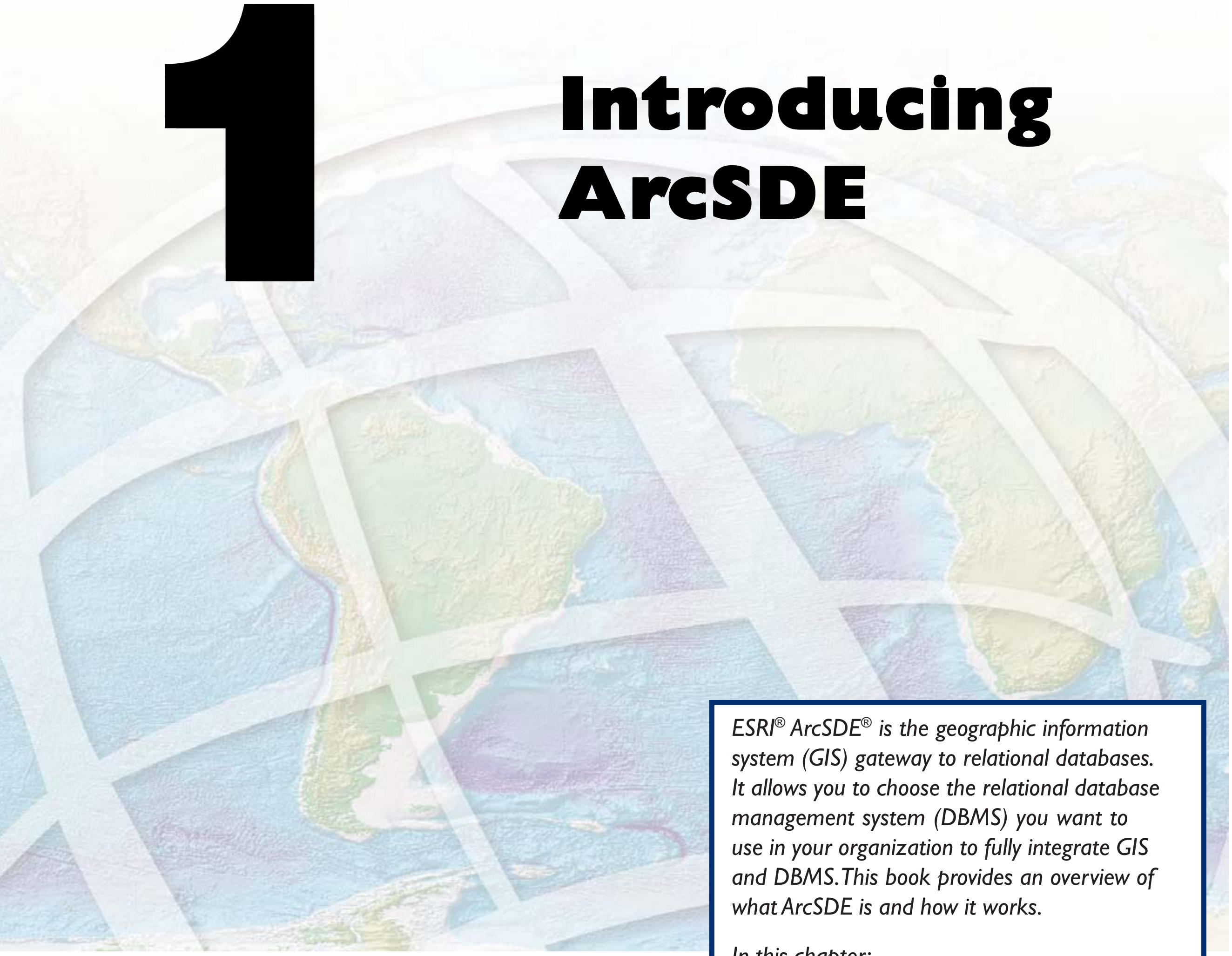
ArcGIS® 9

Understanding ArcSDE®

Contents

1 Introducing ArcSDE



ESRI® ArcSDE® is the geographic information system (GIS) gateway to relational databases. It allows you to choose the relational database management system (DBMS) you want to use in your organization to fully integrate GIS and DBMS. This book provides an overview of what ArcSDE is and how it works.

In this chapter:

- *What is ArcSDE?*
- *Why ArcSDE?*
- *Configuration options*
- *ArcSDE for developers*

WHAT IS ARCSDE?

Welcome to ArcSDE, the GIS gateway to your DBMS. ArcSDE is the software that allows you to use the following ESRI products:

- ArcGIS® suite (Desktop, Server, Engine)
- ArcIMS®
- ArcInfo™ Workstation
- ArcView®
- ArcSDE CAD Client
- MapObjects® and MapObjects—Java™

to store, use, and manage all your GIS data (including feature geometry) in one of the following commercial DBMSs:

- IBM® DB2®
- IBM Informix®
- Microsoft® SQL Server™
- Oracle®

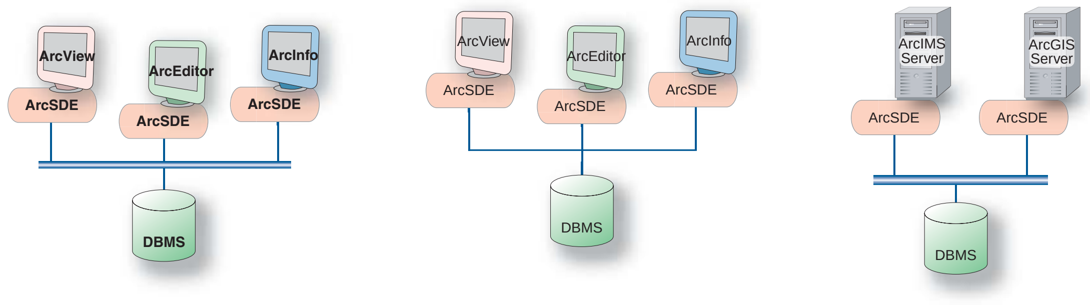
on your choice of several operating systems and scaling from work groups to large enterprise databases. It also provides other choices for developing custom applications as well as how the geometry of your features is stored.

ArcSDE is a key component in a multiuser GIS because it allows you to "marry" world-class GIS technology with world-class relational database management system technology. You get to use what a GIS does best along with what a DBMS does best. The combination provides you with:

- Support for concurrent multiuser editing with long transactions, allowing support for many editing and critical GIS data management work flows.
- No limits on the size of your spatial database.
- The ability to serve many concurrent users anywhere on the network or the Internet.
- The ability to build custom applications from the ArcSDE C and Java application programming interfaces (APIs) as well as ArcGIS, ArcObjects™, and ArcIMS.
- A number of configuration choices.

All of this multi-DBMS, multiplatform/OS, multiclient, and multiple developer choice flexibility adds up to an open, scalable solution, with more choices for users and better interoperability.

In this book, you'll get an overview of what ArcSDE is for and how it relates to the ArcGIS geodatabase, as well as a review of how ArcSDE works and information on where to go next for more in-depth information on deploying a system with ArcSDE.



ArcSDE acts as the gateway between ArcGIS and your relational database. This graphic illustrates several ways to configure ArcSDE.

A common question to ask is: Do I need ArcSDE? Answering that requires answering another question first: Do I need a DBMS?

If you are unfamiliar with DBMS technology, here is a very brief overview. Please consult with DBMS vendors to get more information on their particular product. Chapter 2 goes into a little more detail on how data is stored in a DBMS.

Why use a DBMS?

Multiuser GIS requires a DBMS and comprehensive GIS tools that work with the geographic data. As your organization evolves, your spatial databases will grow in size and number of users. Using a DBMS is a practical means for sharing and protecting your data investments.

For years, file-based formats have been used to manage and share information using coverages, shapefiles, grids, TINs, computer–aided design (CAD) drawings, and many raster formats. There were some clear advantages to doing this. First, it was easy and inexpensive. Everyone could edit and manage file-based data. No DBMS investment was necessary. Plus, DBMSs lacked the ability to manage the larger, more complex data types and support for operations required by GIS. In recent years, however, the use of DBMS in GIS has become more widespread.

As DBMS capability has evolved into more powerful technology that could support complex data objects, larger queries, and stronger transactional support, the use of DBMS for geographic data management has become much more practical. Most multiuser GIS systems today manage their GIS data in a DBMS, and many smaller sites are beginning their migration to a DBMS.

As with other information resources in an organization, GIS data can benefit from DBMS use. It makes good business sense to manage your GIS resources in a relational database. Some of the reasons to use relational databases are as valid for GIS as they are for other information systems:

- A single data store for attribute and spatial data
- Base relational model (so GIS is compliant with other information technology [IT] system requirements)
- Concurrency management in a multiuser environment
- Standard data management practices, such as backup, recovery, and replication
- Performance for any number of users
- The need for managed and organized data
- Scalable data volumes with no size limitations
- Centralized systemwide or companywide access to the data
- Data maintenance over long time periods, spanning personnel changes and hardware and software upgrades
- System failure and recovery mechanisms
- Industry-standard client/server and Internet architectures (e.g., Web services)

The right tool for the job

Responsibility for management of geographic datasets is shared between GIS software and generic relational database management system software. Certain aspects of geographic dataset management, such as disk-based storage, definition of attribute types, associative query processing, and multiuser transaction processing, are delegated to the DBMS. Some DBMS engines have been extended with support for spatial types, with associated indexing and search functionality.

The GIS application retains responsibility for defining the specific DBMS schema used to represent various geographic datasets and for domain-specific logic, which maintains the integrity and utility of the underlying records. In effect, the DBMS is used as an implementation mechanism for geographic datasets.

ArcSDE is part of the multitier architecture (application and storage), where aspects related to data storage and retrieval are implemented in the storage (DBMS) tier, while high-level data integrity and information processing functions are retained in the application and domain software (for example, ArcGIS).

ArcSDE enables the same capabilities on all DBMSs

Although all relational database vendors support SQL and process SQL queries in similar ways, there are significant differences among database vendors in the details of their database server implementation. These relate to performance and indexing, the supported data types, the integrity management tools, and the execution of complex queries. These also relate to support for spatial types in the DBMS.

Standard SQL does not support spatial data. The ISO SQL/MM Spatial and Open GIS Consortium, Inc. (OGC), simple feature SQL specifications extend SQL by defining standard SQL language for vector geometry types. DB2 and Informix support these standard SQL types. Oracle has implemented its own spatial type, while Microsoft SQL Server has no spatial type support. Spatial geometry in Microsoft SQL Server is stored as type “image”. ArcSDE provides the flexibility to leverage the capabilities that each DBMS vendor offers so client applications will work the same regardless of what spatial storage method is used.

ArcSDE ensures that full spatial functionality is available

regardless of the capabilities in the underlying DBMS. For example, if you build an ArcGIS application using ArcSDE for Oracle, that same application will work if used with ArcSDE for IBM DB2, Informix, or Microsoft SQL Server.

One of the roles of ArcSDE is to deal with the diversity and complexity in the underlying DBMS. ArcSDE is like an “adapter” for client applications to use when they want to store and manage their spatial data in a commercial DBMS. A client application is built using the ArcSDE API, and that client application, with little or no modification, can work on any one of the four DBMSs or spatial storage methods supported by the host DBMS.

Fitting GIS into an IT strategy

Many GIS users require that their GIS fits into a coherent information technology strategy for their organization. Simply put, their GIS must adhere to IT standards, the GIS data should be managed as an integral part of the organization’s valuable data holdings, the data must be secure, and access to the data must be restricted to only those users that need it. These are standard advantages of a DBMS that GIS users need. Since ArcSDE provides the means for storing and using GIS data in an DBMS with various GIS and non-GIS applications, ArcSDE plays an important role in fitting GIS into an IT strategy.

Data interoperability in ArcSDE

Data interoperability is the ability of multiple applications to use the same data. Sometimes that data is shared by the multiple applications, and sometimes interoperability tools (translators) are required. Many sites have multiple applications using the same data, so interoperability is an important issue. Other sites have multiple applications using different data types, so it’s important that applications be able to use them all.

ArcSDE also plays a role in GIS data interoperability by use of spatial types in IBM DB2, IBM Informix, and Oracle. Sites can, for example, use Oracle Spatial or Locator products as the spatial geometry storage method, allowing ESRI’s ArcSDE client applications, as well as non-ESRI applications, to use the same spatial data without making a copy of it as long as both applications follow the host DBMS’s interoperability rules for application development.

ArcSDE connection configuration options

The ArcSDE gateway can be multitier and will run as part of a GIS client application or in a centralized server. ArcSDE can be flexibly configured to connect client applications directly to a DBMS (a two-tier configuration) or to run as an application server near the DBMS (a three-tier configuration). There are advantages to each configuration, depending on your needs.

Direct connection configuration

This configuration is useful in many situations because it allows for increased scalability by offloading work from the server to each connected client. It also can be useful in many failover environments because it reduces the number of single points of failure.

Application server configuration

Currently, the most common ArcSDE configuration includes an ArcSDE Application Server. The application server allows you to serve geographic data from a large, central geodatabase on UNIX, Linux®, and Windows® servers. The ArcSDE Application Server is typically located on the same hardware platform as your DBMS, ArcIMS, or ArcGIS Server. The application server configuration can offer performance advantages; during editing, it may be faster due to the intelligent client/server data communications.

See Chapter 4, 'The ArcSDE architecture', for more information about two-tier and three-tier configurations.

ArcSDE comes with high-level APIs for querying and working with information in geodatabases. These include:

- ArcSDE Client API for C and Java developers
- COM API (ArcObjects) for use with the ArcGIS suite of products

ArcSDE Client API

ArcSDE ships with an ArcSDE Developer Kit CD–ROM. This CD–ROM has C and Java Software Development Kit (SDK) from which applications can be built. The ArcSDE Client API provides many advanced GIS functions. All ESRI client applications that work with the ArcSDE gateway (for example, ArcGIS) use either the C or Java API. ArcSDE client applications, such as ArcGIS, are large and use a great deal of the available ArcSDE API. If you are building custom applications from one of these ESRI products (for example, ArcGIS Engine), you won't need to use the C or Java API. You'll use ArcObjects with one of the ArcGIS Application Development Frameworks (ADF). Non-ESRI client applications can be built using a small portion of the API to build small, focused, mission-critical applications—for example, in emergency response and customer care.

These two ArcSDE APIs allow developers to build custom applications to work with any DBMS supported by ArcSDE. ArcSDE has a significant third-party developer community providing application solutions for many industries.

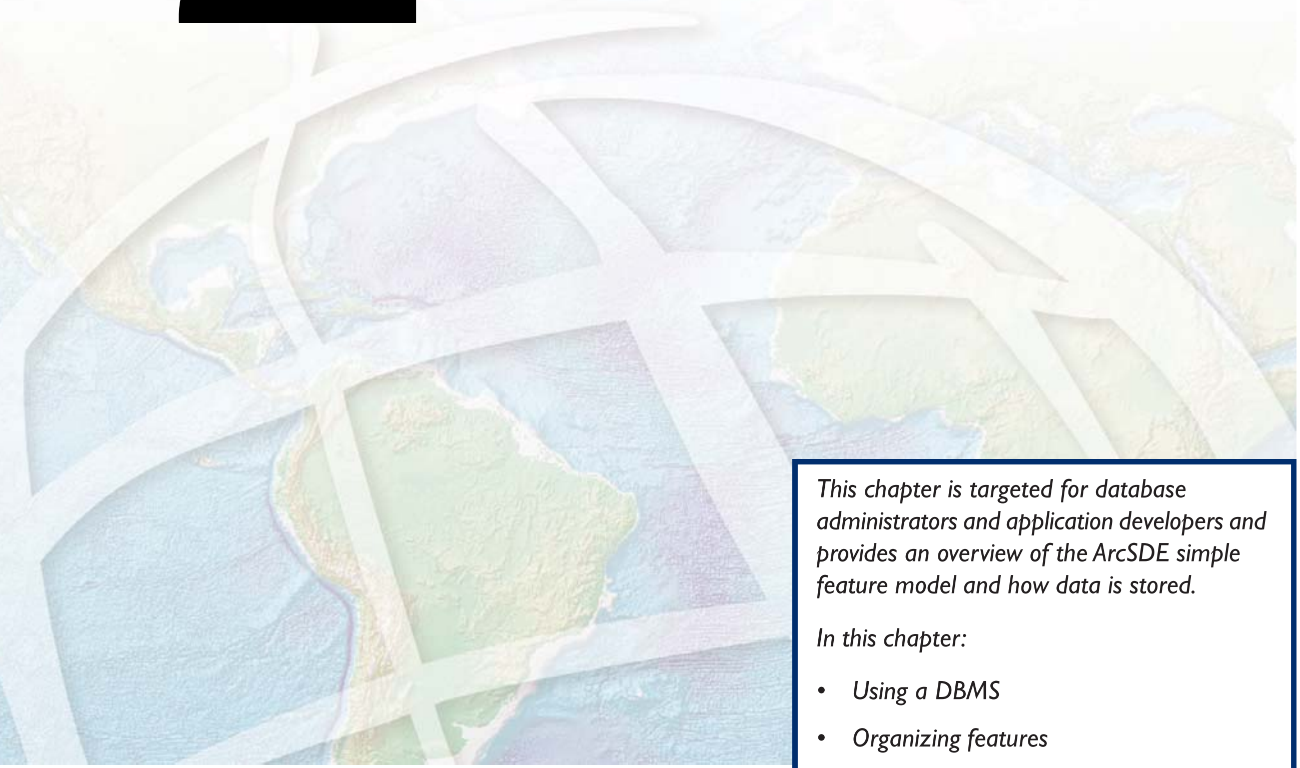
ArcObjects

If you want to build or customize an ArcGIS application, you'll use ArcObjects. ArcObjects is the COM developer's API for ArcGIS Desktop, Server, and Engine. It provides the ability to access and work with the contents of ArcGIS geodatabases as objects with advanced GIS behavior and relationships. A Java ADF and a .NET ADF are available for use with ArcObjects. Refer to ArcGIS documentation for more information on ArcObjects development.

