Understanding **ArcSDE** Table Relationships

By Keldah Hedstrom, ESRI Educational Services

Effective database design typically includes normalizing data to minimize storing the same data multiple times. In order to access data in all related database tables, links or associations can be created between tables. These links are created on common columns between the tables that will be associated.

Associations can be established between multiple data formats such as shapefiles, ArcInfo tables, feature classes in personal geodatabases, or tables in an ArcSDE enterprise geodatabase. These relationships can be temporary or permanent. Temporary relationships exist within a specific ArcMap document while permanent relationships are available to multiple geodatabase users.

There are different ways to create relationships between tables in ArcGIS. In ArcMap, relationships can be created by joining or relating tables together. Relationship classes can be created in ArcCatalog. Tables can also be associated by creating ArcSDE spatial and nonspatial views.

The Basics of Table Associations

This article briefly discusses the basics of table association, different methods available for creating relationships, and how to optimize the performance of these relationships. Although most relationships can be applied to both ArcSDE and non-ArcSDE data, this article addresses relationships applicable to ArcSDE data.

The basics of creating relationships between tables include finding the common information between the tables as well as understanding how the values in each join column match. In order to associate tables, common information has to exist in both tables and be stored as the same data type in each. The names of the columns to be related do not have to be the same; however, the data type and values in both join columns must be the same.

Types of Cardinality

You need to know how individual record values in the common column of each table will connect to one other. The way tables are related to one another is called cardinality. There are four cardinality types: one-to-one, one-to-many, many-to-one, and many-to-many. It is important to understand which relationship type is present between the tables to prevent potential record omission errors.

For example, if a one-to-many relationship exists but the tables are connected using a one-to-one relationship, information will be omitted from the connected table because a one-to-one search stops looking for more matches after the

Attributes of PARCELS

PARCELS_ID	APN	GENPLAN	LANDUSE	ZONING	BOOK	PAGE	BLOCK
2103	017102103	620	1	C-4	0171	02	1
2111	017102111	620	0	C-4	0171	02	1
2110	017102110	210	5	I-3	0171	02	1
2109	017102119	620	0	C-4	0171	02	1

Attributes of LANDUSE

LANDUSE	LU_DESC	NO_PARCELS	LAND_VAL	PROP_VAL
0	Vacant Industrial	41	871027	270,642
11	City Parks	7	0	0
14	City-Owned Property	47	0	0

An ArcSDE geodatabase feature class called Parcels can be associated with the ArcSDE geodatabase attribute table called Landuse based on a common column called LANDUSE, which stores the same type of values or data type in both tables.

Attributes of PARCELS

PARCELS.PARCELS_ID	PARCELS.GENPLA	LANDUSE.LU_DESC	LANDUSE.NO_PARCEL
2103	620	City Parks	7
2111	620	Vacant Industrial	41
2110	210	Vacant Industrial	41
2109	620	City Parks	7

The ArcSDE Parcels feature class and the ArcSDE Landuse attribute table are joined based on the LANDUSE column. All columns from both tables are combined into one single join output table. A specific naming convention helps to prevent duplicate field names when the target table and a join table have common field names. The naming convention for columns in the output join table is TableName.FieldName.

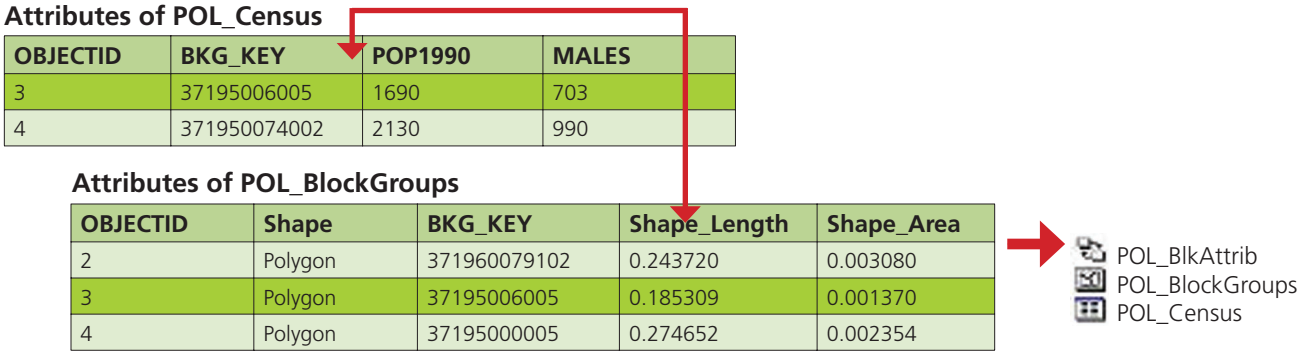
Attributes of Coffee

FID	Shape	ID	NAME	Coffee_ID
2	Point	2	The Coffee Place	2
3	Point	3	Coffee Corner	3
6	Point	4	The Percolator	4
7	Point	5	Java Quik	5

Attributes of Percent_own

COFFEE_ID	OBJECTID	OWNER_ID	PER_OW	Own_ID
4	5	40	100	40
5	6	40	75	40
5	7	50	25	50

The Coffee feature class and the Percent_own attribute table are related together based on the COFFEE_ID column from both tables. When the relate is created, the tables remain separate and are not virtually appended together. The related Percent_own table is viewed by pressing the Options button and then choosing Related Tables and choosing the relate that was previously created.



