

中华人民共和国国家标准

GB/T 30320—2013

地理空间数据库访问接口

Geospatial database call-level interface

2013-12-31 发布

2014-06-01 实施

中华人民共和国国家质量监督检验检疫总局 发布
中国国家标准化管理委员会

目 次

前言	V
1 范围	1
2 规范性引用文件	1
3 术语和定义	1
4 符号和缩略语	3
4.1 符号	3
4.2 缩略语	3
4.3 UML 图示符号	4
4.4 UML 多样性描述	4
5 空间数据库数据访问对象逻辑关系	5
5.1 概述	5
5.2 数据源	5
5.3 数据集	5
5.4 矢量数据集	5
5.5 要素集	6
5.6 栅格数据集	6
5.7 空间数据模型	6
6 空间数据库数据访问对象描述	6
6.1 数据源	6
6.2 数据连接	9
6.3 数据集	10
6.4 数据集类型	12
6.5 矢量数据集	13
6.6 要素集	15
6.7 查询对象	17
6.8 要素	20
6.9 字段信息	21
6.10 字段信息集合	24
6.11 栅格数据集	24
6.12 波段信息	25
6.13 像素格式	26
6.14 像素	27
6.15 扫描线	27
6.16 数据块	28
7 元数据访问对象	29

7.1 元数据对象逻辑关系	29
7.2 元数据访问对象定义	30
8 空间参照系	33
8.1 空间参照系对象逻辑关系	33
8.2 空间参照系对象定义	34
9 数据提供者加载与卸载	35
9.1 概述	35
9.2 数据提供者管理器	36
9.3 数据提供者	37
参考文献	38
图 1 GDB-CLI 数据访问对象 UML 图	5
图 2 数据源相关对象 UML 图	7
图 3 数据源对象属性和方法	7
图 4 连接对象属性	9
图 5 数据集相关对象 UML 图	10
图 6 数据集对象属性和方法	11
图 7 数据集类型	12
图 8 矢量数据集相关对象 UML 图	13
图 9 矢量数据集属性和方法	14
图 10 要素集对象的方法	16
图 11 查询对象 UML 图	18
图 12 要素对象 UML 图	21
图 13 字段信息对象 UML 图	22
图 14 栅格数据集对象 UML 图	24
图 15 波段信息对象 UML 图	25
图 16 像素格式枚举对象 UML 图	26
图 17 像素对象 UML 图	27
图 18 扫描线对象 UML 图	28
图 19 数据块对象 UML 图	28
图 20 元数据对象 UML 图	30
图 21 元数据库对象属性和方法	31
图 22 元数据集对象属性和方法	32
图 23 空间参考系对象 UML 图	34
图 24 数据提供者管理器类图	36
表 1 符号对照表	3
表 2 缩写对照表	3
表 3 UML 图符号对照表	4
表 4 UML 图多样性描述	4
表 5 数据集与空间对象的存储对应关系	6

表 6	数据集类型描述	12
表 7	查询选项枚举	19
表 8	查询类型枚举	19
表 9	游标类型枚举	20
表 10	字段标记列表	23
表 11	字段类型描述列表	23
表 12	像素格式描述	26



前 言

本标准按照 GB/T 1.1—2009 给出的规则起草。

本标准由国家测绘地理信息局提出。

本标准由全国地理信息标准化技术委员会(SAC/TC 230)归口。

本标准起草单位:北京超图软件股份有限公司、武大吉奥信息技术有限公司、武汉中地数码科技有限公司、中遥地网信息技术有限公司、北京大学遥感与地理信息系统研究所。

本标准主要起草人:李绍俊、邓跃进、刘冬林、周芹、陈俊华、曾志明、陈国雄、李文龙、江志斌、高凤飞、谢林。



地理空间数据库访问接口

1 范围

本标准规定了地理空间数据库的数据访问对象模型及接口,对满足本标准接口的空间数据提供者的加载和卸载作出规定。

本标准适用于通过标准接口访问或共享地理空间数据库的应用,侧重于从数据使用者和数据提供者角度描述访问对象模型及接口,不适用于坐标参照系和元数据的具体内容。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件,仅注日期的版本适用于本文件。凡是不注日期的引用文件,其最新版本(包括所有的修改单)适用于本文件。

GB/T 19710—2005 地理信息 元数据(ISO 19115:2003,MOD)

ISO 19111:2007 地理信息 基于坐标的空间参照(Geographic information—Spatial referencing by coordinates)

ISO 19125-1:2004 地理信息 简单要素存取 第1部分:通用框架(Geographic information—Simple feature access—Part 1:Common architecture)

3 术语和定义

下列术语和定义适用于本文件。

3.1

地理空间数据库 geospatial database

用来存贮、管理和检索空间数据,表达空间实体的位置、形状、大小及其分布特征等方面信息的数据库。

3.2

数据提供者 data provider

实现了地理空间数据库访问接口以及相关的配置说明的对象。

3.3

数据提供者管理器 data provider manager

管理数据提供者的对象,包括对数据提供者的加载、卸载、查找等操作。

3.4

数据使用者 data customer

调用地理空间数据库访问接口操作数据提供者的空间数据的对象。数据使用者可以使用标准的接口访问不同数据提供者的空间数据。

3.5

要素 feature

现实世界现象的抽象。

[ISO 19101]

注：要素对应于数据集或要素集中的一条记录，要素实例中最多有一个几何空间对象或栅格空间对象，但可以有一个属性值。

3.6

空间对象 spatial object

代表要素空间特征的对象。

[GB/T 23707—2009]

3.7

栅格数据 raster data

被表示成有规则的空间阵列的数据。

3.8

矢量数据 vector data

由几何元素所表示的数据。

3.9

数据集 dataset

可以标识的数据集合。

[ISO 19101]

注：数据集是由同种类型数据组成的数据集合，也就是一组数据对象的集合。

3.10

矢量数据集 vector dataset

一组矢量数据要素(feature)的集合，具有相同的几何对象类型、相同的空间参照系和相同的字段信息结构。

3.11

栅格数据集 raster dataset

具有长度和宽度范围的一组有规则的空间阵列数据的集合，这些阵列数据有相同的空间参照系，阵列数据值即代表对应空间位置的属性且所有值描述的是同类属性，比如温度、高程值或其他。

3.12

数据源 data source

由数据提供者提供的各种类型的数据集(如点、线、面、影像、格网数据集)组成的数据集集合。一个数据源可包含一个或多个不同类型的数据集；一个数据源可以对应一个数据文件或者数据库。

3.13

要素集 featureset

对矢量数据集进行条件查询的结果集。查询条件可以是空间关系或者属性条件。

3.14

游标类型 cursor type

标识要素集遍历数据的方式。

3.15

字段 field

表示属性信息的集合。

3.16

字段信息 field info

字段的相关信息，如名称、类型、长度等。

3.17

坐标 coordinate

用来指示 N 维空间中点的位置的数值序列。

[GB/T 23707—2009]

3.18

坐标参照系 **coordinate reference system**

通过基准与现实世界相关的坐标系。

[ISO 19111:2007]

3.19

大地坐标系 **geodetic coordinate system**

位置由大地经度和纬度及大地高程(在三维的情况下)确定的坐标系。

[ISO 19111:2007]

3.20

投影坐标系 **projected coordinate system**

由地图投影产生的二维坐标系。

[ISO 19111:2007]

3.21

元数据 **metadata**

关于数据的内容、质量、状况和其他特性的描述性数据。

[GB/T 19710—2005]

4 符号和缩略语

4.1 符号

表 1 描述的符号适用于本文件。

表 1 符号对照表

符号	含 义
~	表示在符号前后对象之间取值
	表示符号前后项任选其一
[]	表示括号内的对象为可选
n	表示数字
(,)/[,]	表示在符号左端数值到右端数值的区间中任取其一,()表示开区间;[]表示闭区间

4.2 缩略语

表 2 描述的缩略语适用于本文件。

表 2 缩写对照表

描述符	中文名称	英文名称
GDB-CLI	空间数据库访问接口	Geospatial Database Call-Level Interface
DDL	数据定义语言	Data Definition Language
DML	数据操纵语言	Data Manipulation Language

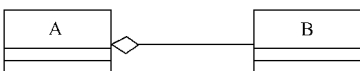
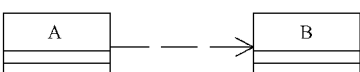
表 2（续）

描述符	中文名称	英文名称
UML	统一建模语言	Unified Modelling Language
SRID	空间参照系唯一标识	Spatial Reference System Identifier
UUID	通用唯一识别码	Universally Unique Identifier

4.3 UML 图符号

本标准出现的图用 UML 静态结构表示，所有数据模型 UML 图示中符号表示的规定见表 3。

表 3 UML 图符号对照表

符号	名称	说明
	双向关联	表示 A、B 两个类之间的一般关系，两个类都知道另一个类的公共属性和方法
	单向关联	表示 A、B 两个类之间的关联关系，A 类知道 B 类的公共属性和方法，但 B 类不知道 A 类的公共属性和方法
	聚合	A 对象拥有 B 对象，A 对象可以包含 B 对象，但 B 对象不是 A 对象的组成部分，二者生命周期可以不同
	组合	A 对象拥有 B 对象，是整体和部分的关系，且生命周期一致
	泛化	B 对象继承 A 对象，即 B 对象由 A 对象派生
	依赖	A 类依赖于 B 类，B 类的变化将影响 A 类。如果 A 类依赖 B 类，则 B 可以体现为 A 的局部变量、方法的参数或者静态方法的调用

4.4 UML 多样性描述

本标准涉及的 UML 图中多样性描述含义见表 4。

表 4 UML 图多样性描述

多样性	意 义
0..1	0 个或 1 个
1	只能 1 个
0..n	0 个或多个
1..n	1 个或多个

5 空间数据库数据访问对象逻辑关系

5.1 概述

GDB-CLI 的空间数据库访问对象如图 1 所示。数据源 (GDBDataSource) 是各种数据集 (GDBDataset) 的容器, 统一管理所有的数据集。数据源和数据集可以关联相关的元数据 (GDBMetadata), 数据集还可以具有相应的空间参照系统 (GDBCoordReferenceSys)。

