

第 1 章 Geodatabase

1、1 Geodatabase 概述

Geodatabase 概述

ArcGIS 操作基于 GIS 文件格式和存储于 geodatabases 中的地理信息。Geodatabase 是 ArcGIS 的本地数据结构，是用于编辑和数据管理的基本数据格式。

Geodatabase 能将空间数据存储在文件、MDB 文件或者大型 DBMS 中。以上三种存储方式的区别在于可存储数据量的不同以及可支持的并发用户数量不同，能够实现从小数据量、单用户的文件数据库到大数据量、多用户并发编辑的企业级 DBMS 的不同层次的应用。

理解 geodatabase

简介：geodatabase 是大量不同类型的地理数据集的集合。在这一章节可以学习 geodatabase 的基础，这些概念能够为学习和有效使用 geodatabase 打下坚实的基础。

Geodatabase 定义：geodatabase 是大量不同类型的地理数据集的集合，这些地理数据集可以存储在普通的文件中、微软的 Access 数据库中或者多用户的关系数据库（比如 Oracle, Microsoft SQL Server, 或者 IBM DB2）。

Geodatabase 中主要数据集：数据集是 geodatabase 中的一个重要概念。它是 ArcGIS 用来组织和运用地理信息的基本机制。Geodatabase 包含三种主要的数据集类型：要素类、栅格数据集和表。

在创建 Geodatabase 时，首先生成不同的 Dataset 类型，然后添加或者扩展 Geodatabase 基本要素的能力，例如添加拓扑、网络、子类以实现 GIS 行为建模、维护数据完整性以及建立空间关系。

Geodatabase 的存储

Geodatabase 的存储不仅包括简单的空间坐标和属性数据的表格，还包括这些地理数据集的模式和规则。Geodatabase 的三种基础数据集（要素类，属性表和栅格数据集）和其他的 geodatabase 元素都以表格的形式存储。在地理数据集中空间表示或者以矢量要素的形式存储，或者以栅格数据存储。几何对象和传统的属性字段一起存储在表的列中。

Geodatabase 将地理要素以表格的形式存储，每行记录代表一个要素。下表是多边形要素，在 Shape 列为每个要素存储了多边形类型，值 Polygon 用于描述 Shape 列的几何坐标和几何形状，表示在每一行定义了一个 polygon。

Geodatabase 的一个关键策略就是利用 RDBMS 来管理从简单要素集到海量数据集，多用户并发操作的 GIS 数据集。二维表为几何数据集提供了基本的存储机制。SQL 语言具有强大的查询和操作表的功能，geodatabase 正是基于想利用这些功能而设计的。

空间数据在下列数据库中支持 Sql 访问空间信息：

可带或不带 oracle spatial 和 oracle locator 的 oracle

IBM DB2

IBM Informix

内在的 SQL API 是基于 标准 Sql Spatial 和 OGL 的简易特征 SQL 规范的，这一规范对空间向量类型以标准 Sql 扩展。

用高级数据类型扩展要素类、栅格数据集和属性表

大量的 geodatabase 元素用来扩展简单的表，要素类、栅格数据集，包括添加丰富的行为，数据完整性、数据管理等能力。Geodatabase 的模式包括定义、完整性规则和行为。

具体包括 coordinate systems, coordinate resolution, feature classes, topologies, networks, raster catalogs, relationships, domains 等等。模式信息存储在 DBMS 的 geodatabase 元数据表的集合

中，这些表定义了数据的完整性和行为。

Geodatabase 元素

不管 ArcGIS 用户使用何种操作系统，他们都操作三种的基本数据集类型。即一系列的要素类（如 ESRI 的 shapefiles），或者大量的属性表（如 dBase 文件、Microsoft Access 表、Excel 电子表格、DBMSs 等等），还有大量的影像和栅格数据集。

原则上，所有的 geodatabase 都包含相同类型的内容，设计 GIS 数据库的第一步就是生成不同类型的数据集。

实际上，用户需要扩展他们的数据模型以支持特定的能力，geodatabase 有大量数据元素和数据集类型用来扩展基本的数据集类型。

Geodatabase 的事务处理和版本管理

ArcSDE geodatabases 支持版本和长事务处理

Geodatabase 支持丰富的数据类型，如 annotation, topology, networks, terrains, and address locators 等等，这些类型都可以应用于海量、高性能的数据库。Geodatabase 同时还支持长事务框架，以支持多种数据管理工作流和操作。

大多数情况下需要多用户并发编辑

以 Check out 和 check in 方式更新

多个复制品之间由共享只供改变更新的多个同步拷贝可以是 DBMS 类型的任何数字（譬如 Oracle 和 SQL 服务器）并且不需要被连接。

创建、管理和使用历史归档数据

1、2 表基础

表基础

Geodatabase 中的属性存储在二维表中，建立在以下一系列简单但根本的关系数据概念：

表包含行。

表中所有的行含有相同的列。

每一列具有一个数据类型，如 integer, decimal number, character, and date。

一系列的相关函数和操作符（如 SQL）可以用于操作表和表的数据元素。

和在传统的数据库中的应用一样，表和关系在 ARCGIS 中同样扮演着重要的角色。表中的行可以用于存储地理对象的所有属性，包括在 Shape 列存储和管理要素的几何图形。

下表解释了两个表如何通过使用一个公共的字段使它们的记录相互关联。

Geodatabase 支持的属性数据类型

在 geodatabase 中支持很多数据类型存储和管理属性列，包括很多数值类型、文本类型、日期类型、二进制大对象类型和全球唯一标识码（GUIDS）。

在 Geodatabase 的数据表支持的数据类型包括：

Number: 包括 short integers, long integers, single-precision floating-point numbers (通常称为单精度浮点类型)和 double-precision floating-point numbers (通常称为双精度浮点类型)。

Text: 任何一定长度的文字数字式字符的集合。

Date: 存储日期和时间数据。

BLOBs: 二进制大对象用于存储和管理二进制信息，如符号和 CAD 几何图形。

Global Identifiers: GlobalID 和 GUID 数据类型存储用波形括号括起来的由 36 个字符组成的 registry style 字符串。

XML 列类型也可以通过程序接口支持，XML 列可以包含任何格式化 XML 内容（如元数据 XML）。

扩展表

在 geodatabase 中表为要素类、栅格数据集和传统的属性表提供了详细的信息描述，用户也可以实现对表的关系操作。

在 geodatabase 中有以下一些可选的功能用于扩展表的能力，它们包括以下几种：

1、3 要素类基础

要素类基础

要素类是具有相同几何类型和属性的要素的集合。在 geodatabase 中常用的要素类有四种：点、线、多边形和注记。

在下面的图示中，点、线、多边形和注记用于表示同一地区的四种数据集。（1）入孔井盖用点表示（2）下水道用线表示（3）地块用多边形表示（4）街道名称用注记表示。

在这个图示中，也许已经注意到一些潜在的对高级要素属性建模的要求。比如，下水道的管线和入孔井盖组成了一个雨水管线网络。同时，相邻的地块共享了公共的边界，大多数地块用户通过使用 topology 来保持在数据集中共享要素的完整性。

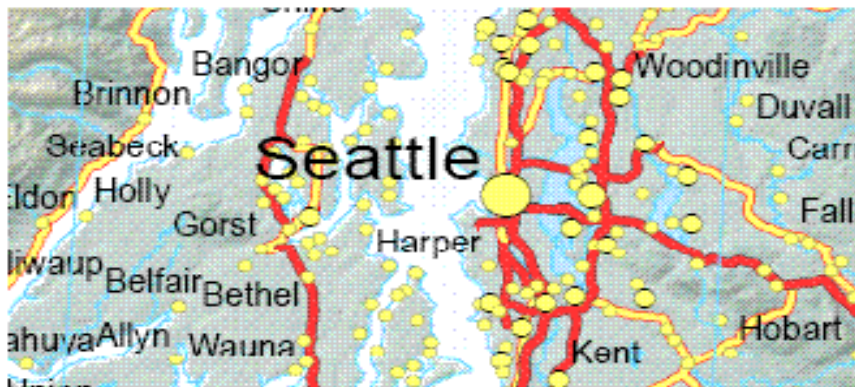
Geodatabase 中要素类的类型

矢量要素是最常用的表达地理数据的数据格式，它适合于表达离散的有明显边界且稳定存在的地理对象。如井、街道、河流、州和地块。一个要素是一个简单对象，它存储其几何图形，它可以是一个点、线或者面，作为这条记录中的一个属性或者字段。在 ArcGIS 中，要素类是具有相同空间参考和属性设置的相同要素的集合。例如，用一个线要素来表达道路中心线。

注意：当在 geodatabase 中创建一个要素类时，需要设置要素类型以定义要素类的类型（点、线、多边形等等）。

通常，要素类是点、线或多边形的集合，具体有 7 种要素类型。

- 1、点—用来表达那些很小且不能用线或多边形来表示的地理要素（如 GPS 观测站）。
- 2、线—用来表达那些长条形的，非常狭窄也不能用多边形表示的地理要素，如街道中心线和溪流。线也可以表达那些有长度却没有面积的要素，如轮廓线和边界。
- 3、多边形—是一个封闭的图形，用来表示均质要素的形状和位置，如州、县、土地、土壤类型或者土地利用类型区等。
- 4、注记—地图文本，包括文本如何组织的属性；例如，除了每个注记的文本字符串。还包括其他属性，如放置文字形状，字体，字体大小和其他显示属性。注记也可以是 feature-linked，也可以包含子类。



5、维度——一种特殊的注记，它显示具体的长度或距离；例如，为了显示一个建筑物或一块地的一条边的长度，或者两个不同要素之间的距离。维度经常用于 GIS 设计、工程和工具应用中。

6、多点——由不止一个点组成的要素。Multipoints 通常用于管理大量点集合的数组，如激光雷达点串，它们可以包含几十亿个点。使用一个单独的列表示这样的点几何是不可行的。聚类这些多点行可以使 geodatabase 处理大块点集合。

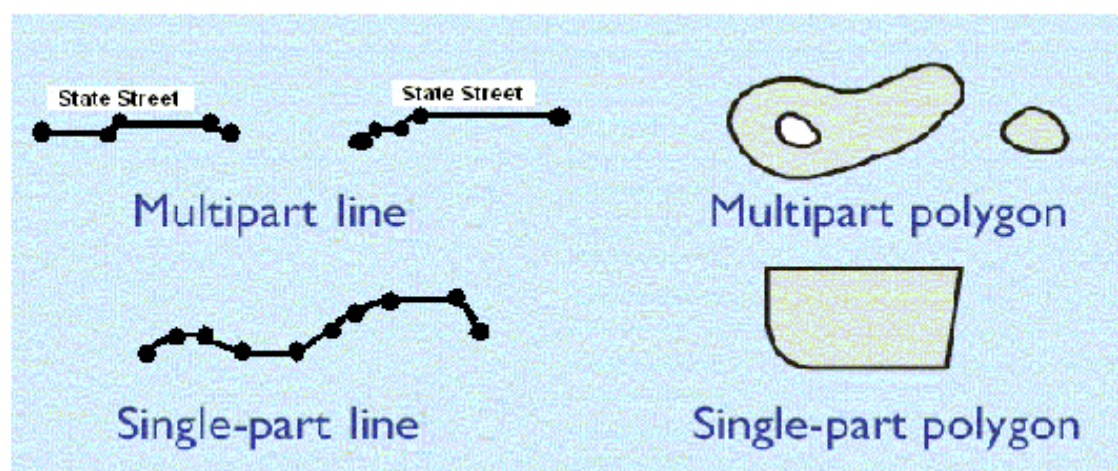
7、Multipatches——一个 3D 几何图形用于表达要素的外部表面或者外形，这个要素占用一个离散的二维面积或者三维空间的体积。Multipatches 由平面的 3D 环和三角形组成。Multipatches 可以被用于表达任何简单到复杂的对象，如球体和立方体到表面和建筑物。

