid connectivity, it is no longer connected to its other edges,  $e2$  and  $e3$ , in the check-out geodatabase. As these edges were not connected to junction  $j1$  at the time it was moved, network connectivity is broken when the data is checked in to the master geodatabase.

Once the changes are checked in, the connectivity in the modified section of the network is rebuilt. This will result in a new junction,  $j9$ , being created automatically to maintain the correct precheck-out network connectivity between edges  $e2$  and  $e3$ .

In the next example, a collection of edges and junctions in a network is checked out. In the check-out geodatabase, a complex edge that connected numerous simple edges to the rest of the network is deleted. The data is then checked back in to the master geodatabase. In this case, connectivity is again broken, but editing the data in the check-out resulted in the same connectivity as if the edge had been deleted in the master geodatabase.



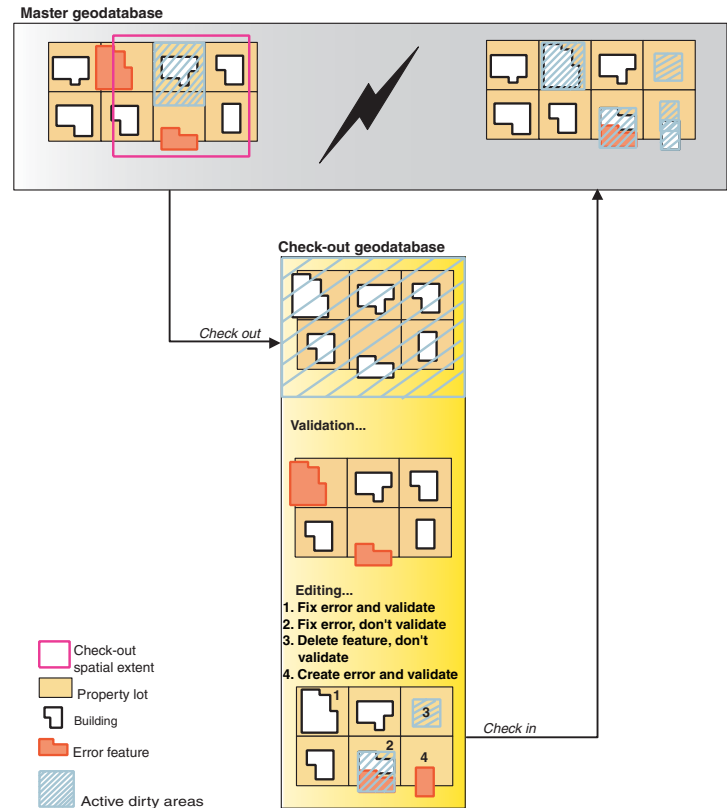
### Checking in network features where one edge has been deleted

If the modifications in the check-out geodatabase result in network connectivity being broken in the master geodatabase, the responsibility for correcting this remains with the user or data administrator.

### Checking in topologies

After topologies have been checked in, all the changes (inserts, updates, and deletes) are flagged as dirty areas that require revalidation.

If you have included reconcile and post as part of the check-in, the standard topology and version reconciliation rules will apply. For more information, see the 'Topology' chapter in this book.



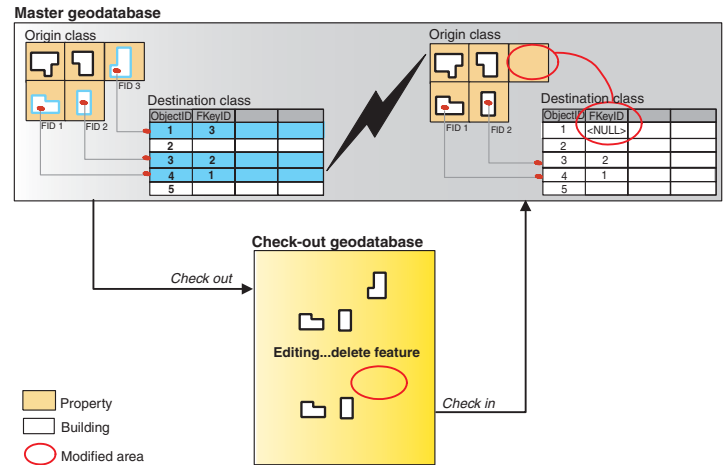
### Checking in topological data

## Checking in related data

When checking in data, there are some important considerations to remember if related data was not included in the check-out. The check-in process is a full geodatabase check-in, which means the process assumes that edits made in a check-out geodatabase were made with a geodatabase-aware editor, such as ArcMap, or any custom editing tool written using the geodatabase application programming interface. The check-in process also assumes that required related records were included in the check-out. For example, if a new feature was created in the check-out geodatabase, the foreign keys of a relationship involving a new feature are automatically updated on check-in.

However, if all the related objects were not included in the check-out, the check-in process will ensure that relationships pertaining to objects deleted in the check-out geodatabase are also deleted during check-in. This may result in the nulling of foreign keys in classes in the master geodatabase that were excluded from the check-out.

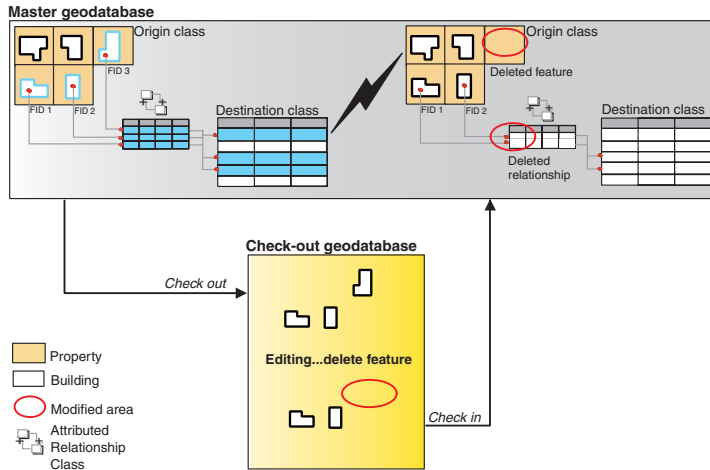
In this first example, some features in an origin class, buildings, were selected for checking out. The buildings are related, in a nonattributed relationship, to attribute records in a table that were excluded from the check-out. While the data was checked out, a building was deleted. On check-in, to void the relationship with the feature that was deleted, the corresponding entry in the foreign key field in the related destination class, the table, is set to NULL.



*Checking in nonattributed relationships when related objects were excluded from the check-out*

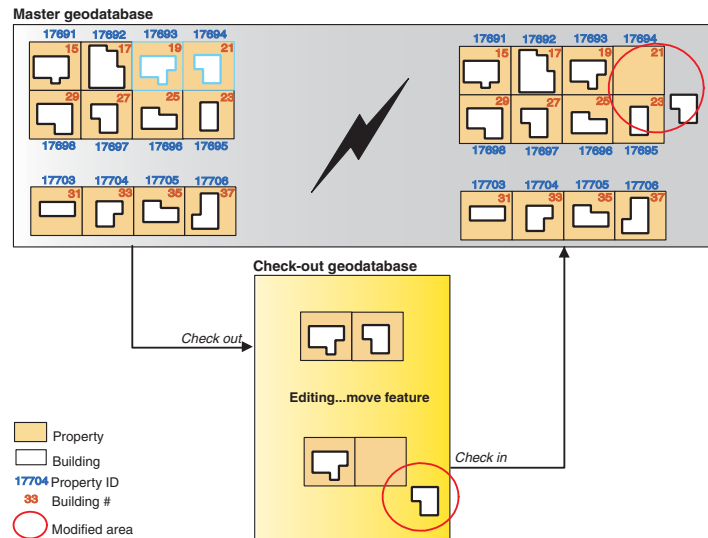
This check-in behavior may also result in the deletion of rows representing relationships in an attributed relationship class table. In the next example, the relationship between the origin feature class and the destination class table is attributed, which means the relationship itself has an associated table. Both the relationship and the destination class were excluded from the check-out. Edits made to the origin feature class resulted in one feature being deleted. On check-in, the row in the attributed relationship class table representing this feature's relationship to an object in the destination class is removed.

Only relationships are deleted on check-in; related objects themselves are never deleted.



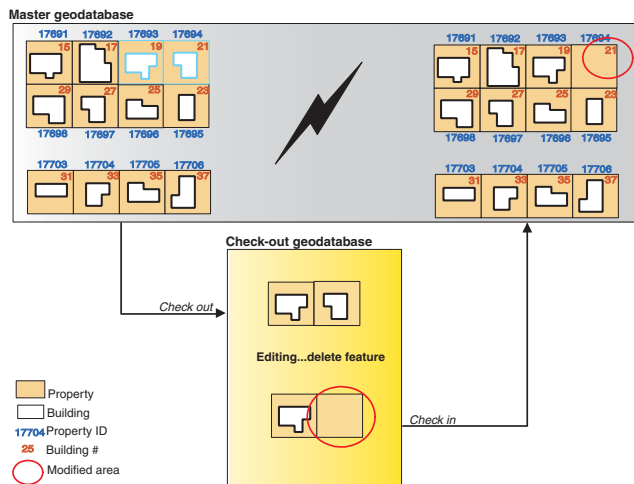
Managing attributed relationships on check-in when related objects are excluded in the check-out

In the next example, some features are selected for checking out without their related feature-linked annotation. The position of one feature is altered in the check-out geodatabase. On check-in, the annotation that relates to the modified feature in the master geodatabase is unaltered and remains in its original position.



Checking in modified features where the related annotation was excluded from the check-out

Similarly, not including related objects in the check-out and deleting a feature in the check-out geodatabase would result in orphaned annotation after the data has been checked back in.



*Checking in deleted features where the related annotation was excluded from the check-out*

## The check-in process

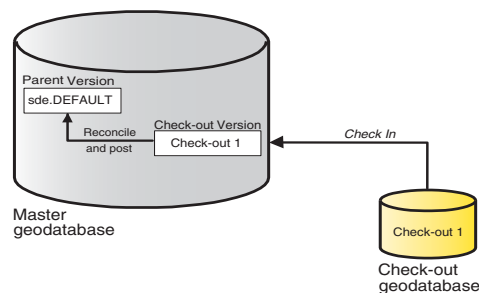
As with checking out data, the check-in process involves a number of automated steps. This process begins by establishing what changes have been made to the data in the check-out geodatabase; only these changes will be checked back in to the master geodatabase. If the check-out geodatabase is an ArcSDE geodatabase, a comparison is made between the edited check-out version and the static synchronization version to identify what is different. If the check-out geodatabase is a personal geodatabase, the changes are recorded in a separate DBMS table.

The changes are then transferred directly to the check-out version in the master geodatabase; there is no version

reconciliation at this point with the check-out version. If the check-out version has been modified since the data was checked out, these changes may be overwritten.

Once the data has been checked back in to the master geodatabase, all associated check-out information will be removed, such as the list of datasets checked out, from both the master and the check-out geodatabases. Any versions created in an ArcSDE check-out geodatabase will also be removed. However, the copy of data that was checked out is not removed from the check-out geodatabase. The responsibility for deleting any residual copies of the data once check-in has completed remains with the user or data administrator.

The check-in process may optionally involve an additional step of reconciling and posting the changes from the check-out version to its parent version in the master geodatabase.



*The check-in with reconcile process*

For further information on the reconcile and post processes, see the chapter 'Working with a versioned geodatabase'. If any conflicts are detected at this stage, they must be resolved using standard geodatabase version reconciliation tools. If no conflicts were detected, the check-out version in the master geodatabase is removed.

## Working with ArcGIS 8.3 and 9 geodatabases

Geodatabases built using previous versions of ArcGIS do not support some of the newer functions of ArcGIS. Consequently, if your organization has both ArcGIS 8.3 and 9 geodatabases, consider the following:

- You can check out and check in between geodatabases of the same version. With ArcGIS 9 for example, you can check out and check in between an 8.3 master geodatabase and an 8.3 check-out geodatabase.
- You can check out and check in between an 8.3 master geodatabase and a 9 check-out geodatabase. When editing the check-out geodatabase, be careful not to introduce edits that are not supported in 8.3. For example, annotation classes were introduced in 9 and, therefore, are not supported in 8.3.
- You cannot check out from a 9 geodatabase to an 8.3 geodatabase, since 9 geodatabases support data types that 8.3 geodatabases do not. Similarly, you cannot check in from an 8.3 geodatabase to a 9 geodatabase.
- If you check out to an 8.3 geodatabase and upgrade the master geodatabase to 9, you will need to upgrade the check-out geodatabase before checking in. If checking in from a delta file, you will need to upgrade the check-out geodatabase and reexport the delta file.



# Checking out data from a geodatabase

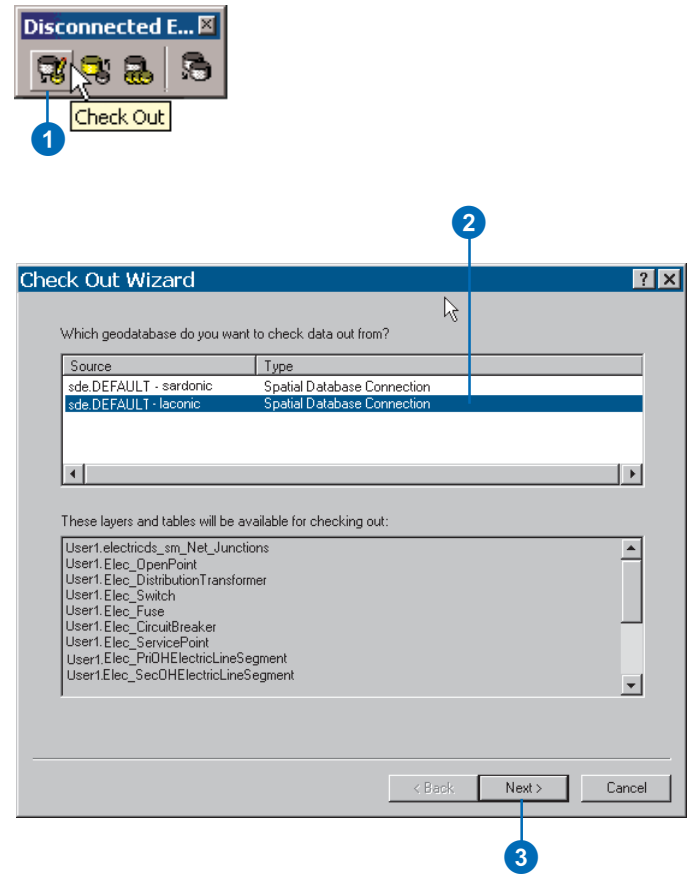
You can check out data from a geodatabase using the check-out wizard in ArcMap. The wizard supports a range of default and advanced check-out options.

The default behavior for each check-out will be:

- To include all the data, visible and invisible, in the active document for the chosen geodatabase.
- To maintain any selections and definition queries applied to that data.
- To restrict the spatial extent of the check-out to the current view extent of the document or the boundary of a selected graphic.
- To include any directly related objects.
- To expand the list of layers and tables to check out to include dependent datasets—for example, all feature classes in a geometric network will be included if just one feature class in the geometric network is selected. ►

## Creating a check-out using the check-out wizard

1. Click the Check Out command on the Disconnected Editing toolbar to activate the check-out wizard in ArcMap.
2. You will be prompted to choose which ArcSDE geodatabase you want to work with if the current ArcMap document includes data from more than one geodatabase. You can only check out data from one ArcSDE geodatabase at a time.
3. Click Next. ►



- If there are several layers representing one feature class in the document, only the top layer in the table of contents (TOC) is used in the check-out.

If you just want to add new features or create a template for future check-outs, use the Schema Only option. This will create the tables for your data in the check-out geodatabase, but will not copy any data.

Each check-out name must be unique to the user creating it. User1 and user2 could create a check-out called MyCheckOut, but neither could create multiple check-outs with that name. With each check-out a new version is created with the name of the check-out. The combination of user name and check-out name must be unique when creating versions.

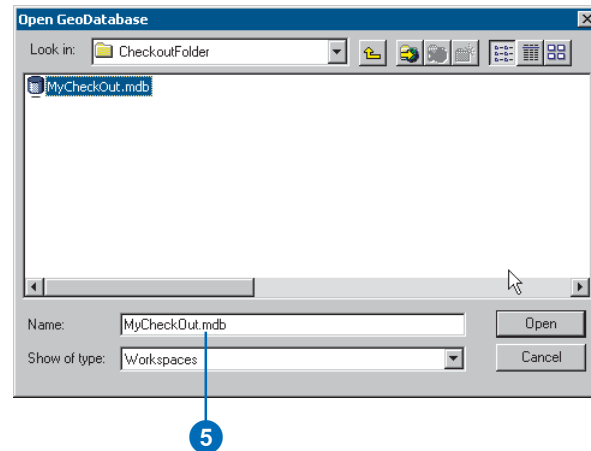
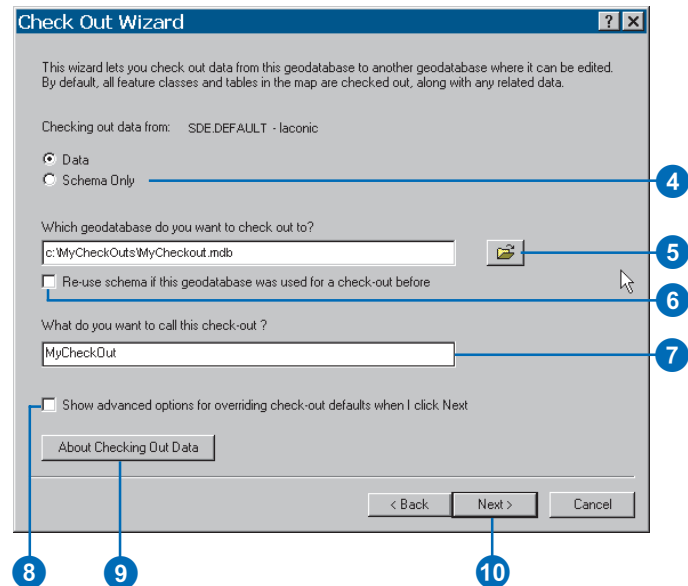
The reuse schema option allows you to reuse a geodatabase that contains the schema of the data you want to check out. This will reduce the amount of time required to check the data out.

## Tip

### Edit sessions

You may also check out data in an edit session. The check-out will represent the current state of the map cache. Creating the check-out will not save pending edits to the master geodatabase.

4. Click Schema Only if a *schema-only check-out* is required. The first panel of options has one default setting: to check out data from the current ArcMap document.
5. Enter the name of, or click the Browse button to navigate to, a personal geodatabase or ArcSDE connection to check data out to. If the personal geodatabase does not already exist, it will be created.
6. Check Re-use schema if you wish to reuse schema in an existing geodatabase.
7. A default name for the new check-out is provided. The name of each check-out must be unique to the user creating the check-out.
8. Check Show advanced options if you wish to override the check-out defaults.
9. Click About Checking Out Data to learn more about checking out data.
10. Click Next. If you are not making any change to the default options, skip to step 16. ▶



# Customizing a check-out

The advanced check-out options allow you to customize the check-out. The advanced panel will initially reflect the current check-out defaults and the expanded list of all the datasets that will be included in this check-out. Any data that is not versioned or for which you do not have read-write permissions will be automatically excluded from the list.

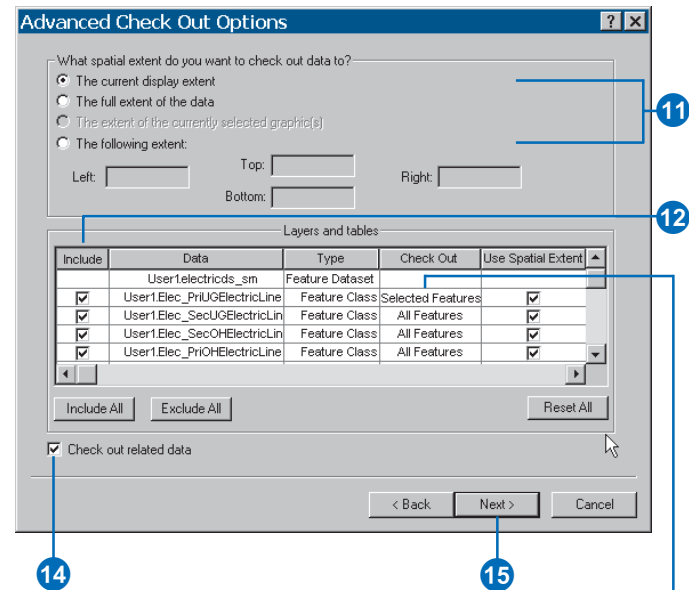
The extent of the check-out area may be determined by one of the following:

- The current view extent (the default)
- The full extent of the data
- The boundary of a currently selected graphic
- User-defined coordinates

If a schema-only check-out was selected previously, this top section will be unavailable.