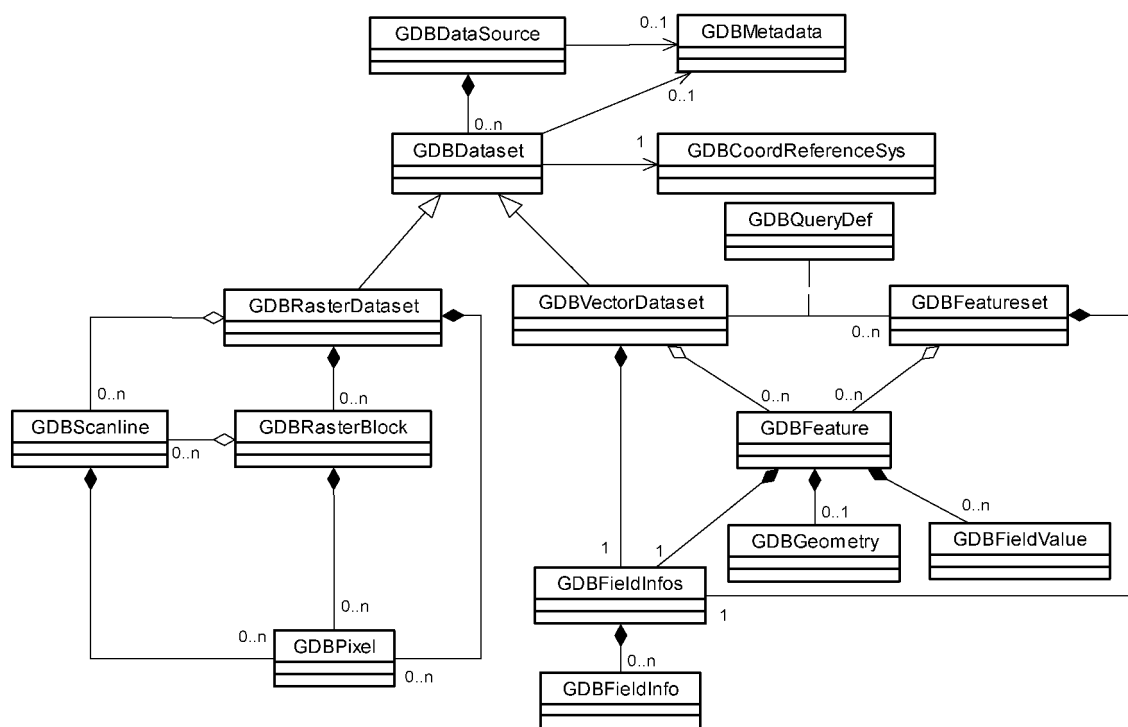


图 1 GDB-CLI 数据访问对象 UML 图

5.2 数据源

数据源 (GDBDataSource) 是由各种类型的数据集 (如点、线、面、影像、格网数据集) 组成的数据集集合。一个数据源可包含一个或多个相同类型的数据集, 也可以同时管理多个不同类型的数据集。

5.3 数据集



数据集 (GDBDataset) 是由相同特征的空间对象组成的数据集集合。根据对空间信息的描述方式不同而派生两大子类: 矢量数据集 (GDBVectorDataset) 和栅格数据集 (GDBRasterDataset)。

5.4 矢量数据集

矢量数据集是矢量要素 (GDBFeature) 的聚合, 每个要素可以有几何 (GDBGeometry) 和属性 (GDBFieldValue) 数据。

对矢量数据集的查询结果可以用要素集 (GDBFeatureset) 表示。

5.5 要素集

要素集是要素的聚合,它是数据集中数据的一个投影或访问窗口。矢量数据集和要素集使用字段信息集合(GDBFieldInfos)表示属性结构信息。

矢量数据的编辑操作包括添加或修改要素的空间或属性信息、删除要素。编辑操作提供两种方式:

- 要素集方式。对矢量数据集进行查询得到要素集(GDBFeatureSet),通过要素集分别操作空间数据和属性数据。参见 6.6.2。
- 对象方式。以要素(GDBFeature)为基本单元的数据操作。通过操作矢量数据集或要素集中的要素(GDBFeature)来操作空间数据和属性数据。参见 6.5.3 和 6.6.2。

5.6 栅格数据集

栅格数据集是按规则排列的一系列像素点或数值的聚合,它用一个规则格网来描述与每一个格网单元位置相对应的空间现象特征的位置和取值,每一个格网单元为一个像素。栅格数据集常用来存储影像、数字高程模型(DEM)、格网(GRID)等数据。

GDB-CLI 通过数据块(GDBRasterBlock)、扫描线(GDBScanline)、像素点(GDBPixel)三种模式来访问栅格数据集中的数据。

5.7 空间数据模型

GDB-CLI 的空间数据模型采用 ISO 19125-1:2004 规定的简单对象模型,各空间对象(GDBGeometry)与数据集的存储对应关系见表 5。表中空间对象的 is3D()和 isMeasured()应与数据集的 hasZDimension 和 hasMDimension 属性对应,即如果空间对象的 is3D()方法返回 TRUE,则应存储在 hasZDimension 属性为 TRUE 的数据集中,对 Measure 属性类同。

表 5 数据集与空间对象的存储对应关系

数据集类型	空间对象类型
Point	Point, MultiPoint
Line	LineString, Line, LinearRing, MultiLineString
Region	Polygon, Triangle, MultiPolygon
Model	PolyhedralSurface
Text	Annotation Text
Compound	以上各种空间对象, GeometryCollection

6 空间数据库数据访问对象描述

6.1 数据源



6.1.1 对象描述

数据源的职责包括:1)建立到后台数据库或数据文件的访问链接;2)对各数据集的组织和管理(如对数据集元数据的查询、修改,以及对数据集的创建、删除等);3)数据源自身的元数据说明。数据源的对象 UML 图见图 2。

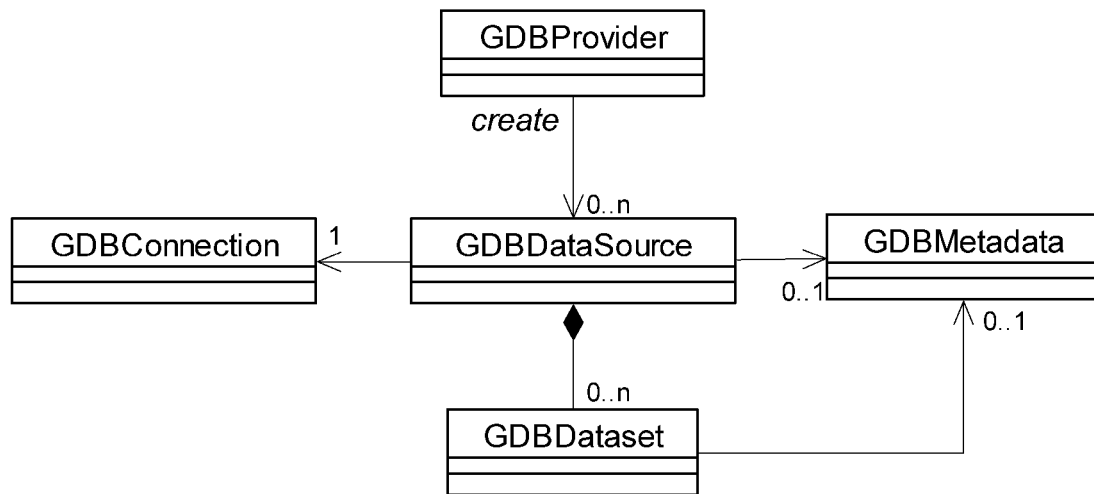


图 2 数据源相关对象 UML 图

数据源对象的属性和方法图见图 3。

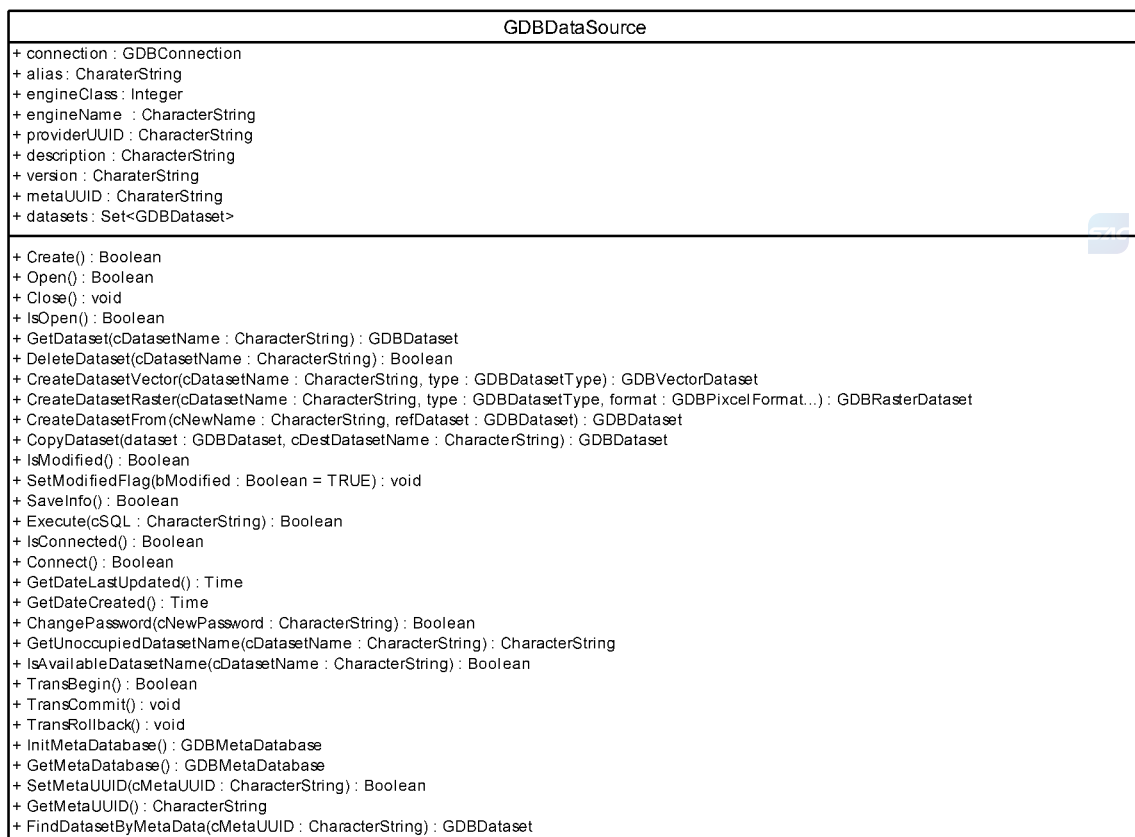


图 3 数据源对象属性和方法

6.1.2 属性

数据源对象的属性如下：

- `connection`: `GDBConnection`—数据源的连接信息,通过此信息打开数据源;
- `alias`: `CharaterString`—数据源别名,是当前环境下数据源实例对象的唯一标识;
- `engineClass`: `Integer`—数据源的引擎类型,与对应的数据提供者的该属性值相等,见 9.3.3;
- `engineName`: `CharacterString`—数据源的引擎名称;
- `providerUUID`: `CharacterString`—数据源的提供者的唯一标识,与对应数据提供者的 `uuid` 属性值相等,见 9.3.3;
- `description`: `CharacterString`—数据源描述信息;
- `version`: `CharaterString`—数据源的版本名称,与对应数据提供者的 `version` 属性值相等,见 9.3.3;
- `metadataUUID`: `CharaterString`—关联的元数据的唯一标识字符串,可以为空,为空则表示该数据源对象没有关联元数据;
- `datasets`: `Set<GDBDataset>`—数据源中的所有数据集的集合。