SQL

There are SQL APIs available for working with spatial data. The SQL interface provided by your DBMS can be used to work with the contents of spatial databases. Since spatial databases use standard DBMS columns, the DBMS's SQL API is used. For a spatially enabled DBMS (for example, IBM DB2, IBM Informix, and Oracle Spatial/Locator), a SQL API is available to perform spatial queries directly in the database. This allows SQL query access to feature geometry using DBMS applications. See your host DBMS documentation for SQL spatial query syntax.

2 Data storage



This chapter is targeted for database administrators and application developers and provides an overview of the ArcSDE simple feature model and how data is stored.

In this chapter:

This chapter reviews the basics of the ArcSDE simple feature model and how the data is physically stored in the DBMS. This chapter is an important building block for understanding the overview of the ArcGIS geodatabase in Chapter 3, 'The ArcGIS geodatabase'.

The first thing you should know is that data is never stored *in* ArcSDE. Data is stored in tables in a DBMS. ArcSDE is the tool that allows you to use that data with GIS applications.

Second, all data is stored in standard DBMS tables using data types available for the host database. ArcSDE uses and complements the base DBMS capability by adding a spatial component to the database. A key part of using ArcSDE is tuning the host DBMS. There can be many performance variables, but ArcSDE performance hinges largely on how well the host database is tuned.

Key DBMS concepts

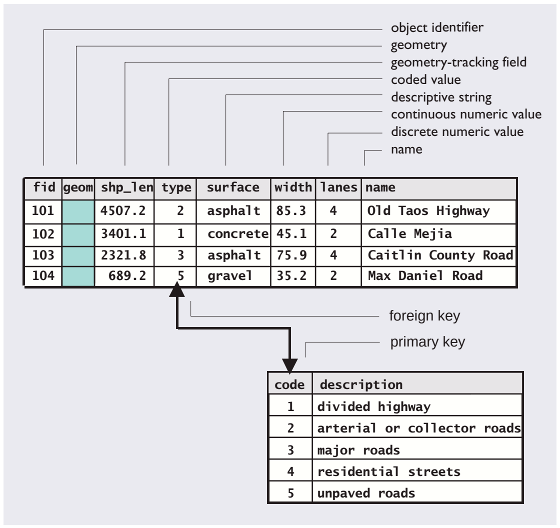
In a relational database management system model, data is stored in tables consisting of rows and columns. The cell defined by the intersection of a row and column is called a field; the data contained in the field is a value. A row represents a particular occurrence, or instance, of a feature, while the columns contain the attributes of the feature, such as owner name for a parcel. Attributes can have many types, such as dates, text strings, or numbers. A geometric shape of a feature is another type of value, stored in a column that defines an abstract geometric data type.

SQL provides an interface to relational tables that allows you to select rows based on the values contained in the fields. A SQL statement can range from simple to complex, allowing you to compose virtually any type of query from basic column types.

A query may return columns from any number of tables by joining the tables together on key columns. A primary key (one or more columns) uniquely identifies rows in a table. The same column or columns, duplicated in another table, is called the foreign key. These keys allow tables to be joined.

The result of a query is a set of rows meeting the criteria established by the SQL statement. This set is called a cursor. An application can reference a number of active cursors. The application proceeds through a cursor, looking at each individual row. As each successive row is requested, the appropriate values of each field are made available to the application.

ArcSDE extends SQL by providing tools to work with spatial data. You can also use standard SQL queries in the ArcSDE API to perform attribute-only queries.



fid	geom	shp_len	type	surface	width	lanes	name
101	🟦	4507.2	2	asphalt	85.3	4	Old Taos Highway
102	🟦	3401.1	1	concrete	45.1	2	Calle Mejia
103	🟦	2321.8	3	asphalt	75.9	4	Caitlin County Road
104	🟦	689.2	5	gravel	35.2	2	Max Daniel Road

code	description
1	divided highway
2	arterial or collector roads
3	major roads
4	residential streets
5	unpaved roads

Organizing geographic features

ArcSDE organizes features in layers or *feature classes*. A feature class is a collection of one or more features of one geometric type. A feature is a geometric representation of a spatial object (for example, a road is represented in the database as a line feature), defined as a sequence of one or more x,y coordinates and the attributes for that geometry. Features are stored so that one row in a table equals one feature.

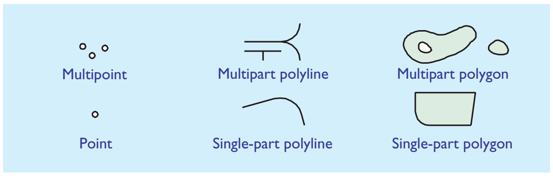
Logical elements	Database elements
Object	Row
Attribute	Column, Field
Class	Table

Users may logically think of a feature class as one single table. However, ArcSDE implements a feature class as one or more tables, depending on the DBMS, the column type used for storing the geometry, and whether or not the data will be used in long transactions.

ArcSDE doesn't change existing DBMSs or affect current applications. It simply adds a spatial column to tables and provides tools (an API) for a client application, such as ArcMap™, to manage and access the geometry data referenced by that column.

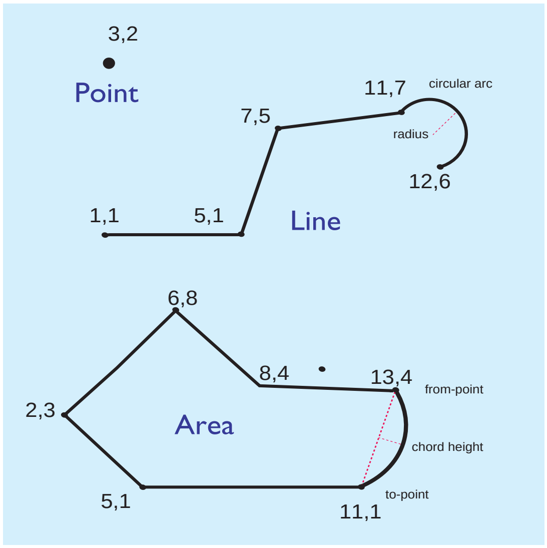
When you add a spatial column to a table (sometimes referred to as a business table), you spatially enable it. The ArcSDE software manages spatially enabled tables by storing information, such as the name of a feature class; its owner; x,y extent; type of geometry allowed in the layer; and many other pieces of information in ArcSDE metatables. These ArcSDE metatables are stored in the host DBMS and are created when ArcSDE is installed. They are owned, managed, and populated exclusively by ArcSDE. The schema of these tables is available in the ArcSDE Developer Help (located on the ArcSDE Developer Kit CD–ROM). Under no circumstances should any ArcSDE metatable be directly modified with SQL. These tables are managed totally by ArcSDE.

This chapter has discussed features in a table. So what geometry types are supported with ArcSDE? The following graphic illustrates the feature geometries you may store in your DBMS with ArcSDE.



Basic feature storage

ArcSDE stores geometric shapes as x,y coordinates and true curves. Points are recorded as a single x,y coordinate, lines as a series of ordered x,y coordinates and curves, and areas as sets of lines composed of x,y coordinates and curves that have the same starting and ending point.



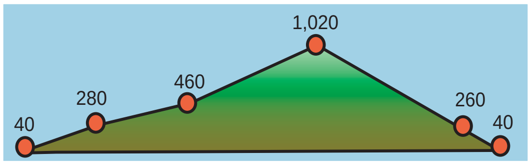
ArcSDE stores a list of x,y coordinates that define the location and shape of each geographic feature.

Every geometry type in ArcSDE has a set of strict verification rules that determine whether a geometry is geometrically correct before it is stored. Verification rules for each shape type are described in the ArcSDE Developer Help (available on the ArcSDE Developer Kit CD–ROM).

These simple features that ArcSDE stores in a DBMS are the building blocks on which the ArcGIS geodatabase is built.

Z-values

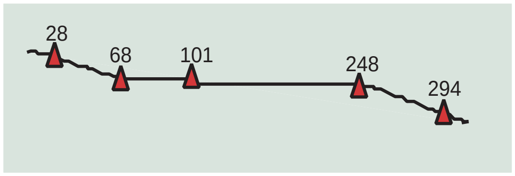
ArcSDE allows you to add z-values to the x,y coordinates. Z-values can represent height or depth. Geometry may be either two-dimensional (x,y) or three-dimensional (x,y,z).



Z-values, such as elevations or depths, can be stored in a feature's geometry. In this case, points on a mountain have z-values containing their elevation, in addition to their x,y position.

Measures

Measures represent a distance, time, address, or some other event at given points along a feature. You can add measure values (m-values) to any geometry type. Identifying an event by road name and distance from a known location (for example, Highway 20, kilometer 38) is a common method for locating highway information, such as sign or accident locations, exit ramps, pavement quality, and speed zones.



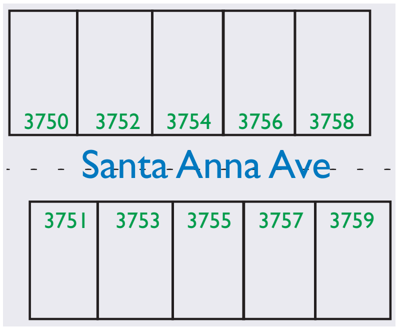
Measures on a simple line shape mark traffic accidents on a highway.

Measure values are independent of a geometry's coordinate system. The x,y coordinate of a point could be (529482, 5109382) with a measure value of 248. Although many applications use measures to represent increasing linear distances along a line (see the graphic above), measure values can arbitrarily increase, remain constant, or decrease.

Like z-values, geometry can contain a measure value (x,y,m). You could also have four-dimensional geometry by adding z- and m-values to the geometry (x,y,z,m).

ArcSDE annotation

Annotation is text that labels features for cartographic display. It helps identify places and features. For example, a road map can have labels that identify the names of the roads, the distances between intersections, and so on.



The street name and the addresses are attached to the street and parcel features, respectively.

Annotation is a feature attribute. ArcSDE stores annotation as a feature attribute to ensure a direct link between the text and the feature it labels. Annotation properties can be stored in a single binary large object (BLOB) column as part of the feature table, or they can be stored in one or more related tables. See the ArcSDE Developer Help for more details on annotation.

Annotation is usually thought of as text attached to a feature or coordinates on a map. Maps often include other text to label nongeographic map components, such as key legends, map titles, North arrows, and scalebars. This text has no geographic coordinates and isn't stored by ArcSDE.

ArcSDE annotation is not the same as ArcGIS geodatabase annotation. Refer to *Building a Geodatabase* to create and maintain geodatabase annotation. ArcGIS Desktop applications can view ArcSDE annotation but cannot edit it. If you need to edit your annotation with ArcEditor™, for example, you will need to convert your ArcSDE annotation to the ArcGIS geodatabase annotation format.

CAD data

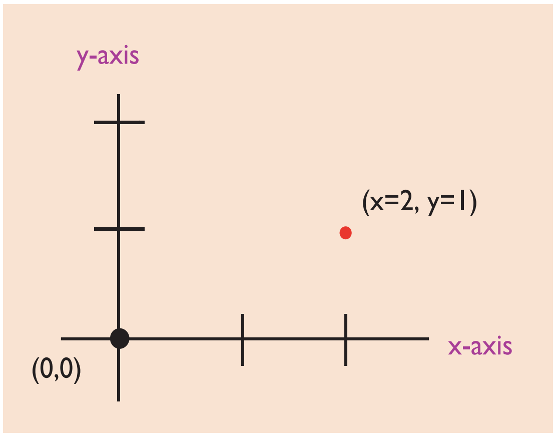
ArcSDE layers can also contain CAD entities, with the aid of ArcSDE CAD Client. When a CAD entity is stored in a layer, an ArcSDE feature representing the entity is also stored. Only ArcSDE CAD Client can work with CAD entities, but other ArcSDE client applications can use the corresponding features. Refer to 'Using ArcSDE CAD Client' (usingcadclient.pdf in the documentation folder of the ArcSDE Developer Kit CD–ROM) for more information.

XML data

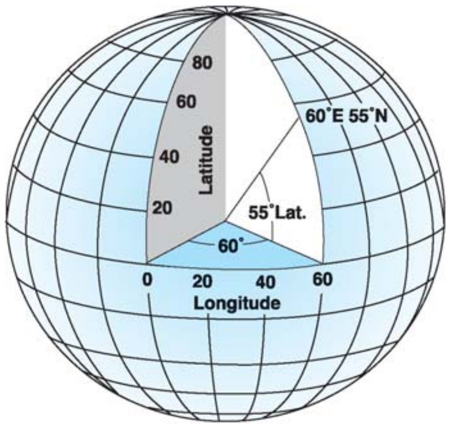
Much like ArcSDE extends traditional DBMS functionality by adding support for spatial and raster columns, it also supports columns containing XML documents. Users can add a column of type SE_XML_TYPE to a table, insert XML documents into it, index the column, and search the table for XML documents that meet specific criteria. Business tables can have both a spatial column and an XML column, so XML documents may be associated with geometric shapes. Searches can be executed using both columns. For example, a user can search for all features in a given area that have a keyword of "precipitation" in the XML document. The Metadata Server extension to ArcIMS uses this XML type. ArcGIS does not yet support the ArcSDE XML type, however.

Coordinate reference

Each ArcSDE feature class contains coordinate reference information for your data. The coordinate reference includes the coordinate system and the information needed to convert from real-world coordinates to internal ArcSDE storage values. ArcSDE stores its coordinates as positive integer values internally because they take less room to store in the database and expedite calculations.



The position of spatial data is defined by its coordinate system, usually projected (planar) or geographic.



The projected coordinate system, usually measured in meters or feet, defines locations on a two-dimensional plane using two axes: the x-axis, representing east–west; and the y-axis, representing north–south. They intersect at the origin (0,0). Locations are defined relative to the origin.

Points above the x-axis or to the right of the y-axis have positive values. Points below or to the left are negative.

Measures and z-coordinates are referenced independently of the x,y coordinate system, allowing you to assign to them whatever values you want.

The geographic coordinate system defines locations on a spheroid, a three-dimensional surface. When storing geographic coordinates, longitude values correspond to x, while latitude values correspond to y.

Understanding Map Projections, one of the books in the ArcGIS documentation set, is a crucial book for anyone needing an indepth understanding of coordinate reference.

ArcSDE manages the physical storage of geometry for features using standard data types provided by the host DBMS. Some DBMSs have spatial data types, while others provide standard binary or BLOB storage types.