The options in the grid determine how many records from each layer or table will be checked out. In addition to excluding layers and tables from the list of data to check out, you may also override the defaults for individual layers and tables. For example, if a selection set exists for a ►

11. Click the appropriate extent option to modify the spatial extent of the check-out.
12. Uncheck the check box associated with that layer or table to exclude individual layers or tables from the check-out.
13. Each entry in the Check Out column is a combo box of options. The options always include All Features and Schema Only. If a particular layer or table has a selection set or definition query defined, the options may also include Selected Features Only, All Features in Def. Query, and Selected Features in Def. Query. Select the All Features option and uncheck the Use Spatial Extent check box if you do not wish to apply any data filters.
14. Uncheck the Check out related data check box if you do not wish to check out any related data.
15. Click Next. ►



Type	Check Out
Feature Class	All Features
Feature Class	Selected Features
Feature Class	Selected Features Only
Feature Class	All Features in Def. Query
Geometric Net	All Features
	Schema Only

layer, you may choose to disregard that selection for this check-out. By default, all layers will be filtered by geometry, selection, and/or definition query. For tables, if no other filters have been applied—for example, a selection—the default filter is Schema Only; only the schema for the table will be checked out.

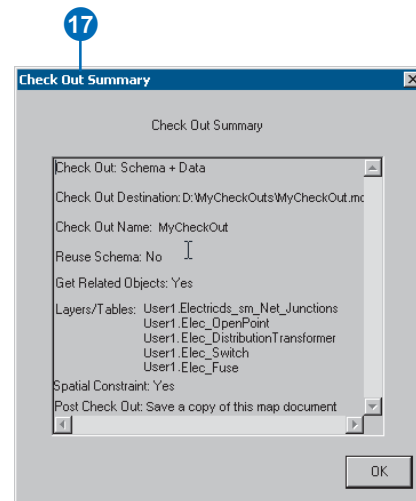
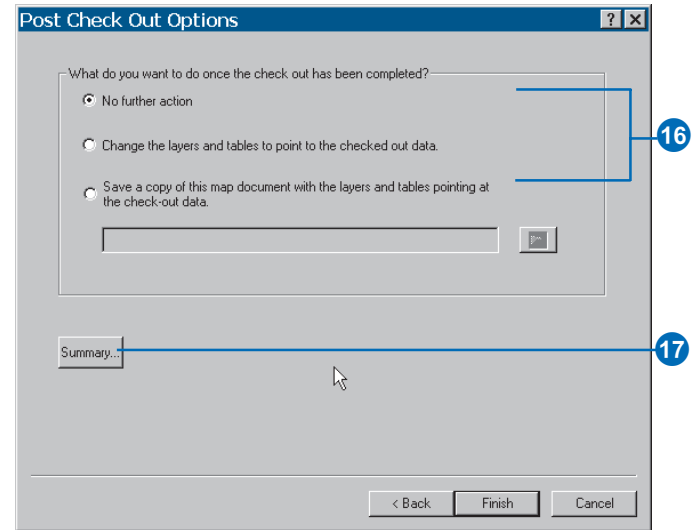
If you wish to exclude a geometric network or topology from a check-out, you must exclude all the participating layers; including just one layer will result in the whole network or topology being checked out.

The final panel includes some post check-out options:

- No further action (the default)—the current document will not be modified and no new documents created.
- The current ArcMap document will be modified to point to the data in the check-out geodatabase, preserving all symbology.
- A new ArcMap document, referencing the data in the check-out geodatabase, again with symbology preserved, will be created.

The summary report is available for reference.

16. Select an appropriate post check-out option.
17. Click Summary to review the parameters for the current check-out. ►



## Tip

### Related objects

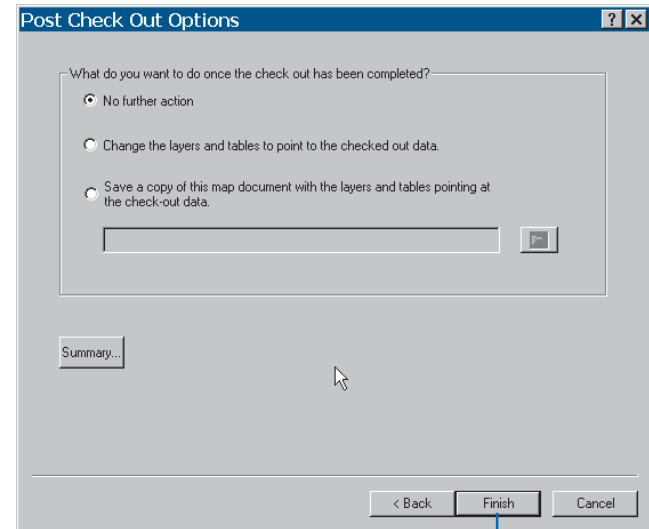
*It is always recommended that you include related objects in the check-out. This will ensure the integrity of the data throughout the check-out/check-in process. If you choose not to include related objects, you may introduce some inconsistencies in the data that may result in unexpected behavior.*

## Tip

### Saving new map documents

*If you opt to save a new map document with your check-out, there are two settings you can use to make the new document more portable. First, modify your existing document to store relative pathnames, and second, under the map document page setup (click the File menu, then click Page Setup), uncheck the Same as Printer option. Unchecking this option prevents information about the local default printer from being saved with the new document.*

18. Click Finish to start checking out the data. The status of the check-out operation will be monitored in a progress dialog box.



## Checking in data to a geodatabase

You can check in data from a geodatabase using the check-in wizard. The wizard supports both the *pull* and *push* models for checking in data. This utility is available in both ArcCatalog and ArcMap.

You may also check in data from a delta database or delta XML file. They both contain changes exported from a check-out geodatabase.

### Tip

#### Reconcile following check-in

*If you choose to reconcile the changes with the parent version once the data has been checked in, any conflicts encountered must be resolved using standard geodatabase version reconciliation tools. If no conflicts are encountered, the changes will be posted to the parent version.*

## Checking in data—pull operation

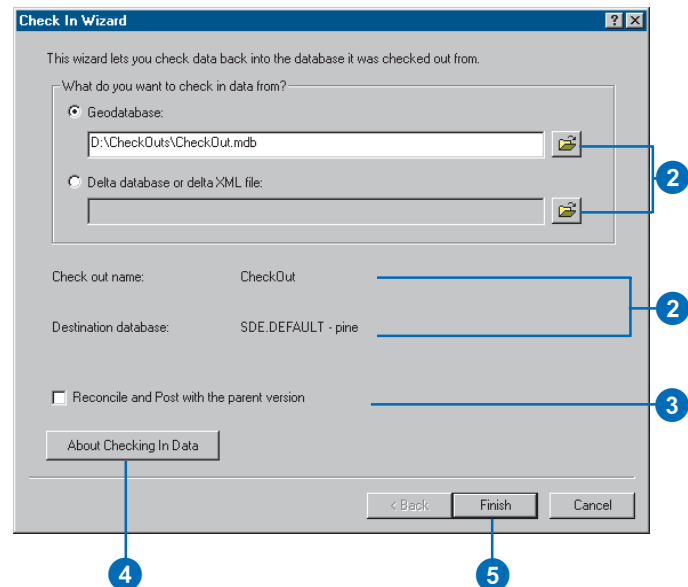
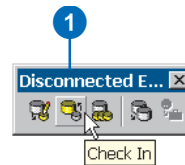
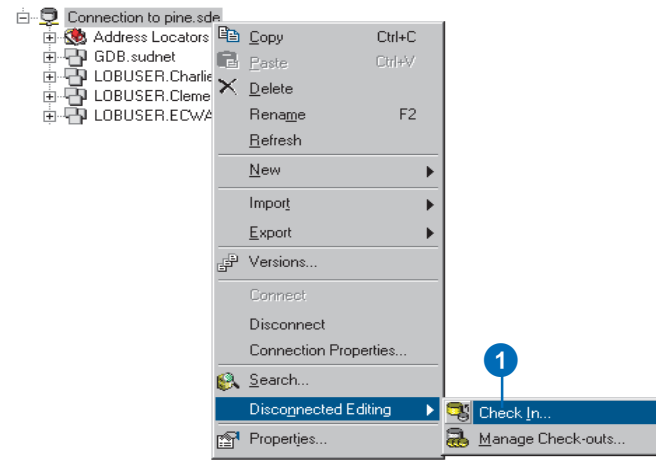
1. Browse for the master geodatabase in ArcCatalog to check data back in and right-click the connection item. Point to Disconnected Editing, then click Check In to activate the wizard.

In ArcMap, click the Check In command on the Disconnected Editing toolbar. If more than one master geodatabase is present, you will be prompted to choose one.

2. Enter the name of, or click the Browse button to navigate to, a check-out geodatabase, delta database, or delta XML file from which you want to check in the changes.

The name of the check-out and the master geodatabase you are checking in to will be listed for reference.

3. Check the Reconcile and Post box to reconcile and post the changes to the parent version once the check-in is complete.
4. Click About Checking In Data to learn more about checking in data.
5. Click Finish. The check-in status will be monitored in a progress dialog box.



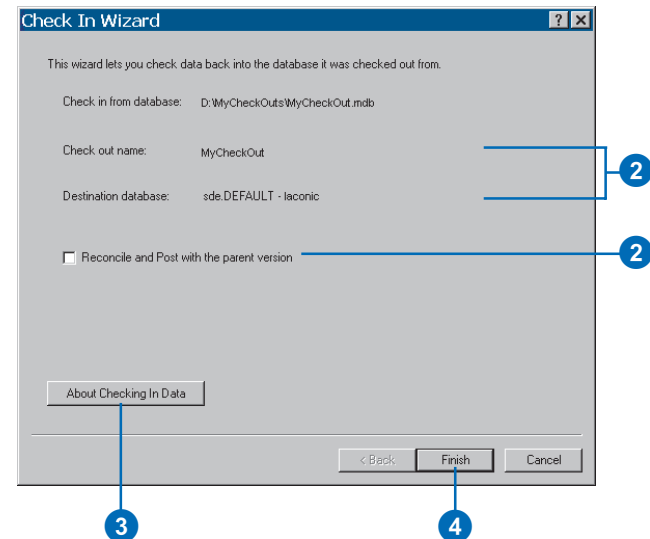
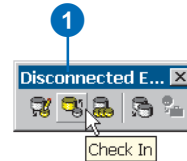
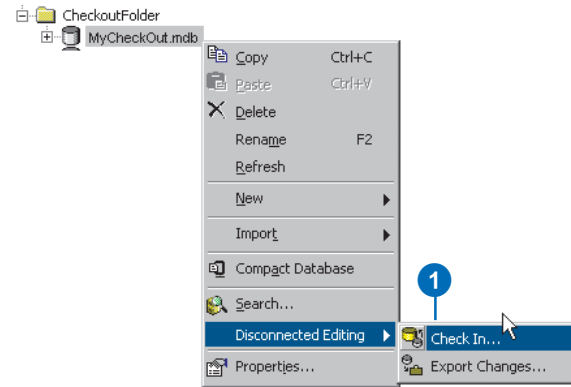
## Checking in data—push operation

1. Select your check-out geodatabase in ArcCatalog and right-click the database object. Point to Disconnected Editing, then click Check In.

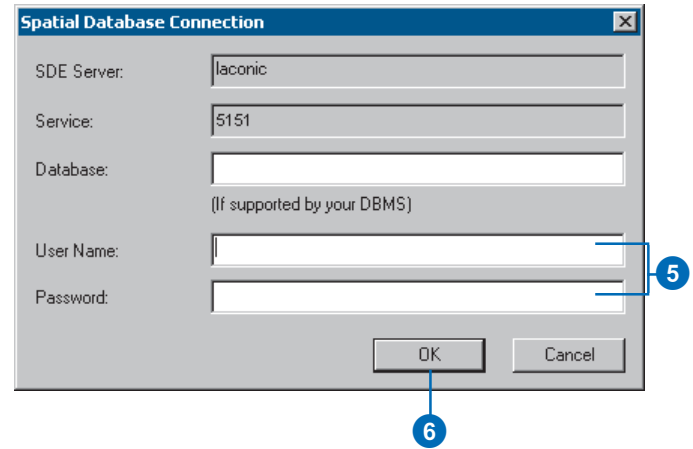
In ArcMap, click the Check In command on the Disconnected Editing toolbar. If more than one check-out geodatabase is present in your document, you will be prompted to choose one.

The name of the check-out and the master geodatabase you are checking in to will be listed for reference.

2. Check the Reconcile and Post box if you want to reconcile and post the changes to the parent version once the check-in is complete.
3. Click About Checking In Data to learn more about checking in data.
4. Click Finish.



5. Supply the necessary information to complete the connection to the master geodatabase in the SDE connection dialog box.
6. Click OK. The status of the check-in will be monitored in a progress dialog box.





# Managing check-outs

You can manage the check-outs in a master geodatabase using the check-out manager. With this utility, you can rename, refresh, and review the properties of each check-out and unregister as required. This utility is available in both ArcCatalog and ArcMap.

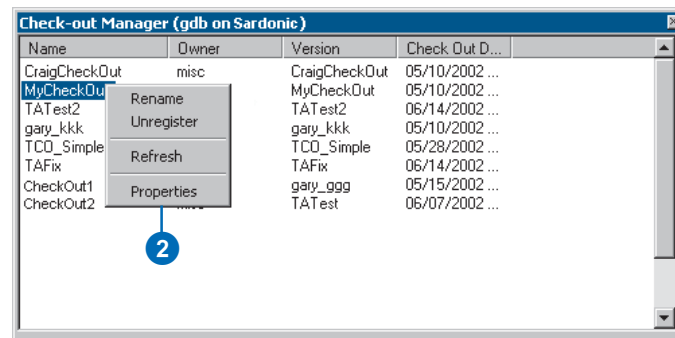
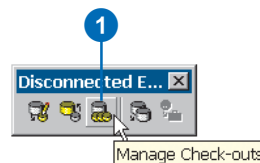
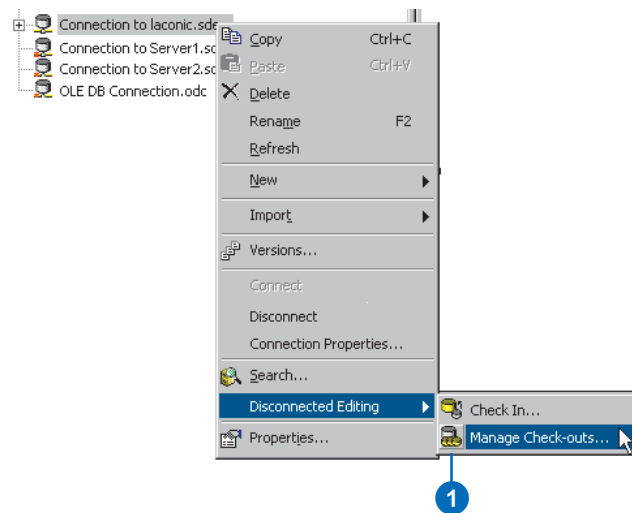
The general geodatabase property dialog box provides the same management utilities as the check-out manager for a check-out geodatabase. From this dialog box, the check-out may be renamed, its properties may be reviewed, or the check-out may be unregistered.

## Managing check-outs in a master geodatabase

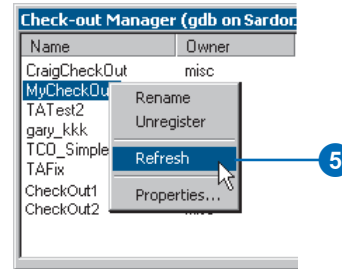
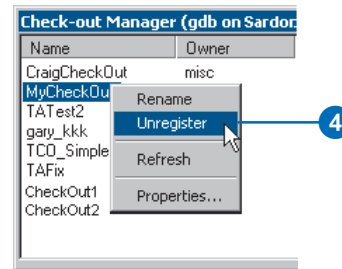
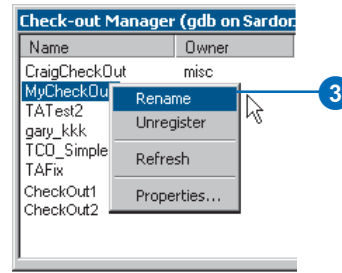
1. Right-click your database connection in ArcCatalog. Point to Disconnected Editing and click Manage Check-outs.

In ArcMap, click the Manage Check-outs command in the Disconnected Editing toolbar. If more than one master geodatabase is present in your document, you will be prompted to choose one.

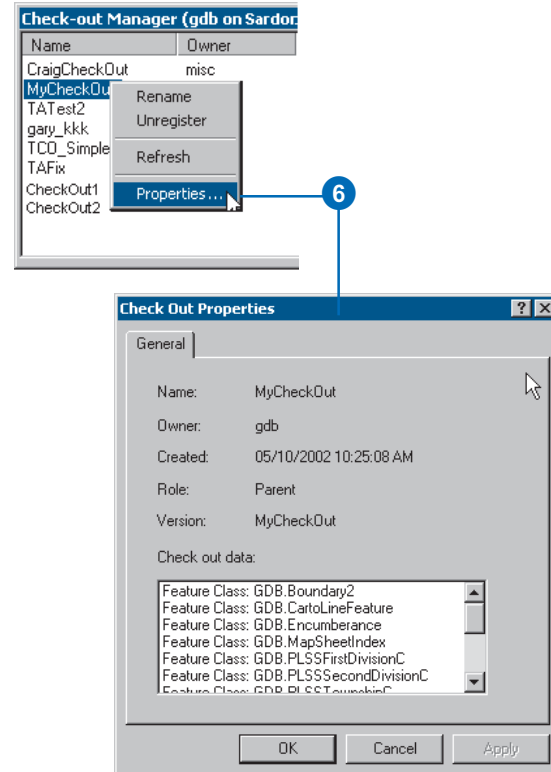
2. Select a check-out and right-click to review the options. ▶



3. To rename a check-out, right-click the check-out you want to rename and click Rename. Enter a new name and press Enter. This will automatically rename the associated check-out version at the same time.
4. To unregister a check-out, right-click the check-out you want to unregister and click Unregister. The version associated with this check-out will also be removed at the same time.
5. To refresh a check-out, right-click the check-out you want to refresh and click Refresh. The latest state of the properties will be displayed. ▶

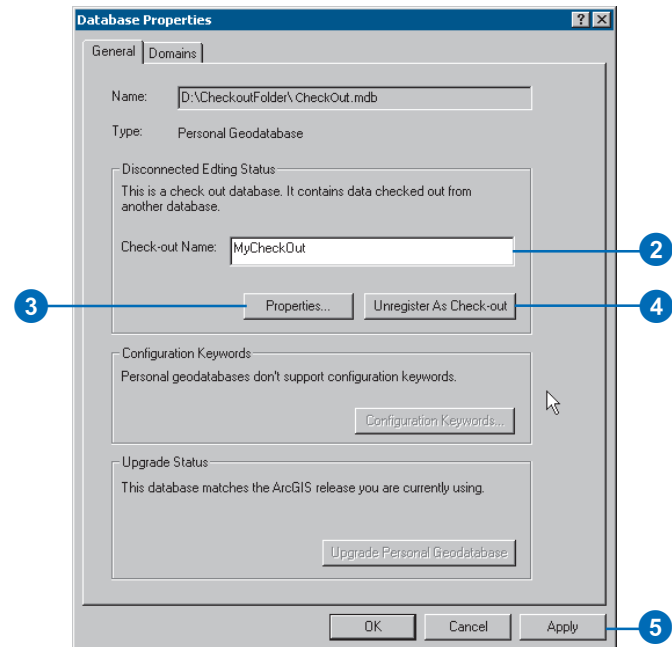
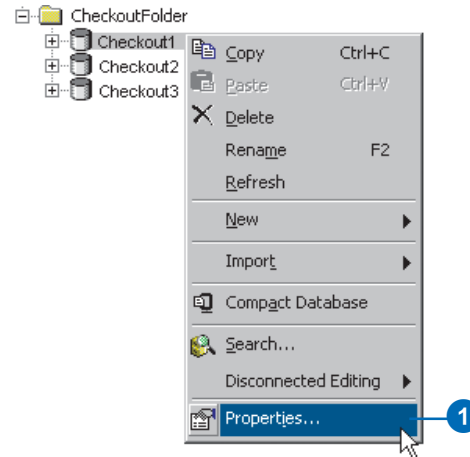


6. Right-click the check-out and click Properties to view the properties of a check-out. The properties include information on when the check-out was created, who created it, and the associated check-out version. A full list of the data checked out is also included.



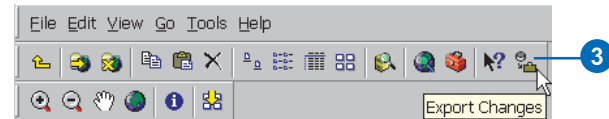
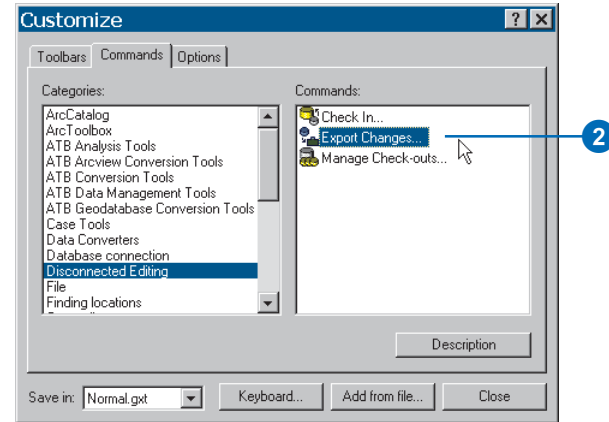
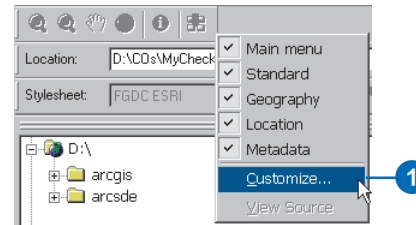
## Managing a check-out in a check-out geodatabase

1. Right-click your check-out geodatabase in ArcCatalog and click Properties.
2. Enter a new name in the name field to rename a check-out.
3. Click Properties to review the properties of a check-out.
4. Click Unregister As Check-out to unregister a check-out in a check-out geodatabase. This will remove all the information about the check-out in the geodatabase; it will not delete the data that was checked out. If the check-out geodatabase is an ArcSDE geodatabase, the check-out version in that geodatabase will be removed.
5. Click Apply.