6.1.3 方法

数据源对象的方法如下:

- `Create()`: `Boolean`—创建数据源,需要预先设置好 `connection` 属性。
- `Open()`: `Boolean`—连接数据库,打开数据源,需要预先设置好 `connection` 属性。
- `Close()`: `void`—关闭数据源。
- `IsOpen()`: `Boolean`—判断数据源是否已经打开。
- `GetDataset(cDatasetName: CharacterString)`: `GDBDataset`—通过指定名称返回数据集对象。
- `DeleteDataset(cDatasetName: CharacterString)`: `Boolean`—通过指定名称删除数据集。
- `CreateDatasetVector(cDatasetName: CharacterString, type: GDBDatasetType)`: `GDBVectorDataset`—创建新的矢量数据集。矢量数据集可以有自己的子数据集,参见 6.3.2。
- `CreateDatasetRaster(cDatasetName: CharacterString, type: GDBDatasetType, format: GDBPixelFormat, nWidth: Integer, nHeight: Integer, nBlocksizeX: Integer, nBlocksizeY: Integer, nBandCount: Integer)`: `GDBRasterDataset`—创建新的栅格数据集。栅格数据集也可以有自己的子数据集,参见 6.3.2。
- `CreateDatasetFrom(cNewName: CharacterString, refDataset: GDBDataset)`: `GDBDataset`—通过模板数据集创建新的数据集, `newName` 为新数据集名称, `refDataset` 为模板数据集。
- `CopyDataset(dataset: GDBDataset, cDatasetName: CharacterString)`: `GDBDataset`—复制数据集,源数据集 `srcDataset` 可以来自不同的数据源, `datasetName` 是复制后的数据集名称。
- `IsModified()`: `Boolean`—判断数据源对象是否有信息需要保存。
- `SetModifiedFlag(bModified: Boolean = TRUE)`: `void`—设置数据源修改标记,默认 `bModified` 为 `TRUE`,即标识数据源已被修改,有信息需要保存。
- `SaveInfo()`: `Boolean`—保存数据源信息。
- `Execute(cSQL: CharacterString)`: `Boolean`—执行 SQL 语句,主要用于在数据库级执行的数据操作,用以执行 DDL 和 DML,不返回结果集。
- `CheckConnection()`: `Boolean`—判断数据源是否仍然处于连接状态以及连接是否可用,主要用于数据库型的数据源。
- `Refresh()`: `Boolean`—刷新数据源以获取最新信息;此外,如果数据库型数据源因网络连接断开导致 `CheckConnection()` 方法检查到数据源的连接已经失效,恢复网络连接后,可以用 `Refresh`

()方法刷新数据源后继续访问数据。

- `GetDateLastUpdated():Time`—获取数据源最后修改的时间。
- `GetDateCreated():Time`—获取数据源创建的时间。
- `ChangePassword(cNewPassword;CharacterString):Boolean`—修改数据源连接密码。本地文件型数据源可以使用此接口修改数据源连接密码。
- `GetUnoccupiedDatasetName(cDatasetName;CharacterString):CharacterString`—获取唯一数据集名,返回数据集名称,该名称是该数据源中未被使用过的。
- `IsAvailableDatasetName(cDatasetName;CharacterString):Boolean`—判断数据集名是否可用。
- `TransBegin():Boolean`—开始事务。
- `TransCommit():void`—提交事务。
- `TransRollback():void`—回滚事务。
- `InitMetaDatabase():Boolean`—初始化与数据源对象关联的元数据库对象。
- `GetMetaDatabase():GDBMetaDatabase`—获取元数据库对象。
- `SetMetaUUID(cMetaUUID;CharacterString):Boolean`—给数据源设置元数据的 UUID,通过该标识就能在数据源关联的元数据库中找到对应的元数据。
- `GetMetaUUID():CharacterString`—获取数据源的元数据对象的 UUID。
- `FindDatasetByMetaData(cMetaUUID;CharacterString):GDBDataset`—通过元数据的 UUID 获取数据集,返回的数据集对象。如果该数据源中没有数据集的元数据 UUID 与传入的 `strMetaUUID` 相等,则返回空对象。

6.2 数据连接

6.2.1 对象描述

数据连接对象(GDBConnection)的属性见图 4。数据源通过 GDBConnection 提供的信息与数据库建立连接。

GDBConnection
+ server : CharacterString + user : CharacterString + password : CharacterString + readOnly : Boolean + transacted : Boolean + exclusive : Boolean + authorization : Boolean + database : CharacterString + driver : CharacterString + pipe : Boolean + port : Integer + connMaxNum : Integer + connMinNum : Integer + connIncNum : Integer

图 4 连接对象属性

6.2.2 属性

数据连接对象的属性如下：

- server:CharacterString—服务名。例如：对网络数据库型数据源，表示 Oracle 数据库的实例名，或者 SQL Sever 的服务器名；对本地文件型数据源，该属性为文件路径；对网络服务型数据源，该属性为网络服务地址。
- user:CharacterString—用户名。
- password:CharacterString—密码。
- readOnly:Boolean—是否只读打开。只读打开后数据所有的信息都不可修改。
- transacted:Boolean—是否以事务方式打开。
- exclusive:Boolean—是否独占打开。
- authorization:Boolean—数据库是否需要密码打开。
- database:CharacterString—数据库名。
- driver:CharacterString—驱动程序名称，例如不同数据库的 ODBC 驱动名称。
- pipe:Boolean—是否用命名管道中连接方式，MySQL 数据库连接时使用。
- port:Integer—连接端口，例如 MySQL 中普通连接方式的端口号。
- connMaxNum:Integer—数据库连接池的最大连接数。
- connMinNum:Integer—数据库连接池的最小连接数。
- connIncNum:Integer—数据库连接池的增长连接数。

6.3 数据集

6.3.1 对象描述

数据集与相关对象关联的 UML 图见图 5，数据集对象的方法和属性见图 6。

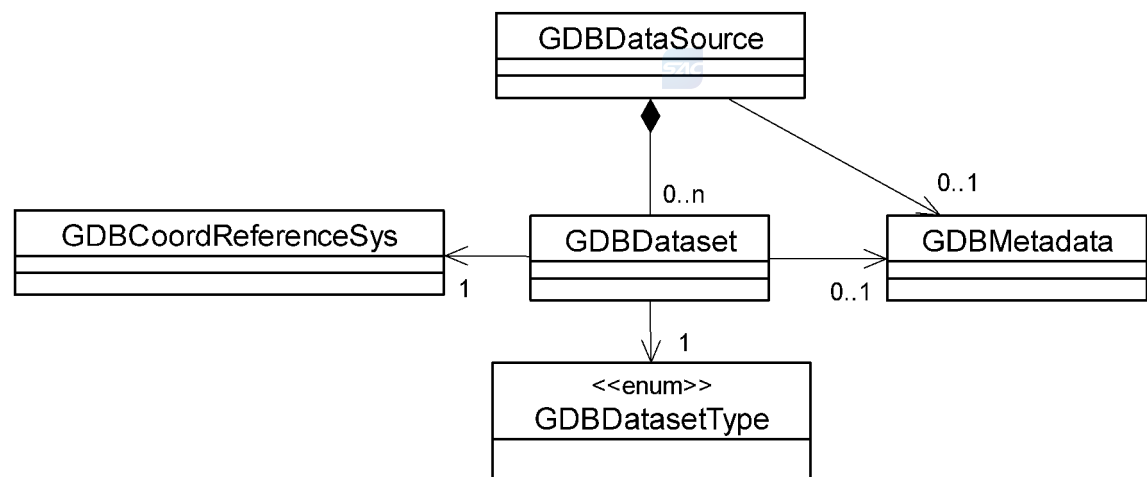


图 5 数据集相关对象 UML 图

GDBDataset
+ datasetName : CharacterString + tableName : CharacterString + datasetType : GDBDatasetType + rcBounds : GDBGeometry + isRaster : Boolean + parentDataset : GDBDataset + childDatasets : Set<GDBDataset> + datasource : GDBDataSource + spatialReferenceSys : GDBCoordReferenceSys + description : CharactorString + metadataUUID : CharacterString
+ Open() : Boolean + Close() : void + IsOpen() : Boolean + IsReadOnly() : Boolean + Rename() : Boolean + ComputeBounds () : Boolean + IsModified() : Boolean + SetModifiedFlag(bModified : Boolean = TRUE) : void + SaveInfo() : Boolean + RefreshInfo() : Boolean + GetDateLastUpdated() : Time + GetDateCreated() : Time + GetMetadata() : GDBMetadata

图 6 数据集对象属性和方法

6.3.2 属性

数据集对象的属性如下：

- datasetName:CharacterString—数据集名称。
- tableName:CharacterString—数据集在数据库中的数据表名称。
- datasetType:GDBDatasetType—数据集类型,为枚举值。见 6.4。
- rcBounds:GDBGeometry—数据集范围。
- isRaster:Boolean—是否是栅格数据集。
- parentDataset:GDBDataset—数据集的父数据集。
- childDatasets:Set<GDBDataset>—数据集的子数据集集合。子数据集强依赖于父数据集,如果父数据集消亡,则其子数据集也随之消亡,例如:栅格数据集的影像金字塔,可以将金字塔作为原栅格数据集的子数据集集合存储和管理,从而维护二者的关系。
- datasource:GDBDataSource—数据集所在的数据源对象。
- spatialReferenceSys:GDBCoordReferenceSys—数据集空间参照系信息。
- description:CharactorString—数据集的描述信息。
- metadataUUID:CharacterString—数据集关联的元数据的唯一标识。

6.3.3 方法

数据集对象的方法如下：

- Open():Boolean—打开数据集;
- Close():void—关闭数据集;
- IsOpen():Boolean—判断数据集是否已经打开;
- IsReadOnly():Boolean—数据是否能够被修改;
- Rename():Boolean—数据集重命名;
- ComputeBounds():Boolean—计算数据集范围,一般在数据集中元素被删除或修改后使用;
- IsModified():Boolean—判断数据集对象是否有信息需要保存;
- SetModifiedFlag(bModified:Boolean=TRUE):void—设置数据集修改标记,默认 bModified 为 TRUE,即标识数据集已被修改,有信息需要保存;
- SaveInfo():Boolean—保存数据集信息;
- RefreshInfo():Boolean—刷新数据集信息;
- GetDateLastUpdated():Time—获取数据集最后修改的时间;
- GetDateCreated():Time—获取数据集创建的时间;
- GetMetadata():GDBMetadata—获取数据集元数据对象。