ArcSDE geometry storage depends on the DBMS you use. The options for each DBMS are:

- ArcSDE compressed binary. This is stored as "long raw" or "BLOB" in Oracle and as type "image" in SQL Server.
- Spatial types. A spatial type embeds support for GIS feature geometry into the DBMS kernel. Some DBMSs that support spatial types comply with the Open GIS SQL specification for user-defined types (UDTs) and the ISO SQL Multimedia Spatial Standard. These standards define columns capable of storing spatial data, such as the location of a landmark, a street, or a parcel of land. Use of these spatial types integrates geometry and nonspatial attributes, providing a single point of access inside the DBMS through a SQL API. IBM (via DB2 and Informix) and Oracle support spatial types.
- OGC Well Known Binary. This is stored as "long raw" in Oracle and as type "image" in SQL Server.

ArcSDE client applications see the data as feature layers, regardless of the geometry storage type. In the case of Oracle, you have the option to choose the storage methods for any feature class. You may choose to store a point layer as Oracle Spatial geometry types and a polygon layer as ArcSDE compressed binary. The decision on how to store your geometry should be based on the DBMS you use and the requirements specific to your implementation.

Details on these geometry storage types and how you can define the storage type before loading data can be found in the respective configuration and tuning guides for each DBMS. Check the ESRI Library CD–ROM for the form ArcSDE_Config_GD_<your dbms>.pdf.

RDBMS	GEOMETRY STORAGE	COLUMN TYPE
Oracle	ArcSDE Compressed Binary	Long Raw or BLOB
	Oracle Spatial / Locator - Geometry Type	SDO_Geometry
	OGC Well Known Binary	Long Raw
Microsoft SQL Server	ArcSDE Compressed Binary	Image
	OGC Well Known Binary	Image
IBM DB2	Spatial DataBlade - Geometry Type	ST_Geometry*
IBM Informix	Spatial Extender - Geometry Type	ST_Geometry*

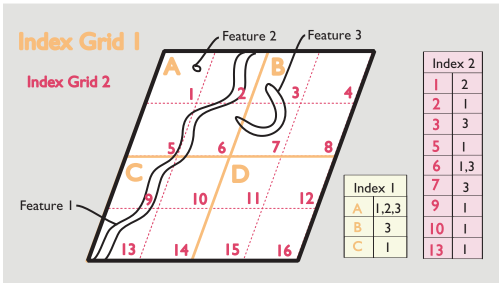
Summary of geometry storage and column types available by DBMS

*ST_GEOMETRY is a superclass of several subclasses (for example, ST_polygon).

ArcSDE implements a continuous data model; it doesn't tile or split your data. For example, you could store all the parcels for the United States in one continuous feature class. Each parcel would be stored as a complete polygon, quickly retrieved by a single disk access.

Grid index

To support the use of databases with millions of spatial records, ArcSDE spatially indexes features in each feature class for rapid search and retrieval. ArcSDE builds a spatial index by applying a grid to the feature class. It records features that fall within each grid cell in an index table (the S table of the feature class schema). A feature



Index 1	
A	1,2,3
B	3
C	1

Index 2	
1	2
2	1
3	3
5	1
6	1,3
7	3
9	1
10	1
13	1

The feature class is overlaid by grid cells to create the spatial index.

that falls in more than one cell is listed in each. Grid cells with no data are not included in the table.

A layer of data may have up to three index grids of different resolutions, although in practice a single grid is usually sufficient.

Other spatial indexing methods

There is more than one way to create a spatial index. ArcSDE spatial indexing does not apply to all databases or geometry storage methods. Oracle Spatial and Informix use other methods for spatial indexing, while DB2 stores the grid spatial index differently from Microsoft SQL Server. Refer to the configuration and tuning guide for your database as well as the spatial documentation provided with Oracle Spatial, IBM Informix, and DB2 for more information.

Spatial indexing is an advanced, but important, topic. Other performance variables notwithstanding, a poorly defined spatial index can make data retrieval unacceptably slow. The ArcSDE spatial indexing method and some guidelines on how to determine your spatial index (grid size) are described in detail in the configuration and tuning guide for your database (ArcSDE_Config_GD_<your_dms>.pdf on the ESRI Software Documentation Library CD–ROM).

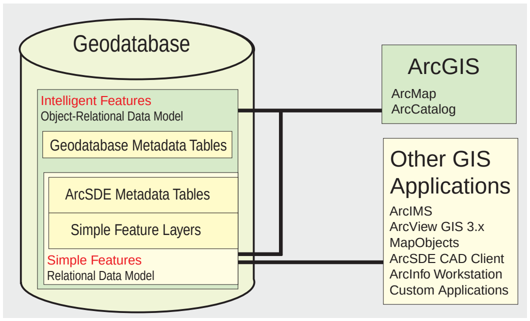
RDBMS	GEOMETRY STORAGE	SPATIAL INDEX METHOD
Oracle	ArcSDE Compressed Binary	Multi-level grid
	Oracle Spatial / Locator - Geometry Type	Rtree or quadtree
	OGC Well Known Binary	Multi-level grid
Microsoft SQL Server	ArcSDE Compressed Binary	Multi-level grid
	OGC Well Known Binary	Multi-level grid
IBM DB2	Spatial DataBlade - Geometry Type	Multi-level grid
IBM Informix	Spatial Extender - Geometry Type	Rtree

Summary of geometry storage and spatial indexing methods available by DBMS

There is a great deal of metadata that must be maintained about spatial data in a DBMS. ArcSDE and the ArcGIS geodatabase have a set of metatables for managing this spatial data (see Chapter 3 for an overview of the ArcGIS geodatabase). However, these metatables should never be managed or edited directly. Only ArcSDE populates and manages the ArcSDE tables, and only ArcGIS applications manage the geodatabase tables. End users do not need to be aware that these tables exist. More information about these tables may be found in the ArcSDE Developer Help (on the ArcSDE Developer Kit CD–ROM) and is provided for database administrators (DBAs).

Relational view and object relational access

You might logically think of the ArcSDE and ArcGIS geodatabase system tables as one set of metadata tables. However, there is an important distinction based on the type of access an application has to the data. Some applications have relational access to data in a geodatabase, while others have object relational access. Relational access is access to the simple features of ArcSDE. Object relational access is access to the simple features, plus the intelligence for the features stored in the geodatabase metatables.



Relational access and object relational access depend on whether the client application has access to the geodatabase metatables. Applications based on ArcObjects, such as ArcGIS Desktop and ArcGIS Server, have object relational access.

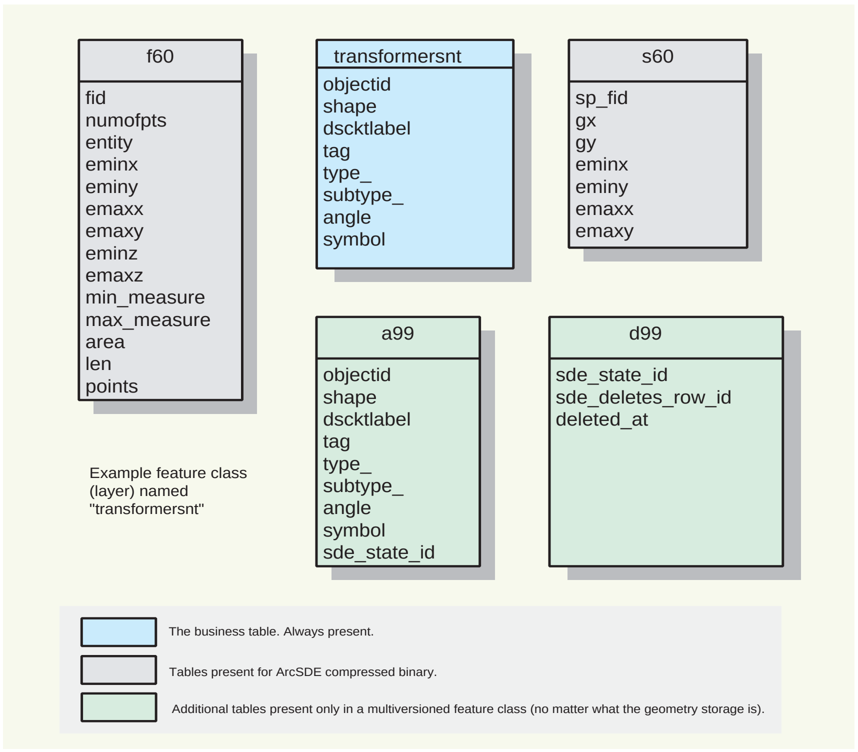
The geodatabase metatables are accessible and used only by applications with object relational access—the ArcGIS Desktop products. This means that applications such as ArcView GIS 3.x, MapObjects 2.x, ArcInfo Workstation (for example, ArcPlot™), ArcSDE CAD Client, and third party applications built with the C or Java APIs cannot take advantage of the data stored in those geodatabase metatables. However, these applications do have access to the underlying simple features.

It is also important to note the ArcGIS family of products can use the simple features without having to populate any of the geodatabase metatables first. However, this use is limited to basic display and query functionality.

When users access feature data in an application, they'll see a layer or feature class name as data to add to their map. In the database, the physical representation of that feature class consists of one or more tables.

- Business table—This is the table that logically represents the feature class and what users will see. It is not necessary for users to be aware of the existence of any of the other tables that make up a feature class.
- Feature table—Always prefaced with an "f", this table has the geometry stored in it along with other metadata about the layer. Feature classes based on IBM DB2 and Informix or Oracle Spatial/Locator do not have this table. They have the feature column as part of the business table. Each table has a "layer identification" number as part of the table name. For example, f60 is the feature table for a feature class with an ID of 60.
- Spatial Index table—Always prefaced with an "s", this table contains the feature classes' spatial index. Feature classes based on IBM DB2 and Informix or Oracle Spatial/Locator do not have this table. Their spatial indexes are stored differently. As with feature tables, the layer ID of the spatial index tables is part of the spatial index table name.
- Delta tables—If a feature class is multiversion enabled (see Chapter 3 for information on versioning), the feature class will have the following tables:
  - Adds table—Always prefaced with an "a", this table stores all additions to the feature class.
  - Deletes table—Always prefaced with a "d", this table stores all deletes performed on the feature class.

  Both tables will have a registration identification number as part of the table name.



An example of the physical table schema of a single feature class for geometry

Raster data represents a significant portion of the total data used in a GIS. ArcSDE provides support for rasters in a number of different formats.

ArcSDE handles raster data much like vector data. When a business table is created with a column of raster type, ArcSDE will reference this raster column as a raster dataset. Raster catalogs can have multiple rasters. Information about the raster column is maintained in one of the ArcSDE system tables called raster_columns.

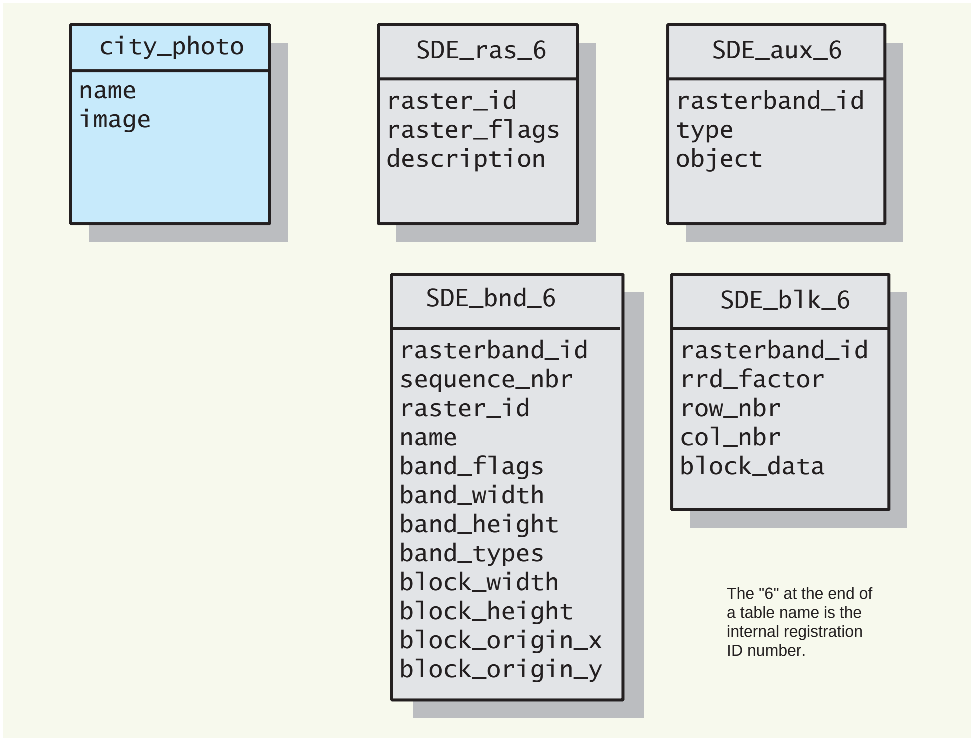
Here's a brief description of the tables in raster layer schema.

For each raster type column in a business table, ArcSDE will automatically create four additional tables. They are:

- Metadata table for raster (SDE_RAS_<id#>)
- Metadata table for raster band (SDE_BND_<id#>)
- Auxiliary table for raster band (SDE_AUX_<id#>)
- Block table (SDE_BLK_<id#>)

The first two metadata tables are used to store information about a raster and a raster band such as the raster dimensions and the pixel depth.

The auxiliary table is used to save additional information about a raster band, such as a colormap and raster statistics. The block table is where the actual pixel blocks are stored.



Tables that define an ArcSDE raster feature class. In this example, city_photo is the feature class users would see to add to their map. Users do not need to be aware of the existence of the other tables.

ArcSDE handles XML data nearly in the same way it handles spatial GIS data. A business table can have one or more columns of type XML. These columns can store XML documents that contain information such as descriptions of spatial features. XML documents are useful for storing longer text descriptions than are typically stored in a column, and text indexes built on those descriptions let you search for features using their content.

An XML column will always have a full text index that lets you search for a word anywhere in a document. The column may optionally have an XPath text index that lets you search for a word in a specific XML element or attribute.

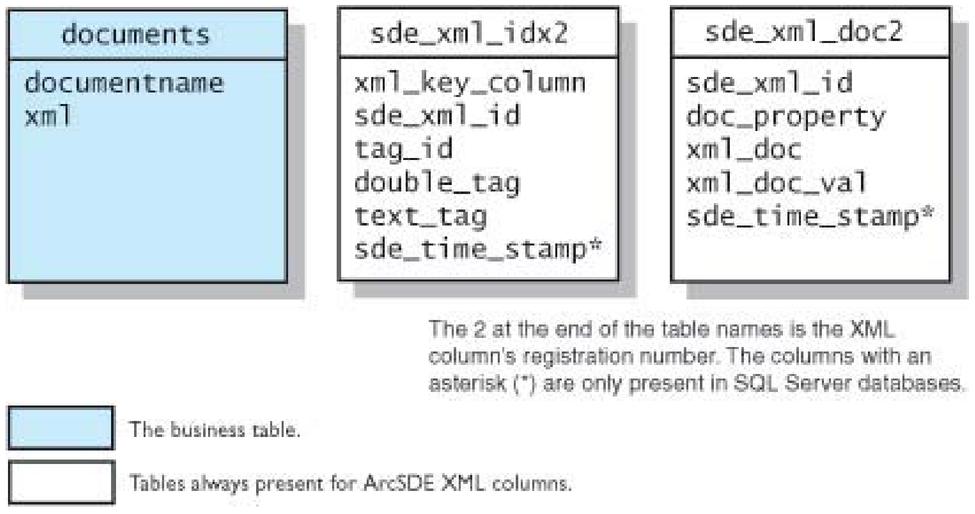
Information about an XML column is maintained in an ArcSDE system table named xml_columns. Information about its text indexes is maintained in the ArcSDE system tables named xml_indexes and xml_index_tags.

For each XML column, ArcSDE creates additional tables, which are used to store and index XML documents.

- **XML document table** (sde_xml_doc<id#>)—This table stores the XML document and maintains a full text index on the document's content.