## Exporting changes from a check-out geodatabase to a delta database or XML file

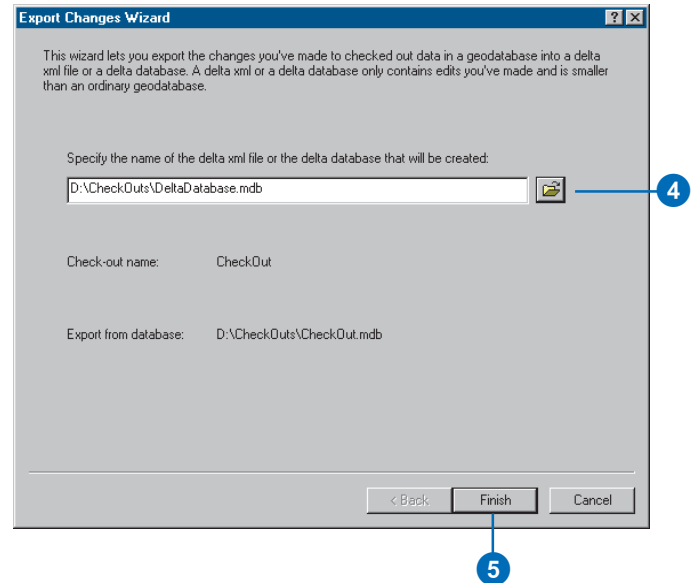
1. Right-click one of the toolbars in ArcCatalog or ArcMap to bring up the Customize dialog box.
2. Click the Commands tab to see a list of all the commands. Navigate to the Disconnected Editing category and select the Export Changes command.
3. Drag the command onto a toolbar. To enable this command, select a check-out geodatabase. Once enabled, click to bring up the Export Changes dialog box. ►



- Specify the path and name of the new delta database or XML file you will create.

The name of the check-out and the name of the check-out geodatabase from which you are exporting the changes are included for reference.

- Click Finish.



# Building a raster geodatabase

# 12

## IN THIS CHAPTER

- **Rasters and the geodatabase**
- **Importing and loading raster data**
- **Attributes of type raster**
- **Converting raster formats**
- **Mosaicking raster datasets**
- **Raster data and disconnected editing**
- **More about rasters in ArcGIS**

Raster data comes in many forms, and the pixels they contain describe many different types of phenomena. Each pixel contains information describing the spatial area it represents. These pixels can depict many different types of information, such as a reflectance value (that is, a measure of light) or a thematic value (for example, land use type). When grouped together, these pixels create images depicting information that can be used in analysis or as a backdrop to a map. Raster images can originate from satellite imagery, aerial photography, scanned images, or as the result of analysis, using calculations such as surface interpolation, slope, hillshades, or aspect.

This chapter will discuss raster-related tasks and concepts within the realm of geodatabases. After reading this chapter, you will be able to create and manage your raster data in both a personal and ArcSDE geodatabase.

# Rasters and the geodatabase

## Why use rasters in a geodatabase?

There are many advantages to working with rasters in a geodatabase:

- Enterprise or personal geodatabase solutions
- Large data holdings easily built, modified, and utilized
- Choice of creating mosaics or raster catalogs
- Fast raster dataset display at any scale
- Enhanced raster catalog functionality
- Raster data extraction easily facilitated
- Raster data compression, for example, lossy or lossless (for ArcSDE)
- Taking advantage of the relational database management system's (RDBMS) security, multiuser access, user rights, recoverability, and so on.

## Raster datasets and catalogs

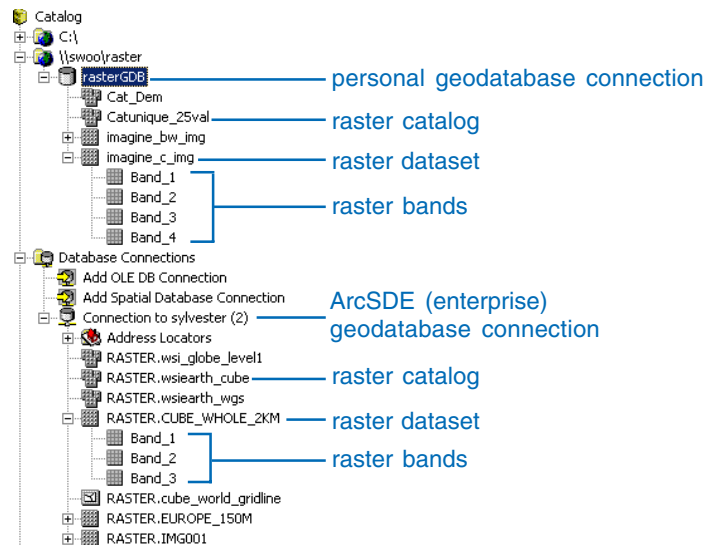
A raster dataset is any valid raster format that is organized into bands. Each band consists of an array of pixels, and each pixel has a value. A raster dataset has at least one band. More than one raster dataset can be mosaicked together into a larger, single continuous raster dataset.

A raster catalog is a collection of raster datasets defined in a table format in which each record identifies the individual raster datasets included in the catalog. A raster catalog is most often used to display adjacent, fully overlapping, or partially overlapping raster datasets without having to mosaic them into one large raster dataset.

When working with multiple raster datasets, there are three possible storage methods—you can store each raster dataset individually, mosaic them all into one large raster dataset, or store

them all inside a raster catalog container. Storing the raster datasets individually is the best method when the datasets are not adjacent to each other and when the datasets are rarely used on the same project. Mosaicking your inputs together to form one large, single extent of raster data is appropriate for most applications, but a raster catalog may be desired for one or more of the following reasons:

- The extents of the raster datasets partially or fully overlap, and you want the common areas to be preserved.
- The extents of the raster datasets fully overlap and are part of a time series.
- You do not need to see the entire area at one time. Raster catalogs display a wireframe at smaller scales.
- There are times when it is important to preserve each piece of metadata associated with each raster dataset.





Rasters can also be an attribute to a feature. This means that a field of type raster can exist as one of the columns within a feature class. This is similar to having a hyperlink of a file-based image in a field, except the raster image is stored and protected within the geodatabase.

## Enterprise or personal geodatabase?

Raster data stored within the ArcSDE database offers an enterprise level of functionality, such as security, multiuser access, data sharing, and so on. Raster data is imported into ArcSDE and rebuilt as ArcSDE raster data, so it can be accessed from the underlying database in the most efficient manner. Because of its new storage structure, the raster data is said to be managed, or fully controlled, by the geodatabase.

In the personal geodatabase, the raster data is always stored as referenced raster dataset files on disk. The personal geodatabase can either manage or not manage the raster data it is referencing. When the raster data is managed, it is converted from its original location and format into an ERDAS IMAGINE (.img) file inside a special folder located next to the personal geodatabase. Deleting managed raster data deletes the associated ERDAS IMAGINE files, which are stored in the special raster geodatabase folder.

The personal geodatabase can also reference raster dataset files of any supported format from their existing location without having to manage the files. This type of reference does not require the raster data to be imported, and it can significantly save time. Prototyping is easily done using the nonmanaged approach, but because of its nonmanaged nature, there is potential insecurity in this scenario. Deleting a row (raster dataset) of a nonmanaged geodatabase will not delete the referenced raster files.

Because personal geodatabase raster datasets are likely to have additional data mosaicked into them, they are always managed by

the geodatabase. However, raster catalogs and raster attributes stored in a personal geodatabase can either be designated as managed or nonmanaged geodatabase entities.

A personal geodatabase is good for the following reasons:

- Raster catalogs can be managed or unmanaged.
- It works well to allow for quick raster catalog prototyping.
- Only a moderate number of users read the data.
- It does not involve an enterprise RDBMS; therefore, there is a lower cost for maintenance, administration, and the RDBMS itself.
- You do not already have an ArcSDE geodatabase, or you do not need its higher performance.

An ArcSDE enterprise geodatabase is good for these reasons:

- It has the best display performance on large raster datasets (can be multiple terabytes of data).
- It is the preferred option for raster dataset mosaics, which grow larger than 2 GB (uncompressed).
- It allows multiuser access for updating and reading.
- It allows for incremental updates.
- It allows for lossy or lossless compression.
- If you already use ArcSDE, then it is recommended that you make the best use of it by adding raster data, thus creating centralized data storage.

## Importing rasters

Raster data is imported into a geodatabase via the user interface in several ways. Raster data can be imported into either a personal or enterprise geodatabase using the Import context menu by clicking a geodatabase workspace. Data can also be

loaded into a raster dataset or raster catalog in a geodatabase with the Load Data context menu choice found in ArcCatalog. The Copy Raster geoprocessing tool can be used to import raster data. Finally, raster data can be loaded into the ArcSDE enterprise geodatabase using the SDERASTER command line loader. The most efficient way to load many raster datasets to the geodatabase is to use geoprocessing scripting and SDERaster batch files (ArcSDE only) within ArcObjects™ programming.

## Enterprise geodatabase raster storage

In the enterprise geodatabase, raster data is stored in a structure where the data is tiled, indexed, pyramided, and most often compressed. Because of tiling, indexing, and pyramiding, each time the raster data is queried, only the tiles necessary to satisfy the extent and resolution of the query are returned, instead of the whole dataset. Compression, which is highly recommended, reduces the amount of data transferred between the client and the server, making it possible to store large, seamless raster datasets and raster catalogs as large as several terabytes and serve them quickly to a client for display.

There are several storage parameters from which you can choose when storing enterprise geodatabase raster data: pyramids, tile size, and data compression.

Pyramids are reduced resolution representations of your dataset used to improve performance. Pyramids can speed up the display of raster data because the server returns only the data at a specified resolution that is required for the display. Pyramids are created by resampling the original data into several different layers, each representing an increasingly larger resolution. The resampling methods instruct the server how to resample the data to build the pyramids. Nearest neighbor should be used for nominal data or raster datasets with color maps, such as land use or pseudocolor images. Bilinear interpolation or cubic

convolution should be used for continuous data such as satellite imagery or aerial photography. Prototyping the most appropriate resampling technique for your particular data is highly recommended. Remember that pyramid resampling only affects the display, not the original data.

The tile size controls the number of pixels you want to store in each database memory block. This is specified as a number of pixels. The default tile size is 128 by 128, and most applications do not warrant deviating from these default values.

Data compression compresses the tiles of raster data before storing them in the geodatabase. The compression used can be lossy (JPEG and JPEG 2000) or lossless (LZ77). Lossless compression means the values of cells in the raster dataset will not be changed. The amount of compression will depend on the type of pixel data; the more homogeneous the image, the higher the compression ratio.

The primary benefit of compressing your data is that it requires less storage space. An added benefit is the overwhelmingly improved performance, because you are transferring fewer packets of data from the server to the client application.

Lossy compression should be chosen for the following reasons:

- If the rasters are only background images, and there will be no analysis on the raster data.
- Data loading and retrieval are faster.
- Less storage space is needed since the compression ratios can be 5:1 or 10:1 (choose JPEG 2000 for higher compression ratios such as 10:1 or 20:1).

Lossless compression should be chosen for the following reasons:

- The raster datasets are to be used for deriving new data or even visual analysis.

- The compression required is only between 1:1 and 3:1.
- You don't plan to retain the original data.
- Your inputs are already lossy compressed.

Although the enterprise geodatabase storage can accommodate raster data without compression, compression is recommended. If you are unsure of which to use, always use the default of *LZ77* (lossless).

# Importing and loading raster data

Whether you create a new geodatabase or are working with an existing one, you will most likely need to import some data. Importing data allows you to take existing data and make it available for use within a geodatabase.

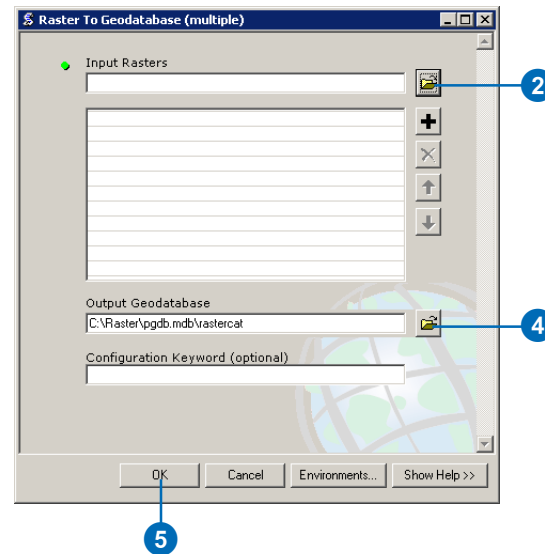
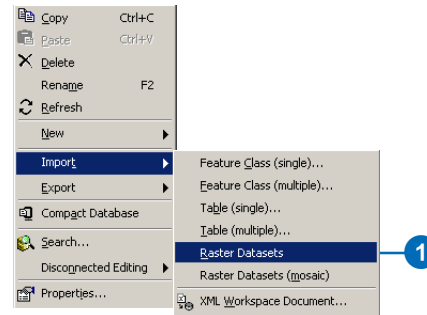
You are able to import rasters to a raster dataset or a raster catalog. Importing rasters to a raster dataset will add the raster dataset into an empty raster dataset or mosaic it with a raster dataset that is already present. Importing a raster dataset into a raster catalog will add the raster dataset into the raster catalog as a new raster dataset item, or you may choose to mosaic it into an existing raster dataset item.

Raster data can be loaded into a geodatabase in several ways including using the Import Raster Datasets (geodatabase workspace context menu), Copy Raster (geoprocessing tools), or Load Data (ArcCatalog dataset context menu) commands.

## Importing raster datasets into a geodatabase

1. Right-click the geodatabase, point to Import, and click Raster Datasets.
2. Click the Browse button to navigate to the raster datasets you want to import.
3. Highlight the raster datasets and click Add.
4. Click the Browse button to set the output geodatabase if it has not already been specified.
5. Click OK.

This will create individual raster datasets in the geodatabase using the default parameters. To set different parameters, click the Environments button and edit the settings for Raster Geodatabase.



## Tip

### The raster database naming convention

The raster dataset name cannot have spaces. You can use underscores to separate letters.

## Tip

### Load raster data into an empty raster dataset

Once an empty raster dataset is created, you can load raster data into it.

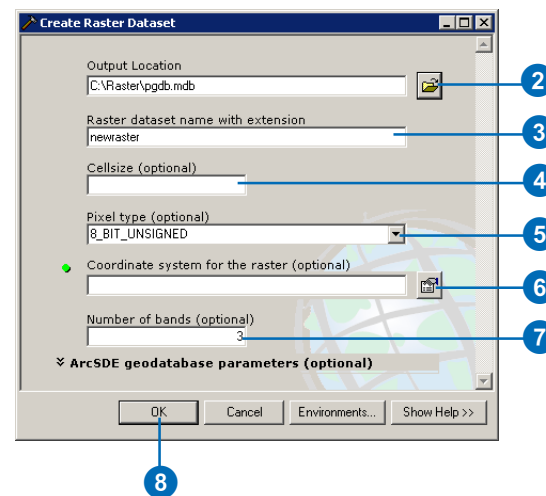
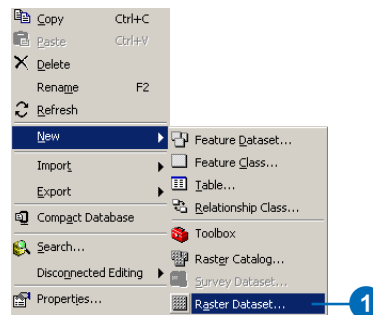
## Tip

### Blank spatial reference

If you leave the spatial reference blank, the raster dataset will use the spatial reference of the raster that you load.

## Creating an empty raster dataset in a personal geodatabase

1. Right-click the personal geodatabase, point to New, then click Raster Dataset.
2. Make sure that the output geodatabase is correct. If not, navigate to the proper geodatabase.
3. Type in the name of the new raster dataset. No extension is needed, since it will be stored in a geodatabase.
4. Set the cell size of the geodatabase raster dataset. If you do not specify a cell size, the cell size of the raster dataset that you load into it will be used.
5. Set the pixel type for the geodatabase raster dataset.
6. Click the Browse button to set the spatial coordinate system.
7. Type the number of bands that the raster dataset will contain.
8. Click OK.



## Tip

### Personal versus enterprise

*Different parameters can be set when loading raster data to either a personal or ArcSDE geodatabase. In general, an ArcSDE geodatabase contains more raster storage control by incorporating more parameters.*

## Tip

### Cube projection and the Create Raster Dataset dialog box

*If the user defines the cube projection as the raster dataset spatial reference, the predefined pyramid origin will be used, so the settings in the ArcSDE parameters for pyramids will be ignored. If the cell size is specified, it will be snapped to one of the predefined cell sizes that is closest to the specified cell size. If the cell size is not specified, it will be determined by the cell size of the first dataset mosaicked into it. It will be one of the predefined cell sizes that is closest to the cell size of the first dataset that is projected to cube.*

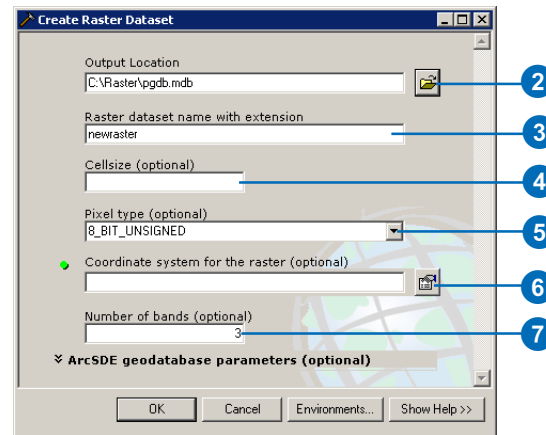
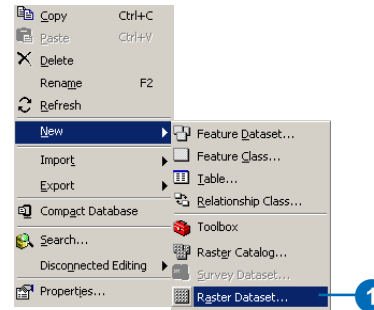
## Tip

### ArcSDE licensing

*You will need an ArcSDE license to create or edit in ArcSDE.*

## Creating an empty raster dataset in an ArcSDE geodatabase

1. Right-click the ArcSDE geodatabase, point to New, then click Raster Dataset.
2. Make sure that the output geodatabase is correct. If not, navigate to the proper geodatabase.
3. Type in the name of the new raster dataset. No extension is needed, since it will be stored in an ArcSDE geodatabase.
4. Set the cell size of the geodatabase raster dataset.
5. Set the pixel type for the geodatabase raster dataset.
6. Click the Browse button to set the spatial coordinate system.
7. Type the number of bands that the raster dataset will contain. ▶



## Tip

### JPEG compression quality

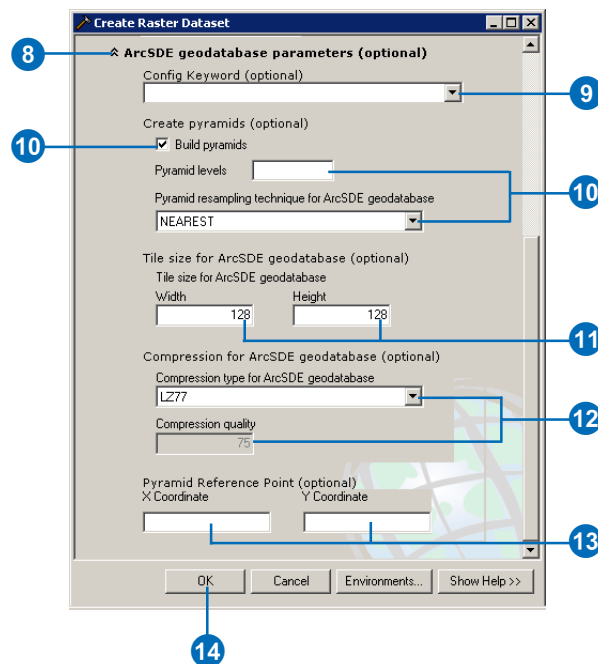
The quality factor can be set between the values of 5 and 95. The larger the value, the higher the quality, the larger the storage size, and the smaller the compression ratio. Prototyping is the key with this parameter.

## Tip

### Pyramid Reference Point

The pyramid reference point is an offset that you can create. It is helpful if you plan on mosaicking additional raster datasets. It is not necessary that you set the X and Y coordinates, but it is necessary if you plan on utilizing partial pyramid updates.

8. Optionally, you can configure the SDE geodatabase parameters by clicking the arrows to expand the SDE geodatabase parameters.
9. Optionally, set a configuration keyword.
10. Toggle the checkbox to either create or not create pyramids. If you choose to create pyramids, you may specify the number of pyramid levels to create and an appropriate resampling method.
11. Optionally, set the tile size. It is recommended that you simply use the defaults, but feel free to experiment and prototype different solutions.
12. Optionally, set the compression type. If you choose JPEG or JPEG 2000 compression, you may also set the compression quality.
13. Optionally, set the X and Y coordinates for the Pyramid Reference Point.
14. Click OK.