6.4 数据集类型

在 GDB-CLI 中矢量数据集包括 7 种:纯属性数据集、点数据集、线数据集、面数据集、模型数据集、文本数据集、复合数据集;栅格数据集包括 2 种:GRID 数据集、Image 数据集。数据集类型以枚举对象提供,见图 7。每种类型的描述信息见表 6。

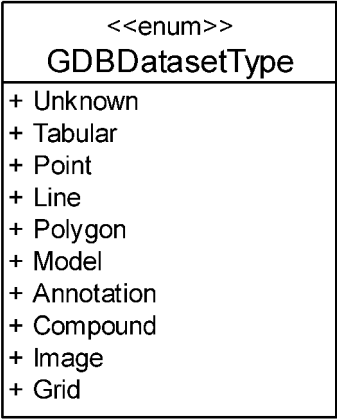


图 7 数据集类型

表 6 数据集类型描述

类 型	描 述
Unknown	未知类型,用于标记不识别的数据集
Tabular	仅包含属性信息,没有空间数据的数据集
Point	点数据集,存储点对象

表 6 (续)

类 型	描 述
Line	线数据集,存储线对象
Polygon	面数据集,存储面对象
Model	模型数据集,存储三维场景中的模型对象,一般包括模型的骨架和纹理等信息
Annotation	文本标注数据集,存储文本对象
Compound	复合数据集,可以存储多种类型的要素对象
Image	影像数据集,像素值代表颜色值
Grid	格网数据集,像素值具有具体的含义,如代表高程值、降雨量或光照量等

6.5 矢量数据集

6.5.1 对象描述

矢量数据集是矢量要素(Feature)的集合,同一数据集中的要素具有相同的属性特征。矢量数据集相关的对象 UML 图见图 8。矢量数据集的属性和方法见图 9。

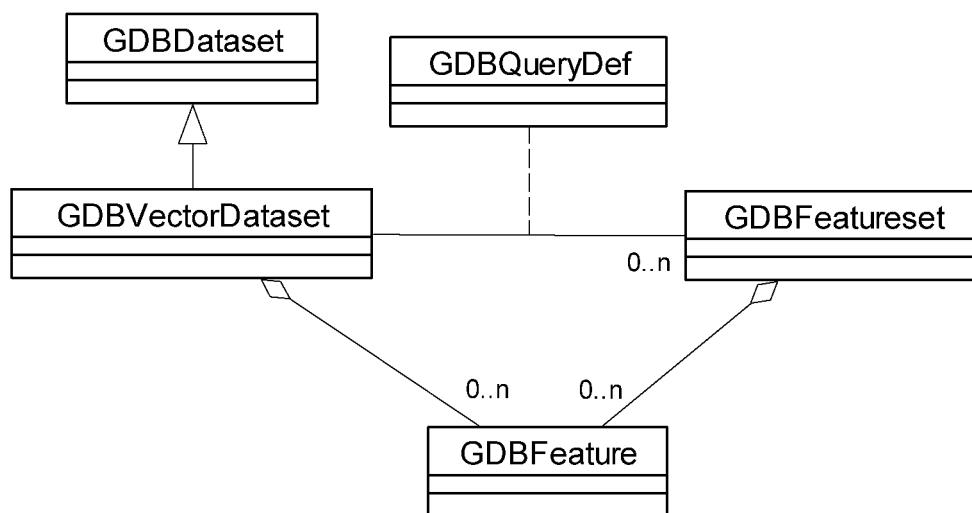


图 8 矢量数据集相关对象 UML 图

GDBVectorDataset
+ featuresets : Set<GDBFeatureset> + hasZDimension : Boolean + hasMDimension : Boolean
+ GetFeatureCount() : Integer + AddFeature(feature : GDBFeature) : Boolean + UpdateFeature(feature : GDBFeature) : Boolean + DeleteFeature(nID : Integer) : Boolean + DeleteFeatures(IDs : Set<Integer>) : Boolean + GetFeature(nID : Integer) : GDBFeature + Append(featureset : GDBFeatureset) : Boolean + DeleteAllFeatures() : Boolean + BuildSpatialIndex() : Boolean + DropSpatialIndex() : Boolean + IsSpatialIndexDirty() : Boolean + Query(querydef : GDBQueryDef&) : GDBFeatureset + ReleaseFeatureset(pRecordset : GDBFeatureset) : Boolean + ReleaseAllFeaturesets() : void + UpdateField(cFieldName : CharacterString, cExpress : CharacterString, cFilter : CharacterString) : Boolean + CopyField(cSrcFieldName : CharacterString, cDestFieldName : CharacterString) : Boolean + CreateFieldIndex(cFieldNames : Set<CharacterString>, cIndexName : CharacterString) : Boolean + DropFieldIndex(cIndexName : CharacterString) : Boolean + CreateFields(fieldInfos : GDBFieldInfos) : Boolean + DeleteField(cFieldName : CharacterString) : Boolean + GetFieldInfos(fieldInfos : GDBFieldInfos, bSystemFieldExcluded : Boolean = FALSE) : Boolean + GetFieldInfo(cFieldName : CharacterString, fieldInfo : GDBFieldInfo) : Boolean + SetFieldInfo(cFieldName : CharacterString, fieldInfo : GDBFieldInfo) : Boolean + Statistic(cFieldName : CharacterString, nMode : GDBFeatureset::GDBStatisticMode) : Variant + GetUnoccupiedFieldName(cFieldName : CharacterString) : CharacterString + IsAvailableFieldName(cFieldName : CharacterString) : Boolean

图 9 矢量数据集属性和方法

6.5.2 属性

矢量数据集对象的属性如下：

- featuresets: Set<GDBFeatureset>—数据集查询出来的要素集的集合。由数据集对象统一管理。
- hasZDimension: Boolean—是否带有 Z 属性, 如果带有 Z 属性则存储带高程值(Z)的空间对象。
- hasMDimension: Boolean—是否带有 M 属性, 如果带有 M 属性则存储带度量值(M)的空间对象。

6.5.3 方法

矢量数据集对象的方法如下：

- GetFeatureCount(): Integer—数据集中要素个数。
- AddFeature(feature: GDBFeature): Boolean—向数据集中添加新的 Feature。
- UpdateFeature(feature: GDBFeature): Boolean—根据 feature 的 ID 更新要素。
- DeleteFeature(nID: Integer): Boolean—删除指定 ID 的要素。
- DeleteFeatures(IDs: Set<Integer>): Boolean—删除 ID 包含在传入 ID 集合中的要素。
- GetFeature(nID: Integer): GDBFeature—获取指定 ID 的要素。
- Append(featureset: GDBFeatureset): Boolean—追加要素集。
- DeleteAllFeatures(): Boolean—清空数据集中所有要素。
- BuildSpatialIndex(): Boolean—创建空间索引。

- DropSpatialIndex(): Boolean—删除空间索引。
- IsSpatialIndexDirty(): Boolean—判断空间索引是否为脏。
- Query(querydef: GDBQueryDef&): GDBFeatureset—根据指定条件查询数据集。
- ReleaseFeatureset(pRecordset: GDBFeatureset): Boolean—释放指定的要素集。
- ReleaseAllFeaturesets(): void—释放所有查询出来的要素集。
- UpdateField(cFieldName: CharacterString, cExpress: CharacterString, cFilter: CharacterString): Boolean—根据表达式和过滤条件更新字段值。
- CopyField(cSrcFieldName: CharacterString, cDestFieldName: CharacterString): Boolean—复制字段值, 目的字段名应都存在且目的字段要与源字段类型匹配。
- CreateFieldIndex(cFieldNames: Set<CharacterString>, cIndexName: CharacterString): Boolean—创建字段索引。
- DropFieldIndex(cIndexName: CharacterString): Boolean—删除指定索引名称的字段索引。
- CreateFields(fieldInfos: GDBFieldInfos): Boolean—创建字段。
- DeleteField(cFieldName: CharacterString): Boolean—删除指定名字的字段。
- GetFieldInfos (fieldInfos: GDBFieldInfos, bSystemFieldExcluded: Boolean = FALSE): Boolean—获取字段信息。bSystemFieldExcluded 表示是否将系统字段一起返回。
- GetFieldInfo(cFieldName: CharacterString, fieldInfo: GDBFieldInfo): Boolean—根据字段名获取字段信息。
- SetFieldInfo (cFieldName: CharacterString, fieldInfo: GDBFieldInfo): Boolean—设置字段信息。
- Statistic(cFieldName: CharacterString, nMode: GDBFeatureset::GDBStatisticMode): Variant—统计字段信息。
- GetUnoccupiedFieldName(cFieldName: CharacterString): CharacterString—获取新的合法的字段名。
- IsAvailableFieldName(cFieldName: CharacterString): Boolean—判断输入的字段名称是否可以作为新的字段名。

6.6 要素集

6.6.1 对象描述

要素集(GDBFeatureset)是矢量数据集的一个子集。要素集可以是根据一定的条件对矢量数据集(GDBDatasetVector)进行查询得到的结果集。要素集是访问矢量数据的主要对象, 可以通过要素集获取、修改和存储矢量数据。要素集对象的方法见图 10。