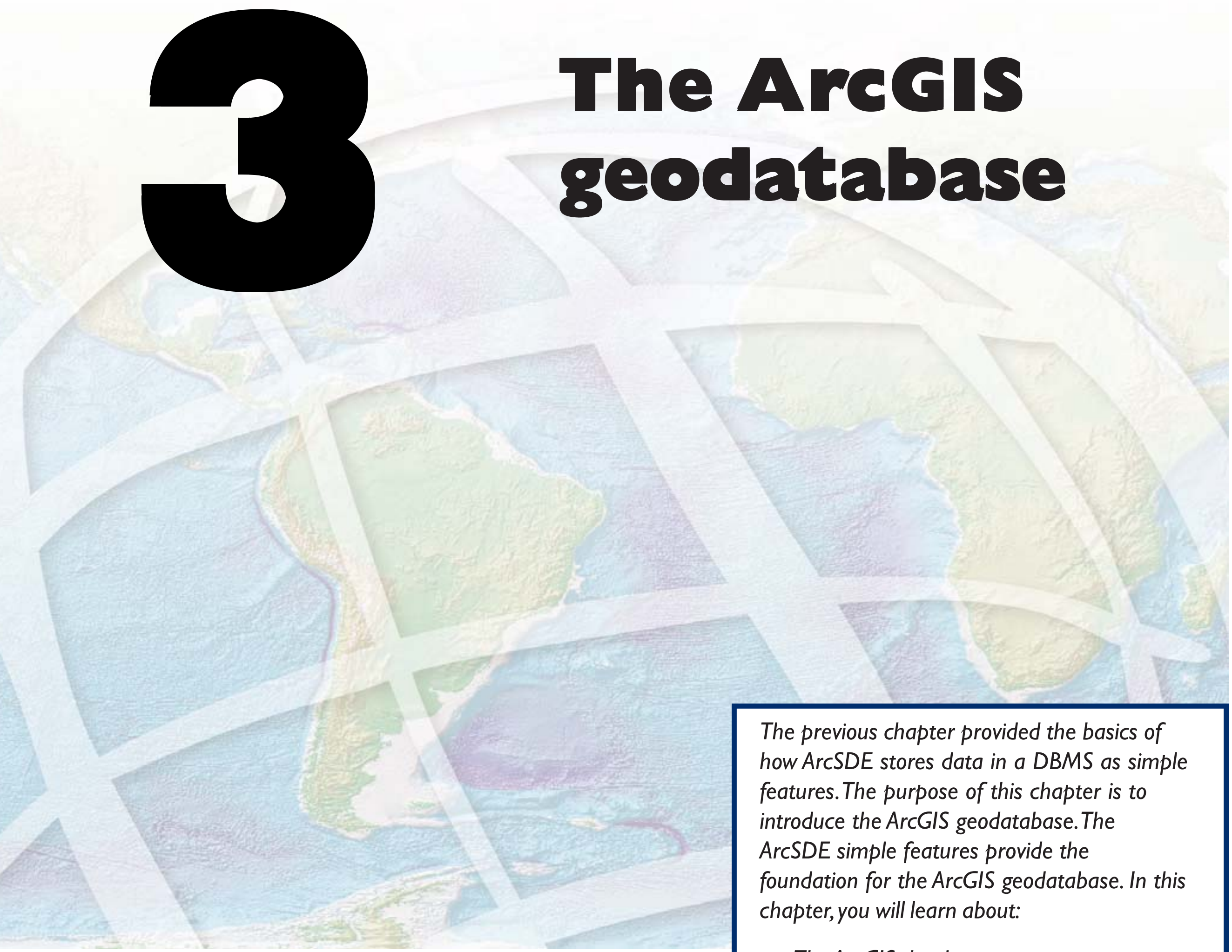
- **XML index table** (sde_xml_idx<id#>)—This table is created for XML columns that have an XPath text index. This table stores the text or number content for each XPath that is indexed.

The ID number in the table names of the XML index table and XML document table is the internal registration number for the XML column.


documents
documentname
xml
sde_xml_idx2
xml_key_column
sde_xml_id
tag_id
double_tag
text_tag
sde_time_stamp*
sde_xml_doc2
sde_xml_id
doc_property
xml_doc
xml_doc_val
sde_time_stamp*
The 2 at the end of the table names is the XML column's registration number. The columns with an asterisk (*) are only present in SQL Server databases.
The business table.
Tables always present for ArcSDE XML columns.

An example of a physical table schema for a table with an XML column. In this example, users would only see the table named "documents".

3 The ArcGIS geodatabase



The previous chapter provided the basics of how ArcSDE stores data in a DBMS as simple features. The purpose of this chapter is to introduce the ArcGIS geodatabase. The ArcSDE simple features provide the foundation for the ArcGIS geodatabase. In this chapter, you will learn about:

- *The ArcGIS database*
- *What is the geodatabase?*
- *Geodatabase architecture*
- *Geodatabase application logic*
- *Vector and raster data*
- *Geodatabase storage*
- *Versioned geodatabases*

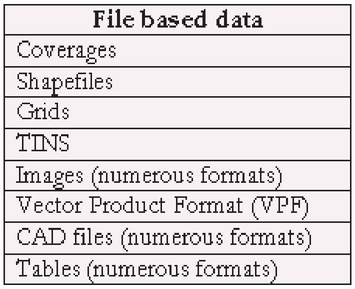
ArcGIS supports GIS data in files and DBMSs

A key cornerstone of ArcGIS is the ability to access GIS data in any format and to use multiple databases and file bases concurrently.

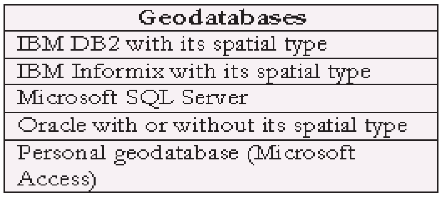
ArcGIS has a high-level generic geographic data model for representing spatial information as features, rasters, and other spatial data types. ArcGIS supports an implementation of the data model for both file systems and relational database management systems. The support for spatial data in a DBMS is provided via ArcSDE.

Support for file-based models includes access to numerous GIS datasets, such as coverages, shapefiles, grids, images, and triangulated irregular networks. The geodatabase model manages the same types of geographic information in a relational database.

Both the file-based datasets and the DBMS-based datasets define a generic model for geographic information. This generic model can be used to define and work with a wide variety of GIS applications. By defining and implementing the behavior of a generic geographic data model, geographic information in ArcGIS can be multipurpose, shareable, and standards based. Most important, a comprehensive series of tools is available to work with the generic data types. Thus, ArcGIS provides a robust platform for virtually any GIS application.



File based data
Coverages
Shapefiles
Grids
TINS
Images (numerous formats)
Vector Product Format (VPF)
CAD files (numerous formats)
Tables (numerous formats)



Geodatabases
IBM DB2 with its spatial type
IBM Informix with its spatial type
Microsoft SQL Server
Oracle with or without its spatial type
Personal geodatabase (Microsoft Access)

Some of the common GIS data formats that can be used directly in ArcGIS. Access to numerous additional formats is supported through data conversion and interoperability extensions.

The geodatabase is a data model for representing geographic information using standard relational database technology. The geodatabase—short for *geographic database*—supports the storage and management of geographic information in standard relational database management system tables via ArcSDE.

Geodatabases can scale from smaller, single-user databases built on the Microsoft Jet Engine database up to larger work group, department, and enterprise databases accessed by many users. Two types of geodatabase architectures are available: personal geodatabases and multiuser geodatabases.

Personal geodatabases, which are freely available to all ArcGIS users, use the Microsoft Jet Engine database file structure to persist GIS data in smaller databases. Personal geodatabases are much like file-based workspaces and hold databases up to 2 GB in size. Microsoft Access is used to work with attribute tables in personal geodatabases.

Personal geodatabases are ideal for working with smaller datasets for GIS projects and in small work groups or projects of short duration. Typically, users will employ multiple personal geodatabases for their data collections and access these simultaneously for their GIS work. Personal geodatabases support single-user editing. No concurrent multiuser editing support is provided.

Multiuser geodatabases are primarily used in a wide range of work groups, departments, and enterprise settings. They take full advantage of their underlying DBMS architectures to support:

- Extremely large, continuous GIS databases
- Many simultaneous users
- Long transactions and versioned work flows that are critical in GIS

GIS database sizes and the numbers of supported users can be much larger than GIS file bases.

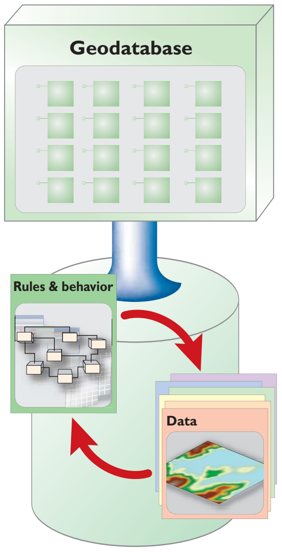
Geodatabase type	DBMS	Notes
Personal geodatabase	Microsoft Jet Engine (Access)	· Single-user editing · 2 GB size limit · No versioning support
Multiuser, versioned geodatabase	· Oracle · Oracle with Spatial or Locator · IBM DB2 · IBM Informix · Microsoft SQL Server	· Requires ArcSDE Gateway · Multiuser editing · Version-based work flows · Database size and number of users up to DBMS limits

Summary of personal and multiuser geodatabases

Responsibility for managing geographic datasets is shared between GIS software and generic DBMS software. Certain aspects of geographic dataset management, such as disk-based storage, definition of attribute types, associative query processing, and multiuser transaction processing, have been delegated to the DBMS. The GIS application retains responsibility for defining the specific DBMS schema used to represent various geographic datasets and for domain-specific logic, which maintains the integrity and utility of the underlying records.

In effect, the DBMS is used as one of a series of implementation mechanisms for persisting geographic datasets. However, the DBMS does not fully define the semantics of the geographic data. This could be considered a multitier architecture (application and storage), in which aspects related to data storage and retrieval are implemented in the storage (DBMS) tier as simple tables, while high-level data integrity and information processing functions are retained in the application and domain software (GIS).

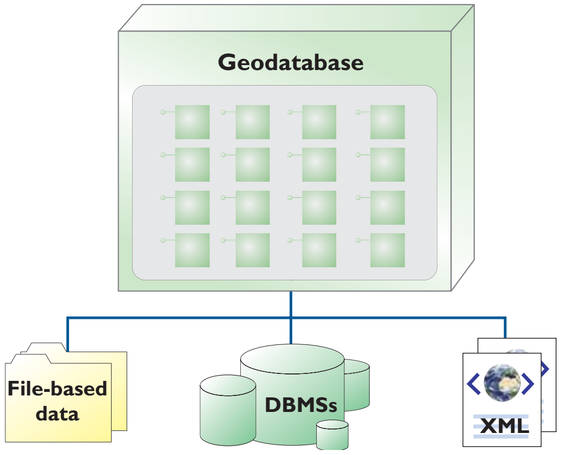
The geodatabase is implemented using the same multitier application architecture that you find in other advanced DBMS applications. The geodatabase objects are persisted as rows in DBMS tables that have identity, and the behavior is supplied through the geodatabase application logic.



The geodatabase architecture is based on simple relational storage and comprehensive application logic.

At the core of the geodatabase is a *standard* relational database schema, or a series of standard DBMS tables, column types, indexes, and so forth. This simple physical storage works in concert with and is controlled by a set of higher-level application objects hosted in the application tier, which can be an ArcGIS client or an ArcGIS Server. These geodatabase objects define a generic GIS information model, which is shared by all ArcGIS applications and users. The purpose of the geodatabase objects is to expose a high-level GIS information model to clients and to persist the detailed implementation of this model in any appropriate storage model.

All ArcGIS applications interact with this generic GIS object model for geodatabases, not directly with the actual SQL-based DBMS instance. The geodatabase software components implement behavior and integrity rules implicit in the generic model and translate data requests to the appropriate physical database design.



The separation of geodatabase logic from storage enables support for numerous file types, DBMSs, and XML instances.

Accessing and using geodatabase application logic

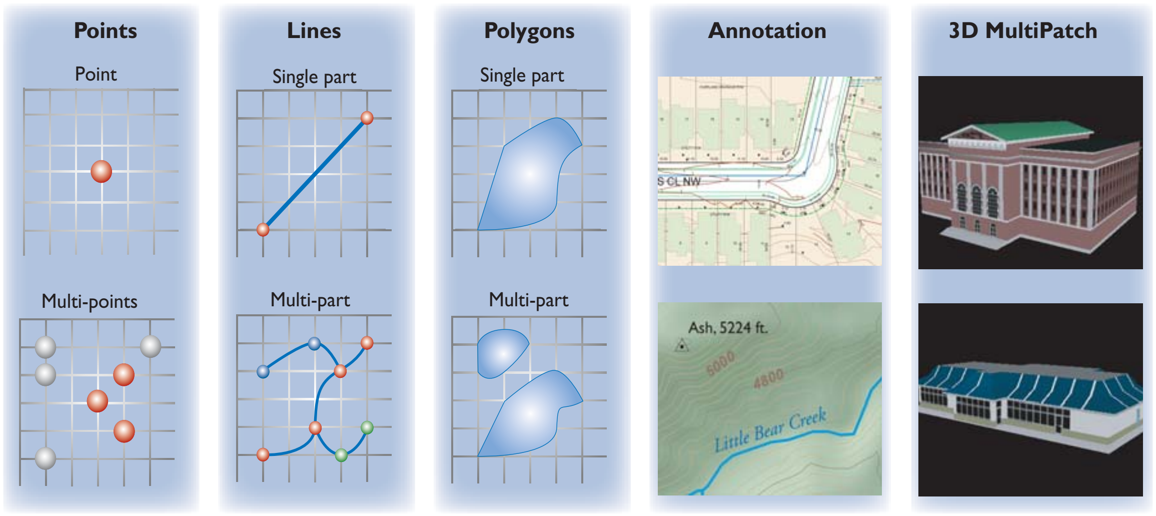
Chapter 2 , 'Data storage', had a section that reviewed relational access and object relational access. The discussion was about geodatabase metadata stored in tables in the database and how ArcGIS applications used and managed that data while non-ArcGIS applications were unable to. Another way to look at that topic is to discuss application logic. One view of the geodatabase is that it is a series of software components that provides complete logic for implementing, compiling, and managing GIS information objects. These geodatabase software objects are accessible through a number of ArcGIS technologies:

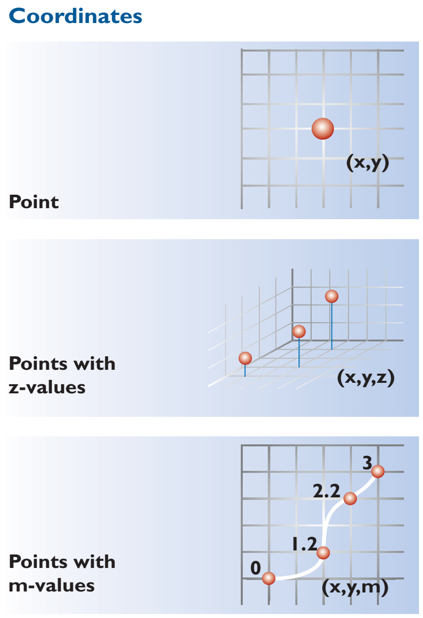
- Interactive application interfaces in ArcGIS Desktop
- Enterprise and Web services in ArcGIS Server
- Embeddable developer components for C++, Java, COM, and .NET developers in ArcGIS Engine

Separating of the geodatabase information model from a particular DBMS schema allows support for multiple storage models for the geodatabase, including multiple DBMSs as well as personal geodatabases with Microsoft Access.

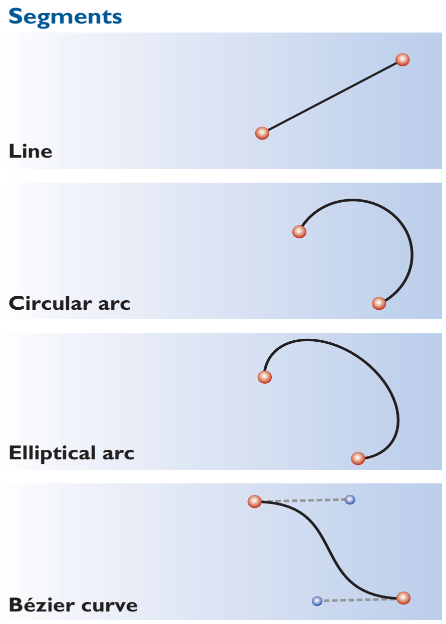
Vector features (geographic objects with vector geometry) are a versatile and frequently used geographic data representation, well-suited for representing features with discrete boundaries, such as wells, streets, rivers, states, and parcels. A feature is simply an object that has a location stored as one of its properties (or fields) in the row.