## Tip

### Raster data loading options

Make sure that the Raster data loading options have been set to the proper setting before you load any data. You can find these settings in the Raster Geodatabase settings in the Environment settings.

## Tip

### Mosaicking tolerance

The mosaicking tolerance controls whether pixel resampling takes place. If the difference in pixel alignment between the incoming dataset and the target dataset is less than the tolerance, no resampling is performed; instead, a shift is performed. The unit of tolerance is the pixel, which has valid values between 0 and 1.

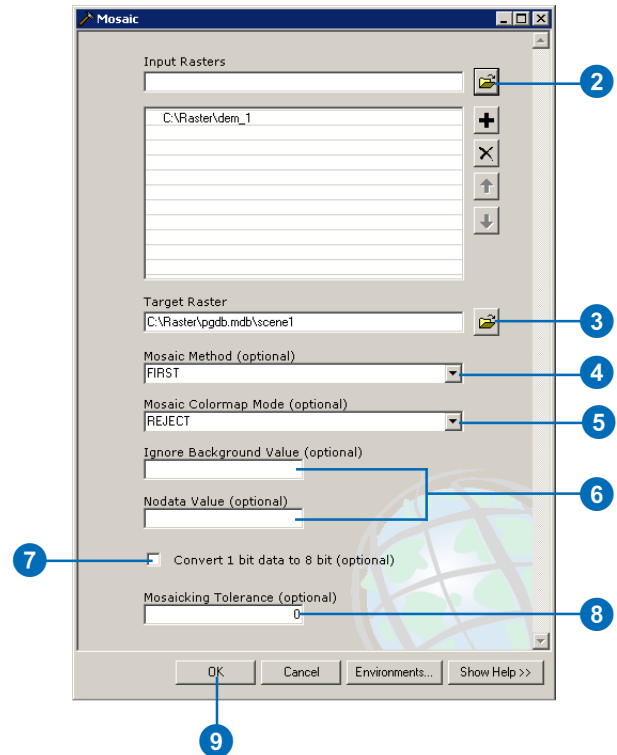
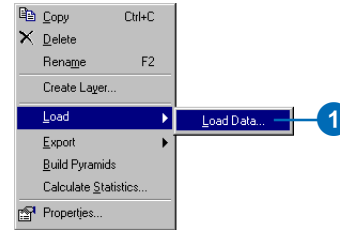
## Tip

### Analyze your raster data

After loading your data to an ArcSDE enterprise geodatabase, be sure to analyze your raster dataset or raster catalog. This will allow the geodatabase to perform at its optimal level, increasing display speeds dramatically.

## Loading a raster dataset into an empty geodatabase raster dataset

1. Right-click the empty geodatabase raster dataset, point to Load, then click Load Data.
2. Navigate to the location of the raster dataset you want to load and select it.
3. Make sure that the output location is correct. If not, navigate to the proper location.
4. Choose the rule for overlapping areas if you are loading more than one raster dataset (mosaicking).
5. Choose the rule for dealing with colormaps if you are loading raster datasets that have colormaps.
6. Optionally, you can set a background value to ignore and a Nodata value.
7. Optionally, check the box to convert your 1-bit data to 8-bit data.
8. Optionally, you can set the mosaicking tolerance.
9. Click OK.





## Tip

### Loading raster datasets into your new raster catalog

Once you have created your raster catalog, you can load raster datasets into it using the Load Data command from the context menu in ArcCatalog.

## Tip

### Managed by the personal geodatabase

Raster datasets within raster catalogs can be managed in two ways by the personal geodatabase: managed by the geodatabase or not managed by the geodatabase. To have the raster catalog managed by the geodatabase means that the raster datasets will be copied and stored alongside the personal geodatabase. When a row is deleted from the catalog, it is deleted from the raster geodatabase. When you do not have your raster managed by the geodatabase, there will only be a pointer connecting the raster catalog row to the file-based raster dataset.

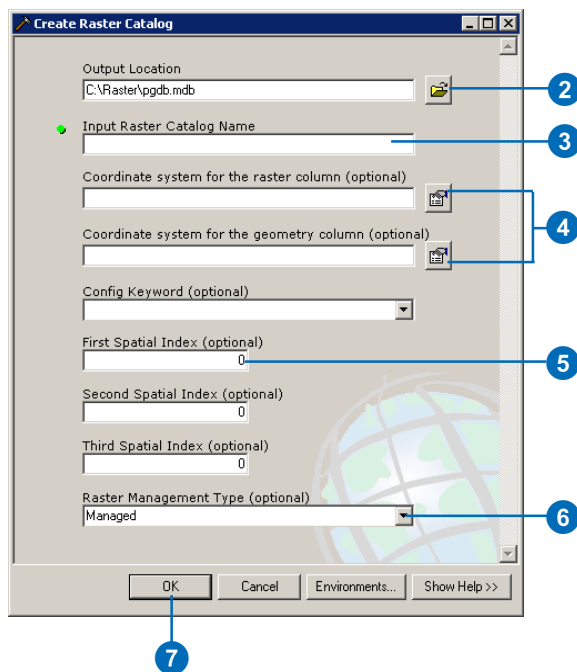
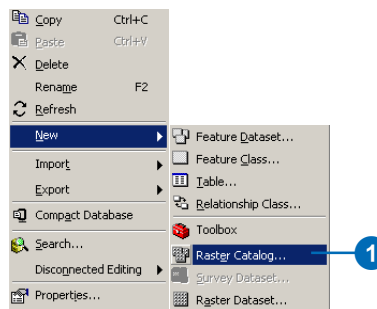
## Tip

### Setting the domain

Setting the domain for a raster catalog works in the same way as it would for a feature class. For more information, refer to 'Setting an appropriate geodatabase spatial domain' in this book.

## Creating a raster catalog in a personal geodatabase

1. Right-click the personal geodatabase, point to New, and click Raster Catalog.
2. Make sure that the output geodatabase is correct. If not, navigate to the proper geodatabase.
3. Type in a name for the new raster catalog.
4. Set the coordinate system for the raster column. Set the coordinate system and domain for the geometry column. Ensure that the domain is set to adequately surround the entire set of raster datasets that will comprise the raster catalog.
5. Optionally, set one spatial index.
6. Choose whether you want your raster catalog to be managed by the geodatabase.
7. Click OK.



## Tip

### The raster database naming convention

The raster dataset name cannot have spaces. You can use underscores to separate letters.

## Tip

### Upgrading ArcSDE 8.x embedded raster catalogs

To upgrade an ArcSDE 8.x embedded raster catalog to a new geodatabase raster catalog, right-click the old raster catalog and click Register with Geodatabase. This can only be performed with an ArcSDE geodatabase.

## Tip

### SDERASTER command-line loader

The SDERASTER command-line loader loads raster catalogs with the 8.x schema. These raster catalogs must be upgraded with Register with Geodatabase, found on ArcCatalog's context menu.

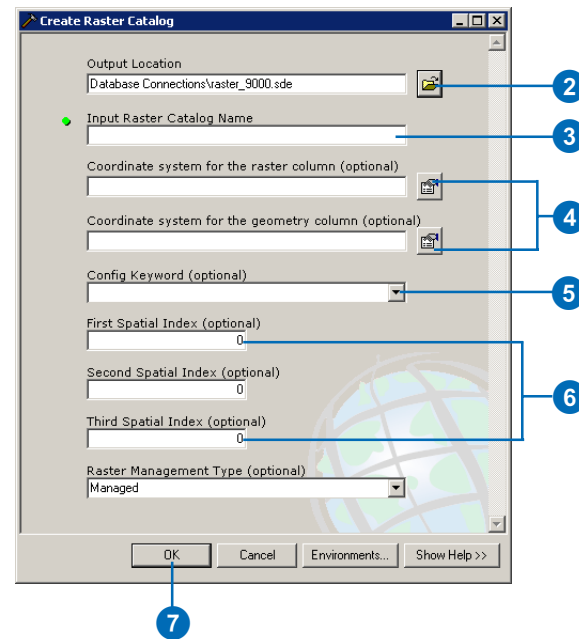
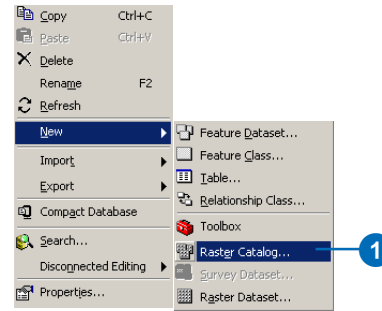
## Tip

### Setting the domain

Setting the domain for a raster catalog works in the same way as it would for a feature class. For more information, refer to 'Setting an appropriate geodatabase spatial domain' in this book.

## Creating a raster catalog in an ArcSDE geodatabase

1. Right-click the database connection, point to New, and click Raster Catalog.
2. Make sure that the output geodatabase is correct. If not, navigate to the proper geodatabase.
3. Type in a name for the new raster catalog.
4. Set the coordinate system for the raster column. Set the coordinate system and domain for the geometry column. Ensure that the domain is set to adequately surround the entire set of raster datasets, which will comprise the raster catalog.
5. Optionally, you can set a configuration keyword.
6. Optionally, set a spatial index (up to three can be set).
7. Click OK.



## Tip

### Loading data into a raster catalog

Raster data can be loaded into a geodatabase in several ways including using the *Import Raster Datasets* (geodatabase workspace context menu), *Copy Raster* (geoprocessing tools), or *Load Data* (ArcCatalog dataset context menu) commands.

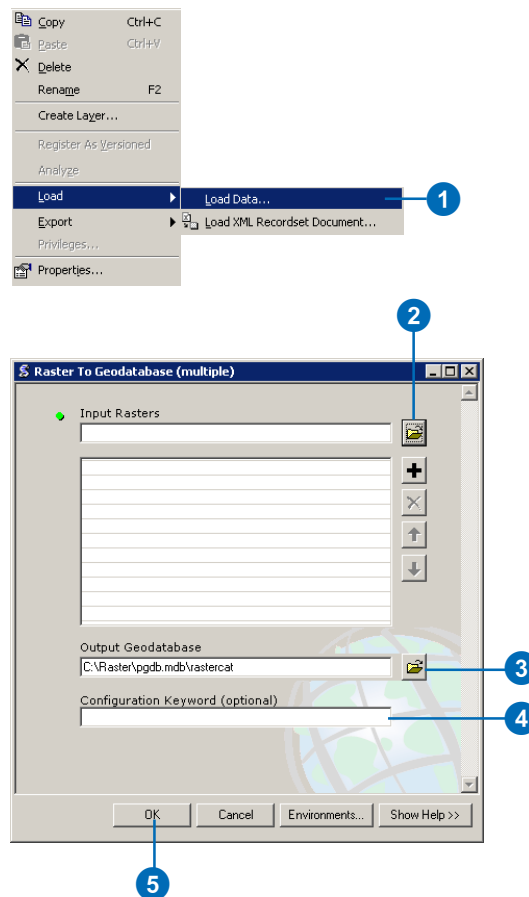
## Tip

### Building pyramids for raster catalogs

You cannot build pyramids on a raster catalog; however, you can build pyramids for each raster dataset within a raster catalog, which is recommended.

## Loading data into a raster catalog

1. Right-click the raster catalog, point to Load, then click Load Data.
2. Navigate to the location from which you want to load and select the raster datasets to load into the raster catalog.
3. Make sure that the output raster catalog is correct. If not, navigate to the proper raster catalog.
4. Optionally (for ArcSDE), choose a configuration keyword.
5. Click OK.



## Attributes of type raster

A feature class can have many fields or attributes. There are many valid data types that a field can possess, including text, integer, date, float, and even raster. Unlike a hyperlink that simply links a feature's field to an image, a field of type raster can actually store the raster data within or alongside the geodatabase.

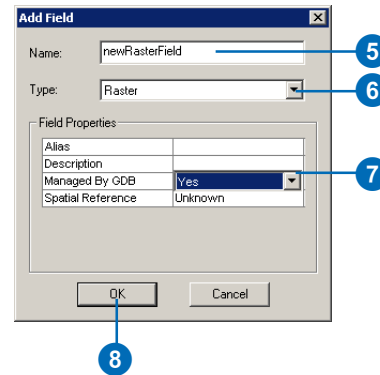
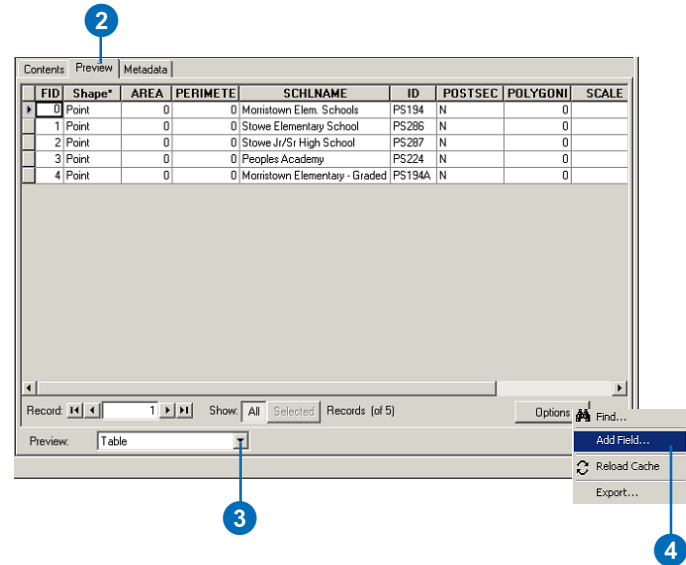
### Tip

#### Only one raster field

*Only one field of type raster may be used on each feature class or table. Use a different table with a related field to associate more than one raster dataset with each feature.*

1. Click a feature class within a personal geodatabase.
2. Click the Preview tab.
3. Click the Preview dropdown arrow and click Table.
4. Click the Options button and click Add Field.
5. Type a name for the new field of type raster.
6. Click the Type dropdown arrow and click Raster.
7. Edit any of the properties that may need editing.
8. Click OK.

To add or edit a raster in the raster field, see the section 'Updating raster attributes in ArcMap' in the online Help.



## Converting raster formats

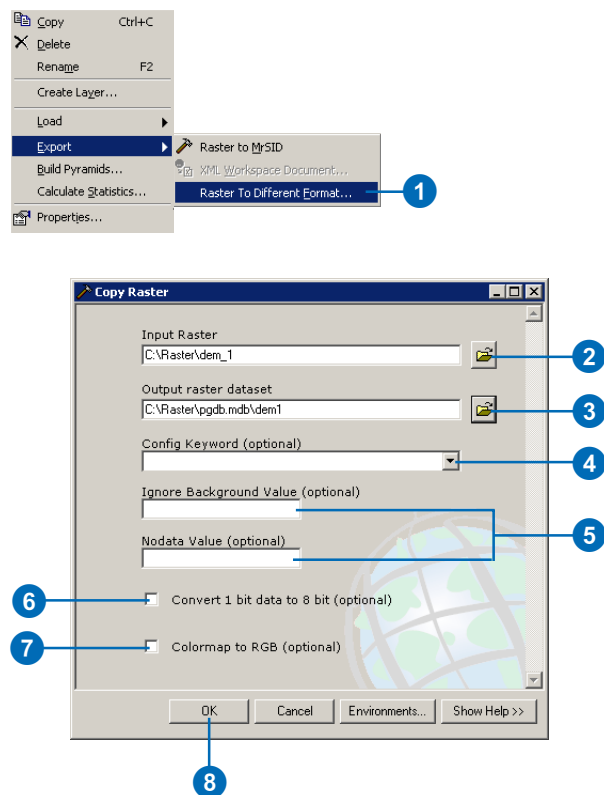
There are many different raster dataset formats. These formats are normally differentiated using different file extensions. ArcGIS is able to view many different raster file formats; however, it is only able to output a raster dataset as an ESRI GRID, ERDAS IMAGINE, or a TIFF. You can store these rasters in a file-based system or a personal geodatabase (even in geodatabase raster catalogs). You can also save your raster dataset or raster catalog to an ArcSDE geodatabase.

### Tip

#### Naming rasters in a geodatabase

*When you save to an ArcSDE or personal geodatabase, you do not need to specify a format—the format will be converted to a geodatabase format.*

1. Right-click the raster dataset, point to Export, and click Raster To Different Format.
2. Make sure that the input raster dataset is correct. If not, navigate to the proper raster dataset.
3. Navigate to the output location and type in the name of the output raster dataset. The output raster dataset format can be chosen on the Output Raster Dataset dialog box by choosing a geodatabase workspace or an appropriate file extension (within a file-based workspace).
4. Optionally, choose a configuration keyword (ArcSDE only).
5. Optionally, you can set a background value to ignore and a Nodata value.
6. Optionally, check the box to convert your 1-bit data to 8-bit data.
7. Optionally, check the box to convert your colormap raster into a red, green, and blue (RGB) raster.
8. Click OK.



# Mosaicking raster datasets

Mosaicking allows you to take two or more raster datasets and combine them into an existing raster dataset to create a single, seamless raster dataset. Mosaicking creates a raster dataset that is no different in schema from any other raster dataset.

All the raster datasets and the output raster mosaic must have the same number of bands and the same cell size; otherwise, the mosaic cannot be created.

## Tip

### Mosaicking tolerance

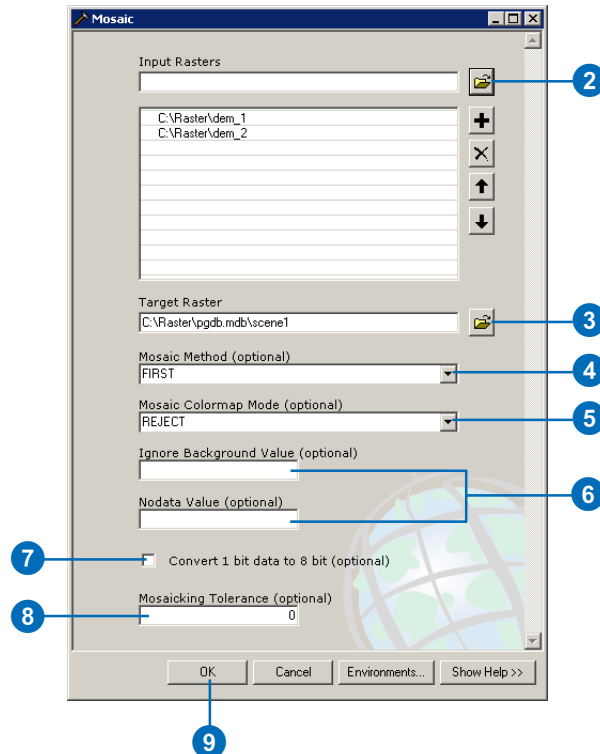
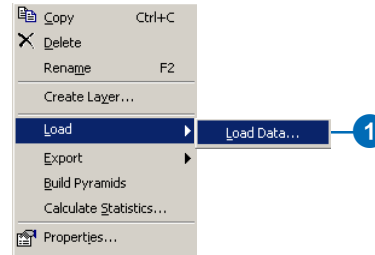
The mosaicking tolerance controls whether pixel resampling takes place. If the difference in pixel alignment between the incoming dataset and the target dataset is less than the tolerance, no resampling is performed; instead, a shift is performed. The unit of tolerance is the pixel, which has valid values between 0 and 1.

## Tip

### Mosaicking into a new raster dataset

Often, the output of a mosaic operation needs to be a new raster dataset. This can be accomplished by using the Mosaic To New Raster tool.

1. Right-click an existing raster dataset to contain the mosaic, point to Load, then click Load Data.
2. Navigate to the location of the raster datasets to be mosaicked and select them.
3. Make sure that the target raster dataset is correct. If not, navigate to the proper raster dataset.
4. Choose the rule to use when adjacent raster datasets are overlapping.
5. Choose the rule to use if colormaps are present.
6. Optionally, you can set a background value to ignore and a Nodata value.
7. Optionally, you can check the box to convert your 1-bit data to 8-bit data.
8. Optionally, you can set the mosaicking tolerance.
9. Click OK.



# Raster data and disconnected editing

Raster extraction allows you to clip a raster dataset or a raster catalog. Raster extraction performed on a raster dataset will output a raster dataset with only the extracted (clipped) portion of the dataset. Raster extraction that is performed on a raster catalog will output a raster catalog with each of the raster datasets that intersect (or are contained in) the extraction envelope.