The relationship class POL_BlkAttrib is created between the POL_Census attribute table and the POL_BlockGroups feature class based on the BKG_KEY column from both tables. In ArcMap the relationship class behaves like a relate as both tables remain separate.

first match is found. After designing tables that include common values, you can start creating the relationships between the tables.

Using Joins in ArcMap

Joins in ArcMap are the first type of table relationship between ArcSDE feature classes and tables. When tables are joined in ArcMap, all columns from both tables are virtually appended together into one large table without affecting the source data. The tables are not appended together in the database; it is just a temporary association created in the map document. This join can be saved in an individual ArcMap document or as a layer file, which allows you to create multiple joins between ArcSDE feature classes and attribute tables.

When creating joins in ArcMap, only one-to-one or many-to-one cardinality is permitted. Incorrect cardinality results in omitted records. Joining tables using correct cardinality will allow you to symbolize or label features based on attributes from the joined table. Joins can be used to complete a dataset. For example, a feature class of GPS points and a table of attributes can be joined, and the table can be exported to make a new permanent dataset.

Using Relates

The second type of relationship that can be created is an ArcMap relate. By relating tables, a relationship is established between the tables. However, unlike a join, the columns from related tables are not virtually appended together and the tables are viewed separately.

Relates differ from joins in that they can be created regardless of the cardinality between tables without the danger of omitting information. Relates can be created between tables of all cardinality types. However, relates will not allow you to symbolize or label features based on attributes from the related table.

By default, when a selection is made in one table, the related records are not automatically updated in the corresponding table. The relationship must be refreshed. This is done by selecting the Options button and selecting the specific relationship to see the related records. However, the Auto Relate Tool, available from the ArcObjects Online Web site, can be added into the ArcMap interface. This tool will automatically refresh the selection between related tables.

Relates, like joins, are temporary associations saved in an ArcMap document that can also be applied across multiple data formats. It is advantageous to use relates when you have redundant information in both tables and do not want to make one large table with repeating values. Relates are often used with lookup tables to maintain a relationship between the table that has all of the descriptive information and the table with the corresponding values.

Using Relationship Classes

A relationship class is the third type of relationship that can be created for ArcSDE data. A relationship class behaves like a relate in that it is an association that has been created between two tables. However, a relationship class is a permanent connection between tables that must be physically removed. This allows you to create a persistent connection between tables that can be accessed by many geodatabase users or used in multiple ArcMap documents.

Relationship classes, or stored relationships, are created in ArcCatalog and used in ArcMap. After a relationship class is created, the permanent relationship is symbolized and stored in the geodatabase. This relationship class object stores the properties of the permanent relationship in the geodatabase.

Unlike relates, a relationship class can only be created from tables with one-to-one, one-to-many, or many-to-many cardinality and is only available to one geodatabase. This means that you can only create a relationship class between tables with the correct cardinality residing within the same geodatabase. Relationship classes between different geodatabases cannot be created.

A relationship class is more sophisticated than a relate. The user can apply behaviors and rules to tables connected in the relationship class. For example, a relationship class between a Fire Station feature class and a Fire Station Employees attribute table has been created. Relationship rules can be set that allow the user to specify that each fire station has to have between five and 10 employees.

Relationship classes are also very useful when editing data. When participating in a relationship class, the properties of all of the objects that are related can also be edited. Moreover, breaking or creating new relationships between the features can be performed.

Using ArcSDE Views

The fourth type of relationship, an ArcSDE spatial or nonspatial view, can only be created between ArcSDE data. ArcSDE views can associate multiple ArcSDE tables that will appear as a single unit to the end user. ArcSDE imposes the limit of one spatial column per table; that limit also applies to views created by the create_view operation.

When creating a spatial view, you must include the spatial column from the ArcSDE feature class. If a spatial column is not included or if the view is made between two attribute tables, then the view will be nonspatial (i.e., a typical DBMS view).