要素的几何特征系统

要素类不仅包含每一要素的几何形状，还包含描述要素的属性。一个要素可以用点、线或多边形三种要素类型来定义，但是其他的地理属性也可以定义，如要素是单部件或多部件，可以有 3D 定点，可以有线性测量（称为 m-值），可以曲线。这一部分对这些功能作了简单介绍。

单部件和多部件的线，多边形

Geodatabase 中线和多边形要素可以有单部件或多部件组成。如，一个州可以包含多部件（夏威夷岛），但是通常认为它是一个单独的要素。



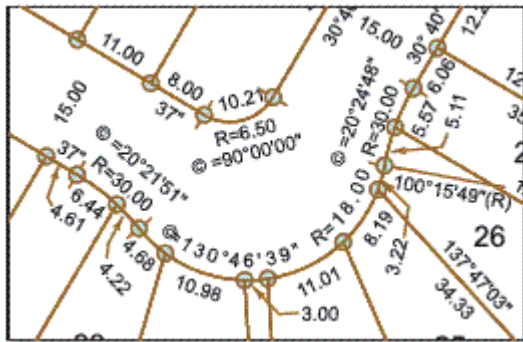
顶点、线段、高程值和测量值

要素几何图形主要是由坐标顶点组成的。线和多边形要素中的线段跨多个顶点。线段可以是直线边线或者是参数定义的曲线。要素中的顶点也可以包含表达高程值的 Z 值和表达沿着线要素的度量值的 M 值。

线段类型

线段和多边形可以由两个关键元素定义（1）定义线段或多边形的几何形状的一系列有序定点（2）连接每对定点之间的线段类型。因此，每一条线段或一个多边形可以认为是由一系列可以连接的有序定点形成的几何形状。还有一种表达线或多边形的方法是：每一条线段或一个多边形是一系列有序的可以连接的线段，每一线段可以是以下类型中的一种：直线

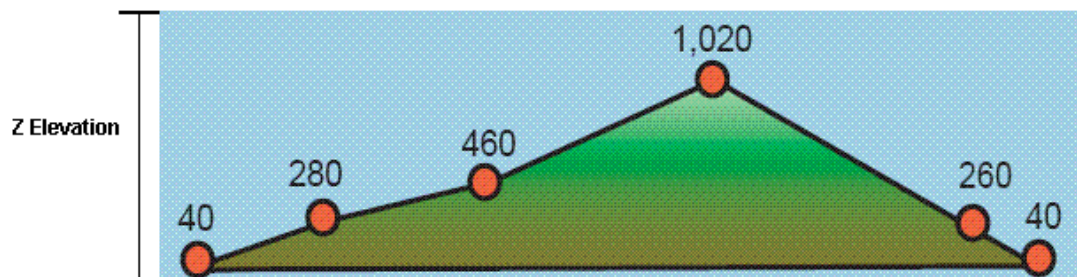
段、圆弧、椭圆弧或 Bezier 曲线。



默认的线段类型是直线段，但是，当需要定义 curves 或 parametric shapes 时，可以有三种其他线段类型选择：圆弧、椭圆弧或 Bezier 曲线。这些几何形状常常用于表达建筑环境，如地块边界和车道线。

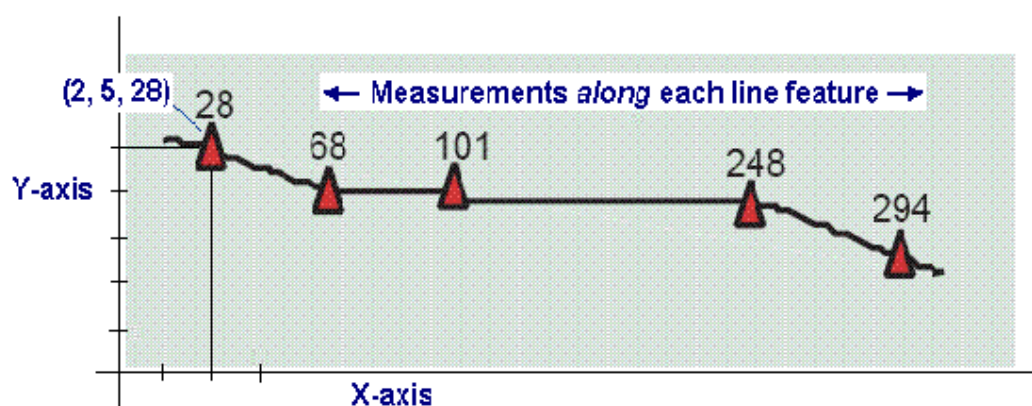
用 Z 值作垂直量度

要素的几何坐标可以包括 x、y 或 x、y、z。Z 值大多数情况下表示高程值，但它们也可以用来表示其他的属性，如降雨量等或空气质量值。



用 m 值作线性量度

线性要素的每一个顶点也可以指定一个 M 值或线性量度。一些 GIS 应用利用线性量度方法沿着线性要素内插距离，如道路、溪流和管线。可以给点、多点、折线或多边形的每一个点指定一个 M 值或线性量度。线性度量方法的一个典型例子就是公路或运河沿线的里程碑或码头。



含线性量度的顶点的坐标可以是 (x, y, m) 或者 (x, y, z, m)。

支持这些数据类型通常称为线性参照 (Linear Referencing)。沿着这些量度系统的发生的地理定位事件的过程称为动态分割 (Dynamic Segmentation)。

在 ArcGIS 的线性参照的实现中，术语 route 引用任何线性要素，如一个城市街道、高

速公路、河流或者管道，它具有唯一一个标识符和沿着每个线性要素的一个通用度量系统。用一个通用度量系统的 routes 集合建立在一个线性要素类上，如下所示：

要素容差

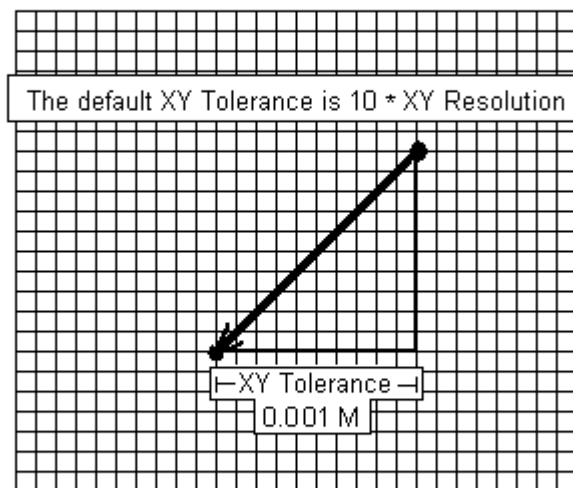
在各类 ArcGIS 操作中，使用一些关键的地理属性来处理和管理要素坐标。这些属性在每个要素类或者要素数据集创建的时候被定义。

X,y tolerance

当创建一个新的要素类时，将被提示设置 x,y tolerance。x,y tolerance 被用于在聚类操作中设置坐标之间的最小距离，如拓扑验证、缓冲区生成和多边形叠加等编辑操作。

要素处理操作是被 x,y tolerance 影响的，它决定了在这些操作中所有要素坐标的最小距离。通过定义，它也定义了聚类操作中一个坐标在 x 或 y 方向（或者两者）可以移动的距离。

Points within both the X and the Y dimension of the XY Tolerance are moved to the same location during clustering operations.



The minimum distance between coordinates that have been clustered will be:

$$\sqrt{2} * XY \text{ Tolerance}$$

x,y tolerance 是及其小的距离。它用于在聚类操作中解决坐标之间不精确的相交位置。当使用几何操作处理要素类时，如拓扑验证、多边形叠加、缓冲分析和裁剪等，坐标之间的 x 距离和 y 距离落在 x、y tolerance 内时，则被认为是一致的点。因此，聚类坐标被移动到一个公共的位置。通常，较低精确的坐标被移动到较高精度坐标的位置，或者一个新的位置在聚类的坐标之间作为一个加权平均值距离被计算。在这种情况下，加权平均值距离是基于聚类坐标的精度级别的。

聚类处理通过移动整个地图内的落入 x、y tolerance 的坐标来进行。ArcGIS 使用这个运算法则发现、清除和管理要素之间的共享几何图形。这意味着这些坐标被认为是一致的。这是许多 GIS 操作和概念的根本。

在这些操作中一个坐标可以移动到新位置的最大距离是 $\sqrt{2}$ 倍 XYTolerance。

默认的 x、y tolerance 是 0.001 meters（地理坐标系）。例如，如果是在 state plane feet 坐标系中，默认的 x、y tolerance 是 0.0003281 feet (0.003937 inches)。如果是 latitude-longitude 坐标系，默认的 x、y tolerance 是 0.000000008982995 degrees。

x、y tolerance 的默认值是 10 倍的默认 x,y resolution，这是在大多数情况下推荐使用的。

注意：x、y tolerance 不是打算用于生成几何形状。相反，它是用于在拓扑操作中集成线和边界的。这意味着集成彼此落入非常小距离的坐标。

有用的提示：

一般地，使用 10 倍 x,y resolution 的一个 x,y tolerance；

为了保持较小的移动，保持 x,y tolerance 很小。但是，一个太小的 x,y tolerance 可能不能准确地集成一致性边界的线元素和坐标；

相反，如果 x,y tolerance 太大，要素坐标可能相互覆盖。这可能降低要素边界表达的准确性；

x,y tolerance 应该永远不要接近 data capture resolution；

在拓扑中，可以设置每个要素类的 coordinate rank。

X,y resolution

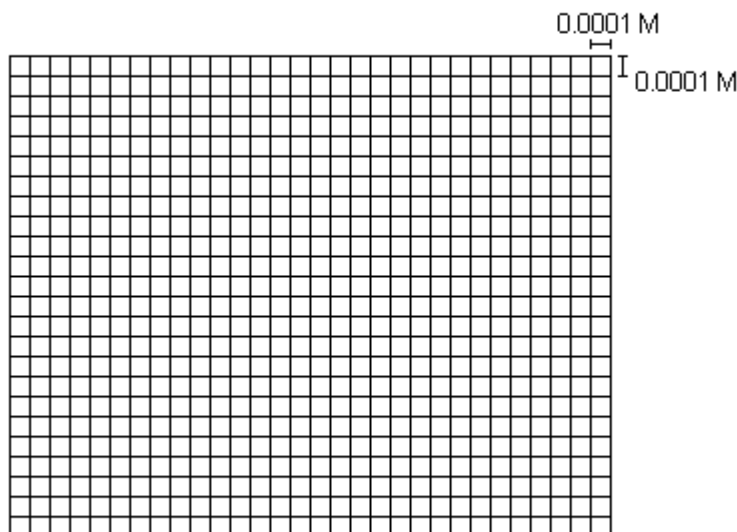
要素类或者要素数据集的 x,y resolution 是用于存储 x、y 坐标值的数值精度。精度对准确的要素表达、分析和制图是很重要的。

x,y resolution 定义了小数点后可以保留的小数位，或者存储要素坐标的有效数。坐标值在 ArcGIS 以 integers 被精确存储和操作。

默认的要类 x,y resolution 在 ArcGIS9.2 中是 0.0001 meter (地理坐标)。如果要素是被存储在 State Plane feet 中，默认的 x,y resolution 是 0.0003281 feet (0.003937 inches)。如果是 latitude-longitude 坐标系统，则默认的 x,y resolution 是 0.000000001 degrees。

以下是一个关于 x,y resolution 的概念视图的图表。

The XY resolution defines the fineness of a grid mesh that covers the extent of your feature class or feature dataset.