## Tip

### Valid raster extraction inputs and outputs

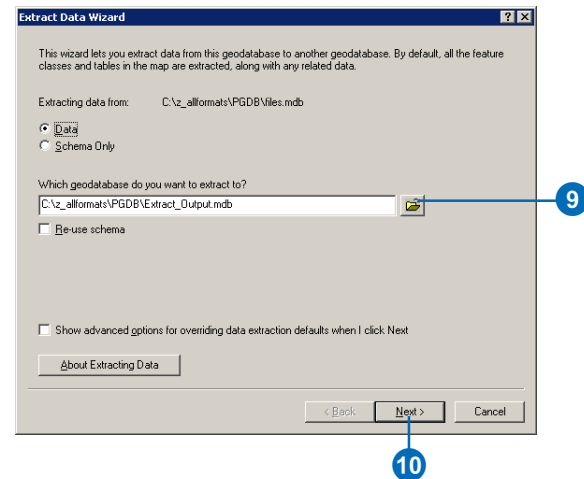
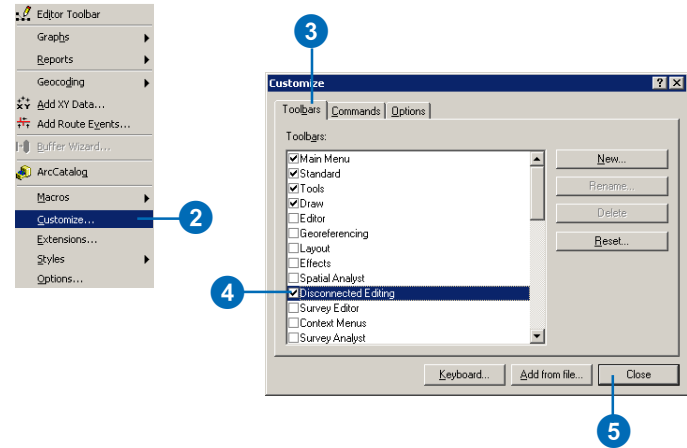
Only raster datasets and raster catalogs stored in a geodatabase are valid for performing raster extractions. The output of the extraction must also be stored within a geodatabase.

## Tip

### Raster check-out support

When performing a check-out of your data, raster data can be included. Although the raster data cannot be edited or checked back in, it will be extracted instead. Raster check-out only supports rasters created in ArcGIS 9.x.

1. Add a raster dataset or raster catalog contained within a geodatabase.
2. Click Tools on the main menu and click Customize.
3. Click the Toolbars tab.
4. Check the Disconnected Editing check box.
5. Click Close.
6. Click the Rectangle Draw tool.
7. Draw a rectangle around the area from which you want to extract data.
8. Click the Extract Data button on the Disconnected Editing toolbar.
9. Click the Browse button to specify the output geodatabase.
10. Click Next.
11. Click Finish.



# More about rasters in ArcGIS

## Speed

There are several ways to speed up your raster data within a geodatabase. Pyramids will help you display your raster data faster, since they display the most appropriate level of detail. Data compression compresses data in tiles before it is stored. Since ArcSDE normally serves data over a network with a specified and perhaps shared bandwidth, the fewer the data packets, the faster the transfer.

## Sharing

Data can be shared between users through a geodatabase or via an ArcIMS® server. Sharing on ArcSDE is especially useful since it has multiuser capabilities. All sorts of themes and layers can be shared, so duplicates of file-based raster datasets and raster catalogs are not necessary. If raster catalogs are distributed to other users, it is a good idea to create them using relative paths.

## Scale of use

The scale of use usually pertains to the cell size (pixel resolution) of the raster datasets you are using in your project. For example, it doesn't make sense to use a scale of pixels equal to 1 km to adequately map the streets of a city, nor does it make sense to use two-foot resolution data to map the soils of an entire country. One is as inadequate as the other is overkill. Scale of use also means ensuring that a scanned map that was built to satisfy a certain scale (for example, 1:24,000) is not used for projects requiring 1:100,000 scale mapping. Finally, scale of use can also refer to prototyping how much lossy compression will affect your data (for example, 1-meter orthophotos). Since higher compression ratios can degrade the display quality of the raster data, prototyping needs to be conducted to ensure that a subjective quality threshold has not been crossed.

## Raster catalogs

File-based raster catalogs are obviously useful; however, geodatabase raster catalogs have many additional benefits. A geodatabase raster catalog can contain raster datasets that are not homogeneous, whereby they can still be displayed properly. Using heterogeneous raster catalogs allows you to mix raster datasets of different cell sizes and number of bands. Geodatabase raster catalogs also allow the user to select raster datasets based on attribute or spatial queries.

## Loading procedures

Raster data can be loaded into a geodatabase in several ways including using the Import Raster Datasets (geodatabase workspace context menu), Copy Raster (geoprocessing tools), or Load Data (ArcCatalog dataset context menu) commands.

## Custom formats

Raster support in ArcGIS is built using a publicly available API. With this open structure, users can add their own format support into ArcGIS by creating a customized raster format dynamic link library (DLL) and plugging it into ArcGIS. The customized raster format will be treated the same as other raster formats (varying degrees of support may be programmed). The customized DLL created in ArcGIS can be plugged into ArcIMS 4.0.1 or later and customized format support can be served through ArcIMS. To learn more about creating your own format, refer to the technical documents section at [www.esri.com/arcobjectsonline](http://www.esri.com/arcobjectsonline) and look for the document 'Creating a customized raster format dll'.



# Glossary

## **absolute mode**

See digitizing mode.

## **active data frame**

The data frame currently being worked on—for example, the data frame to which layers are being added. The active data frame is highlighted on the map, and its name is shown in bold text in the table of contents.

## **alias**

An alternative name specified for fields, tables, and feature classes that is more descriptive and user-friendly than the actual name of these items. On computer networks, a single e-mail alias may refer to a group of e-mail addresses. In database management systems, aliases can contain characters, such as spaces, that can't be included in the actual names.

## **aligned dimension**

A drafting symbol that runs parallel to the baseline and indicates the true distance between beginning and ending dimension points. ArcInfo supports aligned dimension and linear dimension.

## **annotation**

Descriptive text used to label features on or around a map. Information stored for annotation includes a text string, a position at which it can be displayed, and display characteristics.

## **annotation class**

A subset of annotation in the standard or feature-linked geodatabase annotation feature class that contains properties that determine how the subset of annotation will display. A standard or feature-linked geodatabase annotation feature class may contain one or more annotation classes.

## **annotation feature class**

A geodatabase feature class that stores text or graphics that provide additional information about features or general areas of a map (annotation). An annotation feature class may be linked to another feature class, so that edits to the features are reflected in the annotation (feature-linked annotation). Annotation in a geodatabase is edited during an edit session, using the tools on the Annotation toolbar.

## **ArcInfo workspace**

A file-based collection of coverages, grids, triangulated irregular networks (TINs), or shapefiles stored as a directory of folders in the file system.

## **arc-node topology**

The data structure in a coverage used to represent linear features and polygon boundaries and to support analysis functions, such as network tracing. Nodes represent the beginning and ending vertices of each arc. Arcs that share a node are connected, and polygons are defined by a series of connected arcs. An arc that intersects another arc is split into two arcs. Each arc that defines all or part of a polygon boundary records the number of the polygon to its left and to its right, giving it a direction of travel.

## **ArcSDE**

Server software that provides ArcSDE client applications (such as ArcGIS Desktop, ArcGIS Server, and ArcIMS) a gateway for storing, managing, and using spatial data in one of the following commercial database management systems: IBM DB2 UDB, IBM Informix, Microsoft SQL Server, and Oracle.

## **asynchronous**

Not synchronous; that is, not happening, existing, or arising at the same time. For example, in disconnected editing, modifying the properties of a check-out is an asynchronous operation; changes made to the check-out in a master geodatabase do not affect the associated check-out in a check-out geodatabase.

## **attribute**

1. Information about a geographic feature in a GIS, generally stored in a table and linked to the feature by a unique identifier. For example, attributes of a river might include its name, length, and average depth.
2. In raster datasets, information associated with each unique value of raster cells.

## **attribute domain**

In a geodatabase, a mechanism for enforcing data integrity. Attribute domains define what values are allowed in a field in a feature class or nonspatial attribute table. If the features or nonspatial objects have been grouped into subtypes, different attribute domains can be assigned to each of the subtypes.

## **attribute table**

A database or tabular file containing information about a set of geographic features, usually arranged so that each row represents a feature and each column represents one feature attribute. In raster datasets, each row of an attribute table corresponds to a certain region of cells having the same value. In a GIS, attribute tables are often joined or related to spatial data layers, and the attribute values they contain can be used to find, query, and symbolize features or raster cells.

## **attributes dialog box**

In ArcMap, a dialog box that displays attributes of selected features for editing.

## **azimuth**

The angle, measured in degrees, between a baseline drawn from a center point and another line drawn from the same point. Normally, the baseline points true north, and the angle is measured clockwise from the baseline. Azimuth is often used to define an oblique cylindrical map projection or the angle of a geodesic between two points.

## **band**

A set of adjacent wavelengths or frequencies with a common characteristic, such as the visible band of the electromagnetic spectrum.

## **behavior**

The way in which an object in a geodatabase functions or operates. Behavior rules define how geodatabase objects can be edited and drawn. Defined behaviors include, but are not limited to, validation rules, subtypes, default values, and relationships.

## **bilinear interpolation**

A resampling method that uses an average of the four nearest cells to determine the new value.

## **buffer**

A zone around a map feature measured in units of distance or time. A buffer is useful for proximity analysis.

## **CAD**

A computer-based system for the design, drafting, and display of graphical information. Also known as computer-aided drafting, such systems are most commonly used to support engineering, planning, and illustrating activities.

## **CAD feature class**

A read-only member of a CAD feature dataset, comprising one of the following: polylines, points, polygons, multipatch, or annotation. The feature attribute table of a CAD feature class is a virtual table comprising select CAD graphic properties and any existing field attribute values.

## **centroid**

The geometric center of a feature. Of a line, it is the midpoint; of a polygon, the center of area; of a three-dimensional figure, the center of volume.

## **check-in**

The procedure that transfers a copy of data into a master geodatabase, overwriting the original copy of that data and reenabling it so it can be accessed and saved from that location.

## **check-out**

A procedure that records the duplication of data from one geodatabase to another and disables the original data so that both versions cannot be accessed or saved at the same time.

## **check-out geodatabase**

A personal or ArcSDE geodatabase that contains data checked out from a master geodatabase.

## **check-out version**

The data version created in a check-out geodatabase when data is checked out to that database. This version is created as a copy of the synchronization version. Only the edits made to this check-out version can be checked back in to the master geodatabase.

See also check-out geodatabase, master check-out version.

## **circular arc**

A line with two vertices, one situated at each endpoint, rather than a line composed of numerous vertices with line segments between them.

## **clip**

In ArcInfo, a command that extracts the features from one coverage that reside entirely within a boundary defined by features in another coverage (the clip coverage).

## **cluster tolerance**

In geodatabase feature classes, a definition for the minimum tolerated distance between vertices in the topology. Vertices that fall within the set cluster tolerance will be snapped together during the validate topology process.

## **clustering**

A part of the topology validation process in which vertices that fall within the cluster tolerance are snapped together.

## **coincident**

Occupying the same space. Coincident features or parts of features occupy the same space in the same plane. In geodatabase feature classes, vertices or boundaries that fall within the set cluster tolerance of one another are coincident; they are snapped together during the validate topology process.

## **colormap**

A set of values that are associated with colors. Colormaps are most commonly used to display a raster dataset consistently on many different platforms.

## **column**

The vertical dimension of a table. Each column stores the values of one type of attribute for all the records, or rows, in the table. All the values in a given column are of the same data type—for example, number, string, BLOB, date.

## **compaction**

See compression.

## **compression**

A reduction of file size for data handling and storage. Examples of such methods include quadtrees, run-length encoding, and wavelet.

## **computer-aided design**

See CAD.

## **conflict**

An opposing action of incompatibles that occurs when multiple users simultaneously edit a version or reconcile two versions. Conflicts occur when the same feature or topologically related features are edited in both the edit and reconciliation versions, and it is unclear in the database which representation is valid.

## **connectivity**

1. In a geodatabase, the state of edges and junctions in a logical network that controls flow, tracing, and pathfinding.
2. In a coverage, topological identification of connected arcs by recording the from-node and to-node for each arc. Arcs that share a common node are connected. See also arc-node topology.

## **connectivity rule**

A rule that constrains the type and number of network features that can be connected to one another in a geodatabase. There are two types of connectivity rules: edge-junction and edge-edge.

## **constraints**

Limits imposed on a model to maintain data integrity. For example, in a water network model, an 8-inch pipe cannot connect to a 4-inch pipe.

## **construct features**

In ArcMap, an edit command that takes selected features from one or more feature classes and creates new features in a target feature class. The Construct Features tool uses the input geometries of the selected features to construct polygons or lines following polygon boundaries, depending on the geometry of the target feature class.

## **control**

In mapping, a system of points with established horizontal and vertical positions that are used as fixed references for known ground points or specific locations. The establishment of controls is one of the first steps in digitizing.

## **control points**

See control.

## **coordinate system**

A fixed reference framework superimposed on the surface of an area to designate the position of a point within it; a reference system consisting of a set of points, lines, and/or surfaces; and a set of rules, used to define the positions of points in space in either two or three dimensions. The Cartesian coordinate system and the geographic coordinate system used on the earth's surface are common examples of coordinate systems.

## **coordinates**

Values represented by x, y, and possibly z that define a position in terms of a spatial reference framework. Coordinates are used to represent locations on the earth's surface relative to other locations.

## **coverage**

A data model for storing geographic features using ArcInfo software. A coverage stores a set of thematically associated data considered a unit. It usually represents a single layer, such as soils, streams, roads, or land use. In a coverage, features are stored as both primary features (points, arcs, polygons) and secondary features (tics, links, annotation). Feature attributes are described and stored independently in feature attribute tables. Coverages cannot be edited in ArcGIS.

## **cracking**

A part of the topology validation process in which vertices are created at the intersection of feature edges.

## **cubic convolution**

A resampling method that uses an average of the nearest 16 cells to calculate the new cell value.

## **current task**

During editing in ArcMap, a setting in the Current Task dropdown list that determines the task with which the sketch construction tools (Sketch, Arc, Distance–Distance, and Intersection) will work. The current task is set by clicking a task in the Current Task dropdown list.

## **custom behavior**

A set of methods, functions, or operations associated with a geodatabase object that has been specifically created or overridden by a developer.

## **custom feature**

In geodatabases, a feature with specialized behavior instantiated in a class by a developer.

## **custom object**

An object with custom behavior provided by a developer.

## **dangle length**

See dangle tolerance.

## **dangle tolerance**

In ArcInfo coverages, the minimum length allowed for dangling arcs by the Clean process. Clean removes dangling arcs shorter than the dangle tolerance.

## **dangling arc**

An arc having the same polygon on both its left and right sides and having at least one node that does not connect to any other arc. It often occurs where a polygon does not close properly, where arcs do not connect properly (an undershoot), or where an arc was digitized past its intersection with another arc (an overshoot). A dangling arc is not always an error; for example, it can represent a cul-de-sac in a street network.

## **data**

Any collection of related facts arranged in a particular format; often, the basic elements of information that are produced, stored, or processed by a computer.

## **data frame**

A map element that defines a geographic extent, a page extent, a coordinate system, and other display properties for one or more layers in ArcMap. A dataset can be represented in one or more data frames. In data view, only one data frame is displayed at a time; in layout view, all of a map's data frames are displayed at the same time. Many cartography texts use the term "map body" to refer to what ESRI calls a data frame.

## **data integrity**

The degree to which the data in a database is accurate and consistent according to data model and data type. Data integrity is maintained through the creation of attribute domains and through mandatory conflict resolution between versions of a dataset.

## **data source**

Any geographic data. Data sources may include coverages, shapefiles, rasters, or feature classes.

## **data type**

The attribute of a variable, field, or column in a table that determines the kind of data it can store. Common data types include character, integer, decimal, single, double, and string.

## **data view**

An all-purpose view in ArcMap and ArcReader™ for exploring, displaying, and querying geographic data. This view hides all map elements, such as titles, North arrows, and scalebars.

See also layout view.

## **database**

One or more structured sets of persistent data, managed and stored as a unit and generally associated with software to update and query the data. A simple database might be a single file with many records, each of which references the same set of fields. A GIS database includes data about the spatial locations and shapes of geographic features recorded as points, lines, areas, pixels, grid cells, or TINs, as well as their attributes.

## **dataset**

Any organized collection of data with a common theme.

## **dataset precision**

The mathematical exactness or detail with which a value is stored within the dataset, based on the number of significant digits that can be stored for each coordinate. In a geodatabase, the precision of the dataset is the number of internal storage units that are allocated to each of the linear units of a coordinate system.

## **decimal degrees**

Values of latitude and longitude expressed in decimal format rather than degrees, minutes, and seconds.

## **default junction type**

In geometric networks, the user-established junction type that automatically connects two edge types in the absence of a current user choice, in cases where two edge types may be connectable through more than one junction type. An edge may also have a default end junction type, used for the free ends of new edges.

## **delta database**

See delta file.

## **delta file**

A file that contains data edits that can be exchanged between geodatabases or between geodatabases and other data stores. The edits can come from a check-out geodatabase, modified rows between source and target versions, or a custom application. Supported delta file formats are XML (delta XML file) and delta database (.mdb file).

## **digitizing**

The process of converting the geographic features on an analog map into digital format using a digitizing tablet, or digitizer, which is connected to a computer. Features on a paper map are traced with a digitizer puck, a device similar to a mouse, and the x,y coordinates of these features are automatically recorded and stored as spatial data.

## **digitizing mode**

One of the ways in which a digitizing tablet operates. In digitizing mode, locations on the tablet are mapped to specific locations on the screen. Moving the digitizer puck on the tablet surface causes the screen pointer to move to precisely the same position.

## **dimension construction methods**

Procedures that dictate what type of dimension feature is created and the number of points required to complete the feature's geometry. Construction methods include simple aligned, aligned, linear, rotated linear, free aligned, and free linear.

## **dimension feature**

A special kind of geodatabase annotation that shows specific lengths or distances on a map. A dimension feature may indicate the length of a side of a building or land parcel, or it may indicate the distance between two features, such as a fire hydrant and the corner of a building. ArcInfo supports aligned dimensions and linear dimensions.

## **dimension feature class**

A collection of spatial data in the geodatabase that shares the same dimension features. Like other feature classes in the geodatabase, all features in a dimension feature class have a geographic location and attributes and can either be inside or outside a feature dataset.

## **dimension style**

Description of a dimension feature's symbology, what parts of it are drawn, and how it is drawn. Every time a new dimension feature is created, it is assigned a particular style according to its shared characteristics. A collection of dimension styles is associated with a dimension feature class. Styles for a dimension feature class are created, copied, and managed using ArcCatalog or the editing capabilities in ArcMap. Styles are then assigned to individual dimension features.

## **Dimensioning toolbar**

A toolbar in ArcMap that facilitates the creation of dimension features.

## **direct connect**

A two-tier configuration for connecting to a spatial database. Direct connect moves processing from the server to the client. It does not require the ArcSDE Application Server to connect to a spatial database. With direct connect, ArcSDE processing still occurs, but it primarily happens on the client side.

## **dirty areas**

Regions surrounding features that have been altered after the initial topology validation process and require an additional topology validation to be performed to discover any errors.

## **disconnected editing**

The process of copying data to another geodatabase, editing that data, then merging the changes with the data in the source or master geodatabase.

## **distance units**

The units of length (for example, feet, miles, meters, or kilometers) that are used to report measurements, dimensions of shapes, and distance tolerances and offsets.

## **domain**

See attribute domain.

## **double precision**

The level of coordinate exactness based on the possible number of significant digits that can be stored for each coordinate. Datasets can be stored in either single or double precision. Double-precision geometries store up to 15 significant digits per coordinate (typically 13 to 14 significant digits), retaining the accuracy at much less than one meter at a global extent.

See also single precision.

## **double-coordinate precision**

See double precision.

## **edge**

A line between two nodes, points, or junctions that forms the boundary of one or more faces of a spatial entity. In an image, edges separate areas of different tones or colors. In a topology, an edge defines lines or polygon boundaries; multiple features in one or more feature classes may share edges.

## **edge element**

A line connecting nodes in the network through which a commodity, such as information, water, or electricity, presumably flows.

## **edge–edge rule**

In geodatabases, a connectivity rule that establishes that an edge of type A may connect to an edge of type B through a junction of type C. Edge–edge rules always involve a junction type.

## **edge–junction cardinality**

In a relationship between objects in a geodatabase, the number of edges of one type that may be associated with junctions of another type. Edge–junction cardinality defines a range of permissible connections that may occur in a one-to-many relationship between a single junction and many edges.

## **edge–junction rule**

A connectivity rule in geodatabases establishing that an edge of type A may connect to a junction of type B.

## **edit cache**

See map cache.

## **edit session**

In ArcMap, the environment in which spatial and attribute editing take place. After starting an edit session, a user can modify feature locations, geometry, or attributes. Modifications are not saved unless the user explicitly chooses to save them.

## **Editor toolbar**

In ArcMap, a set of tools that allows the creation and modification of features and their attributes.



## **ellipse**

A geometric shape equivalent to a circle that is viewed obliquely. It is described mathematically as the collection of points whose distances from two given points add up to the same sum.

## **error**

See topology error.

## **extensible markup language (XML)**

Developed by the World Wide Web Consortium (W3C), XML is a standard for designing text formats that facilitates the interchange of data between computer applications. XML is a set of rules for creating standard information formats using customized tags and sharing both the format and the data across applications.

## **extent**

The coordinate pairs defining the minimum bounding rectangle (xmin, ymin and xmax, ymax) of a data source. All coordinates for the data source fall within this boundary.

## **feature**

1. A representation of a real-world object on a map. Features can be represented in a GIS as vector data (points, lines, or polygons) or as cells in a raster data format. To be displayed in a GIS, features must have geometry and locational information.
2. A group of spatial elements that represents a real-world entity. A complex feature is made up of more than one group of spatial elements: for example, a set of line elements with the common theme of roads representing a road network.