Typically, features are spatially represented as points, lines, polygons, or annotation and are organized into feature classes. Feature classes are collections of features of the same type with a common spatial representation and set of attributes (for example, a line feature class for roads).





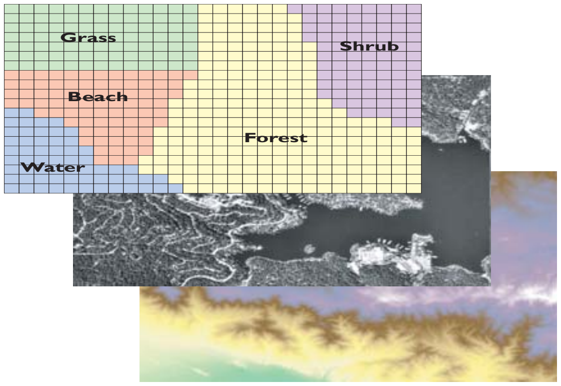
Common vector feature representations



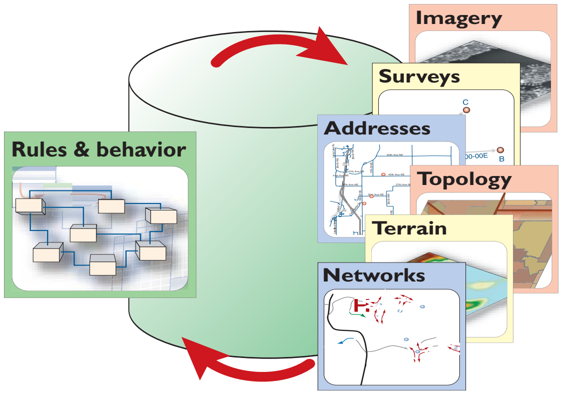
Feature geometry

Raster data

Rasters are used to represent continuous layers, such as elevation, slope and aspect, vegetation, temperature, rainfall, and plume dispersion. Rasters are most commonly used for aerial photographs and imagery of various kinds.



In addition to vector features and raster datasets, all other spatial data types can be managed and stored in the relational tables as well, allowing you the opportunity to manage all geographic data in a DBMS.



Geodatabases are used to manage and store diverse collections of geographic information types.

Geodatabase storage includes the schema and rule base for each geographic dataset as well as the spatial and attribute data. All of the data is stored in the DBMS in tables.

The geodatabase schema includes the definitions, integrity rules, and behavior for each geographic dataset. These include properties for feature classes, topologies, networks, raster catalogs, relationships, domains, and so forth. The schema is persisted in a collection of geodatabase metatables in the DBMS that define the integrity and behavior of the geographic information.

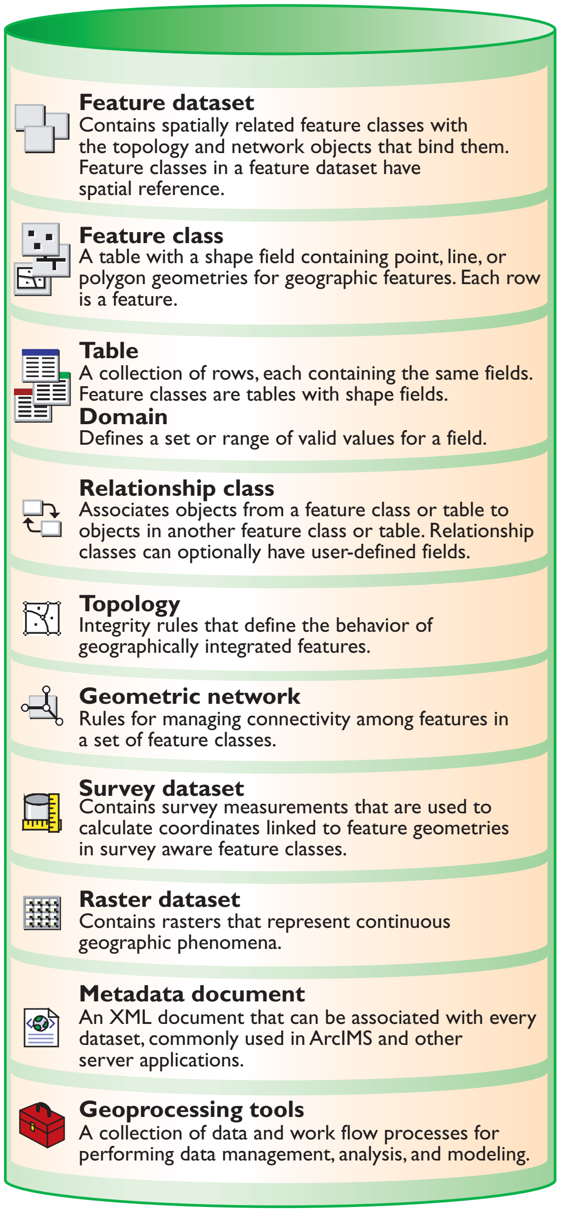
The spatial representations are most commonly stored as vector features or as raster datasets along with traditional tabular attributes. For example, a DBMS table can be used to store a feature collection where each row in the table represents a feature. A shape column in each row is used to hold the geometry or shape of the feature. The shape column holding the geometry is typically one of two column types:

- A binary large object column type
- A spatial column type, if the DBMS supports it

A homogeneous collection of common features, each having the same spatial representation (such as a point, line, or polygon) and a common set of attribute columns, is referred to as a feature class and managed in a single table.

Raster and imagery data types are managed and stored in relational tables as well. Raster data is typically much larger in size and requires a side table for storage. The raster is cut into smaller pieces, called blocks, and stored in individual rows in a separate block table.

The column types that hold the vector and raster geometry vary from database to database. When the DBMS supports spatial type extensions, the geodatabase can readily use them to hold the spatial geometry. ESRI was closely involved in efforts to extend SQL spatially and was one of the primary authors of the SQL 3 MM Spatial and the OGC Simple Features SQL specifications. ESRI has focused on support for these types, as well as the independent Oracle Spatial types, in the persistence of geodatabases using DBMS standards.



Feature dataset
Contains spatially related feature classes with the topology and network objects that bind them. Feature classes in a feature dataset have spatial reference.

Feature class
A table with a shape field containing point, line, or polygon geometries for geographic features. Each row is a feature.

Table
A collection of rows, each containing the same fields. Feature classes are tables with shape fields.

Domain
Defines a set or range of valid values for a field.

Relationship class
Associates objects from a feature class or table to objects in another feature class or table. Relationship classes can optionally have user-defined fields.

Topology
Integrity rules that define the behavior of geographically integrated features.

Geometric network
Rules for managing connectivity among features in a set of feature classes.

Survey dataset
Contains survey measurements that are used to calculate coordinates linked to feature geometries in survey aware feature classes.

Raster dataset
Contains rasters that represent continuous geographic phenomena.

Metadata document
An XML document that can be associated with every dataset, commonly used in ArcIMS and other server applications.

Geoprocessing tools
A collection of data and work flow processes for performing data management, analysis, and modeling.

A geodatabase is a store of geographic data implemented with the relational database of your choice. All geodatabase elements are managed in standard DBMS tables using standard SQL data types. These are some of the structural elements of a geodatabase that you will use to develop your geographic data model.

GIS data, like other information, must be continually maintained and edited. Hence, geodatabases are designed to be *transactional*. The geodatabase was designed from the beginning to be edited by many users; to scale to very large, continuous sizes; and support a number of common GIS application scenarios.

GIS data compilation work flows and data sharing require a long transaction model for numerous editing and data replication needs. A long transaction in the GIS context is one that spans multiple days or even weeks, including when the user disconnects from the database. In the DBMS context, transactions are short in the sense that no DBMS transaction persists when the user disconnects from the database.

In GIS, a single edit operation is typically combined with a series of other edits to define a complete transaction. For example, a typical update in land records applications is a parcel split. This involves three steps: deleting the old parcel, creating two new parcels, and updating the tax rolls with new corresponding parcel and owner information. In this simple case, a single GIS update transaction is actually composed of three or more DBMS transactions. In addition, GIS users often need to:

- Undo or redo individual editing operations during an editing session.
- Create a historical archive of updated features (for example, “retired” parcels and their lineages).

In a multiuser database, the GIS transactions must be orchestrated on the DBMS’s short transaction framework. ArcSDE plays a key role during these operations by managing the higher-level complex GIS transactions on the short duration DBMS transaction framework.

GIS users have many such cases in which long transaction work flows are critical. In most cases, these are made possible through the use of a multiuser DBMS and ArcSDE for managing updates to the central GIS database:

- Multiple edit sessions. A single GIS database update may require numerous changes that span multiple edit sessions occurring over a few days or weeks.
- Multiuser editing. Multiple editors often need to concurrently update the same spatially integrated features. Each needs to work with his or her own database state, viewing their own individual updates and ignoring updates by other editors. Eventually each user needs to post their updates, reconcile them with the other editors, and identify and resolve any conflicts.
- Check-out, Check-in Transactions. Often users want to check out a portion of a database for a particular area or district onto their own computer and update that information in a disconnected session that might last for days or weeks. Eventually, they want to post their updates to the main database. In other cases, users will take a portion of a large geodatabase with them into the field for validation and update with field computers.
- History. Some users want to maintain a historical version of each feature that was in their GIS database, even after that version has been updated. They want to maintain a copy of the retired and changed features in a historical archive. Users also often need to track each individual feature’s history (for example, parcel lineage or feature update properties in a national mapping database).
- Transfer of change-only updates. Many users collaborate on data collection and need to share updates (across the World Wide Web) in a well-defined XML schema for sharing change-only updates between databases. These databases can have different GIS architectures.
- Loosely coupled replication across DBMSs. Often users want to periodically synchronize GIS data contents among a series of database copies (called replicas), where each site performs its own updates on its local database. Periodically, the users want to transfer the updates from each database replica to the others and synchronize their contents. Many times the DBMSs are different (for example, to replicate datasets between SQL Server, Oracle, and IBM DB2).

What is versioning?

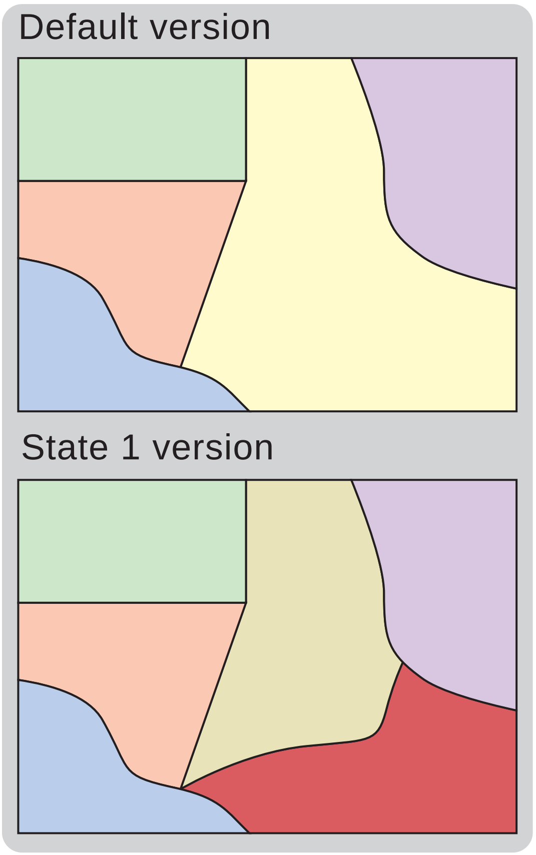
The geodatabase mechanism for managing these and many other critical GIS work flows is to maintain multiple states in the geodatabase, and most importantly, to do so while ensuring the integrity of the GIS database. This ability to manage and view multiple states and to work with them is based on *versioning*. Versioning, as the name implies, explicitly records versions of individual features and objects as they are modified, added, and "retired" through various states. A version explicitly records each state of a feature or object as a row in a table along with important transaction information.

Versions explicitly record the object states of a geodatabase in delta tables called the "Adds" table and the "Deletes" table. Simple queries are used to view (and work with) any desired state of the geodatabase (for example, to view the database state for a point in time or to see a particular user's current version with his or her edits).

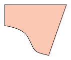
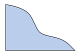
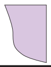
This delta table schema is used to support simultaneous editing of datasets as well as numerous version-based GIS work flows, such as the examples described earlier, for multiple edit sessions, history, and cross-DBMS replication.

ArcSDE plays a critical role in versioned geodatabase applications and is used to manage long transactions in each DBMS as well as across different systems.

The goal of versioning is to support these and many other critical GIS data work flows to enable enterprise GIS implementations.



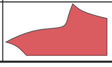
Base table

ID	Shape	Area	Coverage	R-ID
1		15	vege.	10
2		30	indust.	20
3		14	resident	30
4		12	water	40
5		25	grove	50

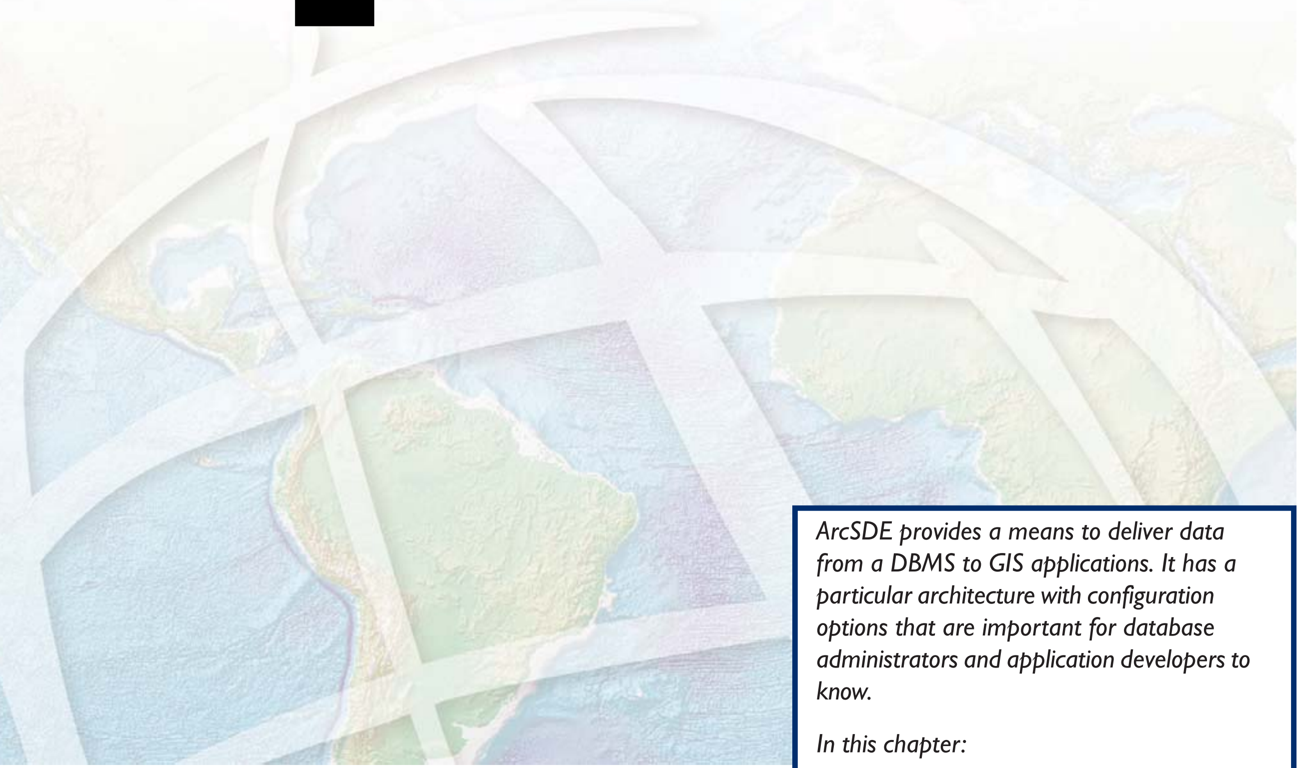
Delete table

ID	Shape	Area	Coverage	R-ID
2		30	indust.	20

Add table

ID	Shape	Area	Coverage	R-ID
6		17	indust.	60
7		13	vege.	70

4 The ArcSDE architecture



ArcSDE provides a means to deliver data from a DBMS to GIS applications. It has a particular architecture with configuration options that are important for database administrators and application developers to know.