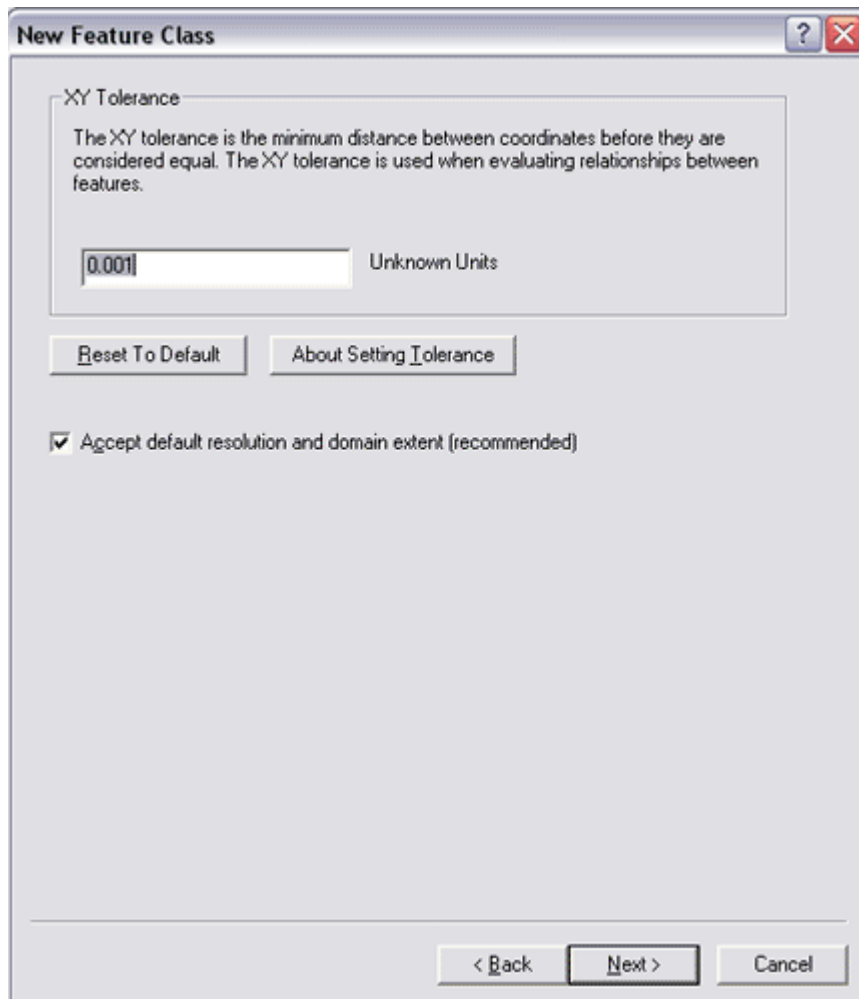
All XY coordinates snap onto this grid mesh.

The XY resolution defines the spacing of the grid mesh.

如果必要，可以不用管默认的 x,y resolution 值，设置要素类或者要素数据集的另一个 x,y resolution 值。设置一个较小的 x,y resolution 值，相比一个较大的 x,y resolution 值，可能潜在地增加了数据存储和处理的时间。

设置 X,y tolerance 和 X,y resolution

当在 ArcCatalog 中建立一个新的要素类，将被询问设置新要素类的 x,y tolerance。数据集坐标系统的默认 x,y tolerance 值如下面板所示：



在大多数情况下，默认的 x,y tolerance 是最好的选择。

在这个面板上，也可以选择接受默认的 x,y resolution，它是 0.0001 meters 或者与坐标系单位相等的量。默认的 x,y resolution 在大多数情况下，是合适的，除非需要比测量和土木工程更好的精度存储坐标值。

为了设置自己的 x,y resolution，取消选择上面面板中的 Accept default resolution 复选框，到下一个面板中设置需要的 x,y resolution 值。

Z-resolution, z-tolerance 和垂直坐标系统

Z-coordinates 主要用于包含高程的 GIS 要素中。但是，Z-coordinates 也可以被用于其他垂直度量，如空气污染观察数据、温度和其他度量值。

当建立一个新要素类时，需要包含 Z-coordinates，可以选择 Coordinates include z values 复选框：

New Feature Class

Name:

Alias:

Type

Type of features stored in this feature class:

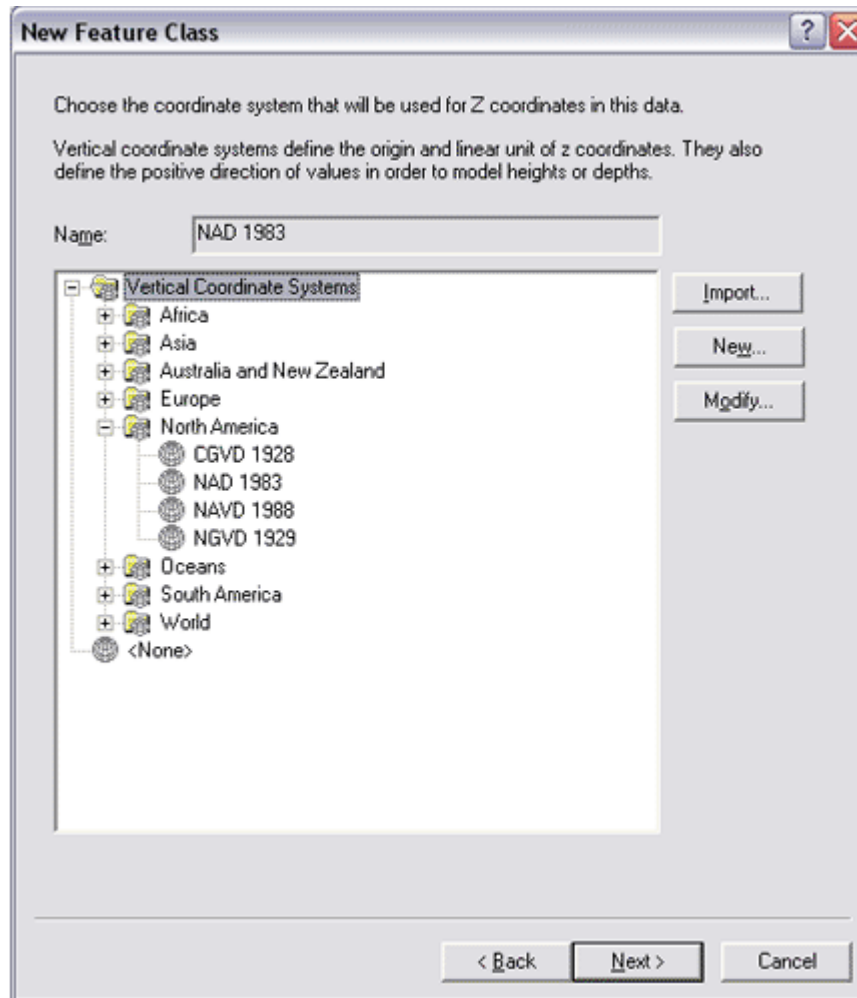
Geometry Properties

☐ Coordinates include M values. Used to store route data.

☒ Coordinates include Z values. Used to store 3D data.

< Back Next > Cancel

当建立了一个含 Z-coordinates 的要素类，将随意设置一个垂直坐标系统(VCS):



指定一个垂直坐标系统的主要目标是数据集文档化，在 VCS 中定义的 z-units 将被用于 ArcGIS 的 3D 处理操作。

大多数 z-units 值以 x,y 坐标相同单位来度量海平面以上或者以下的高度，但是，还有一些其他的 z 度量，它可能需要一个不同的 z-tolerance 和 z-resolution 值。

Z-tolerance 和 z-resolution

z-tolerance 定义了处理过程中聚类 z-values。默认的 z-tolerance 是 0.001 meters 或者定义的 VCS 的单位的相等的量。

如果坐标是一致的，它们的 z-values 落入了 z-tolerance，它们的 z-values 将被设置为相同。

当建立一个新的要素类时，可以接受默认的 z-tolerance 的值，或者选择设置自己的值。

New Feature Class

XY Tolerance

The XY tolerance is the minimum distance between coordinates before they are considered equal. The XY tolerance is used when evaluating relationships between features.

0.001 Degree

Z Tolerance

0.001 Meter

Reset To Default About Setting Tolerance

☒ Accept default resolution and domain extent (recommended)

< Back Next > Cancel

在大多数情况下,默认的 z-tolerance 能很好工作,除非 z-coordinate units 与 x,y coordinates 有很大的不同。

在与上面相同的一个面板上,可以选择接受默认的 z-resolution 值。如果选择修改 x,y resolution、z-resolution 或者两者,取消选择 Accept default resolution and domain extent 复选框,将出现一个面板,在其中可以看到默认的 z-resolution 和 z-values 的最大和最小值。

New Feature Class

All coordinates stored in a feature class are snapped to an underlying coordinate grid. Resolution is the cell size of this grid. Decreasing the resolution may reduce data storage needs but may reduce coordinate accuracy.

The coordinate range or domain extent defines the minimum and maximum coordinate values which can be stored.

XY
XY Resolution: Degree

Z
Z Resolution: Meter

Min: Max:

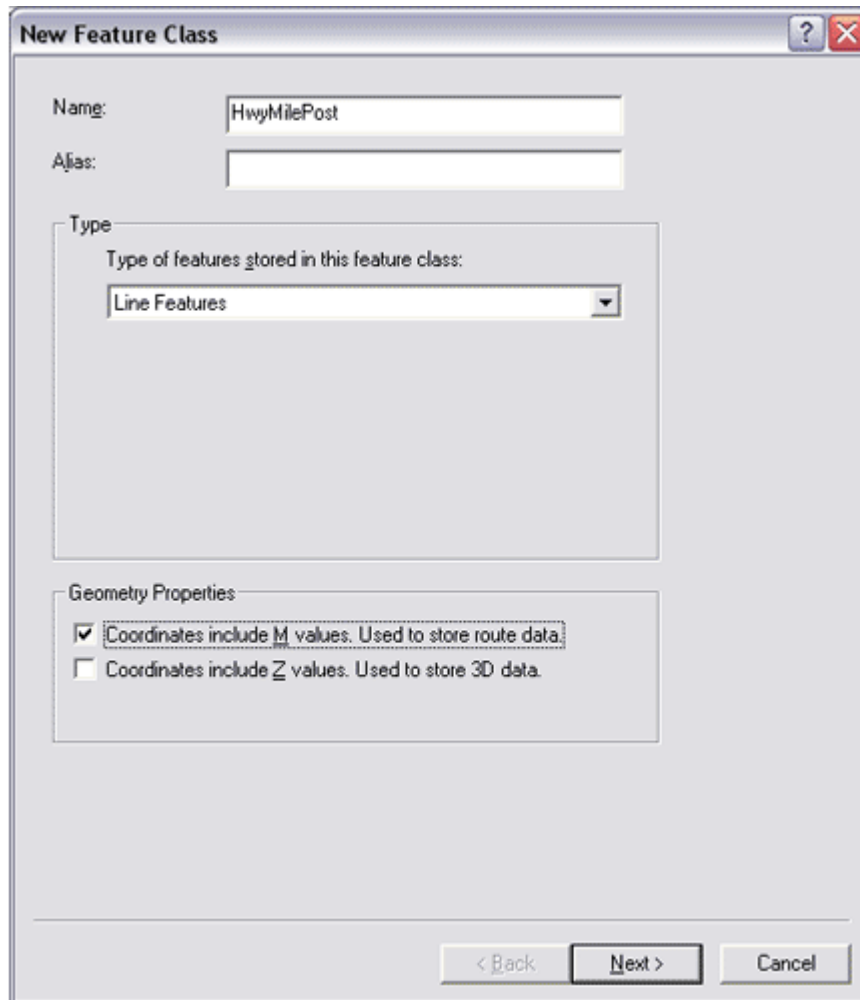
< Back Next > Cancel

对于 z-tolerance，在大多数情况下，接受默认的 z-resolution 和 z-range 将能很好地工作。Z-resolution 定义了 z-values 的 resolution。默认的 z-resolution 是默认的 z-tolerance1/10 倍。

线性参照的 M-resolution 和 m-tolerance

当建立含 m-measures 的要素类，可以设置 m-measures 值的两个属性：

在 ArcCatalog 的 New Feature Class 对话框上，通过选择 Coordinates include M values 复选框指定含 m-measures 的要素类。



The image shows a 'New Feature Class' dialog box with the following fields and options:

- Name:** HwylMilePost
- Alias:** (empty)
- Type:**
 - Type of features stored in this feature class: Line Features
- Geometry Properties:**
 - ☒ Coordinates include M values. Used to store route data.
 - ☐ Coordinates include Z values. Used to store 3D data.