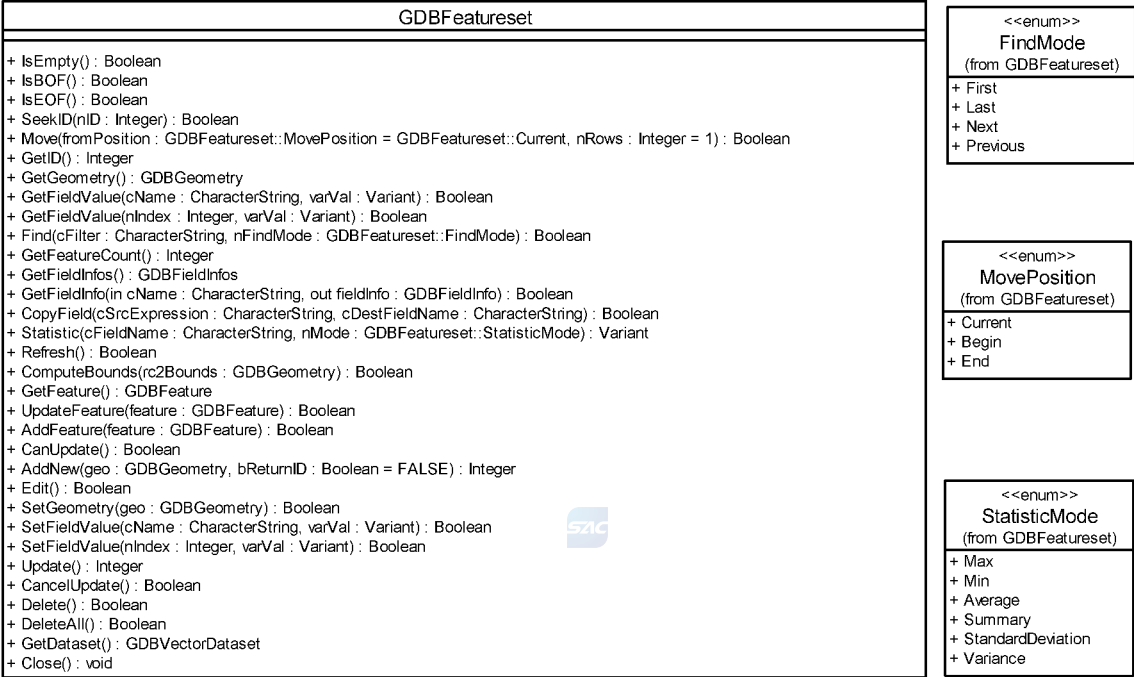


图 10 要素集对象的方法

6.6.2 方法

要素集对象的方法如下：

- IsEmpty(): Boolean—判断要素集是否为空。
- IsBOF(): Boolean—判断游标位置是否已在表头。
- IsEOF(): Boolean—判断游标位置是否已在表尾。
- SeekID (nID: Integer): Boolean—将游标移动到指定 ID 的位置。
- Move (fromPosition: GDBFeatureset::MovePosition = GDBFeatureset::Current, nRows: Integer = 1): Boolean—将游标移动到指定位置。参数 nRows 表示移动的记录数, 大于 0 时向后移动, 小于 0 时向前移动; 参数 fromPosition 为 GDBFeatureset::MovePosition 类型, 表示移动的起始位置, 为枚举常量, 包含: Begin—开始位置, Current—当前位置, End—结尾位置。
- GetID (): Integer—获取当前记录的 ID。
- GetGeometry(): GDBGeometry—获取当前位置的空间对象。
- GetFieldValue (cName: CharacterString, varVal: Variant): Boolean—获取当前名称的字段
- GetFieldValue(nIndex: Integer, varVal: Variant): Boolean—获取当前位置指定字段的值。
- Find (cFilter: CharacterString, nFindMode: GDBFeatureset::FindMode): Boolean—查找指定条件的记录。cFilter 为查询条件, 例如 “ID=5”, 查找 ID 为 5 的记录; nFindMode 为查找模式, 查找当前模式下满足条件的记录。GDBFeatureset::FindMode 为枚举常量, 表示查找模式, 包括: First—第一个, Last—最后一个, Next—下一个, Previous—上一个。
- GetFeatureCount (): Integer—要素数目。
- GetFieldInfos(): GDBFieldInfos—结果集字段信息。

- `GetFieldInfo(in cName:CharacterString,out fieldInfo:GDBFieldInfo):Boolean`—通过字段名获取字段信息。
- `CopyField(cSrcExpression:CharacterString,cDestFieldName:CharacterString):Boolean`—复制字段值。
- `Statistic(cFieldName:CharacterString,nMode:GDBFeatureset::StatisticMode):Variant`—对指定的字段进行数理统计。`GDBFeatureset::StatisticMode` 为枚举类型,包括:Max—最大值,Min—最小值,Average—平均值,Summary—求和,StandardDeviation—标准差,Variance—方差。
- `Refresh():Boolean`—重新执行查询刷新要素集。
- `ComputeBounds(rc2Bounds:GDBGeometry):Boolean`—计算要素集的范围。
- `GetFeature():GDBFeature`—获取当前要素的 Feature。
- `UpdateFeature(feature:GDBFeature):Boolean`—更新当前要素。
- `AddFeature(feature:GDBFeature):Boolean`—添加新的 Feature。
- `CanUpdate():Boolean`—是否能够修改要素集。
- `AddNew(geo:GDBGeometry,bReturnID:Boolean=FALSE):Integer`—增加新记录。
- `Edit():Boolean`—开始修改数据。
- `SetGeometry(geo:GDBGeometry):Boolean`—改写当前记录的几何对象。
- `SetFieldValue(cName:CharacterString,varVal:Variant):Boolean`—根据名称更新当前记录的属性数据。
- `SetFieldValue(nIndex:Integer,varVal:Variant):Boolean`—根据索引号更新当前记录的属性数据。
- `Update():Integer`—更新,提交修改。
- `CancelUpdate():Boolean`—取消当前要素的修改操作。
- `Delete():Boolean`—删除当前记录。
- `DeleteAll():Boolean`—删除所有记录。
- `GetDataset():GDBDatasetVector`—获取要素集对应的数据集。
- `Close():void`—关闭要素集。

6.7 查询对象

6.7.1 对象描述

查询对象(`GDBQueryDef`),用以构造查询参数。`GDBVectorDataset` 的 `Query` 方法中传入此对象进行查询,得到结果要素集 `GDBFeatureset`。查询对象的 UML 图见图 11。

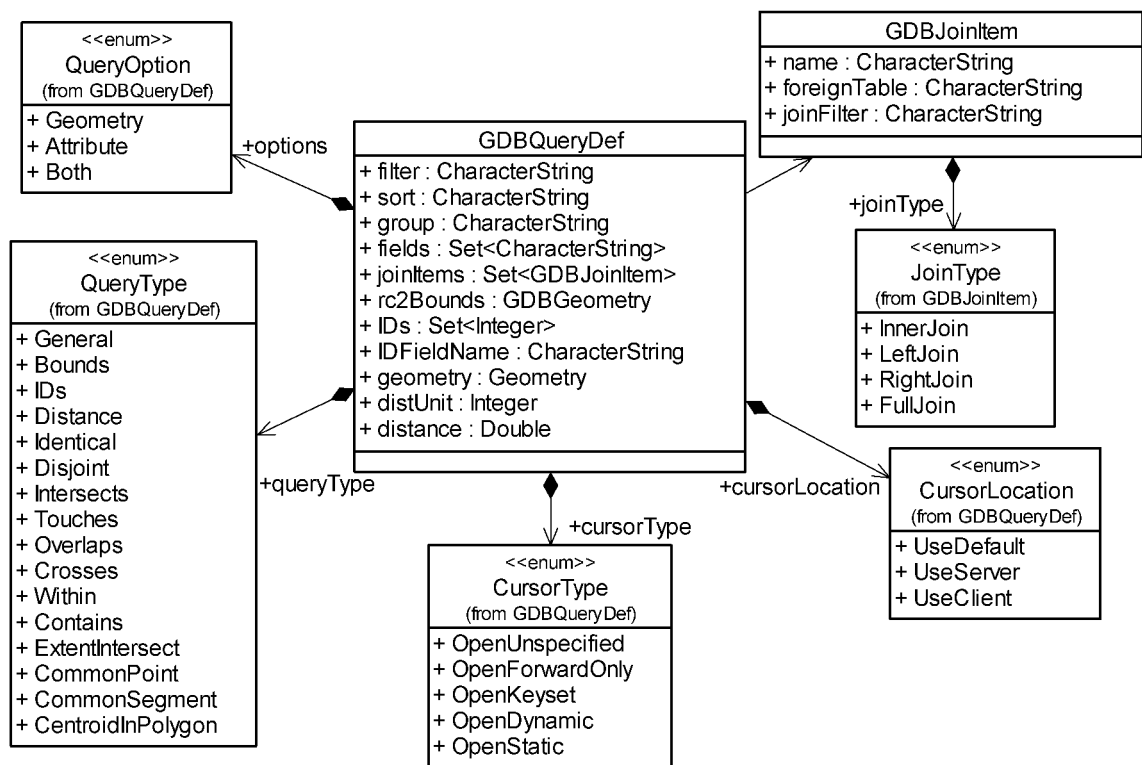


图 11 查询对象 UML 图

6.7.2 属性

查询对象的属性如下：

- filter:CharacterString—查询过滤条件；
- sort:CharacterString—排序子句；
- group:CharacterString—分组子句；
- fields:Set<CharacterString>—需要查询的字段列表；
- joinItems:Set<GDBJoinItem>—关联查询条件设置；
- rc2Bounds:GDBGeometry—二维矩形定义的查询范围，范围查询使用；
- IDs:Set<Integer>—需要查询的 ID 列表；
- IDFieldName:CharacterString—ID 字段名称；
- geometry:Geometry—用来空间查询的空间对象；
- options:QueryType—查询选项；
- queryType:QueryType—查询类型；
- cursorType:CursorType—游标类型；
- cursorLocation:CursorLocation—游标位置；
- distUnit:Integer—距离查询单位；
- distance:Double—距离值，距离查询使用；
- joinItems:Set<GDBJoinItem>—关联查询条件。

6.7.3 查询选项

查询选项(GDBQueryDef;QueryOption)，枚举类型见表 7。

表 7 查询选项枚举

类 型	描 述
Geometry	仅查出空间对象
Attribute	仅查出属性数据
Both	空间对象和属性同时查询

6.7.4 查询类型

查询类型(GDBQueryDef:QueryType),枚举类型见表 8。

表 8 查询类型枚举

类 型	描 述
General	一般的属性条件查询
Bounds	使用地理范围矩形查询
IDs	指定 ID 数组查询
Distance	距离查询
Identical	空间查询:同一。返回被搜索图层中与搜索对象完全相同的对象,包括对象类型和坐标
Disjoint	空间查询:相离。返回被搜索图层中与搜索对象相离的对象
Intersects	空间查询:交叠。如果搜索对象是面,返回全部或部分被搜索对象包含的对象以及全部或部分包含搜索对象的对象;如果搜索对象不是面,返回全部或部分包含搜索对象的对象(面)
Touches	空间查询:接触。返回被搜索图层中其边界与搜索对象边界相触的对象
Overlaps	空间查询:重叠。返回被搜索图层中与搜索对象部分有重叠的对象
Crosses	空间查询:相交。返回被搜索图层中与搜索对象(线或面)相交的所有对象(点、线或面)
Within	空间查询:被包含。返回被搜索图层中包含搜索对象的对象。 如果返回的对象是面,其应全部包含(包括边接触)搜索对象; 如果返回的对象是线,其应完全包含搜索对象; 如果返回的对象是点,其应与搜索对象重合
Contains	空间查询:包含。返回被搜索图层中被搜索对象包含的对象。包括在搜索对象边界上的情况
ExtentIntersect	空间查询:边界矩形相交。返回被搜索图层中其边界矩形与搜索对象的边界矩形相交但其自身不与搜索对象相交的所有对象
CommonPoint	空间查询算子。返回被搜索图层中与搜索对象至少有一个公共节点的所有对象
CommonSegment	空间查询算子。返回被搜索图层中与搜索对象至少有一条公共边(完全重合,即在重合的边上共所有节点(顶点 Vertex 或端点 Endpoint))的所有对象
CentroidInPolygon	空间查询算子。返回被搜索图层中内点在搜索对象内部的面对象