## **feature attribute table**

See attribute table.

## **feature class**

A collection of geographic features with the same geometry type (such as point, line, or polygon), the same attributes, and the same spatial reference. Feature classes can stand alone within a geodatabase or be contained within shapefiles, coverages, or other feature datasets. Feature classes allow homogeneous features to be grouped into a single unit for data storage purposes. For example, highways, primary roads, and secondary roads can be grouped into a line feature class named “roads”. In a geodatabase, feature classes can also store annotation and dimensions.

## **feature dataset**

A collection of feature classes stored together that share the same spatial reference; that is, they have the same coordinate system, and their features fall within a common geographic area. Feature classes with different geometry types may be stored in a feature dataset.

## **field**

A column in a table that stores the values for a single attribute.

See also attribute, column.

## **fuzzy tolerance**

In ArcInfo, the distance within which coordinates of nearby features are adjusted to coincide with each other when topology is being constructed. Nodes and vertices within the fuzzy tolerance are merged into a single coordinate location, connecting previously separate features. Fuzzy tolerance is a very small distance, usually from 1/1,000,000 to 1/10,000 times the width of the coverage extent, and is generally used to correct inexact intersections. The fuzzy tolerance defines the resolution of a coverage resulting from the Clean operation or a topological overlay operation, such as Union, Intersect, or Clip. In geodatabase feature classes, this concept is replaced by cluster tolerance.

## **GDB**

See geodatabase.

## **geodatabase**

An object-oriented data model introduced by ESRI that represents geographic features and attributes as objects and the relationships between objects but is hosted inside a relational database management system. A geodatabase can store objects, such as feature classes, feature datasets, nonspatial tables, and relationship classes.

## **geodatabase data model**

A geographic data model that represents real-world geographic features as objects in an object-relational database. In the geodatabase data model, features are stored as rows in a table, and geometry is stored in a shape field. Objects in the geodatabase data model may have custom behavior.

## **geometric network**

Topologically connected edge and junction features that represent a linear network such as a road, utility, or hydrologic system.

## **georelational data model**

A geographic data model that represents geographic features as an interrelated set of spatial and attribute data. The georelational model is the fundamental data model used in coverages.

## **identity**

A topological overlay that computes the geometric intersection of two coverages. The output coverage preserves all the features of the first coverage plus those portions of the second (polygon) coverage that overlap the first. For example, a road passing through two counties would be split into two arc features, each with the attributes of the road and the county it passes through.

See also intersect, union.

## **index**

A data structure used to speed the search for records in a database or for spatial features in geographic datasets. In general, unique identifiers stored in a key field point to records or files holding more detailed information.

## **instance**

See service.

## **intersect**

A geometric integration of spatial datasets that preserves features or portions of features that fall within areas common to the input datasets.

See also identity, union.

## **IP address**

Internet protocol address. The identification of each client or server computer on the Internet by a unique number. IP addresses allow data to travel between one computer and another via the Internet and are commonly expressed as a dotted quad, with four sets of numerals separated by periods.

## **item**

1. An element in the Catalog tree. Items include data sources, such as shapefiles and geodatabases, and nonspatial elements, such as folders.
2. A column of information in an INFO table.

See also field.

## **junction element**

In a linear network, a network feature that occurs at the intersection of two or more edges or at the endpoint of an edge that allows the transfer of flow between edges.

## **label point**

In a coverage, a feature class used to represent points or identify polygons. When representing points, the x,y location of the point describes the location of the feature. When identifying polygons, the point can be located anywhere within the polygon.

## **layer**

1. A reference to a data source, such as a coverage, geodatabase feature class, raster, and so on, that defines how the data should be displayed on a map. Layers can also define additional properties, such as which features from the data source are included. In ArcGIS 9, layers can be used as inputs to geoprocessing tools. Layers can be stored in map documents (.mxd) or saved individually as layer files (.lyr). Layers are conceptually similar to themes at ArcView 3.x.
2. A standalone feature class in a geodatabase managed with SDE 3.

## **layout view**

In ArcMap and ArcReader, the view for laying out a map. Layout view shows the virtual page upon which geographic data and map elements, such as titles, legends, and scalebars, are placed and arranged for printing.

See also data view.

## **left–right topology**

The topological data structure in an ArcInfo coverage that stores, for each arc, the identity of the polygons to the left and right of it. Left–right topology supports analysis functions such as adjacency.

See also topology.

## **linear dimension**

A measurement of the horizontal or vertical dimension of a feature. Unlike aligned dimensions, linear dimensions do not represent the true distance between beginning and ending dimension points.

## **logical network**

An abstract representation of a network. A logical network consists of edge, junction, and turn elements and the connectivity between them. It does not contain information about the geometry or location of its elements.

## **lossless compression**

Data compression that has the ability to store data without changing any of the values; however, it is only able to compress the data at a low ratio (typically 2:1 or 3:1). In GIS, this type of compression is often used to compress raster data when the pixel values of the raster will be used for analysis or deriving other data products.

## **lossy compression**

Data compression that provides high compression ratios (for example, 10:1 to 100:1); however, lossy compression does not retain all the information in the data. In GIS, lossy compression is used to compress raster datasets that will be used as background images, but this compression is not suitable for raster analysis.

## **map**

1. A graphic depiction on a flat surface of the physical features of the whole or a part of the earth or other body, or of the heavens, using shapes to represent objects and symbols to describe their nature; at a scale whose representative fraction is less than 1:1. Maps generally use a specified projection and indicate the direction of orientation.

2. The document used in ArcMap to display and work with geographic data. In ArcMap, a map contains one or more layers of geographic data, contained in data frames, and various supporting map elements, such as a scalebar.

## **map cache**

A setting used in ArcMap that allows temporary storage of geodatabase features from a given map extent in the desktop computer's RAM, which may result in performance improvements in ArcMap for editing, feature rendering, and labeling.

## **map document**

In ArcMap, the file that contains one map; its layout; and its associated layers, tables, charts, and reports. Map documents can be printed or embedded in other documents. Map document files have a .mxd extension.

## **map feature**

See feature.

## **map projection**

See projection.

## **map topology**

A temporary set of topological relationships between coincident parts of simple features on a map, used to edit shared parts of multiple features.

## **map units**

The ground units—for example, feet, miles, meters, or kilometers—in which the coordinates of spatial data are stored.

## **master check-out version**

The data version in the master geodatabase, created when data is checked out, that represents the state of the data at the time it was checked out.

See also check-out version, master geodatabase.

## **master geodatabase**

A geodatabase from which data has been checked out.

See also check-out geodatabase.

## **merge policy**

In geodatabases, rules that dictate what happens to the respective attributes of features that are merged together during editing in ArcMap. A merge policy can be set to assign a default value to the new attribute, summarize the values of the merged attributes, or create a weighted average from the merged attributes.

## **minimum bounding rectangle**

A rectangle, oriented to the x and y axes, that bounds a geographic feature or a geographic dataset. It is specified by two coordinate pairs: xmin, ymin and xmax, ymax. For example, an extent can define a minimum bounding rectangle for a coverage.

## **mosaic**

A raster dataset that is composed of two or more merged raster datasets—for example, one image created by merging several individual images or photographs of adjacent areas.

## **multipart feature**

A feature that is composed of more than one physical part but only references one set of attributes in the database. For example, in a layer of states, the state of Hawaii could be considered a multipart feature. Although composed of many islands, it would be recorded in the database as one feature.

## **multipoint feature**

A feature that consists of more than one point but only references one set of attributes in the database. For example, a system of oil wells might be considered a multipoint feature, since there is a single set of attributes for multiple well holes.

## **multiuser geodatabase**

A geodatabase in an RDBMS served to client applications—for example, ArcMap—by ArcSDE. Multiuser geodatabases can be very large and support multiple concurrent editors. They are supported on a variety of commercial RDBMSs including Oracle, Microsoft SQL Server, IBM DB2, and Informix.

## **nearest-neighbor resampling**

A technique for resampling raster data in which the value of each cell in an output grid is calculated using the value of the nearest cell in an input grid. Nearest-neighbor assignment does not change any of the values of cells from the input layer; for this reason, it is often used to resample categorical or integer data (for example, land use, soil, or forest type).

See also bilinear interpolation, cubic convolution.

## **network trace**

A function that follows connectivity in a geometric network. Specific kinds of network tracing include finding features that are connected, finding common ancestors, finding loops, tracing upstream, and tracing downstream.

See also geometric network.

## **node**

In a geodatabase, the point representing the beginning or ending point of an edge, topologically linked to all the edges that meet there.

## **null value**

The absence of a recorded value for a geographic feature. A null value differs from a value of zero in that zero may represent the measure of an attribute, while a null value indicates that no measurement has been taken.

## **object**

In GIS, a digital representation of a discrete spatial entity. An object may belong to an object class and will thus have attribute values and behavior in common with other defined elements.

## **object class**

A collection of objects in the geodatabase that have the same behavior and the same set of attributes. All objects in the geodatabase are stored in object classes.

## **overshoot**

The portion of an arc digitized past its intersection with another arc.

See also dangling arc.

## **pan**

To move an onscreen display window up, down, or across a map image without changing the viewing scale.

**password**

A secret series of characters that enables a user to access a computer, data file, or program. The user must enter his or her password before the computer will respond to commands. The password helps ensure that unauthorized users do not access the computer, file, or program.

**personal geodatabase**

A geodatabase that stores data in a single-user relational database management system. A personal geodatabase can be read simultaneously by several users, but only one user at a time can write data into it.

**pixel**

The smallest unit of information in an image or raster map. Usually square or rectangular, pixel is often used synonymously with cell.

**pixel type**

See data type.

**planarize**

The process of creating multiple line features by splitting longer features at the places where they intersect other line features. This process can be useful when you have nontopological line work that has been spaghetti digitized or imported from a CAD drawing.

See also spaghetti data.

**point**

A zero-dimensional abstraction of an object; a single x,y coordinate pair that represents a geographic feature too small to be displayed as a line or area at that scale.

**point digitizing**

See point mode digitizing.

**point feature**

See point.

**point mode digitizing**

A method of digitizing in which a series of precise points, or vertices, are created.

See also stream mode digitizing.

**polygon**

A closed, two-dimensional figure with at least three sides that represents an area. It is used in GIS to describe spatial elements with a discrete area, such as parcels, political districts, areas of homogeneous land use, and soil types.

**polygon–arc topology**

In a polygon coverage, the list of topologically connected arcs that define the boundary of a polygon feature and the label point that links it to an attribute record in the coverage point attribute table.

**port number**

The TCP/IP port number on which an ArcSDE geodatabase service is communicating.

**post**

During versioned geodatabase editing, the process of applying the current edit session to the reconciled target version.

**precision (dataset)**

See dataset precision.

## **preliminary topology**

In coverages, refers to incomplete region topology. Region topology defines region–arc and region–polygon relationships. A topological region has both the region–arc relationship and the region–polygon relationship. A preliminary region has the region–arc relationship but not the region–polygon relationship. In other words, preliminary regions have no polygon topology. Coverages with preliminary topology have red in their icons in ArcCatalog.

## **projection**

A method by which the curved surface of the earth is portrayed on a flat surface. This generally requires a systematic mathematical transformation of the earth’s graticule of lines of longitude and latitude onto a plane. It can be visualized as a transparent globe with a light bulb at its center casting lines of latitude and longitude onto a sheet of paper. Generally, the paper is either flat and placed tangent to the globe (a planar or azimuthal projection), or formed into a cone or cylinder and placed over the globe (cylindrical and conical projections). Every map projection distorts distance, area, shape, direction, or some combination thereof.

## **property**

An attribute of an object defining one of its characteristics or an aspect of its behavior. For example, the Visible property affects whether a control can be seen at runtime. You can set an item’s properties using its Properties dialog box.

## **pseudonode**

In a geodatabase topology, a temporary feature marking the location where an edge has been split during an edit session. This type of pseudonode becomes a vertex when the edit is saved.

## **pull check-in**

A check-in operation initiated from a master geodatabase.

## **push check-in**

A check-in operation initiated from a check-out geodatabase.

## **pyramid**

In raster datasets, a reduced resolution layer that copies the original data in decreasing levels of resolution to enhance performance. The coarsest level of resolution is used to quickly draw the entire dataset. As the display zooms in, layers with finer resolutions are drawn; drawing speed is maintained because fewer pixels are needed to represent the successively smaller areas.

## **query**

A request that selects features or records from a database. A query is often written as a statement or logical expression.

## **radius**

The distance from the center to the outer edge of a circle or circular curve.

## **rank**

A method of assigning an accuracy value to feature classes to avoid having vertices from a feature class collected with a high level of accuracy being snapped to vertices from a less accurate feature class. Vertices from higher ranking feature classes will not be moved when snapping with vertices with lower-ranked feature classes. The highest rank is 1; up to 50 different ranks can be assigned.

## **raster**

A spatial data model that defines space as an array of equally sized cells arranged in rows and columns. Each cell contains an attribute value and location coordinates. Unlike a vector structure, which stores coordinates explicitly, raster coordinates are contained in the ordering of the matrix. Groups of cells that share the same value represent geographic features.

## **raster catalog**

A collection of raster datasets defined in a table of any format, in which the records define the individual raster datasets that are included in the catalog. A raster catalog is used to display adjacent or overlapping raster datasets without having to mosaic them together into one large file.

## **raster snapping**

See snapping.

## **RDBMS**

Relational database management system. A type of database in which the data is organized across several tables. Tables are associated with each other through common fields. Data items can be recombined from different files. In contrast to other database structures, an RDBMS requires few assumptions about how data is related or how it will be extracted from the database.

## **reconcile**

In version management, to merge all modified datasets, feature classes, and tables in the current edit session with a second target version. All features and rows that do not conflict are merged into the edit session, replacing the current features or rows. Features that are modified in more than one version are conflicts and require further resolution via the Conflict Resolution dialog box.

## **record**

A set of related data fields, often a row in a database, containing all the attribute values for a single entity. For example, in an address database, the fields that, together, provide the address for a specific individual comprise one record. In SQL terms, a record is analogous to a tuple.

## **reflectance**

The fraction of the total rate of flow of radiant energy incident upon a surface that is reflected and that varies according to the wavelength distribution of the incident radiation.

## **relate**

An operation that establishes a temporary connection between records in two tables using an item common to both.

## **relational database management system**

See RDBMS.

## **relational join**

An operation by which two data tables are related through a common field, known as a key.

## **relationship**

An association or link between two objects in a geodatabase. Relationships can exist between spatial objects (features in feature classes), nonspatial objects (rows in a table), or spatial and nonspatial objects.

## **relationship class**

An item in the geodatabase that stores information about a relationship. A relationship class is visible as an item in the ArcCatalog tree or contents view.



## **resampling**

The process of extrapolating new cell values when transforming rasters to a new coordinate space or cell size. The three most common resampling techniques are nearest neighbor assignment, bilinear interpolation, and cubic convolution.

## **resolution**

The area represented by each cell or pixel in a raster.

## **row**

1. A record in an attribute table.
2. The horizontal dimension of a table composed of a set of columns containing one data item each.
3. A horizontal group of cells in a raster.

## **scanning**

The process of capturing data in raster format with a device called a scanner. Some scanners also use software to convert raster data to vector data.

## **schema**

The organization and definitions of the feature classes, tables, and other items in a geodatabase. Creating or deleting items or changing their definitions modifies the schema. The schema does not include actual data, only its structure.

## **schema-only check-out**

A type of check-out that creates the schema of the data being checked out in the check-out geodatabase but does not copy any data.

## **segment**

A line that connects vertices. For example, in a sketch of a building, a segment might represent one wall.

## **select**

To choose from a number or group of features or records; to create a separate set or subset.

## **selectable layers**

Layers from which features can be selected in ArcMap with the interactive selection tools. Selectable layers can be chosen using the Set Selectable Layers command in the Selection menu or on the optional Selection tab in the table of contents.

## **selected set**

A subset of features in a layer or records in a table that is chosen by the user.

## **selection anchor**

In an ArcMap editing session, a small “x” located in the center of selected features. The selection anchor is used in the snapping environment or when rotating, moving, and scaling features.

## **server**

A computer in a network that is used to provide services, such as access to files or e-mail routing, to other computers in the network. Servers may also be used to host Web sites or applications that can be accessed remotely.

## **service**

A collection of persistent, server-side software processes that provides data or computing resources for client applications. Examples include ArcSDE Application Server, ArcIMS Application Server, and DBMS server.

## **shape**

The characteristic appearance or visible form of a geographic object. Geographic objects can be represented on a map using one of three basic shapes: points, lines, or polygons.

## **shapefile**

A vector data storage format for storing the location, shape, and attributes of geographic features. A shapefile is stored in a set of related files and contains one feature class.

## **shared boundary**

A segment or boundary common to two features. For example, in a parcel database, adjacent parcels will share a boundary. Another example might be a parcel that shares a boundary on one side with a river. The segment of the river that coincides with the parcel boundary would share the same coordinates as the parcel boundary.

## **shared vertex**

A vertex common to multiple features. For example, in a parcel database, adjacent parcels will share a vertex at the common corner.

## **simple feature**

A point, line, or polygon that is not part of a geometric network and is not an annotation feature, dimension feature, or custom object.

## **single precision**

Refers to a level of coordinate exactness based on the number of significant digits that can be stored for each coordinate. Single precision numbers store up to seven significant digits for each coordinate, retaining a precision of  $\pm 5$  meters in an extent of 1,000,000 meters. Datasets can be stored in either single or double precision coordinates.

See also double precision.

## **single-coordinate precision**

See single precision.

## **sketch**

In ArcMap, a shape that represents a feature's geometry. Every existing feature on a map has this alternate form, a sketch, that allows visualization of that feature's composition, with all vertices and segments of the feature visible. When features are edited in ArcMap, the sketch is modified, not the original feature. A sketch must be created in order to create a feature. Only line and polygon sketches can be created, since points have neither vertices nor segments.

## **sketch constraint**

In ArcMap editing, an angle or length limitation that can be placed on segments created using the Sketch tool.

## **sketch operation**

In ArcMap, an editing operation that is performed on an existing sketch. Examples are Insert Vertex, Delete Vertex, Flip, Trim, Delete Sketch, Finish Sketch, and Finish Part. All these operations are available from the Sketch context menu.

## **snapping**

An automatic editing operation in which points or features within a specified distance or tolerance of other points or features are moved to match or coincide exactly with each other's coordinates.

## **snapping environment**

Settings in the ArcMap Snapping Environment window and Editing Options dialog box that define the conditions in which snapping will occur. These settings include snapping tolerance, snapping properties, and snapping priority.

## **snapping priority**

The order in which snapping will occur by layer during an ArcMap editing session, set from the Snapping Environment window.

See also snapping environment.

## **snapping properties**

In ArcMap editing, a combination of a shape to snap to and a method for what part of the shape will be snapped to. Snapping properties can be set to have a feature snap to a vertex, edge, or endpoint of features in a specific layer. For example, a layer snapping property might allow snapping to the vertices of buildings. A more generic, sketch-specific snapping property might allow snapping to the vertices of a sketch being created.

## **snapping tolerance**

In an ArcMap editing session, the distance within which the pointer or a feature will snap to another location. If the location being snapped to (vertex, boundary, midpoint, or connection) is within that distance, the pointer will automatically snap. Snapping tolerance can be measured using either map units or pixels.

## **spaghetti data**

Vector data composed of simple lines with no topology and usually no attributes. Spaghetti lines may overlap, but there are no relationships between the features.

## **spatial database**

Any database that contains spatial data.

## **spatial domain**

For a spatial dataset, the defined precision and allowable range for x and y coordinates and for m- and z-values, if present. The spatial domain must be specified by the user when creating a geodatabase feature dataset or standalone feature class.

## **spatial join**

A type of table join operation in which fields from one layer's attribute table are appended to another layer's attribute table based on the relative locations of the features in the two layers.

## **spatial reference**

The coordinate system used to store a spatial dataset. For feature classes and feature datasets within a geodatabase, the spatial reference also includes the spatial domain.

## **split policy**

All attribute domains in geodatabases have a split policy associated with them. When a feature is split into two new features in ArcMap, the split policies dictate what happens to the value of the attribute with which the domain is associated. Standard split policies are duplicate, default value, and geometry ratio.

## **SQL**

See Structured Query Language.

## **stream digitizing**

See stream mode digitizing.

## **stream mode digitizing**

A method of digitizing in which points are recorded automatically at preset intervals of either distance or time.

## **stream tolerance**

During stream digitizing, the minimum interval between vertices. Stream tolerance is measured in map units.

## **Structured Query Language (SQL)**

A syntax for defining and manipulating data from a relational database. Developed by IBM in the 1970s, SQL has become an industry standard for query languages in most relational database management systems.

## **subtype**

In geodatabases, a subset of features in a feature class or objects in a table that share the same attributes. For example, the streets in a streets feature class could be categorized into three subtypes: local streets, collector streets, and arterial streets. Creating subtypes can be more efficient than creating many feature classes or tables in a geodatabase—for example, a geodatabase with one dozen feature classes that have subtypes will perform better than a geodatabase with one hundred feature classes. Subtypes also make editing data faster and more accurate because default attribute values and domains can be set up. For example, a Local Street subtype could be created and defined so that whenever this type of street is added to the feature class, its speed limit attribute is automatically set to 35 miles per hour.