Views are permanent SQL queries stored in the database. A view references a collection of related tables and can filter out or look at a subset of data from different tables. Views allow the database administrator to grant specific users

Continued on page 30

```
Figure 1:
Sdetable -o create_view -T road_view -t road_geometry,road_info -c
shape,name -w "road_geometry.road_id = road_info.road_id" -D wilson -u
myusername -p mypassword
```

Understanding **ArcSDE** Table Relationship

Continued from page 29

Attributes of wilson.WILSON.road_geometry

OBJECTID	ID	SHAPE	SHAPE.len
1	180	Polyline	1146.942571
2	183	Polyline	3802.632909
3	208	Polyline	11637.871155
4	209	Polyline	2133.499011
5	228	Polyline	93.204428

The spatial view Road_View was created using the sdetable –o create_view command. Road_View was created between the Road_Geometry feature class and the Road_Info attribute table. Road_View is a spatial view because the SHAPE column has been included from the Road_Geometry feature class. The tables were joined together by relating the ID column from the Road_Geometry table and the ROAD_ID column from the Road_Info table. Road_View only contains the SHAPE column and the NAME column, which were specified in the command when creating the view.

Attributes of wilson.Wilson.ROAD_INFO

OBJECTID	ROAD_ID	MONTH	YEAR	NAME	LEFT	LEFT	RIGHT	RIGHT
1	2	0	1900	US 301	0	0	0	0
2	3	0	1900	US 301	0	0	0	0
3	4	0	1900	US 301	0	0	0	0
4	5	0	1900	RAILROAD ST	0	0	0	0
5	6	0	1900	MAIN ST	0	0	0	0

Attributes of WILSON.WILSON.ROAD_VIEW

shape	name	shape.len	SHAPE.id
Polyline	STRICKLAND RD	1146.942571	1
Polyline	OLD SMITHFIELD RD	3002.632909	2
Polyline	GREEN POND LP	11637.871155	3
Polyline	OLD BAILEY HWY	93.204428	5

access to views without giving them access to the underlying tables or designated columns/rows of the tables that make up the views.

When creating views, remember that ArcMap supports only one-to-one and many-to-one cardinality for views. Views are created using the sdetable–o create_view command as illustrated in Figure 1 and require parameters that specify the name of the view, the tables used, the columns from each table, and how the tables are joined together. Once created, the view is a permanent object in the ArcSDE geodatabase.

Views are accessed in the same manner as a table or feature class stored in the database. A spatial view behaves like a feature class. It can be symbolized, labeled, and queried like any other feature class in ArcMap. A nonspatial view behaves like a table in ArcMap and can be joined or related to other datasets.

Most views accessed using an ArcSDE client, such as ArcMap, are read only and cannot be directly edited in the client. However, you can manipulate the columns participating in the view and these changes will be reflected in the view, when it is refreshed in ArcMap. Changing the participating columns of the view requires accessing the SQL SELECT, FROM, or WHERE statement and updating the statement to pass different columns from a specific table or change how the columns have been joined together.

Optimizing Performance

Once a relationship has been created between ArcSDE tables, indexes are built on specific table columns and statistics are calculated for each table. These procedures optimize the performance of table relationships by helping the database find the best execution plan for retrieving records.

Indexes allow a database to quickly search tables for specific records in the same way an index in the back of a book helps the reader locate information on specific topics. Indexes can be created in ArcCatalog, using the sdetable –o create_index command, or directly in the database. It is recommended that indexes be created on all columns that will be joined between tables and feature classes.

Calculating statistics on the tables and feature classes in the database is another way to enhance performance. Calculating statistics allows the database to choose the best execution plan in searching for records within tables. The database will compute the number of rows in each table as well as the number of unique values in each table. Statistics improve performance when creating joins and searching through the database for specific records.

Statistics can be calculated for tables and feature classes in ArcCatalog, using the sdetable –o update_dbms_statistics command, or directly in the database. Statistics should be updated or recalculated when the data in the tables changes significantly due to appending data into tables or feature classes or after compressing a versioned database. Recalculating database statistics allows the database to reformulate the best execution plan.

Conclusion

Designing a geodatabase often includes associating specific tables and feature classes using different relationship types. The different methods available for creating associations between tables in ArcGIS include ArcMap joins and relates, ArcCatalog relationship classes, and ArcSDE views. Building indexes and calculating statistics after creating a specific relationship between tables are recommended to optimize the performance of associations.

For more information on creating relationships in ArcGIS, refer to ArcGIS online help; ArcSDE online developer help; and *Understanding ArcSDE Table Relationships,* a free training seminar offered by the ESRI Virtual Campus.

About the Author

Keldah Hedstrom received a bachelor of science degree in natural resource management and spatial information management systems from Colorado State University. While in college she worked for the Farm Service Agency of Oregon building a centralized GIS database for multiple counties. She began working for ESRI more than a year ago and currently works in the Educational Services Department as an ArcSDE instructor for the ESRI Redlands campus.