At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

在接下来的面板中，可以设置 m-tolerance 或者接受默认值。m-tolerance 定义了处理过程中 m-values 的 tolerance。它默认值是 0.001 units。如果坐标的 x 和 y 一致，它们的 m-values 落入 m-tolerance，那么它们的 m-values 将被设置为相等。

New Feature Class

XY Tolerance
The XY tolerance is the minimum distance between coordinates before they are considered equal. The XY tolerance is used when evaluating relationships between features.

0.00000008983153 Degree

M Tolerance

0.001 Unknown Units

Reset To Default About Setting Tolerance

☒ Accept default resolution and domain extent (recommended)

< Back Next > Cancel

大多数情况下，接受默认的 m-tolerance 能满足工作。但是，如果 m-measure 值相当小，需要保证 m-measure 不聚类，可以指定自己的 m-tolerance。

在同一个面板中也可以选择修改默认的 m-resolution：取消选择 Accept default resolution and domain extent 复选框。如下所示：

在接下来的面板中可以设置一些其他的 m-coordinate 的属性：

New Feature Class

All coordinates stored in a feature class are snapped to an underlying coordinate grid. Resolution is the cell size of this grid. Decreasing the resolution may reduce data storage needs but may reduce coordinate accuracy.

The coordinate range or domain extent defines the minimum and maximum coordinate values which can be stored.

XY
 XY Resolution: Degree

M
 M Resolution: Unknown Units
 Min: Max:














< Back Next > Cancel

在此，M-resolution 定义了自己 m-values 的 resolution。它的默认值是 0.0001 units。

通过设置 m-values 最小值(Min)和最大值(Max)来设置 m-values 的范围

geodatabase 中的要素类存储

在 geodatabase 中，每个要素类在一个单独的表中被管理。每行中的 Shape 列用于存储每个要素的 geometry 或 shape。

Geographic View		Tables View																											
		<table border="1"> <thead> <tr> <th>Object ID</th><th>Shape</th><th>Name</th><th>LV Code</th><th>Management Agency</th></tr> </thead> <tbody> <tr> <td>1</td><td></td><td>Shady Pines</td><td>20</td><td>Private</td></tr> <tr> <td>2</td><td></td><td>Pinewood Village</td><td>30</td><td>Pinewood Village Association</td></tr> <tr> <td>3</td><td></td><td>Sarah Park</td><td>80</td><td>City Park Board</td></tr> <tr> <td>4</td><td></td><td>Town Park</td><td>99</td><td>City Park Poard</td></tr> </tbody> </table>			Object ID	Shape	Name	LV Code	Management Agency	1		Shady Pines	20	Private	2		Pinewood Village	30	Pinewood Village Association	3		Sarah Park	80	City Park Board	4		Town Park	99	City Park Poard
Object ID	Shape	Name	LV Code	Management Agency																									
1		Shady Pines	20	Private																									
2		Pinewood Village	30	Pinewood Village Association																									
3		Sarah Park	80	City Park Board																									
4		Town Park	99	City Park Poard																									

在要素类表中：

每个要素类是一个表；

单独的要素被存储为行；

Shpfile 存储每个要素的 geometry (point, line, polygon)；

Object ID 列存储每个要素的唯一标识符。

在 ArcSDE geodatabases 中，关系数据库存储每个要素类为一个表。有三种 DBMSs(Oracle, DB2, 和 Informix)提供了 SQL 访问 geodatabase 中的要素的 geometry。

扩展要素类

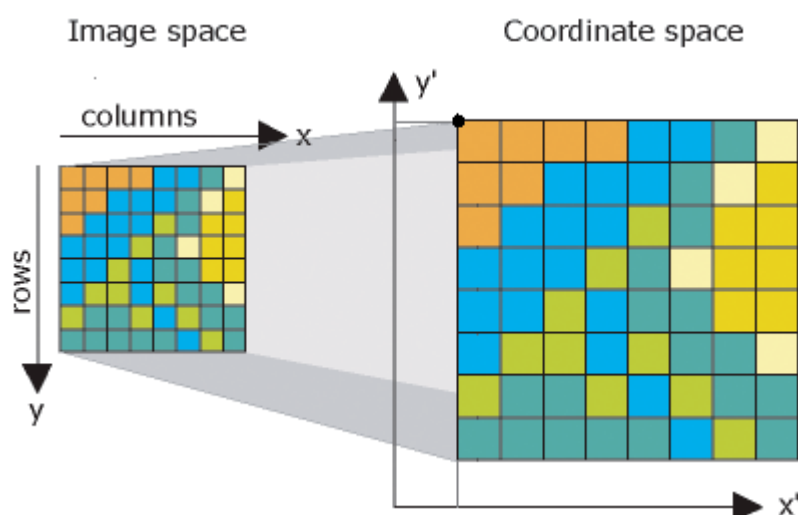
Geodatabase 中的每个要素类是具有相同几何形状、相同属性和相同空间参考的地理要素的集合。要素类可以根据需要被扩展。以下是一些用于要素类扩展的方法：

Use:	If you need to:
Feature Dataset	Hold a collection of spatially related feature classes or build topologies, networks, and terrains.
Subtypes	Manage a set of feature subclasses in a single feature class. This is often used on feature class tables to manage different behaviors on subsets of the same feature types.
Attribute Domains	Specify a list of valid values or a range of valid values for attribute columns. Use domains to help ensure the integrity of attribute values. Domains are often used to enforce data classifications (such as road class, zoning codes, and land-use classifications).
Relationship Classes	Build relationships between feature classes and other tables using a common key. For example, find the related rows in a second table based on rows selected in the feature class and so on.
Topology	Model how features share geometry. For example, adjacent counties share a common boundary. Also, county polygons nest within and completely cover states.
Network Dataset	Model transportation connectivity and flow. (Must have Network Analyst installed to use)
Geometric Network	Model utilities networks and tracing.
Terrain	Model triangulated irregular networks (TINs) and manage large lidar and sonar point collections.
Address Locator	Geocode addresses.
Linear Referencing	Locate events along linear features with measurements.
Cartographic Representations	Manage multiple cartographic representations and advanced cartographic drawing rules.

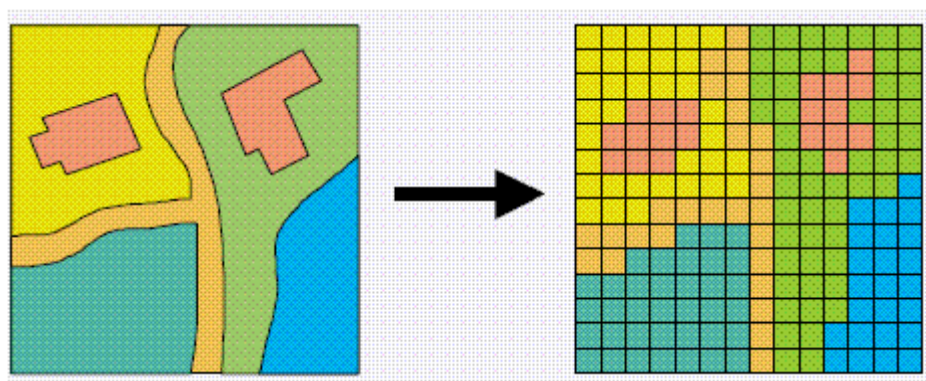
1、4 栅格基础

栅格基础

栅格数据集是采用规则的格网来表示地理要素的，每个规则格网单元用于表达这个位置的一些属性。



栅格数据集通常用于表示和管理影像、数字高程模型和大量其他的现象。通常栅格作为一个表达点、线和多边形要素的方法而使用。在下面的例子中，可以看到一系列多边形被表达为一个栅格数据集。



栅格被感兴趣是基于两个原因：一，它们可以被用于表达所有的地理信息（要素、影像和表面），二，它们具有一套丰富的分析处理运算符。因此，栅格作为一种通用的在 GIS 中存储影像的数据类型，也被用于表达用于基于栅格模型和分析的要素。

Geodatabase 中的栅格

一个栅格就是按行和列分布的一组格网单元，是被经常用于 GIS 的一种数据集。用户通常使用大量栅格文件，但是许多用户需要在一个 DBMS 中同时管理地理信息和栅格数据。Geodatabase 提供了一个非常有效的方法在 file 和 ArcSDE geodatabase 中管理栅格数据。

栅格数据集管理策略

两种栅格数据集管理策略是重要的：

Raster provisioning。

Rasters in the geodatabase。

栅格数据的地理属性

通常有四个地理属性为栅格数据集存储。这些属性在地理参考和帮助解释栅格数据文件如何结构化方面很有用的。

栅格数据集用一个特殊的方法定义地理位置。一旦格网单元或者像素可以被精确引用，然后获得一个栅格中所有格网单元值的一个顺序列表是容易的。这意味着每个栅格数据集通常有一个 header record 存储它的地理属性，数据的主要部分是格网单元值的一个顺序列表。

一个栅格的地理属性通常包括：

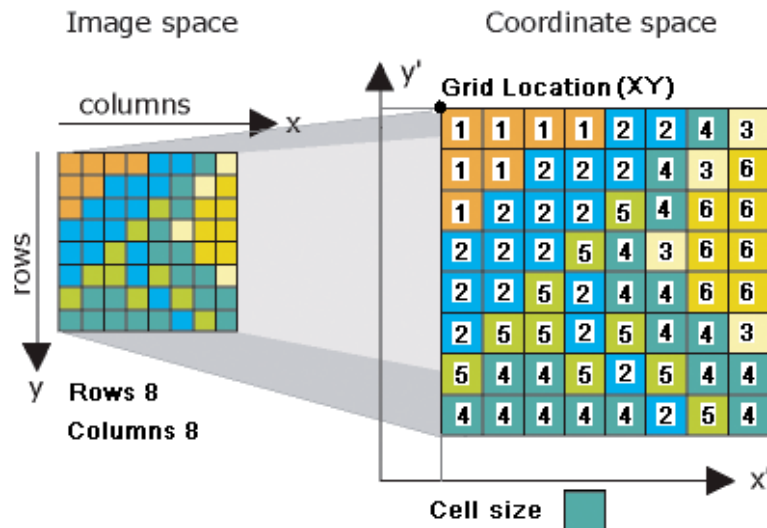
坐标系统;

参考坐标系统或者 x, y 位置 (通常指左上角或者左下角);

一个格网的大小;

行和列的数量。

这个信息可以被用于找到任何具体的格网单元的位置。获得这个信息后, 栅格数据结构容易地列出所有的格网单元值, 从左上角开始, 沿着每一行, 然后到最右下角。如下图所示:



List of cell values

[111122431122243612225466222543662252446625525443544525444444254]

Geodatabase 中的 raster block table

栅格数据通常在大小上比要素要大很多, 需要一个 side table 存储。例如, 一个正常的正射影像有 6700 行*7600 列。

为了取得这些大栅格数据集的高性能, 一个 geodatabase 栅格被分割为较小的片 (称为 blocks), 通常为 128 行*128 列或者 256 行*256 列。这些较小的 blocks 然后存储在 side table 中。如下图所示:

Geographic View

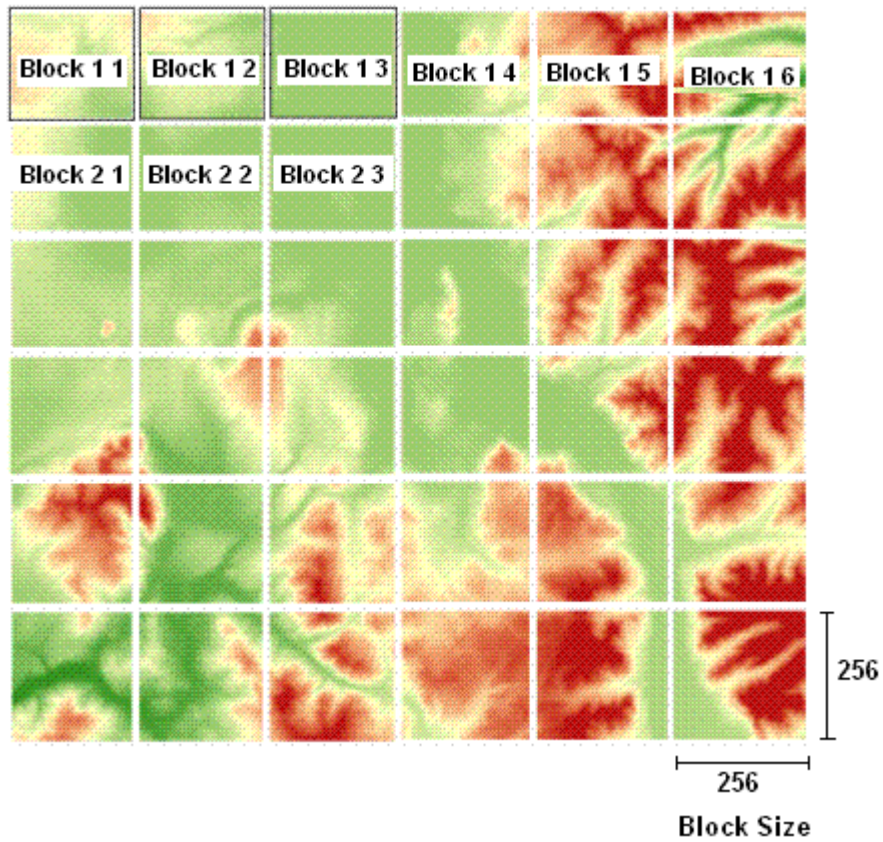









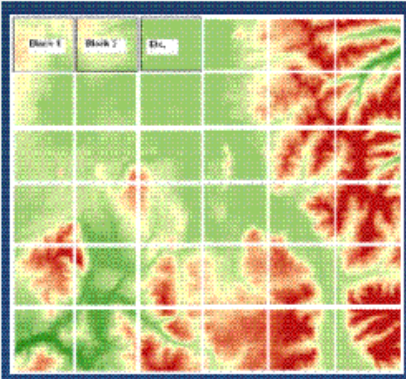





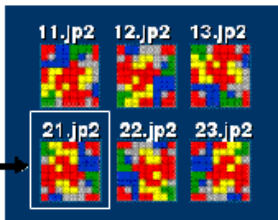


























Table View

BAND	RRD	ROW	COL	BLOCK DATA
1	4	1	1	
1	4	1	2	
1	4	1	3	
1	4	1	5	
1	4	1	5	
1	4	2	1	
1	4	2	2	
Etc.				






扩展栅格数据

栅格数据集被日益广泛地用于 GIS 应用。Geodatabase 可以管理栅格为：一个单独的数据集、数据记录逻辑集合和表中的图片属性。









大量 geodatabase 功能允许用户扩展如何管理这些栅格信息，如下表所示：

Use	If you need to																																			
Raster Datasets	Manage very large, continuous image datasets and image mosaics. 																																			
Raster Catalogs	Accomplish a number of purposes, including <ul style="list-style-type: none">Manage a tiled image layer, where each tile is a separate image. <div><table><tr><th>ID</th><th>Shape</th><th>Image</th><th>Date</th><th>Abstract</th></tr><tr><td>2101</td><td></td><td>11.jp2</td><td>12-12-98</td><td>Stuff</td></tr><tr><td>2102</td><td></td><td>12.jp2</td><td>12-12-98</td><td>Stuff</td></tr><tr><td>2103</td><td></td><td>13.jp2</td><td>12-12-98</td><td>Etc.</td></tr><tr><td>2201</td><td></td><td>21.jp2</td><td>12-15-98</td><td>Etc.</td></tr><tr><td>2202</td><td></td><td>22.jp2</td><td>12-15-98</td><td>Etc.</td></tr><tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td></tr></table></div>	ID	Shape	Image	Date	Abstract	2101		11.jp2	12-12-98	Stuff	2102		12.jp2	12-12-98	Stuff	2103		13.jp2	12-12-98	Etc.	2201		21.jp2	12-15-98	Etc.	2202		22.jp2	12-15-98	Etc.
ID	Shape	Image	Date	Abstract																																
2101		11.jp2	12-12-98	Stuff																																
2102		12.jp2	12-12-98	Stuff																																
2103		13.jp2	12-12-98	Etc.																																
2201		21.jp2	12-15-98	Etc.																																
2202		22.jp2	12-15-98	Etc.																																
...																																
	<ul style="list-style-type: none">Manage any series of images in a DBMS. <div><table><tr><th>ID</th><th>Shape</th><th>Image</th><th>Date</th><th>Abstract</th></tr><tr><td>2101</td><td></td><td>11.jp2</td><td>12-12-98</td><td>Etc.</td></tr><tr><td>2102</td><td></td><td>12.jp2</td><td>02-12-02</td><td>Etc.</td></tr><tr><td>2103</td><td></td><td>13.jp2</td><td>01-13-95</td><td>Etc.</td></tr><tr><td>2201</td><td></td><td>21.jp2</td><td>03-04-98</td><td>Etc.</td></tr><tr><td>2202</td><td></td><td>22.jp2</td><td>06-15-04</td><td>Etc.</td></tr><tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td></tr></table></div>	ID	Shape	Image	Date	Abstract	2101		11.jp2	12-12-98	Etc.	2102		12.jp2	02-12-02	Etc.	2103		13.jp2	01-13-95	Etc.	2201		21.jp2	03-04-98	Etc.	2202		22.jp2	06-15-04	Etc.
ID	Shape	Image	Date	Abstract																																
2101		11.jp2	12-12-98	Etc.																																
2102		12.jp2	02-12-02	Etc.																																
2103		13.jp2	01-13-95	Etc.																																
2201		21.jp2	03-04-98	Etc.																																
2202		22.jp2	06-15-04	Etc.																																
...																																

• Manage a raster time series.

ID	Shape	Image	Time	Abstract
2101		11.jp2	11:00	Etc.
2102		12.jp2	12:00	Etc.
2103		13.jp2	13:00	Etc.
2201		21.jp2	14:00	Etc.
2202		22.jp2	15:00	Etc.
...

Store pictures or scanned documents as attributes in tables

ID	Shape	PIN	Assessed Value	Structure
2101		12-101	150000	
2102		12-102	175000	
2103		13-111	145000	
2201		21-112	173000	

1、5 Geodatabase 类型

geodatabase 类型

Geodatabase 是一个容器用于存储数据集集合。它有三种类型：

1、File Geodatabases—以文件系统中的文件夹存储。每个数据集以一个文件被存储，可达到 TB 级。与 personal geodatabases 相比，推荐使用这种类型。

2、Personal Geodatabases—所有的数据集被存储在一个 Microsoft 的 Access 数据文件中，它的容量限制为 2GB。

3、ArcSDE Geodatabases—使用 Oracle, Microsoft SQL Server, IBM DB2, IBM Informix 存储在一个关系数据库中。这个多用户的 geodatabase 需要 ArcSDE 的使用，在容量和用户数量上没有限制。

三种类型 geodatabase 的比较：

Key Characteristics	ArcSDE Geodatabase	File Geodatabase	Personal Geodatabase
Description	<p>A collection of various types of GIS datasets held as tables in a relational database</p> <p>This is the recommended native data format for ArcGIS stored and managed in a relational database.</p>	<p>A collection of various types of GIS datasets held in a file system folder</p> <p>This is the recommended native data format for ArcGIS stored and managed in a file system folder.</p>	<p>Original data format for ArcGIS geodatabases stored and managed in Microsoft Access data files</p> <p>This is limited in size and tied to the Windows operating system.</p>
Number of Users	<p>Multuser</p> <p>Many readers and many writers</p> <p>ArcSDE can be licensed for use at three levels:</p> <ul style="list-style-type: none"> • Personal ArcSDE • Workgroup ArcSDE • Enterprise ArcSDE 	<p>Single user and small workgroups</p> <p>Some readers and one writer per feature dataset, standalone feature class or table.</p> <p>Concurrent use of any specific file eventually degrades for large numbers of readers.</p>	<p>Single user and small workgroups with smaller datasets</p> <p>Some readers and one writer.</p> <p>Concurrent use eventually degrades for large numbers of readers.</p>
Storage Format	<ul style="list-style-type: none"> • Oracle • Microsoft SQL Server • IBM DB2 • IBM Informix 	<p>Each dataset is a separate file on disk</p> <p>A file geodatabase is a file folder that holds its dataset files.</p>	<p>All the contents in each personal geodatabase are held in a single Microsoft Access file (.mdb).</p>
Size Limits	Up to DBMS limits	<p>One TB for each dataset. Each file geodatabase can hold many datasets</p> <p>Each feature class can scale up to hundreds of millions of vector features per dataset.</p>	<p>Two GB per Access database</p> <p>Effective limit before performance degrades is typically between 250 and 500 MB per Access database file.</p>
Versioning Support	Fully supported across all DBMSs; includes cross-database replication and updates	Not supported	Not supported
Platforms	Windows, Unix, Linux, and direct connections to DBMSs that can potentially run on any platform on the user's local network	Cross-platform	Windows only
Security and Permissions	Provided by DBMS	Operating file system security	Windows file system security
Database Administration Tools	Full DBMS functions for backup, recovery, replication, SQL support, security, and so on	File system management	Windows file system management
Notes	Requires the use of ArcSDE	Allows you to optionally store data in a read-only compressed format to reduce storage requirements	Often used as an attribute table manager (via Microsoft Access). Users like the string handling for text attributes.