## **symbol**

A graphic representation of a geographic feature or class of features that helps identify and distinguish it from other features on a map. For example, line symbols represent arc features; marker symbols, points; shade symbols, polygons; and text symbols, annotation. Many characteristics define symbols including color, size, angle, and pattern.

## **symbolology**

The set of conventions, or rules, that defines how geographic features are represented with symbols on a map. A characteristic of a feature may influence the size, color, and shape of the symbol used.

## **synchronization version**

A data version created in a check-out geodatabase when data is checked out to that geodatabase. This version exists as a copy of the original data and represents the state of the data at the time of the check-out.

See also check-out version, master check-out version.

## **table**

A set of data elements arranged in rows and columns. Each row represents an individual entity, record, or feature, and each column represents a single field or attribute value. A table has a specified number of columns but can have any number of rows.

## **table of contents**

A list of data frames and layers on a map that shows how the data is symbolized.

## **tabular data**

Descriptive information, usually alphanumeric, that is stored in rows and columns in a database and can be linked to map features.

See also table.

## **tagged value**

Used to set additional properties of UML elements. For example, you can set the maximum number of characters in a string field by using a tagged value.

## **target layer**

In an ArcMap editing session, the layer to which edits will be applied. The target layer must be specified when creating new features and modifying existing features.

## **thematic map**

A map designed to convey information about a single topic or theme, such as population density or geology.

**tic**

A registration or geographic control point for a coverage representing a known location on the earth's surface. Tics allow all coverage features to be recorded in a common coordinate system such as Universal Transverse Mercator (UTM). Tics are used to register map sheets when they are mounted on a digitizer. They are also used to transform the coordinates of a coverage, for example, from digitizer units (inches) to the appropriate values for a coordinate system (meters for UTM).

**tiling**

An internal subsetting of a raster dataset into a manageable rectangular set, or rows and columns of pixels, typically used to process or analyze a large raster dataset without consuming vast quantities of computer memory.

**TOC**

See table of contents.

**tolerance**

The minimum or maximum variation allowed when processing or editing a geographic feature's coordinates. For example, during editing, if a second point is placed within the snapping tolerance distance of an existing point, the second point will be snapped to the existing point.

**topological association**

The spatial relationship between features that share geometry such as boundaries and vertices. When a boundary or vertex shared by two or more features is edited using the topology tools in ArcMap, the shape of each of those features is updated.

**topological feature**

A feature that supports network connectivity that is established and maintained based on geometric coincidence.

**topology**

1. In geodatabases, a set of governing rules applied to feature classes that explicitly define the spatial relationships that must exist between feature data.

2. In an ArcInfo coverage, the spatial relationships between connecting or adjacent features in a geographic data layer (for example, arcs, nodes, polygons, and points). Topological relationships are used for spatial modeling operations that do not require coordinate information.

See also arc-node topology, polygon-arc topology.

**topology error**

Violation of a topology rule detected during the topology validation process.

**topology rule**

An instruction to the geodatabase defining the permissible relationships of features within a given feature class or between features in two different feature classes.

**transaction**

1. A group of atomic data operations that comprise a complete operational task, such as inserting a row into a table.

2. A logical unit of work as defined by a user. Transactions can be data definition (create an object), data manipulation (update an object), or data read (select from an object).

**two-tier connection**

See direct connect.

**undershoot**

An arc that does not extend far enough to intersect another arc. See also dangling arc.

## **union**

A topological overlay of two polygonal spatial datasets that preserves features that fall within the spatial extent of either input dataset; that is, all features from both coverages are retained.

See also identity, intersect.

## **username**

The identification used for authentication when a user logs in to a geodatabase.

## **validate (topology)**

The process of comparing the topology rules against the features in a dataset. When you validate a topology, features that violate the rules are marked as error features. Topology validation is typically performed after the initial topology rules have been defined, after the feature classes have been modified, or if additional feature classes or rules have been added to the map topology.

## **validation rule**

A rule applied to objects in the geodatabase to ensure that their state is consistent with the system that the database is modeling. The geodatabase supports attribute, connectivity, relationship, and custom validation rules.

## **version**

In geodatabases, an alternative representation of the database that has an owner, a description, a permission (private, protected, or public), and a parent version. Versions are not affected by changes occurring in other versions of the database.

## **vertex**

One of a set of ordered x,y coordinate pairs that defines a line or polygon feature.

## **virtual page**

The map page as seen in layout view in ArcMap.

## **VPF feature class**

See feature class.

## **wizard**

An interactive user interface that helps a user complete a task one step at a time. It is often implemented as a sequence of dialog boxes that the user can move through, filling in required details. A wizard is usually used for long, difficult, or complex tasks.

## **work flow**

An organization's established processes for design, construction, and maintenance of facilities.

## **work order**

One specific task that proceeds through each stage of an organization's work flow process, including design, acceptance, and construction in the field.

## **workspace**

A container for geographic data. A workspace can be a folder that contains shapefiles, an ArcInfo workspace that contains coverages, a geodatabase, or a feature dataset.

## **XMI**

See XML Metadata Interchange (XMI).

## **XML**

See extensible markup language (XML).

### **XML Metadata Interchange (XMI)**

A standard from Object Management Group (OMG) that specifies how to store a UML model in an XML file. ArcGIS can read models in XMI files.

### **XML recordset document**

An export file containing the features or records from an individual geodatabase feature class or table. Data in the file is encoded in XML and can be imported into an existing feature class or table.

### **XML workspace document**

An export file containing geodatabase feature datasets, classes, and tables. It can include the schema and data or just the schema. The schema and data in the file are encoded in XML and can be imported to a geodatabase.





# Index

## A

- Active data frame
  - defined 345
- Aerial photography 327
- Alias
  - defined 345
  - described 22–24
  - feature class 22, 45, 221
  - field 21, 22, 47
  - table 22
- Aligned dimension
  - defined 345
- AM/FM (Automated mapping/Facilities management) 199
- Analyze command 98
- Angular unit 43
- Annotation
  - and ArcCatalog 235
  - and disconnected editing 300. *See also* disconnected editing: annotation
  - CAD 233
  - classes 231
  - coverage 233, 244–245
  - creating 233–234, 236, 241–242
  - defined 345
  - described 230
  - feature-linked
    - creating 238, 240
    - in ArcMap 240
  - geometric networks 234
  - managing 234
  - performance 231, 232
  - populating 233
  - SDE 3 233
  - text symbols 232–233
  - upgrading 234
  - versioning 233, 284
  - VPF 233
- Annotation class 231
  - defined 345
- Annotation feature class. *See also* annotation: classes
  - defined 345
- Appending data 71–76
- Application programming interface (API) 309
- ArcCatalog
  - creating schema 5
  - customizing 275
  - disconnected editing 318, 319
  - getting help 17
  - schema locking 24
  - tree 9, 110
  - versioning 269, 270, 271
- ArcInfo workspace
  - defined 346
- ArcMap
  - commands 81
  - customizing 81, 143
  - default values 21
  - disconnected editing 296, 313, 318, 319
  - relationships 175
  - table of contents 240, 241
  - versioning 269, 278–279, 281
- Arc-node topology 108
  - defined 346
- ArcSDE 329. *See also* SDE: connection and disconnected editing 295, 304, 320
  - connections 7
    - adding 10
    - testing 10
  - defined 346
  - described 1
  - for coverages 62
  - registering data with the geodatabase 70
- ArcStorm 5, 62
- Area 23
- Area feature
  - in geodatabase
    - attributes 113
    - described 113
    - geometry 113
    - topology 113

Association 175, 176. *See also* relationships

Asynchronous  
defined 346

Attribute  
creating 20  
defined 346  
fields and coverages 111  
mentioned 5, 267, 283, 290  
raster 329  
relationship classes 176, 178, 184, 189  
rules 158. *See also* attribute domains  
validation rule 169. *See also* attribute domains  
mentioned 158

Attribute domains. *See also* domain  
and topology 101  
associating with a field 38, 47, 169  
associating with a subtype 170, 173  
browsing in ArcCatalog. *See also* domain:  
Domain Properties dialog box  
coded value domain  
code description 159, 162, 167  
codes 159, 162, 167  
creating 167  
described 159  
in ArcCatalog 162  
creating 165  
defined 346  
deleting 168  
described 158–161  
field type 162, 165  
mentioned 6, 21, 26, 70, 157, 222  
merge policy  
defining 166, 167  
modifying 168  
properties 162, 168  
range domain  
creating 165–166  
described 159–161  
minimum and maximum value 162, 166

Attribute domains (continued)  
split policy  
defining 166, 167  
mentioned 162

Attribute table  
defined 346

Attributes dialog box  
defined 346

AutoCAD DWG 61

Automated mapping/Facilities management  
(AM/FM) 199

Azimuth  
defined 346

## B

Band  
defined 346

Barrier 202

Behavior 1, 158, 284  
and topology 101  
defined 347

Bilinear interpolation  
defined 347

Binary numbers  
described 25

Buffer  
defined 347

## C

CAD (computer-aided design)  
CAD feature class  
defined 347  
field mapping (table) 62  
importing 59–62, 63, 71–76, 77, 81  
defined 347

Cardinality 176–179

CASE tools  
mentioned 2, 6

Centroid  
defined 347

Check-in  
defined 347

Check-out  
defined 347

Check-out geodatabase  
defined 347

Check-out version  
defined 347

Circular arc  
defined 347

Clip  
defined 347

Cluster tolerance  
defined 347

Clustering  
defined 347  
topology validation 103

Coincident  
defined 348

Colormap  
defined 348

Column  
defined 348

COM (Component Object Model) 24

Compaction. *See* compression

Complex edge 200, 217. *See also* network  
features

Complex junction. *See also* network features

Component Object Model (COM) 24

Compression  
defined 348  
lossless 330  
lossy 330

Computer-aided design (CAD). *See* CAD

Computer-Aided Software Engineering  
(CASE). *See* CASE tools

Configuration keyword 37, 46, 58, 69

Conflict 267, 281, 286–287  
and topology features  
versioned geodatabase 153

- Conflict (continued)
  - defined 348
- Connection file 7
- Connectivity
  - and disconnected editing 307
  - defined 348
- Connectivity rules
  - creating 225
  - default junctions 224
  - defined 348
  - described 224
  - edge-edge rule
    - creating 225
    - described 224
  - edge-junction rule
    - cardinality 226
    - creating 226
    - described 224
  - mentioned 6, 158, 205
- Constraints. *See also* attribute domains
  - defined 348
- Construct features
  - defined 348
- Control
  - defined 348
- Control points. *See* control
- Converting data. *See* importing data; loading data
- Coordinate system
  - and feature datasets 39
  - custom 20, 43
  - defined 349
  - defining 39, 43
  - described 20
  - geographic 43
  - projected 44
  - saving 40
- Coordinates
  - defined 349
- Copy/Paste commands 86, 88–89

- Copying data
  - geodatabase feature classes 86–87, 88–89, 95
  - geodatabase feature datasets 86–87, 88–89, 95
  - geodatabase tables 86–87, 88–89, 95
  - selected features 90, 93
- Copying schema 3, 15–16
- Coverage
  - annotation 233, 244–245
  - data model 1
    - and point features 112
  - defined 349
  - importing 59–62, 63, 71–76, 77, 81, 111
  - items
    - mapping (table) 60
  - mentioned 2, 5–6
  - node 60
  - point 60
  - polygon features
    - topological relationships 113
  - tics 60
- Cracking. *See also* topology: validation
  - defined 349
  - topology validation 103
- Cubic convolution
  - defined 349
- Current task
  - defined 349
- Custom
  - behavior. *See also* behavior
    - defined 349
  - feature. *See also* behavior
    - defined 349
  - object 4. *See also* behavior
    - defined 349
  - rules. *See also* validation rules
- Customization. *See* ArcCatalog:
  - customizing; ArcMap: customizing

## D

- Dangle
  - described 112
- Dangle length. *See* dangle tolerance
- Dangle tolerance
  - defined 349
- Dangling arc
  - defined 349
- Data 267
  - and versioned database 268
  - converters 81
  - defined 350
  - dictionary 22
  - migration
    - to create topology 111
  - model
    - mentioned 296, 300
  - quality and topology 101
  - type 25, 28. *See also* field: properties: data type
    - date 26
    - double 25
    - float 25
    - GlobalID 26
    - GUID 26
    - long integer 25
    - short integer 25
    - text 26
- Data frame
  - defined 350
- Data integrity
  - defined 350
- Data models 3
- Data source
  - defined 350
- Data type
  - defined 350
- Data view
  - defined 350

- Database 267
  - and versioning 268
  - defined 350
- Database management system (DBMS). *See* RDBMS (relational database management system)
- Dataset
  - defined 350
- Dataset precision
  - defined 350
- Date data type
  - described 26
- Datum 43
- DB2 1, 7, 13
- dBASE tables
  - importing 59–62, 67, 71–76, 77, 81
- DBMS (database management system)
  - mentioned 2. *See also* ArcSDE; RDBMS (relational database management system)
  - statistics 98
- Decimal degrees
  - defined 350
- Default junction type
  - defined 350
- Default values
  - and topology 101
  - associating with a field 47, 169
  - associating with a subtype 170
  - described 21, 158
  - mentioned 70
- Delta databases 306, 318, 325. *See also* delta file
- Delta file
  - defined 351
  - XML files 306, 318, 325
- Designing a geodatabase 3, 4, 15
- Destination class
  - and disconnected editing 309
- DGN 61
- Digitizing
  - defined 351
- Dimension construction methods
  - defined 351
- Dimension feature class
  - and ArcCatalog 251
  - creating 252, 252–253
    - by importing a style 255
    - with a custom style 254
  - defined 249, 351
  - Properties dialog box 251
- Dimension features
  - baseline 249
  - creating 247, 250
  - defined 351
  - discussed 248
  - Feature Class Properties dialog box 251
  - mentioned 247, 252
  - performance 250
  - types
    - aligned 248
    - horizontal linear 248
    - rotated linear 248
    - vertical linear 248
- Dimension styles
  - ArcCatalog 251
  - arrow
    - and text fit 250
    - display 250
  - begin symbol 250
  - creating 261–262
    - and managing 257
  - default style
    - mentioned 252
    - setting 265
  - defined 249, 351
  - deleting 266
  - dimension line
    - display 250
    - symbol 250
  - end symbol 250
  - extension line display 250
  - importing 255, 263
  - mentioned 247, 252
- Dimension styles (continued)
  - offset and overshoot 250
  - overriding 257
  - properties
    - lines and arrows 258
    - text 259
    - text and arrow fit 260
  - renaming 264
  - style ID 261
  - text display 250
- Dimensioning toolbar
  - defined 351
- Direct connect 7, 11, 12, 13
  - defined 351
- Dirty areas
  - and topology 106
  - versioned geodatabase 145
  - defined 351
- Disconnected editing
  - and ArcMap 296
  - annotation 301
  - Check In command 318
  - Check In wizard 318
    - About Checking In Data button 318
    - progress dialog box 318
    - Reconcile and Post box 318
  - Check Out wizard 313
    - About Checking Out Data button 314
    - advanced check-out options 314, 315
    - post check-out option 316
    - related data check box 315
    - reusing schema 314
    - summary report 316
  - check-in
    - pull model 306, 318
    - push model 306, 319
  - check-in procedure
    - deletion of rows 309
    - described 309
    - geometric networks 307
    - nulling of foreign keys 309
    - related data 309

- Disconnected editing (continued)
  - check-in procedure (continued)
    - topology 308
  - check-out
    - customizing 315
    - default behavior 300
    - limitations 295
    - manager 321
    - preparing 296
    - renaming 321
    - reviewing properties 321
    - schema-only 314, 315
    - spatial extent 313, 315
    - template 314
    - unregistering 321
  - check-out procedure 295
    - amount to check out 296
    - and ArcMap 313
    - automated processes 296, 300
    - behavior 313
    - dependent datasets 313
    - described 299, 304
    - included information 304
    - related data 300, 304
    - saving new map documents 317
    - topologies 299
  - check-out version 304
  - data filters
    - definition queries 296
    - intersection of data filters 296
    - selections 296
    - spatial extent 296, 300, 315
  - defined 352
  - delta databases 306, 318, 325
  - delta XML files 306, 318, 325
  - described 294
  - destination object 300
  - dirty areas 155
  - extracting data. *See also* exporting data
  - functionality 294
    - restrictions 305
  - geodatabase property dialog box 321

- Disconnected editing (continued)
  - managing check-outs 305, 321
    - exporting changes to a delta geodatabase 325
    - exporting changes to a delta XML file 325
    - in a check-out geodatabase 324
    - in a master geodatabase 321
  - master geodatabase 294
    - check-in models 306
    - independence from check-out geodatabase 305, 306
    - mentioned 304
  - reconcile. *See* Reconcile: and disconnected editing
  - synchronization version 304
  - topology errors 155
  - topology validation 155
- Distance units
  - defined 352
- Domain. *See also* attribute domains
  - defined 352
  - domain description 165
  - Domain Properties dialog box 163
  - name 165
  - type 162, 166, 167
- Double precision
  - defined 352
  - described 25
- Double-coordinate precision
  - defined 352
- Drawing Exchange Format (DXF) 61
- DWG 61
- DXF (Drawing Exchange Format) 61

## E

- Edge. *See also* topology
  - defined 352
  - described 108
- Edge element. *See also* logical network
  - defined 352

- Edge feature class
  - creating 221–222
  - mentioned 225
- Edge features. *See* network features
- Edge–edge rule
  - defined 352
- Edge–junction cardinality
  - defined 352
- Edge–junction rule
  - defined 352
- Edit cache. *See* map cache
- Edit session. *See also* editing; editing in ArcMap
  - and disconnected editing 314
  - and versioning 279, 281, 282, 286, 290
  - defined 352
- Editing 279
- Editing in ArcMap
  - and subtypes 158
  - and versioning 281
  - editing relationships
    - described 197
  - loading objects 82
  - snapping environment 85
  - Target layer 82
- Editor toolbar
  - defined 352
- Ellipse
  - defined 353
- Embedded foreign key 176. *See also* key field
- Error. *See* topology: error features; topology: error summary
- Error features
  - and topology
    - versioned geodatabase 148
- Error inspector
  - mentioned 107
- Errors and exceptions
  - in topology 107
  - report 107
- ESRI Annotation Feature. *See* feature type

- ESRI Complex Edge Feature 221
- ESRI Dimension Feature 252
- ESRI Simple Edge Feature 221
- ESRI Simple Feature. *See* feature type
- ESRI Simple Junction Feature 223
- Exceptions
  - in topology
    - and versioned geodatabase 148
- Exclusive lock. *See* schema locking: exclusive lock
- Export Data command (ArcMap) 86–87, 93
- Exporting data
  - geodatabase feature classes 86–87, 88–89, 95
  - geodatabase feature datasets 88–89
  - geodatabase tables 86–87, 88–89, 95
  - selected features 86–87, 90, 93
  - selected records 86–87
- Exporting schema 15–16
- Extensible markup language (XML)
  - defined 353
- Extent 313, 315
  - defined 353
- Extract Data wizard 15–16, 86–87, 90
- Extracting data. *See* exporting data
- Extracting schema 15–16

## F

- Feature. *See also* feature class; feature type; object
  - defined 353
  - described 1
  - editing. *See also* Editor toolbar
  - mentioned 175, 229, 267, 281, 290
  - selecting 181
  - splitting and merging 160
- Feature attribute table. *See* attribute table
- Feature class
  - adding to topology 133
  - as a template 46
  - attribute domains 169

- Feature class (continued)
  - creating
    - described 20, 45
    - defined 353
    - described 20
    - geometric networks 200–203
    - mentioned 1, 5, 19, 83, 163, 168, 173, 278, 282
    - participating in topology 110
    - registering with the geodatabase 70
    - relationship classes 175
    - removing from topology 133
    - simple 20, 202, 215
    - spatial reference 20, 49–51
    - standalone 20, 45
      - creating 49–51
    - subtypes 158
    - versioned geodatabase 270
  - Feature Class To Feature Class tool 59
  - Feature Class To Geodatabase tool
    - 59, 63, 65, 68
  - Feature classes
    - annotation 230
  - Feature dataset
    - creating 39
    - defined 353
    - described 20, 23
    - geometric networks 204, 205
    - mentioned 5, 19, 45, 114, 211, 215, 221–222, 253, 254, 256
    - modeling 23
    - relationship classes 180
    - topology 102, 110
    - versioned geodatabase 270
  - Feature type 221
    - custom 221
    - ESRI Complex Edge Feature 221
    - ESRI Dimension Feature 252
    - ESRI Simple Edge Feature 221
    - ESRI Simple Junction Feature 223
    - network. *See also* network features

- Feature type (continued)
  - simple 20, 45, 215, 221
  - topological 20. *See also* topology
- Feature-linked annotation 230. *See also* annotation
  - disconnected editing 310
- Field
  - attribute domains 38, 168, 169
  - creating 36
  - defined 353
  - deleting 37
  - described 21
  - indexes 52
  - loading data 77, 81
  - mentioned 19, 283
  - properties
    - data type 36, 158, 189
    - default value 38
    - name 189
    - precision 21
    - scale 21
  - related 181, 197
  - required 22, 202, 222
    - for versioning 270
  - type
    - choosing 21, 25, 28

- Float
  - described 25
- Foreign key 189, 190
- Fuzzy tolerance
  - defined 353