6.7.5 游标类型

游标类型(GDBQueryDef:CursorType),枚举类型见表 9。

表 9 游标类型枚举

类 型	描 述
OpenUnspecified	未定义的游标类型
OpenForwardOnly	只向前游标
OpenKeyset	关键集游标
OpenDynamic	动态游标,可编辑
OpenStatic	静态游标,不可编辑

6.7.6 游标位置

游标位置(GDBQueryDef:CursorLocation),枚举类型如下:

- UseDefault—不指定游标位置;
- UseServer—服务器端游标;
- UseClient—使用客户端游标。

6.7.7 关联条件

关联条件(GDBJoinItem),其属性如下:

- name:CharacterString—关联对象本身的名称,主要起标识作用。
- foreignTable:CharacterString—外部关联表;
- joinFilter:CharacterString—关联条件;
- joinType:JoinType—连接类型。

连接类型(GDBJoinItem :JoinType)的枚举类型如下:

- InnerJoin—内连接;
- LeftJoin—左连接;
- RightJoin—右连接;
- FullJoin—全连接。

6.8 要素

6.8.1 对象描述

要素是矢量数据集中的基本数据单元,一个矢量数据集由多个要素构成。要素的对象 UML 图见图 12。

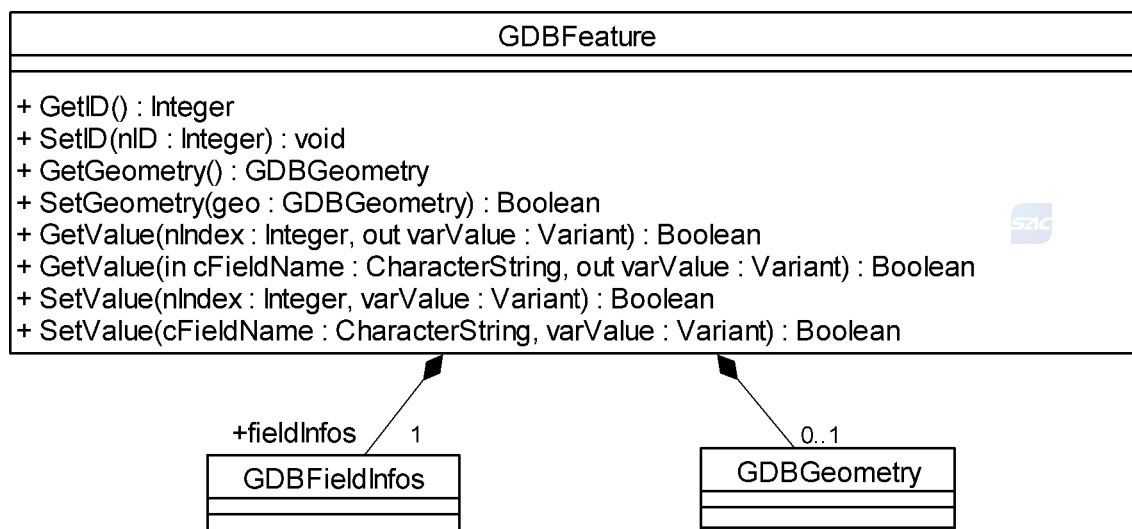


图 12 要素对象 UML 图

6.8.2 方法

要素对象的方法如下：

- `GetID() : Integer`—要素的 ID；
- `SetID(nID : Integer) : void`—设置要素的 ID；
- `GetGeometry() : GDBGeometry`—要素的几何对象；
- `SetGeometry(geo : GDBGeometry) : Boolean`—设置要素的几何对象；
- `GetValue(nIndex : Integer, out varValue : Variant) : Boolean`—根据索引号获取要素的属性字段值；
- `GetValue(in cFieldName : CharacterString, out varValue : Variant) : Boolean`—根据字段值获取要素的属性字段值；
- `SetValue(nIndex : Integer, varValue : Variant) : Boolean`—根据索引号设置字段值；
- `SetValue(cFieldName : CharacterString, varValue : Variant) : Boolean`—根据字段名设置字段值。

6.9 字段信息

6.9.1 对象描述

矢量数据集中的一列为一个字段，字段信息定义了列中的数据的数据类型、长度、是否必填等信息。字段信息的对象 UML 图见图 13。

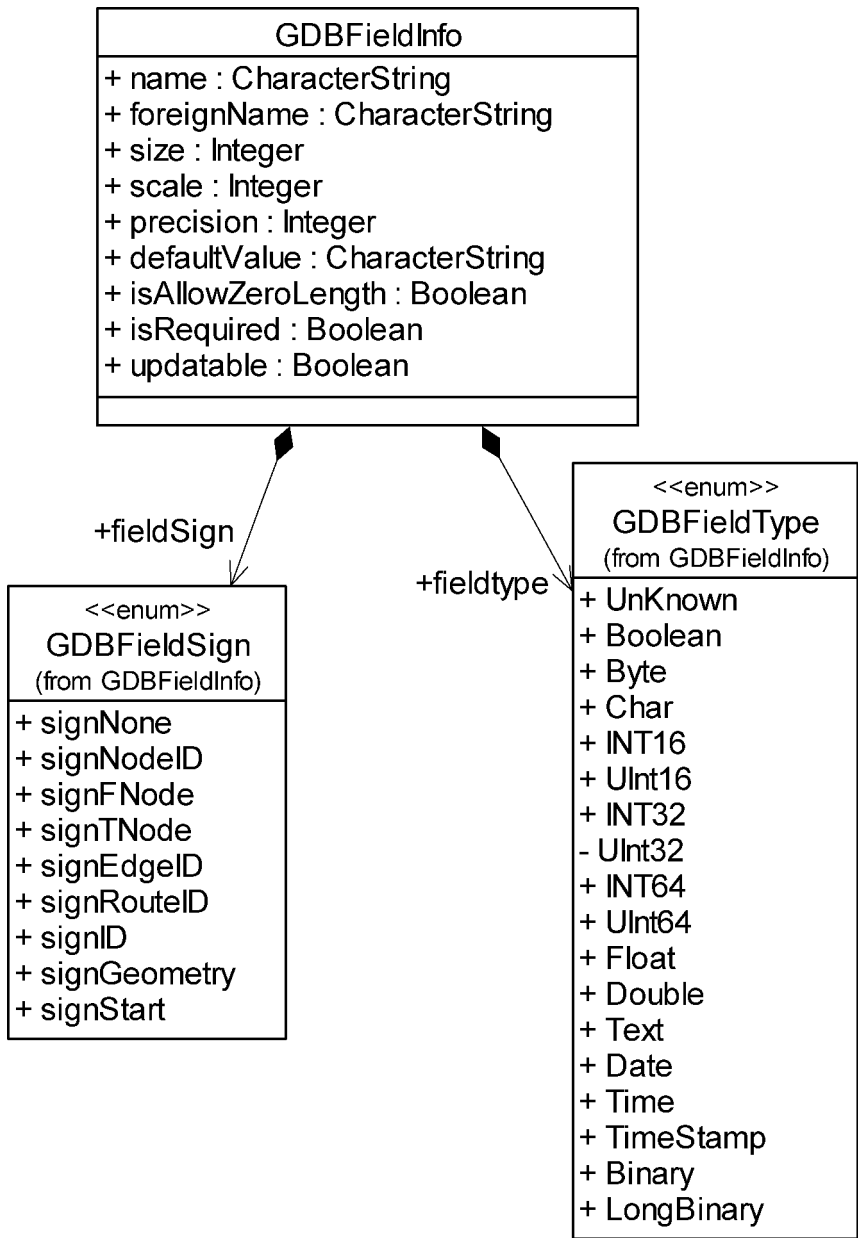


图 13 字段信息对象 UML 图

6.9.2 属性



字段信息对象的属性如下：

- name:CharacterString—字段名称；
- foreignName:CharacterString—字段的外键名；
- size:Integer—字段长度；
- scale:Integer—字段的刻度；
- precision:Integer—字段的精度；
- defaultValue:CharacterString—字段的默认值；
- isAllowZeroLength:Boolean—字段长度是否允许为零；
- isRequired:Boolean—是否为必填字段；

- updatable: Boolean—字段是否可编辑；
- fieldSign: GDBFieldSign—字段标记；
- fieldType: GDBFieldType—字段类型。

6.9.3 字段标记

字段标记(GDBFieldSign)用于标记字段在要素中的意义。其枚举类型见表 10。

表 10 字段标记列表

字段标记	描 述
signNone	普通字段
signNodeID	结点 ID 字段,主要用于标记网络数据集的结点字段
signFNode	弧段起始结点 ID 字段,主要用于标记网络数据集弧段的起始结点的 ID 字段
signTNode	弧段终止结点 ID 字段,主要用于标记网络数据集弧段的终止结点的 ID 字段
signEdgeID	弧段 ID 字段,主要用于标记网络数据集弧段的 ID 字段
signRouteID	路由 ID 字段,主要用于标记路由数据集的 ID 字段
signID	要素的 ID 字段
signGeometry	要素的 Geometry 字段

6.9.4 字段类型



字段类型(GDBFieldType)枚举类型及其描述见表 11。

表 11 字段类型描述列表

类型	长度(字节)	取值范围	描述
Boolean	1	[True False]	布尔型
Byte	1	[0,256]	单字节
Char	n	—	定长字符串
Int16	2	[−32 768,32 767]	短整型
UInt16	2	[0,65 535]	无符号短整型
Int32	4	[−2 147 483 648,2 147 483 647]	整型
UInt32	4	[0,4 294 967 295]	无符号整型
Int64	8	[−2 ⁶³ , (2 ⁶³ −1)]	长整型
UInt64	8	[0, (2 ⁶⁴ −1)]	无符号长整型
Float	4	[−3.4 × 10 ^{−38} , 3.4 × 10 ³⁸]	单精度浮点型
Double	8	[−1.7 × 10 ^{−308} , 1.7 × 10 ³⁰⁸]	双精度浮点型
Text	n	—	变长字符串
Date	4	—	日期型
Time	4	—	时间型
TimeStamp	8	—	日期时间型
Binary	n	—	固定长度二进制型,需指定长度
LongBinary	—	—	不定长二进制类型

6.10 字段信息集合

字段信息集合(GDBFieldInfos),多个字段信息的集合数组,可以获取字段的个数,也可以通过字段名称或序号来获取指定的字段信息。矢量数据集(GDBVectorDataset)获取的字段信息集合为数据集中的所有字段列表,即表结构。