File geodatabases 和 personal geodatabases

File 和 Personal geodatabase，对所有的 ArcGIS 用户都是免费使用的，支持 geodatabase 的全部信息模型。这包括拓扑、栅格目录、网络数据集、表面数据集、地址定位器等。File 和 Personal geodatabase 被设计为由一个单独的用户编辑，不支持 geodatabase 版本。对于 file Geodatabase，可以有多个编辑者同时进行编辑，只要是他们编辑不同的要素数据集、独立要素类或者表。

File geodatabase 是在 ArcGIS9.2 版本中新发布的一种新的 geodatabase 类型。它的目标是：

- 提供所有用户一个广泛可用、简单和可伸缩的 geodatabase 解决方案；
- 提供一个跨操作系统的简便的 geodatabase；
- 提高处理大数据集的能力；
- 提供了优越的性能和可伸缩性；
- 使用一个有效的数据结构，它对数据存储和性能是最优化的；

Personal geodatabases 从 ArcGIS8.0 版本开始就被 ArcGIS 使用，使用 Microsoft 的 Access 数据结构（mdb file）。它们支持的 geodatabase 容量限制在 2GB 或者小于 2GB。但是，有效的数据库大小是比较小的，大概在 250MB—500 MB 之间，超过这个范围之后，数据库性能将开始降低。Personal geodatabases 也仅在 Microsoft 的 Windows 操作系统上被支持。

由于多个原因，ArcGIS 将继续支持 personal geodatabases。但是，它也高度推荐使用 file geodatabase。对于基于文件数据集的 GIS 项目使用 file geodatabase 是理想的，对于比较小的工作组，使用 personal geodatabases 比较理想。

ArcSDE geodatabases

当需要一个可以由多个用户同时编辑和使用的多用户 geodatabase，ArcSDE geodatabase 是一个好的解决方案。它添加了管理一个共享、多用户 geodatabase，以及基于版本的 GIS 工作流的能力。利用企业级的关系数据库的功能是 ArcSDE geodatabase 的最大优势。

ArcSDE geodatabase 使用多种 DBMS 存储模型（IBM DB2, Informix, Oracle, 和 SQL Server）。ArcSDE geodatabases 主要被用于工作组、部门和企业范围。它们利用基本的 DBMS 体系支持：

- 及其大、连续的 GIS 数据库；
- 多个并发用户；
- 长事务和版本工作流；
- 支持 GIS 数据管理的关系数据库；

通过许多 ArcSDE geodatabase 的实现，已经发现 DBMSs 在加载和删除大二进制对象类型的数据很有效。另外，GIS 数据库的容量和支持的用户数量比 GIS 文件系统大得多。

在 ArcGIS 中访问和使用 ArcSDE 有三种级别

ArcSDE geodatabases 包括 Personal ArcSDE geodatabase、Workgroup ArcSDE geodatabase 和 Enterprise ArcSDE geodatabase 三个级别。ArcSDE geodatabase 的功能可以在以下的 ESRI 软件产品中获得：

Personal ArcSDE included with ArcEditor and ArcInfo:从 ArcGIS9.2 开始，ArcEditor 和 ArcInfo 包括免费的 Microsoft SQL Server Express database。这些软件也包括 ArcSDE 的功能支持同时有三个连接的 Personal ArcSDE geodatabase—其中一个可以编辑数据。

SQL Server Express 限制在一个 CPU 上运行，使用 1GB 的内存。SQL Server Express 最大的数据库限制在 4GB。

Personal ArcSDE geodatabase 提供了在 ArcEditor 和 ArcInfo 中使用 SQL Server Express 管理 ArcSDE geodatabases 的功能。

Workgroup ArcSDE included with ArcGIS Server for Workgroups: ArcGIS Server 的 Workgroups 级别包括支持 SQL Server Express。在这个级别上的 ArcSDE，可以在任何数量的用户和编辑者上使用 SQL Server Express，只要他们被配置所支持。

对于 personal ArcSDE, 使用 ArcEditor 或者 ArcInfo 创建、执行和管理 workgroup ArcSDE geodatabases。可以使用 SQL Server Express 在 ArcCatalog 中建立和管理这些 workgroup ArcSDE geodatabases。不需要额外的数据库管理技术。

在这里，可以认为 ArcGIS Server 的 Workgroups 级别是 ArcEditor 或者 ArcInfo 的扩展，帮助管理和服务 workgroup ArcSDE geodatabases。

Enterprise ArcSDE included with ArcGIS Server for Enterprises:这是传统的 ArcSDE 技术，它运行在 Oracle, SQL Server, IBM DB2, 和 IBM Informix，可达到任何容量和任何数量用户的数据库，运行在任何大小和配置的计算机上。

ArcSDE 在 DBMS 的事务框架上提出长和短事务处理

ArcSDE geodatabase 包括对一个多用户 geodatabase 的编辑和更新的高级支持。随着 GIS 日益增加的用户和对来自一组传感器数据的管理的需要，GIS 对事务管理的需要变得更加重要。在 GIS 中，长事务的需求更多。

通常，GIS 用户具有特殊的事务处理需求，其中一个就是跨长时间的（不仅是几秒或者几分钟，有时几个小时，甚至几天和几个月）。

另外，GIS 中一个单独的编辑会话可能包括多个表中的多条记录的变化。用户需要 undo 和 redo 操作。用户在他们提交变化时，需要对待每个编辑会话为一个单独的事务。甚至，编辑通常必须是在一个与中央、共享数据库断开的环境中执行的。

在一个多用户的数据库中，GIS 事务必须与 DBMSs 的短事务框架协调。ArcSDE 通过管理高层次、复杂的 GIS 事务在这些操作中扮演一个关键角色。

ArcSDE 以 delta records 在数据库中存储这些变化的信息；使用版本分离多个编辑会话；支持复杂事务、自动历史数据归档和历史查询。

1、6 常见 geodatabase 的任务

创建和操作 geodatabase 的主要任务包括以下：

设计一个 geodatabase；

建立一个 geodatabase；

加载数据集和栅格到 geodatabase 中；

用高级数据类型扩展要素类；

用域、关联和子类型扩展表；

扩展栅格数据集；

加载和维护 geodatabase 中每个数据集的数据；

管理 geodatabase 的更新和事务；

管理 File 或者 Personal geodatabase；

管理 ArcSDE geodatabase。

第2章 Geodatabase 体系结构

2、1 Geodatabase 体系结构

用户通常认为 geodatabase 是地理信息的物理存储，但从根本来说，是使用 DBMS 或文件来存储的。除了作为数据集集合的一个物理实例，每个 geodatabase 还有其他关键方面特征。

1、 geodatabase 有一个广泛的信息模型来表达和管理地理信息。这个信息模型的实现是通过一系列简单数据表，这些表是存储在要素类、栅格数据集和属性表中的。另外，高级的 GIS 数据对象通过增加 GIS 行为、规则来管理空间完整性，以及增加工具来处理要素、栅格和属性表的大量的空间关系。

geodatabase 的 software logic 提供了通用的 application logic，贯穿整个 ArcGIS，用于访问和处理各种数据格式的地理数据。无疑，它是支持 geodatabase 的，并且支持 shapefiles, CAD 文件, TIN's, grids, CAD 数据, imagery, 还有其他大量的 GIS 数据源。

geodatabase 具有一个事务处理模型来管理 GIS 的数据工作流。

2、2 Geodatabase 基于关系原则存储

用户倾向于认为 DBMS 本质上是开放的，这是因为关系数据模型的简易性和灵活性，使得它能支持广泛的应用。

Geodatabase 的存储模型是以 DBMS 原则为基础，利用了一系列简单却基本的关系数据库概念。DBMS (File geodatabase 的文件系统) 提供了简单而又合适的数据模型存储和操作表。

包括以下的关键概念：

数据被组织为表

表包含行

表中所有的行具有相同的列

每一列具有一个类型，如 integer, decimal number, character, date 等等。

关系类用于一个表中的行和另一个表中的行发生关联，这是以每个表中都有一个公共的列为基础的。

关系完整性规则存在于所有表中。如，每一行总是有相同的列，一个域为每一列指定了有效值或者值范围等等。

对 ArcSDE geodatabases 来说，还有大量其他的 DBMS 功能可以应用：

SQL，一系列关系函数和操作符可以用于操作表和表中的元素。

SQL 操作符同时被设计为可以处理普通的关系数据类型，如 integers, decimal numbers, dates, 和 characters。

Shape	ID	PIN	Area	Addr	Code
	1	334-1626-001	7,342	341 Cherry Ct.	SFR
	2	334-1626-002	8,020	343 Cherry Ct.	UND
	3	334-1626-003	10,031	345 Cherry Ct.	SFR
	4	334-1626-004	9,254	347 Cherry Ct.	SFR
	5	334-1626-005	8,856	348 Cherry Ct.	UND
	6	334-1626-006	9,975	346 Cherry Ct.	SFR
	7	334-1626-007	8,230	344 Cherry Ct.	SFR
	8	334-1626-008	8,645	342 Cherry Ct.	SFR

PIN	Owner	Acq.Date	Assessed	TaxStat
334-1626-001	G. Hall	1995/10/20	\$115,500.00	02
334-1626-002	H. L Holmes	1993/10/06	\$24,375.00	01
334-1626-003	W. Rodgers	1980/09/24	\$175,500.00	02
334-1626-004	J. Williamson	1974/09/20	\$135,750.00	02
334-1626-005	P. Goodman	1966/06/06	\$30,350.00	02
334-1626-006	K. Staley	1942/10/24	\$120,750.00	02
334-1626-007	J. Dormandy	1996/01/27	\$110,650.00	01
334-1626-008	S. Gooley	2000/05/31	\$145,750.00	02

例如，一个要素类以 DBMS 表的形式存储。每一行代表一个要素，每一行中列描述了该要素的各种特征或属性，表中的某一列存储了该要素的几何形状（如 point, line 或 polygon coordinates）。在上面的实例中，shape 字段存储了 polygon 形状。

在 DBMS 中大量的列类型用于 shape 字段。可以是典型的 binary large object (BLOB) 类型或者是扩展的空间类型，它们能够被一些 DBMS 所支持。如，ESRI 提供了一个空间列类型用于在 ArcSDE geodatabases 存储要素，支持这种类型的关系数据库有 Oracle, IBM DB2, 和 Informix。

SQL 操作表中的行、列。这些列类型（numbers, characters, dates, BLOB's, spatial types 等等）在 SQL 代数中被认为是对象。DBMS 管理这些简单的数据类型和表，同时其他应用逻辑实现更复杂的对象行为和完整性约束。

在关系数据库管理系统中实现更高层次的对象和行为

开发人员为了实现更高层次的具有行为和逻辑的对象，就需要写应用程序代码去实现它。如，一个组织可能实现如下名为 EMPLOYEES 的表：

Last Name	First Name	Hire Date	Salary
Brown	Ben	10-10-2001	\$10,000.50
Jones	Betty	06-14-1998	\$22,000.00
Smith	Jason	08-23-1999	\$44,000.75

上面的表是一个简单的关系数据表，包含行和列。每一列的数据都隶属一个独特的数据类型，如 character、date 和 number，DBMS 就是在这个数据类型层次上操作信息的。

然而，简单的给一个 DBMS 增加这些信息，不会使 DBMS 变成一个 payroll 或 employee 管理信息系统。增加一列名为 “Dollars” 用于存储带两位小数的数值，并不意味着将一个 DBMS 变为一个帐务清算系统。要实现这些需要更高层次的应用逻辑。

实现支持雇佣行为的逻辑例子是雇用、实现薪水增长、员工辞职、晋升和奖金管理。业务对象被建模为雇员和他们的姓名、薪水和雇用日期，为了实现这些对象的行为和完整性需要更复杂的和聚焦的应用逻辑。

相似的业务对象也普遍应用于 GIS。如, topologies, networks, linear referencing systems, raster catalogs, annotations, terrains, map layers 等等, 都是高级对象用于实现 GIS 的行为。对于其他的 DBMS 应用, 具有空间属性类型的表是对他们自己的 GIS 应用是不够的。简单的 DBMS 列类型和 geodatabase 应用对象如 topologies 对建立地理信息系统是必须的。

2、3 应用逻辑属于什么?