In this chapter:

- *What is an application server?*
- *Direct connections*
- *Connecting with a direct connect driver*
- *Licensing and authorization*

Architecturally speaking, there are two basic configurations you can use with ArcSDE implementations. You may choose a three-tier architecture, which uses an application server, or a two-tier architecture, which uses what are called direct connect drivers. You can also use a combination of the two.

The application server

ArcSDE is built with client/server architecture—a client application sends requests to the server. In turn, the server receives the request, generates results, and delivers them to the client.

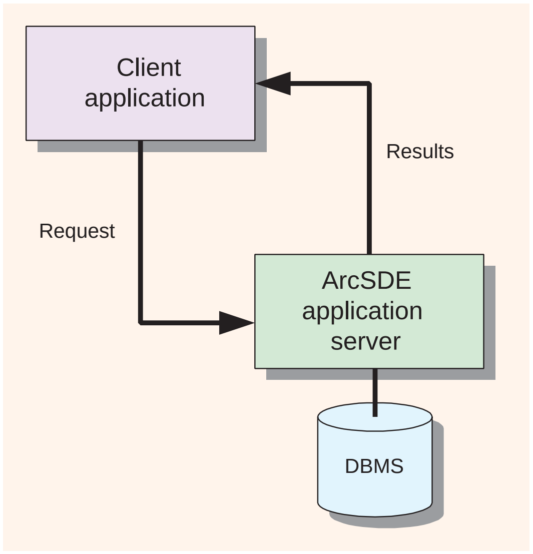
The ArcSDE server accesses spatial data based on highly efficient spatial search functions, provides geometric data validation, and works within heterogeneous hardware and network environments. Data can be delivered to any client from any server anywhere on a network.

In a typical configuration, an ArcSDE application server resides with your relational database on a server platform. The ArcSDE application server performs spatial searches and sends data that meets the search criteria to the client. For example, a common query handled by the ArcSDE application server is to retrieve all the features in a particular map extent to be drawn in the display window.

ArcSDE sends data to the client using "data buffering". Buffering is the process of collecting large chunks of data and sending them all to the client application, rather than sending one record at a time. Processing and buffering data on the server is much more efficient than sending all the data across the network and having the client determine which data to send to the end user application. This becomes critical when applications are simultaneously using thousands of records in the database.

ArcSDE uses cooperative processing, which means that data processing occurs on both the client application machine and the server, depending on which is faster. Some functions require no communication with the server. CPU-intensive tasks, such as polygon overlay and clipping, are best performed by the client application to avoid excessive demands on both the server and the available bandwidth.

The computer network connects many clients to the server. The network must support TCP/IP. It can be a low-speed wide area network (WAN) or a fast local area network (LAN). Network file system mounts are not required for data transfer between the server and the client. This is an important performance and administrative benefit.

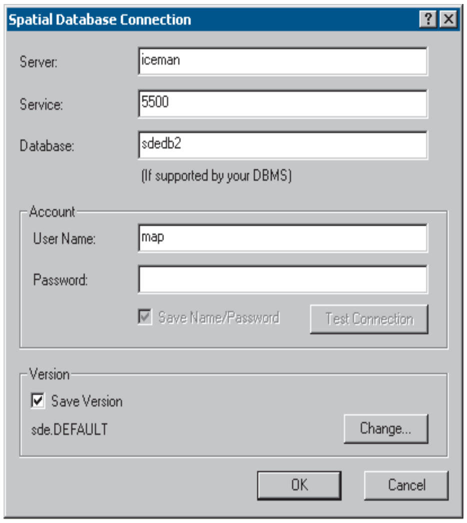


The ArcSDE server handles simultaneous requests from multiple users to update and retrieve information in a geodatabase.

The application service connection process

Geodatabase security is managed via the DBMS. Therefore, accessing data requires a “connection” to the host DBMS. A user must enter connection information including a username and password. This connection information, passed on to the ArcSDE server, is used to log in to the DBMS. If the login connection is successful, the user can then begin working with the geodatabase.

The graphic below illustrates the basic connection process. The giomgr process is the “service” to which a user connects. *Managing ArcSDE Application Servers* describes how to create and manage services. Typically, the ArcSDE service runs on the same machine as the DBMS server. The gsrvr process always runs on the same machine as the giomgr process. Each connection request starts a gsrvr process. Each gsrvr process will use a DBMS connection.


Spatial Database Connection

Server: iceman

Service: 5500

Database: sdedb2

(If supported by your DBMS)

Account

User Name: map

Password:

[✓] Save Name/Password

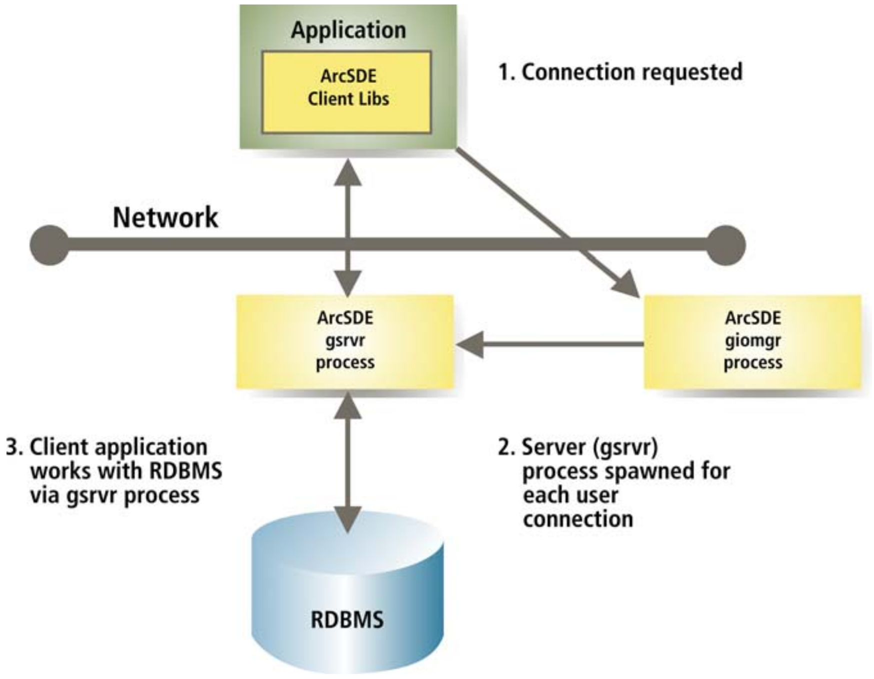
Version

[✓] Save Version

sde.DEFAULT

The Spatial Database Connection wizard in ArcGIS Desktop shows a user named “map” connecting to a service named “5500” (used port number instead of actual name in this case) on a server machine named “iceman”. In this case, the user is connecting to a DB2 database named “sdedb2”.

The illustration below shows the basics of the connection process. Typically, the giomgr process and the DBMS are on the same server machine. The gsrvr process is always on the same machine as the giomgr process.


Application
ArcSDE Client Libs
1. Connection requested
Network
ArcSDE gsrvr process
ArcSDE giomgr process
3. Client application works with RDBMS via gsrvr process
2. Server (gsrvr) process spawned for each user connection
RDBMS

ArcSDE can also be used without the application server, providing you a two-tier configuration option. The two- and three-tier systems are designed to work independently or together, giving the DBA flexibility in system design. When using the two-tier direct connection architecture, you will use direct connect drivers.

When using the two-tier architecture, an application connects directly to the database, without using the ArcSDE application server. The ArcSDE client and server functionality is executed on the same machine as the client application—there is no separate ArcSDE server process running anywhere.

The two-tier architecture provides some flexibility in configuring systems:

- The direct connect database driver doesn't require the administration of the ArcSDE server process. There is no intermediate giomgr process to set up, configure, or start up. Setting up a single-user database using Microsoft Database Engine (MSDE), for example, is easier.
- The direct connect database driver provides additional options for scaling a system. The direct connect architecture moves the ArcSDE server functionality to the desktop. This removes the ArcSDE load from the database server and will allow additional resources to be freed up for the DBMS, which means you get better scalability on the database server.
- The direct connection configuration can be very useful in failover environments. For example, it's easier to configure your system with Oracle RAC using the direct connect driver.

Are there any reasons why you wouldn't want to use the two-tier direct connect architecture?

The following reasons might apply:

- Increased network traffic. This can happen if the spatial filter selectivity is low—for example, finding a very small number of features adjacent to a long sinuous feature that crosses a large area. Increased network traffic can result in poorer overall performance, but the severity of the degradation depends on the bandwidth of your network.
- You do not want the administrative overhead of networking software (for example, Oracle Net).
- If you are building a thin client, executing the ArcSDE server process on the client machine may not be desirable.
- If you have a low-end desktop computer with limited memory and/or chip speed, you may need to use the application server to move that server functionality off your client.

The application server configuration is generally faster, but you should do some prototyping to see which configuration works best for you. However, whether you configure your system with the application server, direct connect drivers, or a mixture of the two, the same client functionality is available to any ArcSDE client application. All ESRI client products are delivered with direct connect drivers.

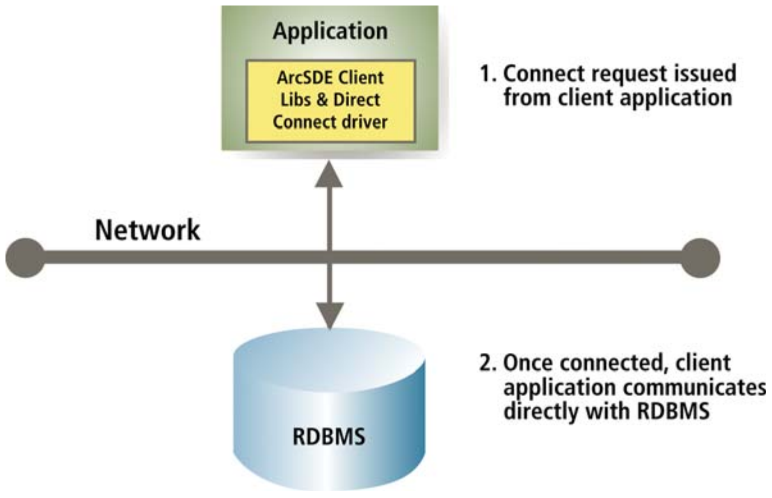
Configuring your database

The ArcSDE direct connect drivers are built from the same software code that is used to build the application server. The difference is the direct connect drivers are built as dynamically linked libraries and execute in the process space of the client application, and the application server is built as an executable program that runs on the server machine. Each user connection in a three-tier configuration has its own application server process.

Since the direct connect drivers are built from the same software code as the application servers, the same database configuration must be done as when setting up a database to use with the application server. The same ArcSDE user and ArcSDE and geodatabase metatables used by the application server configuration must also exist for direct connection configurations. Your administrator must set these up prior to any ArcSDE connection requests. Your client machines must be configured for network access. Read the ArcSDE installation guide closely for setup information as well as supported databases and hardware operating systems. You may also want to read the appendix titled 'Making a Direct Connection' in the *ArcSDE Configuration and Tuning Guide for <DBMS>* PDF file, located in the documentation folder on the ArcSDE CD–ROM. Refer to DBMS documentation for more specifics on client hardware configuration.

Connecting with a direct connect driver

Below is a graphic illustrating the two-tier architecture. The gsrvr functionality is on the client in the form of a dynamically linked library. The connection is still via a DBMS account, and DBMS or operating system security is still used. Once connected, the client works directly with the DBMS without an intermediate application server process.

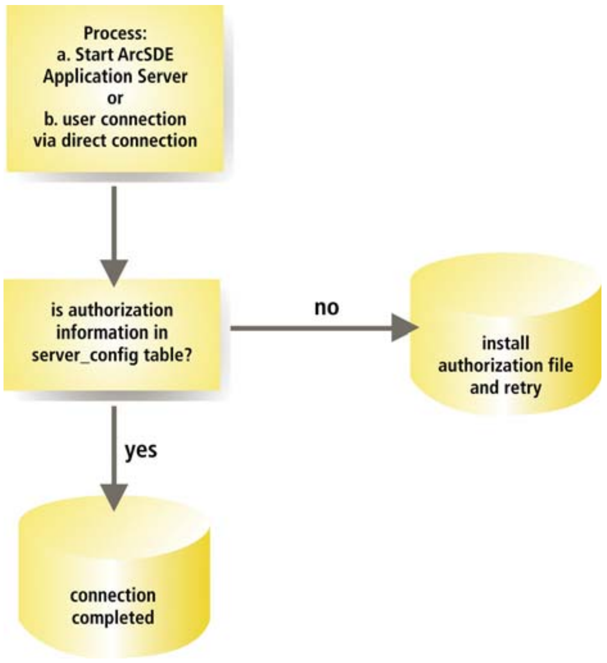

Application
ArcSDE Client Libs & Direct Connect driver
1. Connect request issued from client application
Network
2. Once connected, client application communicates directly with RDBMS
RDBMS

ArcSDE for IBM DB2, IBM Informix, Microsoft SQL Server, and Oracle are licensed products. They are not free. To prevent unauthorized use of the software, an authorization file is required for you to “unlock” the software for use.

After you have installed the software, you will need to get an authorization file. You’ll install that file using the sdesetup<dbms> command (Windows users have a wizard to lead them through the process), and the authorization information goes into a DBMS table called server_config. When a request to start the ArcSDE service (the giomgr process) is issued, it checks the server_config table for the authorization information. Once found, the giomgr process completes its startup and is ready to accept user connections.

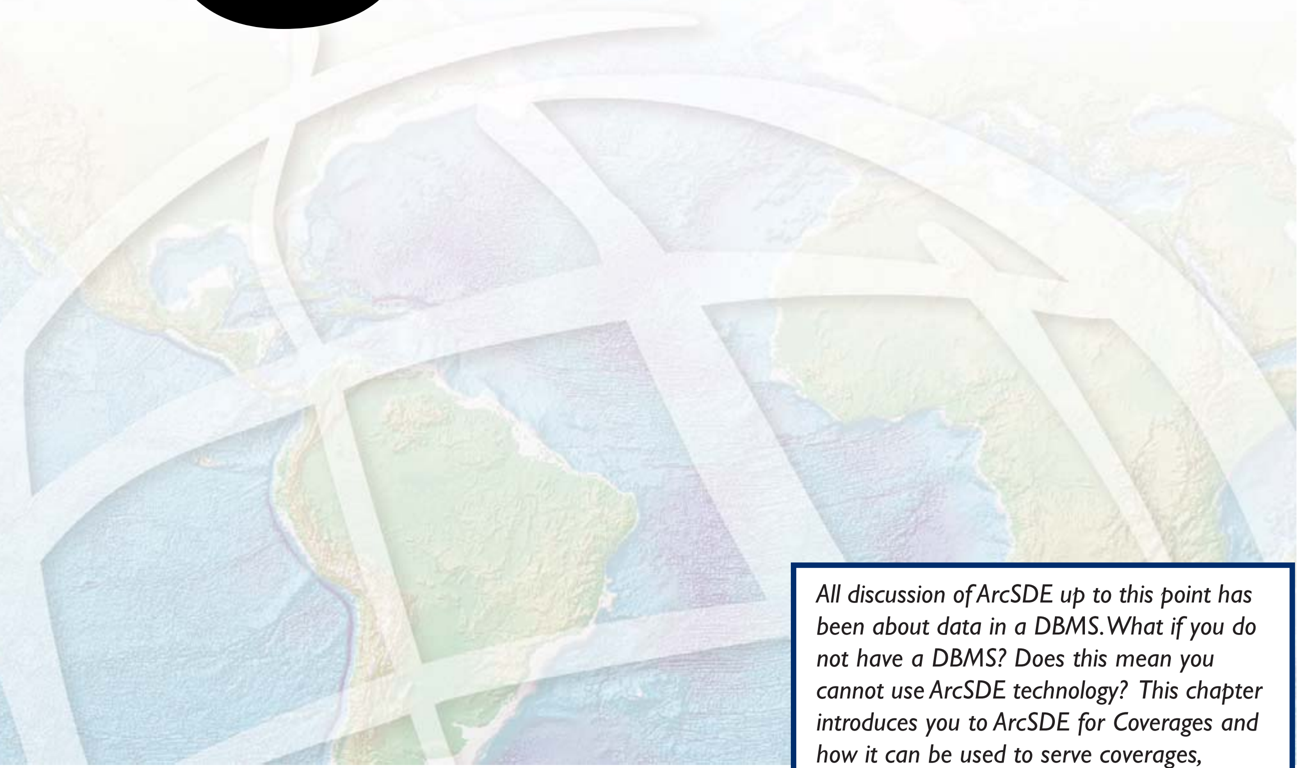
Authorization for direct connect works the same way. When a user issues a connection request via direct connect, the connection process checks to see if the authorization information is present in the DBMS table called server_config and, if so, proceeds with the user’s connection request.