6.11 栅格数据集

6.11.1 对象描述

栅格数据集用一个规则格网来描述与每一个格网单元位置相对应的空间现象特征的位置和取值,每一个格网单元为一个像素。栅格数据集常用来存储卫星影像、数字高程模型(DEM)、格网数据(GRID)。栅格数据集对象的 UML 图见图 14。

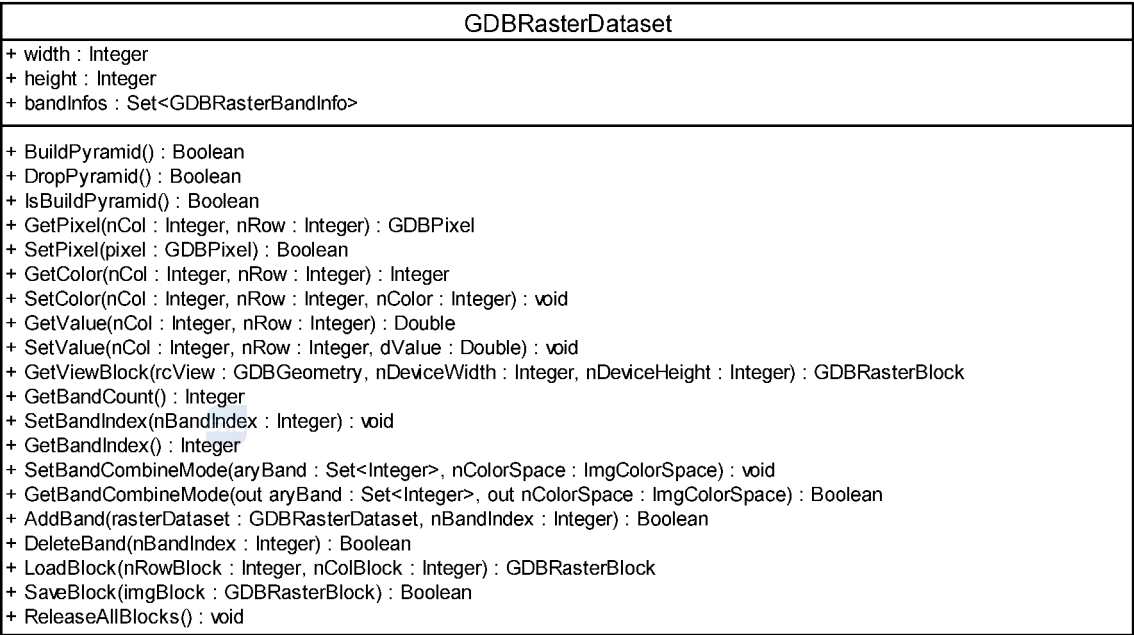


图 14 栅格数据集对象 UML 图

6.11.2 属性

栅格数据集对象的属性如下:

- width:Integer—宽度(像素数);
- height:Integer—高度(像素数);
- bandInfos:Set<GDBRasterBandInfo>—各波段的波段信息集合。

6.11.3 方法

栅格数据集对象的方法如下:

- BuildPyramid():Boolean—建立影像金字塔。
- DropPyramid():Boolean—移除影像金字塔。
- IsBuildPyramid():Boolean—数据集是否已经建立了金字塔。
- GetPixel(nCol:Integer,nRow:Integer):GDBPixel—获取指定位置的像素。

- `SetPixel(pixel:GDBPixel):Boolean`—设置像素,根据像素对象的位置更新栅格数据。
- `GetColor(nCol:Integer,nRow:Integer):Integer`—获取像素的颜色值。
- `SetColor(nCol:Integer,nRow:Integer,nColor:Integer):void`—设置获取像素的颜色值,针对 Image 数据集。
- `GetValue(nCol:Integer,nRow:Integer):Double`—获取像素的值。
- `SetValue(nCol:Integer,nRow:Integer,dValue:Double):void`—设置像素值,针对 GRID 数据集。
- `GetViewBlock(rcView:GDBGeometry,nDeviceWidth:Integer,nDeviceHeight:Integer):GDBRasterBlock`—获取指定视口范围的影像块,主要用于数据显示。
- `GetBandCount():Integer`—获取波段数。
- `SetBandIndex(nBandIndex:Integer):void`—设置当前操作波段的索引号。如果设置了当前波段索引号,则对栅格数据集的操作作用于当前波段;如果不设置当前波段索引号,则是作用于所有波段;GetPixel、GetColor、GetValue 方法应先设置波段索引号。
- `GetBandIndex():Integer`—获取当前波段索引号。返回-1表示未设置当前的波段索引号。
- `SetBandCombineMode(aryBand:Set<Integer>,nColorSpace:ImgColorSpace):void`—设置合成显示的波段和颜色模式。
- `GetBandCombineMode(out aryBand:Set<Integer>,out nColorSpace:ImgColorSpace):Boolean`—获取合成显示的波段和颜色合成模式。
- `AddBand(rasterDataset:GDBRasterDataset,nBandIndex:Integer):Boolean`—将栅格数据集 rasterDataset 的第 nBandIndex 个波段追加到当前数据集中。
- `DeleteBand(nBandIndex:Integer):Boolean`—删除第 nBandIndex 个波段。
- `LoadBlock(nRowBlock:Integer,nColBlock:Integer):GDBRasterBlock`—获取数据块。
- `SaveBlock(imgBlock:GDBRasterBlock):Boolean`—保存数据块。
- `ReleaseAllBlocks():void`—释放所有数据块。

6.12 波段信息

6.12.1 对象描述

波段信息用于描述多波段中每个波段的基本信息,GDBRasterBandInfo 对象描述 UML 图见图 15。

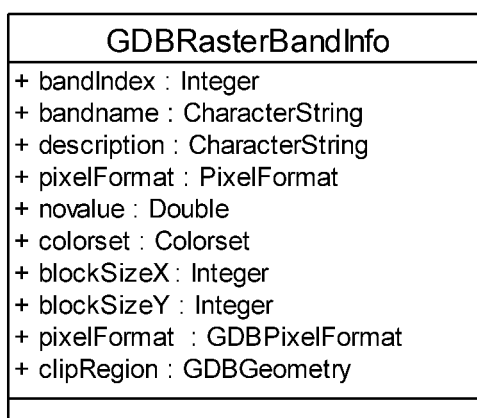


图 15 波段信息对象 UML 图

6.12.2 属性

波段信息对象的属性如下：

- bandIndex:Integer—波段索引号,从 0 开始；
- bandname:CharacterString—波段名称；
- description:CharacterString—波段描述信息；
- pixelFormat:PixelFormat—波段像素格式；
- novalue:Double—无值；
- colorset:Colorset—波段颜色表；
- blockSizeX:Integer—存储块的大小,X 方向(像素数)；
- blockSizeY:Integer—存储块块的大小,Y 方向(像素数)；
- pixelFormat:GDGPixelFormat—像素格式；
- clipRegion:GDGeometry—裁剪多边形,如果设置了裁剪多边形,则仅多边形内的像素是有效的,多边形范围外的像素均视为无值。

6.13 像素格式

像素格式用于描述栅格数据的像素存储方式和含义,用枚举类型 GDGPixelFormat 对象描述 UML 图见图 16。每种像素格式类型的含义及适用情况见表 12。

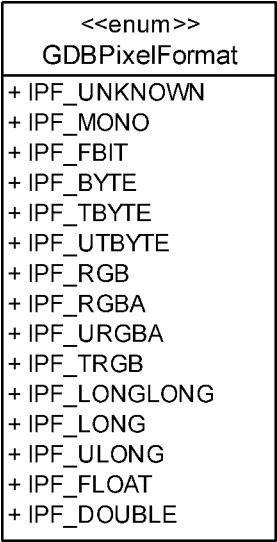


图 16 像素格式枚举对象 UML 图

表 12 像素格式描述

像素格式	描述	适用于 Grid 数据集	适用于 Image 数据集
IPF_UNKNOWN	未知像素格式	—	—
IPF_MONO	1 位	Y	Y
IPF_FBIT	4 位	Y	Y
IPF_BYTE	8 位	Y	Y

表 12 (续)

像素格式	描述	适用于 Grid 数据集	适用于 Image 数据集
IPF_TBYTE	16 位,有符号整数	Y	Y
IPF_UTBYTE	16 位,无符号整数	Y	Y
IPF_RGB	24 位,无符号整数	N	Y
IPF_RGBA	32 位,无符号整数	N	Y
IPF_TRGB	48 位,无符号整数	N	Y
IPF_LONGLONG	64 位,长整型	Y	N
IPF_LONG	32 位,有符号整数	Y	N
IPF_ULONG	32 位,无符号整数	Y	N
IPF_FLOAT	32 位,单精度浮点数	Y	N
IPF_DOUBLE	64 位,双精度浮点数	Y	N

6.14 像素

6.14.1 对象描述

Pixel 即像素对象。Pixel 是基于场模型数据组织的基本元素,不可再拆分。每一个 Pixel 有一个唯一值,在不同应用中具有不同的涵义。Scanline 和 Block 都是由 Pixel 聚合而成。栅格数据集的一个波段只能有一种像素格式,像素格式决定了像素存储空间的大小。像素对象的 UML 图见图 17。

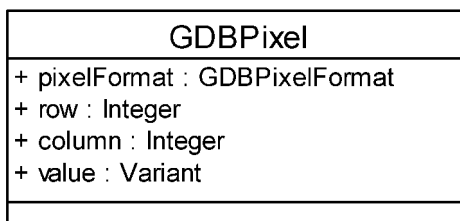


图 17 像素对象 UML 图

6.14.2 属性

像素对象的属性如下:

- pixelFormat:GDBPixelFormat—像素格式;
- row:Integer—所处行位置;
- column:Integer—所处列位置;
- value:Variant—像素值,根据像素点的像素格式不同,有不同的取值,参见表 12。

6.15 扫描线



6.15.1 对象描述

扫描线由像素聚合而成,这些像素是栅格数据集上横向连续像素的集合。它以一维方式访问栅格数据,扫描线的高即为像素高度,最大宽度为栅格数据集的一行所包含的像素数。扫描线对象的 UML