## G

- GDB. *See* geodatabase
- Geodatabase
  - and versions 267
  - creating 9
    - from an existing design 3
    - from scratch 4
  - data model 1, 3, 157
  - defined 354

- Geodatabase (continued)
  - defined 354
  - designing 3, 4, 15
  - disconnected
    - editing 8
  - in ArcCatalog 7, 8
  - schema
    - copying 3, 15–16
    - designing 3, 4
    - downloading from ESRI 3
  - three ways to create 4
  - upgrading 35
- Geographic database 157. *See also* geodatabase
- Geometric network
  - adding feature classes 221
  - build errors 209
  - Build Geometric Network Wizard 215
  - building 205, 208, 215
  - connectivity rules 225
  - creating 205, 211
  - defined 23, 354
  - deleting 130, 227
  - described 200
  - editing 8
  - flow direction 201
  - in ArcCatalog 204
  - loading data 71
  - managing 130, 227
  - mentioned 4, 20–24, 70, 199
  - modeling 202
  - orphan junctions 201–203
  - performance 202
  - properties 224, 225
  - renaming 130
  - snapping models 206
  - sources and sinks 201–203, 218
    - mentioned 222
  - versioning 284, 285
- Geometric relationships 101
- Geometry
  - area and length 22

- Geometry (continued)
  - field 45, 48
  - mentioned 20, 229, 283
  - tracking properties 22
  - type
    - setting 48
- Geoprocessing
  - environment 63
  - model 59
- Georelational data model
  - defined 354
- GlobalID data type
  - described 26
- GUID data type
  - described 26

## H

- Help
  - finding answers to questions 17

## I

- Identity
  - defined 354
- Imagery
  - satellite 327
  - scanned 327
- Importing data
  - annotation 233, 244–245
  - ArcSDE data 70
  - ArcStorm data 62
  - CAD files 59–62, 63, 71–76, 77, 81
  - coverages 59–62, 63, 71–76, 77, 81
  - dBASE tables 59–62, 67, 71–76, 77, 81
  - feature classes 59–62
  - geodatabase feature classes 59–62, 63, 71–76, 77, 81, 86–87, 88–89, 95
  - geodatabase feature datasets 86, 87, 88–89, 95

- Importing data (continued)
  - geodatabase tables 59–62, 67, 71–76, 77, 81, 86–87, 88–89, 95
  - geoprocessing environment 63
  - geoprocessing model 59
  - in batch 59
  - INFO tables 59–62, 67, 71–76, 77, 81
  - Map LIBRARIAN data 62
  - overview 57
  - raster 329
  - shapefiles 59–62, 63, 71–76, 77, 81
  - tables 59–62, 67
  - TIGER files 57

## Index

- attribute
  - ascending 52
  - creating 52
  - deleting 53
  - described 52–53
  - unique 52
- defined 354
- spatial. *See also* spatial index

## INFO tables

- importing 59–62, 67, 71–76, 77, 81

## Informix 7, 13

## Instance

- defined 354

## Intersect

- defined 354

## Invalid features 224

## Invalid objects. *See* invalid features

## IP address 10, 12

- defined 354

## Item

- defined 354

## J

- Junction element
  - defined 354
- Junction feature class. *See also* geometric network

Junction feature class (continued)  
  and connectivity rules 225, 226  
  and disconnected editing 307  
  and versioning 284  
  creating 221, 223  
  described 201

Junction features. *See* network features

**K**

Key field 176, 184, 188, 285. *See also*  
  Foreign key; Primary key

**L**

Label point  
  defined 355  
  migrating into a geodatabase 111

Labels  
  converting to annotation 241–242  
  using related objects 197

Layer  
  defined 355

Layout view  
  defined 355

Left–right topology  
  defined 355

Line 22

Line feature  
  attributes  
    in geodatabase 112  
  geometry  
    in geodatabase 112  
  topological relationships 112  
  topology  
    in geodatabase 112

Linear  
  unit 44

Linear dimension  
  defined 355

Load Data command (ArcCatalog) 71–76, 77

Load Objects command  
  adding to ArcMap 81  
  loading data 82

Load Objects command (ArcMap) 71–76, 81

Loading data. *See also* importing data  
  mentioned 5  
  network feature classes 71  
  Object Loader 71–76, 81, 82  
  relationships 79  
  Simple Data Loader 71–76, 77

Local area network (LAN)  
  mentioned 293

Logical network 200. *See also* geometric  
  network  
  defined 355

Long integer  
  described 25

Lossless compression  
  defined 355

Lossy compression  
  defined 355

**M**

Map 240, 278. *See also* ArcMap  
  defined 356  
  document  
    and disconnected editing 317

Map cache  
  and geometric networks 203  
  defined 356

Map document  
  defined 356

Map feature. *See* feature

Map LIBRARIAN 62

Map projection. *See* projection

Map topology  
  defined 356

Map units  
  defined 356

Master check-out version  
  defined 356

Master geodatabase. *See also* disconnected  
  editing  
  and topology 155  
  defined 356

Merge policy. *See also* attribute domains  
  default value 161  
  defined 356  
  described 160–161  
  geometry weighted 161  
  sum values 161

MicroStation DGN 61

Migrating data  
  to create topology 111  
  to the geodatabase. *See also* importing data

Minimum bounding rectangle  
  defined 356

Mobile users. *See* disconnected editing

Mosaic 329  
  defined 356  
  raster 328

Moving data. *See* copying data

Multipart feature  
  defined 357

Multipoint 23

Multipoint feature  
  defined 357

Multiuser  
  database 24

Multiuser geodatabase  
  defined 357

Multiversioned. *See* version

**N**

Nearest-neighbor resampling  
  defined 357

Network classes  
  disconnected editing 299

Network connectivity 199, 200, 215

Network elements 201



Network feature class. *See also* edge feature class; junction feature class  
creating 215, 221  
managing 227

Network features. *See also* geometric network  
ancillary role 201  
complex 1, 200  
creating 23  
described 200, 200–203  
edges 200, 205, 215, 224  
editing in ArcMap 85  
enabled and disabled 202  
feature class. *See also* edge feature class  
junctions 200, 205, 224  
mentioned 221  
orphan junctions 201–203  
simple 200  
sources and sinks 221, 223  
versioning 284

Network trace  
defined 357

Network weights  
associating with a field 222  
creating 213, 218  
described 201–203  
mentioned 205

Node  
defined 357  
described 108  
topology 109

Null values  
defined 357  
in attribute domains 159  
mentioned 21, 47

**O**

Object  
and relationships 20, 176, 182, 191  
custom. *See also* custom: object  
defined 357

Object (continued)  
described 1  
invalid 158  
mentioned 4, 5, 175  
simple 1  
valid 158

Object class 4, 20  
defined 357  
described 20

Object Loader 71–76, 81. *See also* loading data

ObjectID 21, 38

Object-oriented 1

Oracle 1, 7, 11

Orphan junction feature class 201–203

Overshoot  
defined 357

## P

Pan  
defined 357

Password  
defined 358

Perimeter 23

Permissions. *See also* privileges  
and disconnected editing 306

Personal geodatabase 9, 163, 329  
defined 358  
raster 329

Pixel 328  
defined 358  
type. *See also* data type

Planarize  
defined 358

Point 23  
defined 358

Point digitizing. *See* point mode digitizing

Point events  
and shared geometry 109

Point feature. *See also* point attributes  
in geodatabase 112  
used when creating polygons 111  
geometry  
in geodatabase 112  
topology  
in geodatabase 112

Point mode digitizing  
defined 358

Polygon 22  
creating from lines 141  
defined 358  
Feature Class From Lines tool 141  
topology 108

Polygon-arc topology  
defined 358

Port number  
defined 358

Post 281, 286–287, 290  
and disconnected editing 306  
and versioned geodatabase 268  
defined 358

Precision. *See also* field: properties: precision dataset. *See also* dataset precision  
described 20  
spatial domain 41

Preliminary topology  
defined 359

Primary key 184, 188, 190

Prime meridian 43

Privileges  
and versioning  
described 272, 281, 291  
private 272  
protected 272  
public 272  
delete 55  
described 55  
granting 19, 55  
insert 55  
mentioned 159

- Privileges (continued)
  - revoking 19, 55
  - select 55
  - update 55
- Projection 4, 44
  - defined 359
- Property
  - defined 359
- Pseudonode
  - defined 359
  - described 108, 112
- Pull check-in
  - defined 359
- Pull model. *See* disconnected editing: master geodatabase: check in models
- Push check-in
  - defined 359
- Push model. *See* disconnected editing: master geodatabase: check in models
- Pyramid 330
  - defined 359

## Q

- Query 81
  - defined 359
- Query Builder 79, 84

## R

- Radius
  - defined 359
- Ranks
  - defined 359
  - topology
    - described 102, 105
- Rasters
  - analyze 336
  - ArcSDE geodatabase 329
  - attribute 329, 340
  - bilinear interpolation 330

- Rasters (continued)
  - catalogs 5, 328, 344
    - creating 337, 338
    - defined 360
    - pyramids 339
  - check-out 343
  - compression 330, 335
  - cubic convolution 330
  - customized DLL 344
  - data compression 330
  - datasets 5, 328
    - creating 333, 334
  - defined 360
  - extraction 343
  - field raster 340
  - file 328
  - formats
    - ERDAS Imagine 341
    - ESRI Grid 341
    - TIFF 341
  - geodatabase settings 336
  - importing 329, 332, 344
  - index 330
  - JPEG compression 335
  - loading 332, 333, 334, 336, 337, 339, 344
  - lossless 330
  - lossy 330
  - managed 329, 337
  - mosaic 328, 329, 342
  - nearest neighbor 330
  - personal geodatabase 329
  - pixel 327
  - pyramids 330, 339, 344
  - scale of use 344
  - snapping. *See also* snapping
  - storage 329
    - ArcSDE geodatabase 329
    - personal geodatabase 329
  - tile 330
    - size 329
  - time series 328
  - versioning 8

- Rasters (continued)
  - x,y domain 337, 338
- RDBMS (relational database management system) 1, 22, 52, 328. *See also* ArcSDE; DBMS (database management system): mentioned
  - defined 360
- Reconcile 281, 286–287, 290
  - and disconnected editing 306, 318
  - defined 360
- Record
  - defined 360
- Reference
  - scale 253
    - for annotation 237
- Reflectance
  - defined 360
- Region topology 109
- Registering data 70
- Relate
  - defined 360
- Related objects. *See also* relationship class and disconnected editing 309, 311, 317
- Relational database management system. *See* RDBMS
- Relational join
  - defined 360
- Relationship
  - defined 360
- Relationship class
  - annotation 235, 239
  - attributed
    - described 177–179
  - cardinality 189, 191
  - composite 177–179, 189
  - creating
    - attributed 189
    - composite 186
    - simple 182
  - defined 360
  - deleting 193
  - described 24, 175, 176–179

Relationship class (continued)  
 destination class  
     176, 178, 182, 186, 189, 190, 191  
 disconnected editing 309  
 foreign key. *See also* foreign key  
 in ArcCatalog 180, 182  
 in ArcMap 181, 194, 197  
 managing 193  
 mentioned 20, 24, 157  
 messaging 177, 183, 186  
 modeling 178  
 origin class 178, 182, 186, 189, 190, 191  
 path labels  
     backward path label 183, 186  
     described 177–179  
     editing in ArcMap 194  
     forward path label 183, 186  
 performance 178  
 primary key. *See also* primary key  
 renaming 193  
 simple 177–179, 186, 189  
 versioning 282, 284  
 versus join and relate 181

Relationship rules  
 cardinality 178, 191, 192  
 creating 191  
 described 178–179  
 mentioned 158, 180

Relationships  
 and annotation 230  
 and disconnected editing 300  
 and topology 101  
 and versioning 285  
 creating 181  
 deleting 181  
 described 176–179  
 destination object 192  
 mentioned 1, 4, 6, 70, 99, 175, 199  
 origin object 192  
 related object 284

Remote geodatabase. *See* disconnected editing; SDE: geodatabase

Resampling  
 defined 361

Resolution  
 defined 361

Route topology 108

Row 175, 176. *See also* object  
 defined 361

Rules  
 and topology 104, 124  
 topology  
     mentioned 101

**S**

Satellite imagery 327

Scale. *See* field: properties: scale

Scanned images 327

Scanning  
 defined 361

Schema  
 copying 3, 15–16  
 creating 5  
 defined 361  
 designing 3, 4, 15, 17, 19, 159  
 downloading from ESRI 3  
 exporting 15–16  
 geometric networks 216  
 loading data 77, 81  
 mentioned 2  
 modifying 24

Schema locking  
 and dimension styles 257  
 and geometric networks 130, 210, 227  
 and relationship classes 179  
 and subtypes 161  
 and topology 103  
 described 24  
 exclusive lock 103, 130, 179, 210, 257  
     described 24  
 shared lock 24, 103, 130, 179, 210

Schema-only check-out  
 defined 361

SDE. *See also* ArcSDE  
 connection 7, 10, 11, 12, 13, 163  
 for coverages. *See also* ArcSDE  
 geodatabase 7, 9, 37, 46  
 server 10, 12  
 service 10

Segment  
 defined 361

Select  
 defined 361

Selectable layers  
 defined 361

Selected set  
 defined 361

Selection 240

Selection anchor  
 defined 361

Server  
 defined 361

Service  
 defined 361

Shape  
 defined 361

Shape\_Area 23

Shapefile  
 as a template 20  
 defined 362  
 field mapping (table) 61  
 importing 59–62, 63, 71–76, 77, 81  
 mentioned 2, 5–6, 130, 227  
 polygons  
     and topology rules 113

Shape\_Length 23

Shared boundary  
 defined 362

Shared vertex  
 defined 362

Short integer  
 described 25

Simple Data Loader 71–76, 77

Simple edge. *See also* network features  
 and junction 200

- Simple feature
  - defined 362
- Single precision
  - defined 362
- Single-coordinate precision. *See* single precision
- Sketch
  - defined 362
- Sketch constraint
  - defined 362
- Sketch operation
  - defined 362
- Snapping
  - defined 362
  - features 217
  - snap tolerance 218
- Snapping environment
  - defined 362
- Snapping priority
  - defined 362
- Snapping properties
  - defined 363
- Snapping tolerance
  - defined 363
- Spaghetti data
  - defined 363
- Spatial database
  - defined 363
  - engine (SDE) 306
- Spatial domain
  - defined 363
  - described 20
- Spatial extent 313, 315. *See also* extent
- Spatial index
  - creating 52–53, 54
  - deleting 54
  - described 20
  - grid size 48, 54, 59–62
- Spatial join
  - defined 363
- Spatial reference
  - and feature classes 45
  - and feature datasets 39

- Spatial reference (continued)
  - defined 363
  - described 20
  - importing 39–40
  - precision 20, 41
  - selecting 39–40
  - spatial domain 39, 40
- Spatial relationships. *See* topology: rules
- Split policy. *See also* attribute domains
  - default value 160
  - defined 363
  - described 160–161
  - duplicate 160
  - geometry ratio 160
- Splitting and merging features. *See* merge
  - policy; split policy
- SQL (Structured Query Language)
  - defined 363
  - Server 7, 12
- Standard annotation 230
- Storing rasters 329
- Stream digitizing. *See* stream mode digitizing
- Stream mode digitizing
  - defined 363
- Stream tolerance
  - defined 363
- Subtypes
  - attribute domains 168, 169
  - connectivity rules 224, 225
  - creating 170
  - data loading 77
  - default subtype 171
  - defined 364
  - deleting 173
  - described 158–161
  - mentioned 4, 6, 20
  - modifying 170, 173
  - relationship rules 178, 191
  - subtype
    - code 158, 171
    - description 171, 173
    - field 158, 170

- Subtypes (continued)
  - topology 101
  - rules 129
- Symbol
  - annotation 232–233
  - defined 364
- Symbology 181
  - defined 364
  - using related objects 197
- Synchronization version
  - defined 364

## T

- Table
  - attribute domains 168, 169
  - creating 36
    - from template 37
    - simple objects 36
  - dBASE 5–6, 61
  - defined 364
  - designer 36
  - in UML 130, 227
  - INFO 5–6
  - mentioned 5, 19, 20, 175, 278
  - relationship classes 176, 176–179, 180, 182, 183, 189
  - subtypes 158, 163, 173
  - versioned geodatabase 270
- Table of contents
  - defined 364
  - disconnected editing 314
- Table to Geodatabase tool 67
- Table to Table tool 67
- Tabular data
  - defined 364
- Tagged value
  - defined 364
- Target layer
  - defined 364
- TCP/IP 10

- Text data type
  - described 26
- Text symbols
  - annotation 232–233
- Thematic map
  - defined 364
- Tic
  - defined 365
- Tile 330
- Tiling
  - defined 365
- Time series
  - raster 328
- TOC. *See* table of contents
- Tolerance
  - defined 365
- Topological association
  - defined 365
- Topological feature
  - defined 365
- Topology
  - adding feature classes 114
    - described 119
  - and ArcCatalog 110
  - and ArcSDE 118
  - arc–node 108
  - basics 104
  - building 102
    - resources needed 102
  - cluster tolerance
    - changing 115, 132
    - described 102, 104, 115
    - ranks 102, 105, 116, 134
  - conflicts
    - and versioned geodatabase 153
  - creating 102, 114
    - assigning a name 115
    - in a versioned geodatabase 143
    - mentioned 101
    - setting cluster tolerance 115
  - dangles
    - described 112
- Topology (continued)
  - defined 365
  - described 23, 99, 101
  - dirty areas
    - and versioned geodatabase 145
  - disconnected editing 155, 299, 308
  - error features
    - and versioned geodatabase 148
  - error summary 140
  - exceptions
    - and versioned geodatabase 148
  - feature dataset 23
  - feature geometry 108
  - geometries involved 108
  - geometry
    - edge 108
    - node 108
    - pseudonode 108, 112
  - how to use 101
  - integrated features 281
  - managing 110
  - mentioned 20, 99
  - migrating data 111
    - area features 113
    - coverages 111
    - line features 112
    - point features 111
  - modifying 131
    - adding a feature class 133
    - adding a rule 135
    - changing cluster tolerance 132
    - changing number of ranks 134
    - changing rank of feature class 134
    - getting properties 131
    - removing a feature class 133
    - removing a rule 136
    - renaming 131
  - network topology 130, 200, 205. *See also*
    - network connectivity
  - New Topology wizard 114
  - node 109
  - permissions 143
- Topology (continued)
  - point and line features 109
  - point events 109
  - polygon 108
  - region 109
  - route 108
  - rules 104, 116, 124
    - assigning a feature class 117
    - coverage arc features 112
    - defined 365
    - errors and exceptions 107
    - exceptions 128
    - getting a description 136
    - line rules 125
    - loading a rule set 138
    - mentioned 5
    - point rules 126
    - polygon rules 124
    - polyline features 112
    - Rule Description panel 117
    - saving a rule set 117, 137
    - setting 116
  - sharing geometry 108–109
  - storage 99
  - subtypes 129
  - validation
    - clustering features 103
    - cracking features 103
    - dirty areas 106
    - mentioned 101, 110
    - process 123
    - validating a new topology 118
  - versioned databases 143
    - theory 145
  - view 110
  - wizard
    - mentioned 102, 110
- Topology error
  - defined 365
- Transaction 281, 290–291
  - defined 365
- Transportation networks 199

Two-tier connection. *See* direct connect

## U

UML

mentioned 2, 6

Undershoot

defined 365

Union

defined 366

Upgrading

geodatabase 35

Username

defined 366

## V

Validate (topology)

defined 366

Validation rules. *See also* attribute

domains; connectivity

rules; relationship rules; topology:

validation

and importing data 70

and loading data 85

and topology 101

defined 366

described 158

mentioned 4

Vector data 1

Version

administering 271

changing properties 274

compressing database 277, 291

deleting 273

renaming 273

changing in ArcMap 279

conflict. *See also* conflict

displaying 288

feature-linked annotation 284

geometric networks 284

Version (continued)

conflict (continued)

resolving 281–285, 289

connecting to 14

creating 271

DEFAULT 9

defined 366

described 267, 268–269

descriptions 273

disconnected editing 304

editing 281–285, 286

autoreconciliation 281–282, 286

post 282, 287

reconcile 281, 287

in ArcCatalog 270

in ArcMap 278–279

permissions 272, 291

post. *See also* post

preserving edits 270

purpose 8

reconcile. *See also* reconcile

and post a topology 143

refresh 274, 279, 280

registering data 270

relationships

conflicts 285

scenarios 290–291

topology 143

conflicts 285, 289

theory 145

transaction. *See also* transaction

unregistered database 143

Versioned data 159, 193, 216

loading 82, 88

Vertex

creating new

topology validation 103

defined 366

moving

topology validation 103

Virtual page

defined 366

VPF feature class. *See* feature class

## W

Wide area network (WAN)

mentioned 293

Wizard

defined 366

Work flow. *See also* version

common stages 268

defined 366

disconnected editing 293

process 268, 290

Work order. *See also* work flow

defined 366

Workspace 278

defined 366

## X

XMI. *See* XML Metadata Interchange (XMI)

XML. *See also* extensible markup language

(XML)

delta files 306, 318, 325

exporting data to 87, 95

importing data from 87, 95

XML Metadata Interchange (XMI)

defined 367

XML recordset document

defined 367

wizard 87

XML workspace document

defined 367

wizard 87, 95

## Z

ZIP files

exporting data to 87, 95

exporting schema to 15