Refer to the ArcSDE install guides for more information on authorization.


Process:
a. Start ArcSDE Application Server
or
b. user connection via direct connection
is authorization information in server_config table?
no
install authorization file and retry
yes
connection completed

This is a simple schema for how the authorization is checked for starting up the ArcSDE application server (giomgr process) and for when a direct connect request is made. When the connection is completed, the giomgr process is then ready to accept connections, and the direct connect is ready to process user requests (for example, display, query).

5 ArcSDE for Coverages



All discussion of ArcSDE up to this point has been about data in a DBMS. What if you do not have a DBMS? Does this mean you cannot use ArcSDE technology? This chapter introduces you to ArcSDE for Coverages and how it can be used to serve coverages, shapefiles, ArcInfo Librarian™ layers, and ArcStorm™ layers to your client applications.

While ArcSDE provides the gateway to numerous commercial relational DBMSs, another available ArcSDE application server serves file-based spatial datasets. This read-only server, called ArcSDE for Coverages, serves the following file-based vector datasets:

- ArcInfo coverages
- ESRI shapefiles
- ArcInfo Librarian layers
- ArcStorm layers

Many sites have file-based data on UNIX file servers and use Windows client applications. Mounting disks on each Windows machine is administrative overhead, and performance may be poor. ArcSDE for Coverages allows access to the file-based data without requiring network file system (NFS) disk mounts, and it can be significantly faster than NFS access.

ArcSDE for Coverages is crucial to providing access to file-based spatial datasets for some ArcSDE client applications. ArcGIS Desktop applications, as well as ArcGIS Server, MapObjects, and ArcIMS, can only access ArcInfo Librarian and ArcStorm data via ArcSDE for Coverages. In addition, ArcIMS depends on ArcSDE for Coverages to provide access to ArcInfo coverage data.

ArcSDE for Coverages can also be part of your migration strategy from file-based data sources to geodatabases. The most direct way to put ArcInfo Librarian and ArcStorm layers into feature datasets of the geodatabase is by using the ArcSDE for Coverages server. An ArcSDE for Coverages layer can be directly loaded into a geodatabase using ArcCatalog™. Alternatively, you can use the ArcSDE command cov2sde to put ArcInfo Librarian or ArcStorm data into standalone feature classes in the geodatabase. However, further work is required to migrate these standalone feature classes into a feature dataset. Manually creating an intermediate coverage from an ArcInfo Librarian or ArcStorm layer and loading that coverage into the geodatabase may take considerable time and disk space.

Customized applications built for ArcSDE for Coverages can still be used if you migrate your data to a DBMS version of ArcSDE, although attribute queries will need to be rewritten in SQL for your DBMS.

How does ArcSDE for Coverages work?

Your data is not copied, loaded, or otherwise moved anywhere. When your ArcSDE for Coverages server is running, there is a utility provided to "register" data with the ArcSDE server. Essentially, you tell the server where the data is, provide a layer name, and set a few properties. At that point, the data can be served to a client application. You can also set up relates to attribute data in a DBMS.

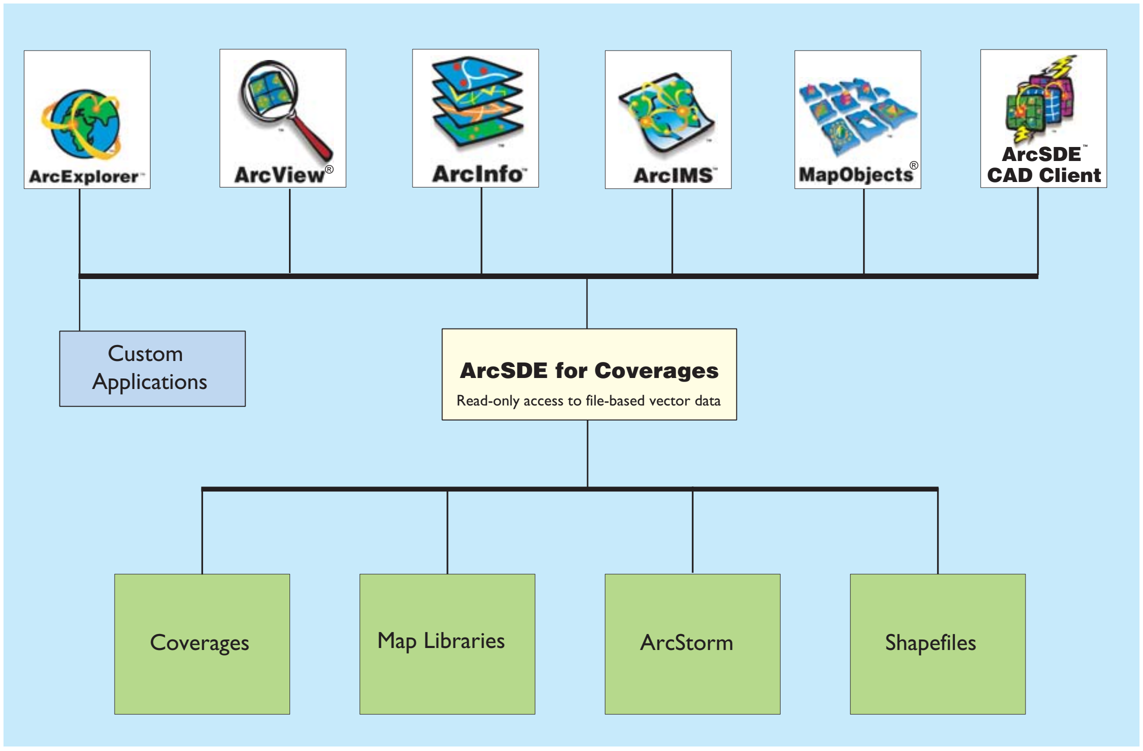
In the case of data split by tiles (ArcInfo Librarian and ArcStorm line and polygon data), the data is "sewn together" on the server if there is a unique ID for each feature. Once a whole feature is assembled, it is sent to the client.

Data transfer is the same as with the DBMS version of ArcSDE. Data is delivered to clients via TCP/IP. This means you need to define a service with a specific name and TCP/IP port number in your services file.

ArcSDE for Coverages is free, and you do not need an authorization file to unlock the software. This makes it an inexpensive way to begin working with ArcSDE technology.

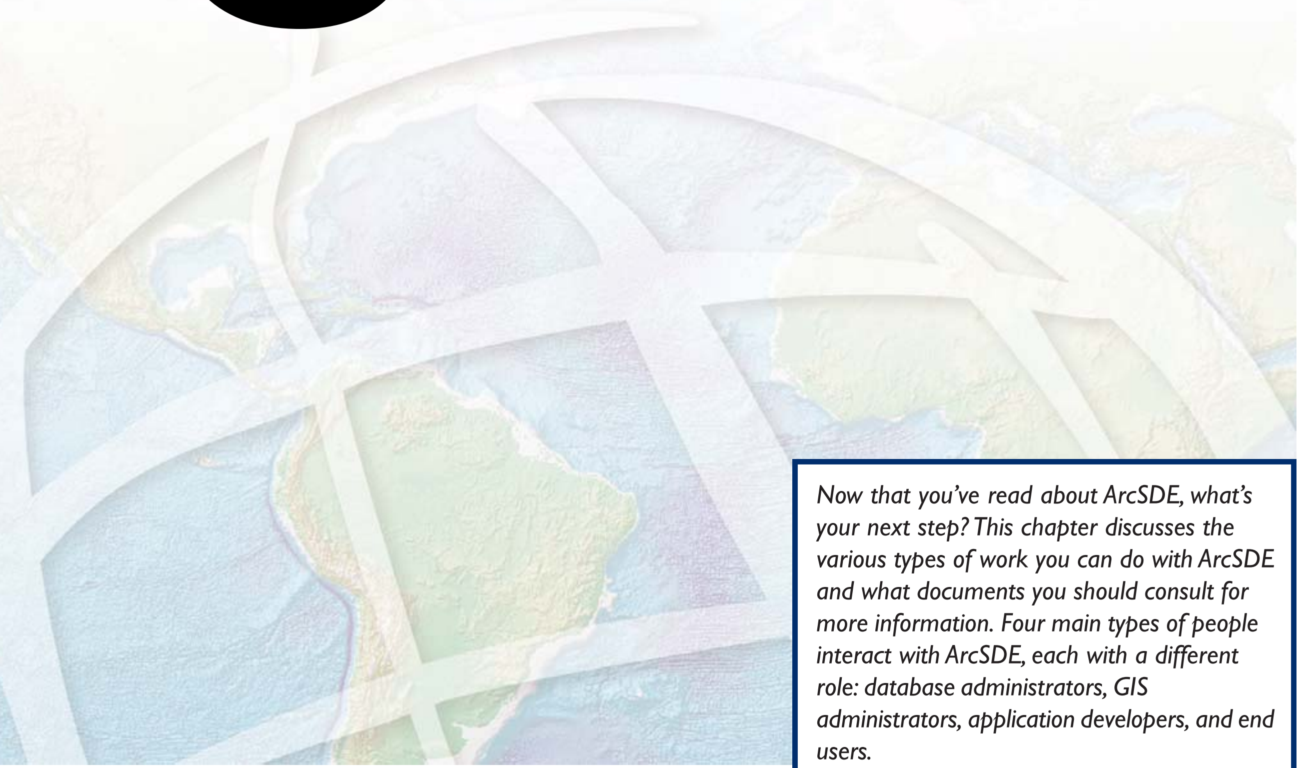
For more information about setting up an ArcSDE for Coverages server, refer to the book *ArcSDE for Coverages Administration Guide*, now in PDF format (arcsde_cov_admin_guide.pdf), located in the documentation folder of the ArcSDE for Coverages CD.

The ArcSDE for Coverages CD–ROM is not in your media kit. If you are new to ArcSDE, you will need to order your CD–ROM through your ESRI regional office or international distributor.



ArcSDE for Coverages acts as a read-only TCP/IP gateway to your file-based data.

6 Where to go next



Now that you've read about ArcSDE, what's your next step? This chapter discusses the various types of work you can do with ArcSDE and what documents you should consult for more information. Four main types of people interact with ArcSDE, each with a different role: database administrators, GIS administrators, application developers, and end users.

In this chapter:

To implement a multiuser GIS, two key roles must be filled: the database administrator and the GIS administrator. Sometimes these are the same person; sometimes multiple people are involved.

The database administrator

The database administrator is responsible for managing the data being served by ArcSDE. When using the ArcSDE application server, the DBA also manages that server and its interface to an organization's relational database. The goal of *Managing ArcSDE Application Servers*, along with the *ArcSDE Installation Guide* (Install_<your_dbms>.htm) and *ArcSDE Configuration and Tuning Guide for <your DBMS>* (ArcSDE_Config_GD_<your dbms>.pdf), is to help database administrators perform a number of key tasks and procedures including:

- Installing the ArcSDE application server
- Configuring the ArcSDE application server for the network
- Starting, stopping, and monitoring ArcSDE server processes
- Tuning the relational database and optimizing ArcSDE performance on the network
- Setting up ArcSDE connections and user access to the database

The ArcSDE application server is often installed on large multiuser computer networks. Managing an ArcGIS system installation is another task for database administrators and requires a good understanding of the key parts of the system. The DBMS and data are located on a UNIX®, Linux, or Microsoft Windows server. The ArcSDE application server is typically on the same server machine as your DBMS, while the ESRI client applications are distributed on many machines across your network. All parts of the ArcGIS software system use some form of software authorization to "unlock" the various applications for use. To effectively manage this system, you should refer to the ArcGIS Desktop Install Guide, the ArcGIS Server Install Guide, the ArcInfo Workstation Install Guide, and a digital book named lmrefguide.htm on the ESRI Software Documentation CD–ROM.

The GIS administrator

The GIS administrator is responsible for designing and implementing an organization's geodatabase. The GIS administrator models a spatial system of features, surfaces, images, and other geographic information to design a geodatabase that supports the work of the organization. This is a key role for a successful multiuser GIS.

It's important to understand the spatial system to be modeled—what its objects are and the relationships between the various objects. For example, designing a geodatabase for land records requires an understanding of the relationships between parcels, surveys, ownership, and important administrative units such as zoning and easements. It also requires a complete understanding of geodatabase concepts and the best practices and methods for GIS database design.

The GIS administrator must be knowledgeable about geodatabase structures and how to use them to represent a real-world system. The GIS administrator must also understand how to apply computer tools, such as UML models and the ArcGIS system, to effectively implement a geodatabase design.

A number of key books are included in the ArcGIS package to help you accomplish these tasks:

- *Modeling Our World—The ESRI Guide to Geodatabase Design* introduces you to geodatabases and their contents. It can help you make decisions about the best strategies to employ in your geodatabase design. It describes in detail the objects and various parts of a geodatabase. *Modeling Our World* will help you with the design process and the key decisions you'll need to make along the way.

- *Building a Geodatabase* is the GIS administrator's guide to implementing a specific geodatabase design. Once you have completed your physical database design, this book contains all the necessary guidance to implement your geodatabase schema. It provides information on using ArcCatalog for these tasks and for using UML. UML is an excellent tool for object modeling and design. It is useful in implementing an integrated design for an entire business database including the geodatabase components.

If you haven't done so already, you might also want to read *Getting Started With ArcGIS*. It gets you started using the software immediately and refers to further ArcGIS documentation. *Getting Started With ArcGIS* contains an overview of the ArcGIS software system and has a series of exercises that leads you through a small GIS project.

Application developers have a number of opportunities to work with geodatabases. In the ArcGIS system, you can build custom applications or simply adjust the look and feel of existing applications with Visual Basic® for Applications (VBA) within the ArcMap and ArcCatalog applications. ArcInfo and ArcView also ship with ArcObjects—the complete set of Component Object Model (COM)-based components for ArcInfo and ArcView software. The ArcInfo superset of COM objects contains hundreds of COM classes with thousands of methods. Developers should look for more information in one of the following:

- *ArcGIS Desktop Developer Guide*
- *ArcGIS Server Developer Guide*
- *ArcGIS Engine Developer Guide*

available on the ESRI Software Documentation Library CD–ROM. ArcGIS Server also comes with a .NET ADF and a Java ADF.

Other ESRI software, such as ArcIMS, ArcView GIS 3, and MapObjects 2, can use the ArcSDE application server directly. In these systems, you can create custom applications that access spatial data through ArcSDE. Each software program provides its own ArcSDE programmer's interface that you can take advantage of as a developer.

ArcIMS

ArcIMS allows direct customization at all levels. At the client level, custom HTML and JavaScript™ can be used to modify the look and feel of the viewer. At the server level, ArcXML can be used to modify map configuration files—to project data, change the appearance of map features, and so on. It is also possible to incorporate custom and advanced services and tools.

ArcIMS supports a number of Internet developer tools including Active Server Pages (ASP) for Microsoft developers, and JavaServer Pages™ (JSP™) to build Web applications using Java 2 Platform, Enterprise Edition (J2EE™), and ColdFusion®. See the ArcIMS documentation set for more information.

ArcView GIS 3

The ArcView GIS 3 object-oriented programming language, Avenue™, can access ArcSDE functionality in addition to customizing the look of ArcView GIS 3. The subsystem that accesses and works with the ArcSDE application server is the Database Access extension. Refer to the book *Using Avenue* in the ArcView GIS package for additional guidance.

MapObjects Windows Edition

You can also develop applications with MapObjects Windows edition using Visual Basic, Visual C++®, and other COM-based developer tools. A part of the MapObjects 2 developer components describes building applications that access and work with ArcSDE. Two MapObjects user guides have more information on building MapObjects applications that use the ArcSDE application server: *Developing Applications With MapObjects* and *MapObjects Programmer's Reference*, both found in the MapObjects package.

ArcSDE CAD Client

With the ArcSDE CAD Client extension, you can work with true CAD entities in a geodatabase. The ArcSDE CAD Client extension has an extensive C language API that has complete access to all ArcSDE functionality. For more information, see *Using ArcSDE CAD Client*, included as a PDF file (usingcadclient.pdf) in the documentation folder on the ArcSDE Developer Kit CD–ROM.