用户可以用很多方法实现这种高层次的逻辑。例如, 应用逻辑可以按如下被实现:

DBMS 中的 Stored procedures 和 database triggers;

扩展 DBMS 的数据类型;

建立一个单独的应用逻辑层用于操作表中的行和列类型。

在过去的二十年中不计其数的 DBMS 实现已经证明使用应用逻辑层对于高级应用是一个明智的选择。如, 通用的 customer information systems (CIS), enterprise resource planning (ERP) systems 和帐务清算包都是在应用逻辑层实现了高级的应用, 它提供了更多的开放性和可扩展性, 高性能操作, 丰富的工具包和大大的灵活性。

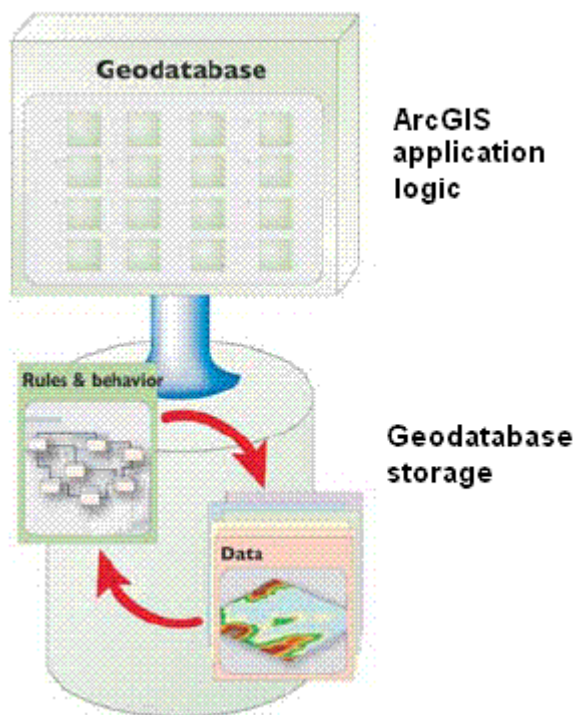
用户通过应用逻辑层与系统交互, 执行版本管理, 实现大量的操作, 仅在 focused activities 中使用 SQL。

分离的应用逻辑层置于数据层之上允许相同的逻辑被应用于 DBMS, files 或其他的数据存储格式。这使得这种体系结构更开放。如, 在 ArcGIS 中 geodatabase 的应用逻辑层也可以用于读取和操作所有的地理数据源 CAD data, shapefiles, MapInfo data, Intergraph GeoMedia files 等等。

2、4 Geodatabase 是 object-relational 的结构

Geodatabase 采用两层结构: 数据存储层和应用层。数据存储层是将 GIS 数据存储为 File、XML、DBMS 等多种格式, 而应用层则是维护数据的高级逻辑和行为, 例如 Feature Classes、Raster Dataset、Topology、Network、Address Locators 等等。

多层的 geodatabase 体系结构有时被称为 object-relational 模型。



Geodatabase 管理地理数据集的功能由 ArcGIS 软件和 DBMS 共同实现。管理地理数据集功能的某些方面, 如 disk-based 存储, 属性类型的定义, 查询处理及多用户版本处理都被

委托给了 DBMS。GIS 应用程序保留了一些功能，如定义具体 DBMS 的 schema，用于表达各种各样的 datasets 和 domain-specific 逻辑，以保持基本记录的完整性和有效性。

实际上，DBMS 作为一种存储地理数据集的实现机制而被使用。但是，DBMS 并没有完全定义地理数据的语义。因此可以认为：geodatabase 采用两层体系结构，数据存储层和应用层，在数据存储层实现数据存储和检索，在应用层实现高层数据完整性和信息处理。

在其他高级 DBMS 应用中 geodatabase 也可以通过这种两层应用结构而实现。Geodatabase 对象在 DBMS 表中以行的形式存储，要素行为由 geodatabase 应用逻辑提供。

所有的 ArcGIS 应用与 geodatabase 的这种通用的 GIS 对象模型通信，而不是实际的基于 SQL 的 DBMS instance。Geodatabase 软件组件实现了这种通用模型的行为和完整性规则，并转换数据请求到合适的物理数据库设计。ArcSDE 软件提供了 ArcGIS 和 DBMS 之间的一个通用网关。

2、5 Geodatabase 在关系数据库存储

在 geodatabase 的核心是标准的关系数据库模式(一系列 DBMS 表、列类型、索引等等)。这种简易的物理存储与宿主在应用逻辑层的高层应用相协调，并受它们的控制，应用层可以是 ArcGIS Desktop、嵌入式 ArcGIS engine logic、或者 ArcGIS server。

这些应用中都包含用于定义通用 GIS 信息模型的 geodatabase 对象，并且为所有的 GIS 应用程序和用户所共享。Geodatabase 对象的目的是暴露高层 GIS 应用模型给客户端，选择合适的存储模型存储应用模型的具体实现细节，如存储在标准的 DBMS 表中、file geodatabases, 和 XML。

Geodatabase 的存储包括基于每个地理数据集的 schema 和 rule，还有简单的空间数据和属性数据的表存储。

Geodatabase 的 schema 包含定义、完整性规则和地理数据集的行为。这些具体包括要素的 properties、topologies, networks, raster catalogs, relationships, domains 等等。Schema 保存在 DBMS 的 geodatabase 元数据表集合中，这些元数据定义了地理信息的完整性和行为。

空间表达要么以矢量要素或栅格数据集的形式用传统的二维表来存储。例如，一个 DBMS 表可以存储一个要素类，表中每一行代表一个要素。每一行的 Shape 列都保存了 geometry 或者是要素的 shape，shape 列通常保存的 geometry 是这两种列类型中的一种：

BLOB column type

spatial column type，如果 DBMS 支持它

要素类是相同类型要素的集合，具有相同的几何类型，如点、线或多边形，具有相同的属性列集合，要素类通常可以由一个独立的表管理。

栅格和影像数据集同样也由关系表存储和管理。栅格数据通常比较大，存储的时候需要一个 Side Table 来存储，将栅格数据集切为小片，或者 Blocks，存储在 Raster 表中，通过 Side 表的每一行记录去记录栅格中的每一个 Block。

列类型存储矢量和栅格的 geometry 随着数据库的不同而不同。当一个 DBMS 支持空间类型扩展时，geodatabase 可以使用这些类型来存储空间几何形状。ESRI 也参与了很多扩展空间 SQL，如作为 SQL 3 MM Spatial 和 OGC Simple Features SQL specifications 的主要创作者。ESRI 不仅关注支持独立的 Oracle Spatial types，同时也支持这些类型，使用 DBMS 的标准来存储 geodatabase。

目前，支持 geodatabase 的含有空间类型的 DBMS 有三种：

使用 ESRI 的 spatial type 的 Oracle 或 Oracle Spatial type

使用 Spatial Extender Geometry Object 的 IBM DB2

使用 Spatial DataBlade Geometry Object 的 Informix。

2、6 Geodatabase 的事务管理

事务是对数据库变动的记录块。GIS 数据库，像其它数据库应用一样，必须支持执行数据完整性和应用行为的更新事务。在很多情况下，用户可以使用 DBMS 的事务处理框架来管理数据编辑和 geodatabase 的更新。

但是，GIS 用户普遍有具体的专业事务处理要求，其中最重要的就是这些事务要跨越很长的时间（有时是几天和几个月）。

另外，大多数 GIS 编辑包括协调（orchestrating）多个表中的多行变化和以单独的统一的事务管理这些变化，用户需要能够撤销或重做这些变化。编辑过程可以跨越几个小时甚至几天。通常编辑过程必须是在与中央共享的数据库断开的状态下进行操作的。

由于 GIS 工作流过程可以跨越几天或几个月，GIS 数据库必须为日常的操作保持持续地可用，每个用户可能有共享 GIS 数据库的个人视图或状态。在一个多用户的数据库中，GIS 事务必须由 DBMS 的短事务处理框架管理。ArcSDE 在简单 DBMS 事务处理框架管理高层次、复杂的 GIS 事务的操作中，扮演着关键的角色。

在很多情况下，GIS 用户选择长事务工作流是必须的，这样才有可能多用户的 DBMS 的使用和用 ArcSDE 来管理中央数据库的更新。下面是一些需要使用长事务处理模型的 GIS 数据编辑工作流的例子：

Multiple edit sessions—一个单独的 GIS 数据库更新需要大量的变动，这些变动跨越多个编辑过程，可能是几天或几个星期。

Multi-user editing—多个用户通常需要同时编辑更新相同的空间要素。每个用户都需要操作私人化的数据库状态，只关注个人的更新情况而忽略其他用户的更新。最后，各个用户提交各自的更新并与其他用户协调更新以确认更新，解决所有冲突。

Checkout/check-in transactions—给特定区域或地区的某个计算机签出（check out）数据库的一部分通常是必要的，它在与数据库断开的状态下进行信息的更新操作，这种情况可能持续几天甚至几个星期的，这些更新最后必须提交到中央数据库。

History—有时候保存一个 GIS 数据库每一个要素的历史版本是很有利的，甚至是在一个特定版本已经被更新后，保存 retired 和 changed 要素的复本在一个历史文档数据中或追踪一个独立要素的历史，例如，在一个国家地图数据库中 parcel lineage 的更新。

Transfer of change-only updates—企业级的数据库和空间数据基础设施的信息被组织内或跨组织共享的，它们合作实现更新操作通过互联网以定义明确的 XML 达到共享。

Distributed geographic database replicas—一个区域数据库可能是某一特定地理区域的中央 GIS 数据库的部分拷贝。定期地，两个数据库必须通过交换更新以达到同步。

Loosely coupled replication across DBMS's—通常地，GIS 数据必须在一系列数据库复本（replicas）中保持同步，每一个复本操作其本地的数据库进行更新。这些数据库只有定期地通过互联网连接。更新按照预定的原则从一个数据库复本转移到其他的数据库复本，保持它们内容的同步。大多数情况下，DBMS 是不同的，例如，可以在 SQL Server, Oracle, 和 IBM DB2 进行复制数据集。

Geodatabase 的事务模型——版本

Geodatabase 管理一些 GIS 工作流的机制就是保持多个状态，并且保证地理信息、规则和行为的完整性。从字面意思来理解，版本就是记录独立要素或对象的修改、增加和删除的各种状态。每一个版本就是明确记录了每一个要素或对象的行状态，同时还包括一些重要的事务处理信息。任何数量的用户可以同时操作和管理多个版本。

版本可以使所有的事务处理随着时间以数据库的一系列变化而记录。这意味着各个用户都可以操作 geodatabase 的多个视图或状态。它的目标就是开放的、高性能的多用户访问。例如，系统必须高速运行，并且必须支持含有成千上万的记录的数据集在数千个用户并发访问时的使用。

Geodatabase 基于版本的事务模型相对简单——更新被记录在变化的表中。

版本在两个 delta tables 明确地记录了 geodatabase 对象状态: Adds table 和 Deletes table。简单的查询用于对任何想要的 geodatabase 状态进行视图；例如，实时对一个点的数据库状态进行视图或特定用户的当前的编辑版本。

ArcSDE 在版本化的 geodatabase 应用中扮演着关键的角色，用于提供对同一或不同的 DBMS 的长事务管理。

Default Version before editing

A diagram showing a grid of parcels. Parcel 45 is highlighted in orange. The grid consists of a 2x2 arrangement of parcels, with the bottom-left parcel (45) highlighted.

Base Table

ObjectID	Perimeter	Bldg_Code	Area
41	30106.25	04	1253459.45
42	27458.37	04	1048592.56
43	32945.09	04	1584562.04
44	30001.55	04	1116459.67
45	30556.38	04	1362965.03

Adds Table

ObjectID	Other Columns	State_ID

Deletes Table

Deleted_at	Deletes_Row_ID	State_ID

Updated Feature after edit session

A diagram showing a grid of parcels. Parcel 47 is highlighted in yellow. The grid consists of a 2x2 arrangement of parcels, with the top-left parcel (47) highlighted.