The ArcSDE application server provides an open client interface to work with the contents of geodatabases as simple features. The ArcSDE simple feature API allows non-ESRI software applications to access and work with geodatabases. The ArcSDE client API is available in both C and Java. Refer to ArcSDE Developer Help, located in the documentation_sdk folder on the ArcSDE Developer Kit CD–ROM.

Typically, end users are those using a client application, such as ArcMap, which may be highly customized. Most end users do not know or even need to know they're using ArcSDE to access data in a DBMS. Working with a multiuser geodatabase is just like working with any other geographic data—use ArcCatalog to explore and work with data holdings; use ArcMap to map, edit, and analyze data from the geodatabase; and use ArcIMS to publish your maps on the Internet.

If you haven't done so already, you might want to read *What is ArcGIS?* and *Getting Started With ArcGIS*. *What is ArcGIS?* is a high-level overview of what ArcGIS software is all about. The *Getting Started With ArcGIS* book gets you started using the software immediately and refers to further ArcGIS documentation. It also contains an overview of the ArcGIS software system and has a series of exercises that leads you through a small GIS project. Other ESRI software programs also have books geared for end users.

ArcSDE resources

This book is designed to provide an introduction to ArcSDE and a brief explanation of its roles and benefits. For more detailed information about ArcSDE, you are encouraged to explore the additional resources available. These are listed and described below.

ArcSDE

ArcSDE Installation Guide: An HTML file that contains detailed installation instructions. The file is called Install_<your dbms>.htm and is located in the root directory of the platform-specific ArcSDE CD–ROM.

Managing ArcSDE Application Servers: Available as a printed book and as a PDF file on the ESRI Software Documentation Library CD–ROM.

ArcSDE Configuration and Tuning Guide for <your DBMS>: This is one of the DBMS-specific PDF files (ArcSDE_Config_GD_<your dbms>.pdf) in the documentation_server folder on the ArcSDE CD–ROM.

ArcSDE Developer Help: Contains developer information on C and Java and command references for ArcSDE administration commands, located in the documentation_sdk folder on the ArcSDE Developer Kit CD–ROM.

ArcSDE Administration Command References: Available as an HTML file called admincmdref.htm in the documentation_server/Admin_Cmd_Ref folder of each ArcSDE CD–ROM. Windows users also have access to a .chm version of the file for use with Internet Explorer.

ArcSDE CAD Client

Using ArcSDE CAD Client: Available as a PDF file (usingcadclient.pdf) in the documentation folder on the ArcSDE Developer Kit CD–ROM.

Glossary

annotation

Descriptive text used to label features. Used for display, not for analysis.

API

Application programming interface. Refers to a defined and documented set of tools or functions that application developers use to build or customize a program or set of programs. APIs can be built for programming languages, such as C, COM, and Java.

application server

A computer program that receives requests from a client application—another computer program that a user interacts directly with—and returns results to the client. See also client/server and server.

ArcSDE

ArcSDE is server software that provides ArcSDE client applications (for example, ArcGIS Desktop, ArcGIS Server, ArcIMS) a gateway for storing, managing, and using spatial data in one of the following commercial database management systems: IBM DB2 Universal Database (UDB), IBM Informix, Microsoft SQL Server, and Oracle.

ArcSDE for Coverages

An ArcSDE server that provides read-only access to ArcInfo coverages, shapefiles, ArcStorm library layers, and Map Librarian layers. Uses the same data transfer technology as ArcSDE for DBMS products.

ArcStorm

ArcStorm is a data storage facility and transaction manager for ArcInfo file-based coverage data. ArcStorm manages a feature-oriented database that can be closely integrated with a DBMS. Transaction management occurs via feature locking.

attribute

1. A characteristic of a geographic feature described by numbers, characters, images, and CAD drawings, typically stored in tabular format and linked to the feature. For example, the attributes of a well might include depth and gallons per minute.
2. A column in a database table.

band

One layer of a multispectral image or raster dataset. For example, a specific range of the electromagnetic spectrum of reflected light or heat—ultraviolet, blue, green, red, near infrared, infrared, thermal, radar, and so on.

behavior

Implementation of an object class method. The behavior is a procedure stored within the object that is activated when the appropriate method is called on the object. The same method on two object classes can have different behavior.

BLOB

Binary large object. The binary data type of a column in a DBMS table that stores large image, text, or geometry data as attributes.

CAD

Computer-aided design. An automated system for the design, drafting, and display of graphically oriented information.

CASE

Computer-Aided Software Engineering. Consists of tools and techniques that automate the process of developing software systems or schemas. Complex tasks that often require many lines of code to be written are simplified with CASE user interfaces and code generators.

client/server

A software system is said to have a client/server architecture when there is a central process (server) that accepts requests from multiple user processes called clients.

column

The vertical dimension of a table that holds attribute values. A column has a name and a data type applied to all values in the column. A table has rows and columns. See also table and row.

COM

Component Object Model. A binary specification from Microsoft Corporation that establishes a common way of building software components. COM objects have interfaces that contain methods and properties.

COM object

An object that implements its methods and properties through interfaces. COM objects have one or more interfaces and behave as servers that interact with clients through an interface. Client code grabs a COM object's interface and tells that object to perform a function or manipulate a property. COM components can be dynamically interchanged in a distributed system with Dynamic Link Libraries (DLLs are in-process servers) and executable programs (EXEs are out-of-process servers).

concurrency management

A database management process for maintaining the consistency of the data while supporting simultaneous access by more than one user. A typical technique is to use a system of locking data to prevent data corruption caused by multiple users editing and reading it.

conflict

In the context of versioning and reconciling edits to a database, a conflict occurs when two users submit edits that are in conflict with each other.

conflict resolution

The process of resolving conflicting edits that have been made in two or more versions of the same dataset. In the version reconciliation process, if the same feature in the source version and destination version have both been edited, the feature is said to be in conflict. The feature's conflict must be reconciled by user intervention.

coordinate

A set of numbers that designates location in a given reference system such as x,y in a planar coordinate system or x,y,z in a three-dimensional coordinate system. Coordinates represent locations on the earth's surface relative to other locations.

coordinate system

A reference system used to measure horizontal and vertical distances on a planimetric map. A coordinate system is usually defined by a map projection, a spheroid of reference, a datum, one or more standard parallels, a central meridian, and possible shifts in the x- and y-directions to locate x,y positions of point, line, and area features.

coverage

A file-based vector data storage format for storing the location, shape, and attributes of geographic features. One of the primary vector data storage formats for ArcInfo.

custom feature

A feature with specialized behavior in the geodatabase created as a COM object by a developer.

data dictionary

A set of tables containing information about data. ArcSDE and the geodatabase have data dictionary tables containing information about the GIS data in the database.

data integrity

Maintenance of data values according to data model and data type. For example, to maintain integrity, numeric columns will not accept character data.

data type

One of the forms of data that can be stored in a database table. Common data types are CHAR, DATE, LONG, NUMBER, and RAW.

database

1. A collection of related data organized for efficient retrieval of information.
2. A logical collection of interrelated information managed and stored as a unit, usually on some form of mass storage system, such as magnetic tape or disk. A GIS database includes data about the spatial location and shape of geographic features recorded as points, lines, areas, pixels, grid cells, or TINs, as well as their attributes.

database administrator

One who manages a database—sets up users, security, backup, and recovery procedures for all data and optimizes physical data storage for best performance.

database connection

A connection to a DBMS server, an ArcSDE application server, or an Object Linking and Embedding (OLE) database.

DB2

A commercial DBMS from IBM that is supported by ArcSDE.

DBMS

Database management system. A set of computer programs for organizing the information in a database. A DBMS supports the structuring of the database in a standard format and provides tools for data input, verification, and storage. Sometimes referred to as RDBMS.

domain

A named constraint in the database. This named constraint can be associated with a field for the subtype of a feature class or table to make an attribute validation rule.

feature

1. A vector object in a geodatabase that has a geometry type of point, line, polygon, or annotation. Features are stored in feature classes.
2. A representation of a real-world object in a layer on a map.

feature class

In a geodatabase, an object class that stores features and has a field of type geometry in a geodatabase. Feature classes can be standalone or integrated with other feature classes in a feature dataset.

feature dataset

A collection of feature classes that share the same spatial reference. Because the feature classes share the same spatial reference, they can participate in topological relationships with each other such as in a geometric network. Object classes and relationship classes can also be stored in a feature dataset.

field

Part of a table that holds one piece of data. The intersection of a row and a column.

geodatabase

An object-oriented geographic database that is hosted inside a DBMS. Object behavior is implemented using validation rules, relationships, and topological associations.

geographic data

The locations and descriptions of geographic features. The composite of spatial data and descriptive data.

geographic database

A collection of spatial data and related descriptive data organized for efficient storage and retrieval by many users.

geolocation

The process of creating features from tabular data by matching the tabular data to a spatial location. One example of geolocation is creating point features from a table of x,y coordinates. Points can also be created by matching addresses to streets.

geometric network

A graph or logical network composed of connected features. For example, networks include road systems, utility networks, and hydrologic networks. Geometric networks contain feature classes that have network roles such as simple junction features, complex junction features, simple edge features, and complex edge features. More than one feature class can have the same role.

grid
A geographic data model representing information as an array of equally sized square cells arranged in rows and columns. Each grid cell is referenced by its geographic x,y location. See also raster and grid cell.

grid cell
A discretely uniform unit that represents a portion of the earth such as a square meter or square mile. Each grid cell has a value that corresponds to the feature or characteristic at that site such as a soil type, census tract, or vegetation class. Additional values of the cell can be stored in a value attribute table (VAT).

image
Represents geographic features by dividing the world into discrete squares called cells. Examples include satellite and aerial photographs, scanned documents, and building photographs. See also raster.

index
A database tool put on a column or columns of a table and used to speed execution and impose uniqueness on data. Provides faster access to data than doing a full table scan.

Informix
A commercial DBMS from IBM that is supported by ArcSDE.

instance
See service.

join
Retrieval of data from more than one table in one query.

LAN
Local area network. A computer data communications technology that connects computers at the same site—for example, all computers in the same building. Computers and terminals on a LAN can freely share data and peripheral devices such as printers and plotters. LANs are composed of cabling and special data communications hardware and software.

layer
1. A feature class in a geodatabase, as in an SDE layer.
2. A cartographic rendering of a feature class, image, TIN, or graphic in a map.

linear feature
A geographic feature represented by a line or set of lines. A line connects two or more x,y coordinates. Rivers, roads, and electric and telecommunication networks are all linear features.

long transaction
An edit session on a feature dataset that may last from a few minutes to several months. Long transactions are managed by the ArcSDE versioning mechanism.

map projection
A mathematical model that transforms the locations of features on the earth's surface to locations on a two-dimensional surface. Because the earth is three-dimensional, a method must be used to depict a map in two dimensions. Some projections preserve shape; others preserve accuracy of area, distance, or direction. See also coordinate system.

metadata
Data about data. For GIS data, metadata typically means data that is designed to help a prospective user find GIS data for a specific purpose.

method
An action that an object is capable of performing. Objects that belong to the same class all have the same methods. For example, all form objects understand a method called "Show and Hide".

Microsoft Access
A commercial DBMS used for a personal geodatabase.

Microsoft SQL Server
A commercial DBMS that is supported with ArcSDE.

MSDE

Microsoft Database Engine. A client/server data engine based on Microsoft SQL Server. Can be used with the ArcSDE for SQL Server product.

multiuser geodatabase

A geodatabase in a DBMS served to client applications (for example, ArcMap) by ArcSDE. Multiuser geodatabases can be very large and support multiple concurrent editors. Supported on a variety of commercial DBMSs including Oracle, Microsoft SQL Server, IBM DB2, and Informix.

network trace

Navigates a network following connectivity defined by the geometric network. Specific kinds of network traces are:

- Find connected
- Find loops
- Find common ancestors
- Trace upstream
- Trace downstream

See also geometric network.

OGC

Open GIS Consortium, Inc. Go to the OGC Web page at www.opengis.org for more information.

Oracle

A commercial DBMS supported by ArcSDE.

personal geodatabase

A geodatabase, usually on the same machine as the client application (for example, ArcMap), that supports one editor at a time. Personal geodatabases are managed in a Microsoft Jet Engine database.

planar topology

Represents collections of topological feature classes that share geometry among their boundaries. One or more feature classes that share geometry participate in a common planar topology. Updating shared boundaries updates all features in the topology.

point

A single x,y coordinate that represents a single geographic feature such as a telephone pole.

polygon

A two-dimensional feature representing an area such as a state or county.

property

An attribute of a control, field, or database object that you set to define one of the object's characteristics or an aspect of its behavior. For example, the "Visible" property affects whether a control can be seen at run time.

pyramid

The storage of image data in multiple resolutions to speed query and retrieval.

query

A SQL expression to retrieve data from one or more tables.

raster

Represents any data source that uses a grid structure to store geographic information. See also grid and image.

RDBMS

Relational database management system. A set of computer programs for organizing the information in a database. An RDBMS supports the structuring of the database in a standard format and provides tools for data input, verification, and storage. See also DBMS.

record

A horizontal dimension to a table. A single row in a table. See also row and column.

referential integrity

The capability to ensure that changes to one table that affect other tables are transmitted automatically to those other tables. For example, a table will not be given a foreign key value that does not exist as a primary key in another table.

relationship
An association or link between two objects. See also relationship class.

relationship class
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 object classes, relationships are stored in relationship classes.

row
1. A record in an attribute table. The horizontal dimension of a table composed of a set of columns containing one data item each.
2. A horizontal group of cells in a grid or pixels in an image.

schema
1. The structure or design of a database or database object such as a table.
2. The definition of the database. The schema can either be modeled in UML using a CASE tool or defined directly within ArcCatalog using wizard dialog boxes.

SDE
See ArcSDE.

server
1. A computer program that receives a request from a client, processes it to generate results, and returns the results to the client.
2. A computer on which a server process runs.

service
A computer program that receives and processes requests from clients.

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.

SQL
Structured Query Language. A syntax for defining and manipulating data from a relational database. Developed by IBM in the 1970s, it has become an industry standard for query languages in most DBMSs.

SQL Server
See Microsoft SQL Server.

table
Information formatted in rows and columns. A set of data elements that has a horizontal dimension (rows) and a vertical dimension (columns) in an DBMS. A table has a specified number of columns but can have any number of rows. See also attribute.

TCP/IP
The Transmission Control Protocol (TCP) is a communication protocol layered above the Internet Protocol (IP). These are low-level communication protocols that allow computers to send and receive data.

topology
The spatial relationship between features. See also planar topology.

transaction
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).

UML
Unified Modeling Language. A standard for representing object designs. See also CASE.

validation rules
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.

vector model

A representation of the world using points, lines, and polygons. Vector models are useful for representing and storing discrete features such as buildings, pipes, or parcel boundaries.

version

A version is an alternative representation of the geodatabase that has an owner, a description, and a level of access (private, protected, and public).

version merging

The process of reconciling two versions of a feature dataset into a common version. If conflicting edits have been made in either of the merged versions, these conflicts are resolved, either automatically or by an interactive process.

version reconciliation

The process of updating a version of a dataset with changes made in another version. Using this technique, a version can remain up-to-date with changes even if it is within a long transaction lasting many months.

vertex

One of a set of ordered x,y coordinates that defines a line or polygon feature.

WAN

Wide area network. A computer data communications technology that connects computers at remote sites (for example, city to city) to computers at other remote sites. WANs are composed of special data communications hardware and software and usually operate across public or dedicated telephone networks. Typically, data access over a WAN is slower than over a LAN. See also LAN.

XML column

A column or columns in a business table of type XML (eXtensible Markup Language), used to store XML documents. XML columns will always have a full text index that lets you search for a word anywhere in a document.

Information about an XML column is maintained in an ArcSDE system table named xml_columns; information about its text indexes is maintained in the ArcSDE system tables xml_indexes and xml_index_tags.

See also XPath index and XML document.

XML document

A textual object made up of entities, elements, tags, attributes, and character data. Used to store such things as metadata documents that are published to an ArcIMS Metadata Service.

XPath index

Enables you to specify and search for content of a specific XML element or attribute in each document. The definition of which elements and attributes are included in or excluded from each XPath index is recorded in the table SDE_xml_index_tags.