Base Table

ObjectID	Perimeter	Bldg_Code	Area
41	30106.25	04	1253459.45
42	27458.37	04	1048592.56
43	32945.09	04	1584562.04
44	30001.55	04	1116459.67
47	43834.07	06	1953473.02

Adds Table

ObjectID	Other Columns	State_ID
47	<...>	47

Deletes Table

Deleted_at	Deletes_Row_ID	State_ID
45	<...>	0

上面是一个例子，一个 parcel（number 45）被更新为 parcel（number 47）。利用版本，原来的 parcel 被保存在 Deletes table 中，新的 parcel 被保存在 Adds table 中。其他的版本信息被记录到 meta tables 中，如时间和更新序列、版本名称、每一个更新的 state ID。每一个版本都有其安全措施和访问优先权。

大量的更新会不利于每个版本的，当用户编辑数据时，他们需要连接操作更新的版本。当用户准备与其他用户共享更新时，就需要执行冲突协调和提交编辑操作，保存在更新版本中的编辑将被提交到默认（Default）的版本，在对冲突的解决方案中确认和协调那些潜在的冲突。

2、7 Geodatabase XML

Geodatabase XML

Geodatabase XML 代表了 ESRI 的 geodatabase 和外部系统之间的开放的信息交换机制。ESRI 以 XML specification 公开了全部的 geodatabase 的 schema 和 content，并且提供了实现例子解释用户如何在不同类型的系统中共享数据更新。

地理信息的 XML 交换很简单地使用了 geodatabase XML specification。外部应用程序可以接受的 XML data streams 包括以下几种：

交换完全无损数据集；

交换简单要素集（如 shapefile 的交换）；

通过使用 XML streams 在 geodatabase 和外部的数据结构之间传递更新和变化，实现 change-only 记录的交换；

Geodatabase XML 不仅是所有 ArcGIS 用户，也是外部用户共享数据的最基本的交换机制。

在 ArcGIS 中可以创建三种类型的 XML documents: a Workspace document, a RecordSet

document, 和 a Data Changes document。

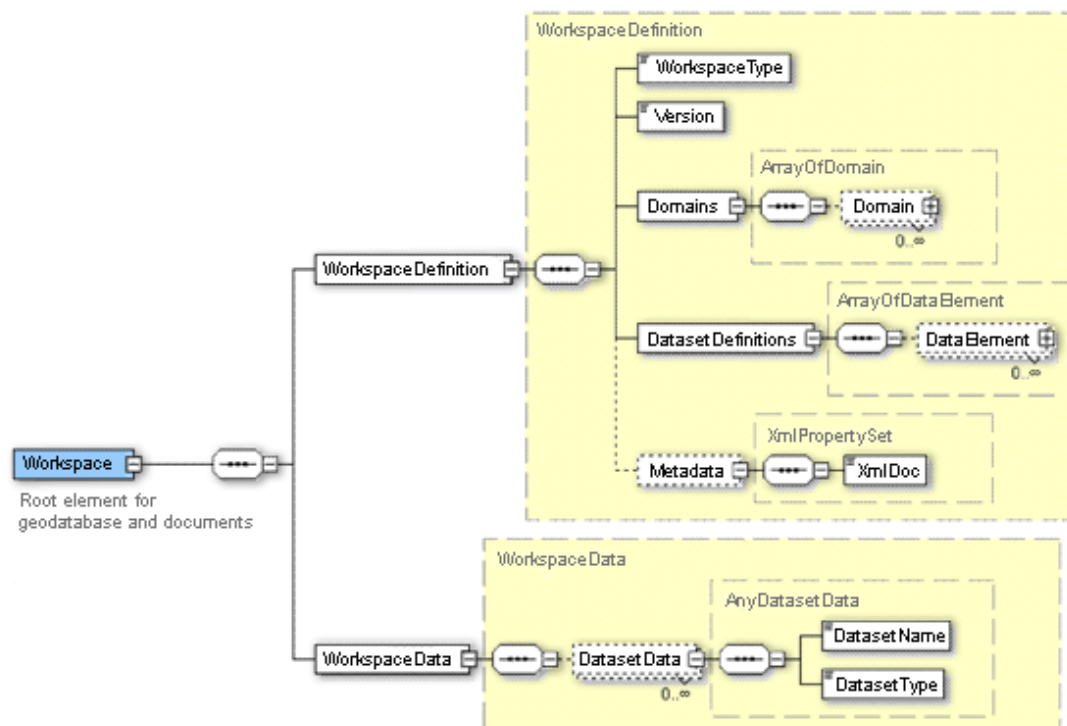
Geodatabase 的 XML workspace document

XML workspace document 保存了一个 geodatabase 中所有内容或其子集。XML workspace document 包含了所有的 schema 信息，数据是随意的。

XML workspace document 用于共享 schema 信息，交换数据集和它们的所有内容，交换数据集和与它们相关的所有数据元素。

Schema、relationship 和 behavior 信息作为 WorkspaceDefinition section 的部分导出，包括简单的用户要素数据、网络和拓扑、网络连接性和拓扑规则、简单的复合关系类和其他任何与地理数据集相关联的信息。因此，所有与 geodatabase 相关联的行为被存储，当 XML documents 被导入时被再创建。

XML workspace document 包含两个元素：WorkspaceDefinition 和 WorkspaceData。



使用 WorkspaceDefinition 和 WorkspaceData，XML workspace document 可以包含 geodatabase 的 schema。与实际数据分开的定义提供了一些好处。一个软件代理仅能读取 workspace document 的定义部分了解它所包含的，使数据部分可选，workspace document 仅能用于转换 schema 信息。

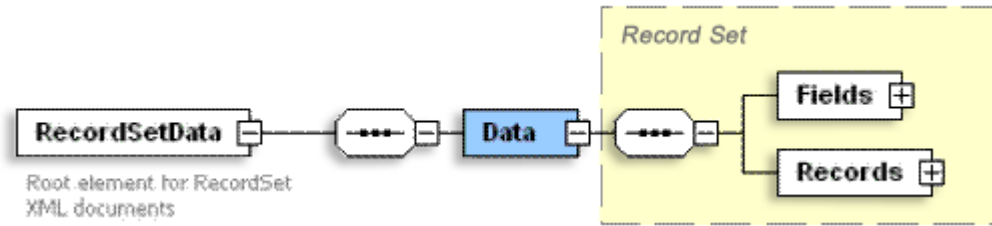
对表和要素类的来说，在定义部分中数据元素的名称和数据部分中的表数据名称是一一对应的。其他数据集，如 topologies, geometric networks, 或 要素数据集，仅在定义部分中出现（它们不能导出数据）。

Geodatabase 的 RecordSet document

Geodatabase 的 RecordSet document 用于从一个独立要素类或表中以简单要素或表记录导出为行集合。

导出到一个 RecordSet document 相当于导出到一个 shapefile 文件。行集合以记录被导出，并且没有附加的 geodatabase-related 信息被写到输出文件。

例如，topologies 和其他要素数据集信息将不会被导出，关系类也不会被导出。



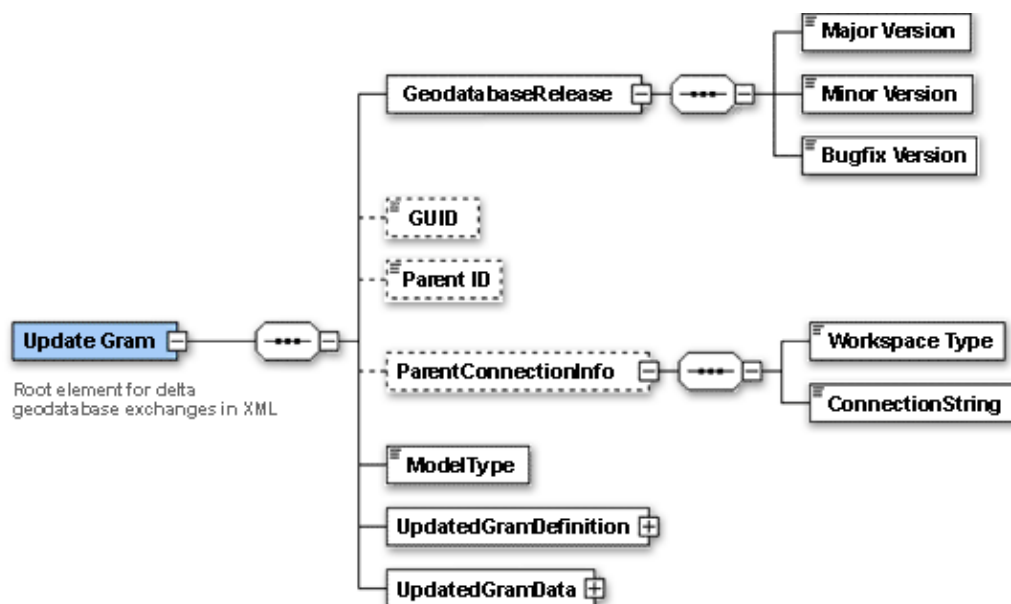
Geodatabase 的 XML Data Changes document

作为 geodatabase 事务和版本管理的一部分, 用户需要共享和操作 change-only 记录集。XML Data Changes document 用于共享 geodatabase 之间、geodatabase 和外部系统之间的变化和更新。要做到如此仅能通过分发变化, 可以通过 geodatabase 的版本追踪这些变化。

ArcGIS 的离线编辑框架允许将数据库中的数据签出一个单独的 geodatabase 中, 然后在与父数据库断开连接的状态下对数据进行编辑。一旦编辑完成, 就可以导出变化 (并不是所有的数据) 到 XML file。这个 XML file 可以随后用于签入变化到父数据库中。

这也是 geodatabase replication 关键的一部分。

XML Data Changes document 的基础信息是 UpdateGram, 在 UpdateGram 中只有变化可以作为 XML document 被传递。

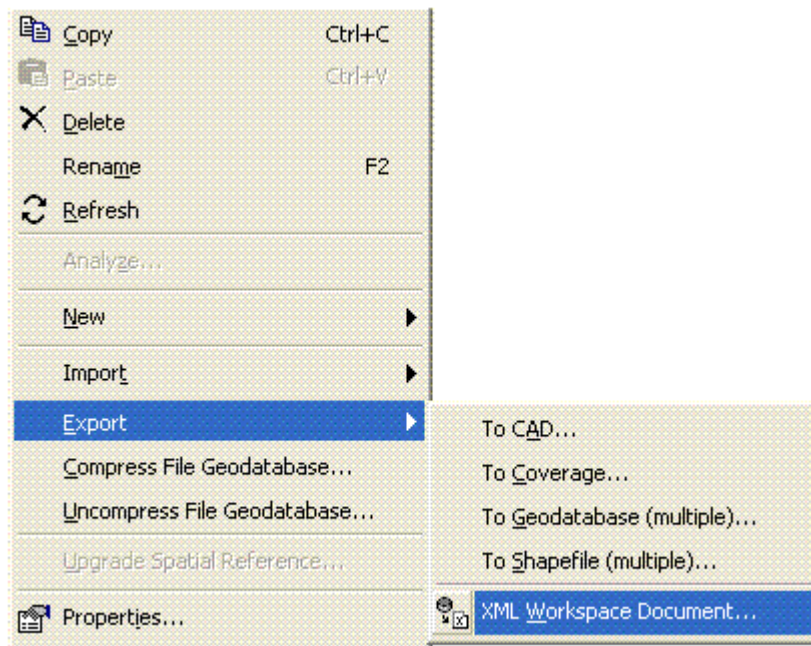


在 ArcCatalog 和 Geoprocessing 中使用 geodatabase XML

在 ArcGIS 中, ArcCatalog 和 Geoprocessing 框架包含大量的功能, 用于导入和导出地理数据。在 ArcGIS 中, 用户可以用 Geodatabase XML 执行的操作包含以下几种:

- 共享 geodatabase 的 schemas;
- 复制整个 geodatabase 或其子集, 如要素数据集和所有相关的信息集合;
- 复制独立的要素类, 表或栅格数据;
- 跨多个 geodatabase 复本同步其内容。

例如, 使用 ArcCatalog, 通过在一个 geodatabase 或要素类上单击右键, 创建一个 XML Workspace Document, 这个 Document 可以用于共享一个 geodatabase 的 schema 或者复制其内容、规则和行为到另外一个 geodatabase。



在 ArcGIS 中，使用 geoprocessing，也可以执行一些操作实现跨 geodatabase 同步和共享变化。