图见图 18。

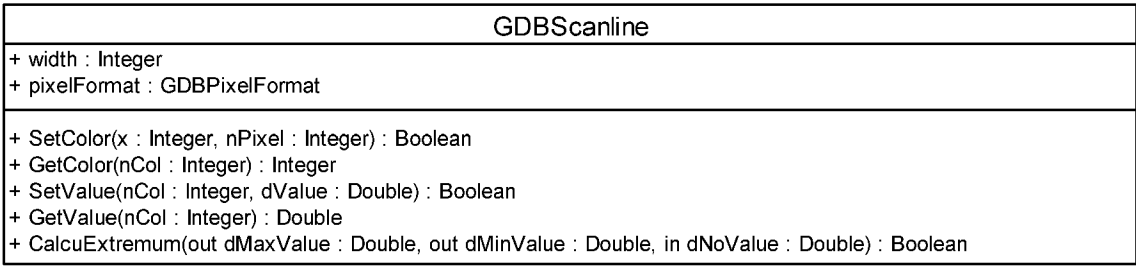


图 18 扫描线对象 UML 图

6.15.2 属性

扫描线对象的属性如下：

- width:Integer—扫描线宽度(像素数)；
- pixelFormat:GDBPixelFormat—像素格式。

6.15.3 方法

扫描线对象的方法如下：

- SetColor(x:Integer,nPixel:Integer):Boolean—设置像素颜色；
- GetColor (x:Integer):Integer—获取像素颜色；
- SetValue(x:Integer,dValue:Double):Boolean—设置像素值；
- GetValue(x:Integer):Double—获取像素值；
- CalcuExtremum(out dMaxValue:Double,out dMinValue:Double,dNoValue:Double):Boolean—计算扫描线的最大和最小值。

6.16 数据块

6.16.1 对象描述

数据块由像素聚合而成,具有行索引、列索引、宽度、高度、有效矩形、地理范围、极值等属性。它是
以二维方式访问栅格数据。数据块对象的 UML 图见图 19。

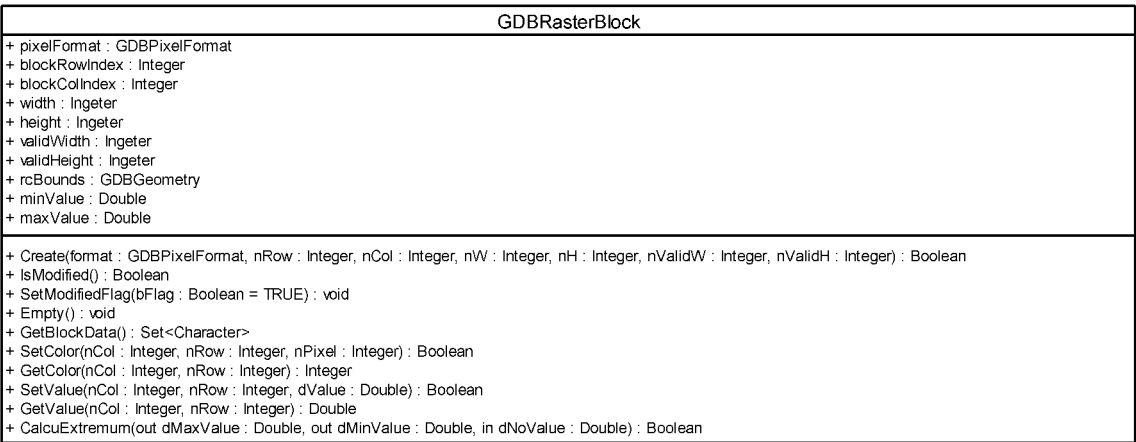


图 19 数据块对象 UML 图

6.16.2 属性

数据块对象的属性如下：

- pixelFormat:GDBPixelFormat—像素格式；
- blockRowIndex:Integer—数据块的行号；
- blockColIndex:Integer—数据块的列号；
- width:Integer—宽度(像素数)；
- height:Integer—高度(像素数)；
- validWidth:Integer—有效的宽度(像素数)；
- validHeight:Integer—有效的高度(像素数)；
- rcBounds:GDBGeometry—范围；
- minValue:Double—最小值；
- maxValue:Double—最大值。

6.16.3 方法

数据块对象的方法如下：

- Create(format:GDBPixelFormat, nRow: Integer, nCol: Integer, nW: Integer, nH: Integer, nValidW: Integer, nValidH: Integer): Boolean—创建 Block；
- IsModified(): Boolean—判断数据块是否被修改；
- SetModifiedFlag(bFlag: Boolean=TRUE): void—设置数据块修改标记；
- GetBlockData(): Set<Character>—获取 Block 的数据内容；
- SetColor(x: Integer, y: Integer, nPixel: Integer): Boolean—设置像素颜色值；
- GetColor(x: Integer, y: Integer): Integer—获取像素颜色值；
- SetValue(x: Integer, y: Integer, dValue: Double): Boolean—设置像素值；
- GetValue(x: Integer, y: Integer): Double—获取像素值；
- CalcuExtremum(out dMaxValue: Double, out dMinValue: Double, in dNoValue: Double): Boolean—计算数据块的极值。

7 元数据访问对象

7.1 元数据对象逻辑关系

在 GB/T 19710—2005 中元数据从描述的内容上自上而下描述为：元数据、元数据包、元数据类和元数据元素。参照该结构，本标准的元数据库对象管理体系包括三个层次：元数据库(GDBMetaDatabase)、元数据集(GDBMetadata)和元数据子项(GDBMetaItem)。见图 20。其中，元数据集(GDBMetadata)与元数据对应；为满足元数据包、元数据类和元数据元素的存储访问要求，由自身可嵌套的元数据子项(GDBMetaItem)提供接口对元数据包、类和元素的操作，参见 7.2.4。

元数据库访问对象的具体关系描述如下：

- 元数据库。专门存储元数据的数据库，元数据库可以独立于数据源所在的数据库存储在数据源之外，也可以与数据源数据存储在数据库。一个元数据库中存储多个数据源相关的元数据。

- 元数据集。由多个元数据子项集合而成。一个元数据集存储与一个数据集相关的元数据，或是与一个数据源相关的元数据。
- 元数据子项。同类元数据信息的集合，元数据子项可以嵌套，即一个元数据子项可以再包含一个或多个元数据子项。一个 0 层嵌套的元数据子项可描述元数据元素。

元数据集提供对元数据项的查找和遍历接口。元数据库支持查询功能，可以通过标题、值、数据集范围及对象名称来进行查询，查询结果为元数据集集合。

GDB-CLI 的空间数据库访问对象中，数据源和数据集（包括矢量数据集和栅格数据集）都可以有自己的元数据，这些对象通过唯一标识与元数据库中的元数据关联。参见 6.1、6.3 和 7.2.3。

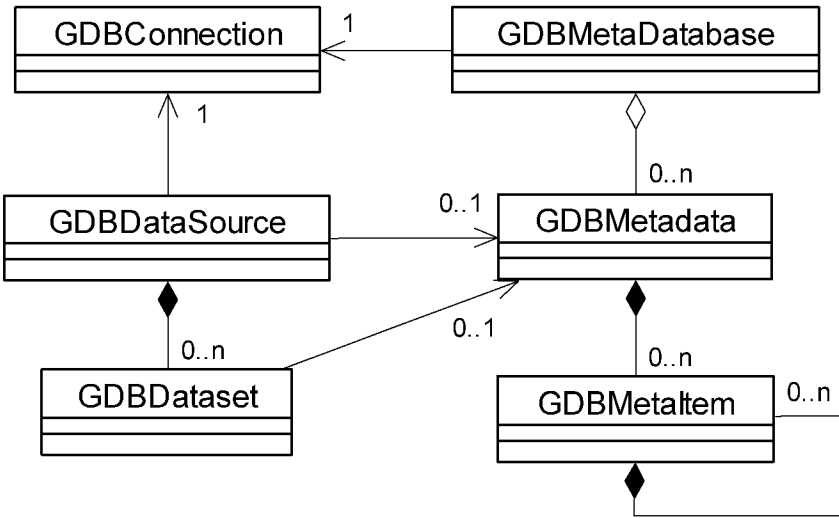


图 20 元数据对象 UML 图

7.2 元数据访问对象定义

7.2.1 元数据库

7.2.1.1 对象描述

元数据库是用来集中存储元数据的数据库。元数据库中以元数据集来存储每个空间数据库对象的元数据。

一个元数据库中存储多个数据源的元数据，可以在元数据库中进行全文查询来查找相关的元数据，返回结果为元数据集的 UUID 集合 Set<CharacterString>。

元数据库对象属性和方法见图 21。

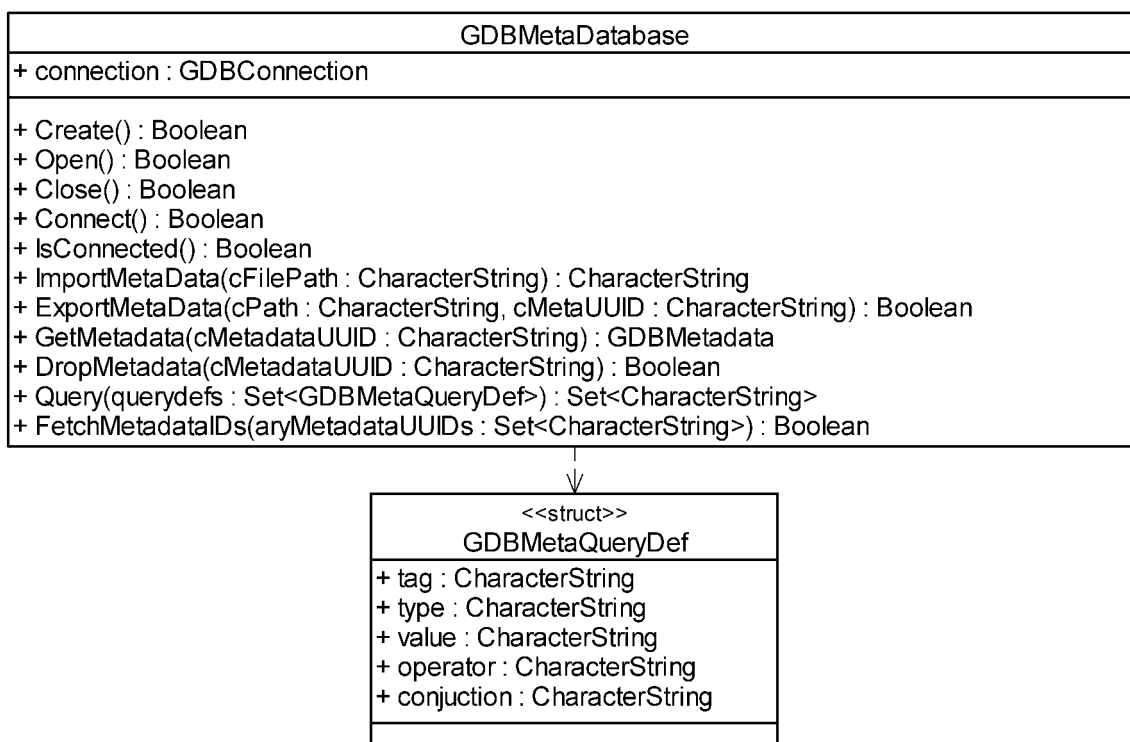


图 21 元数据库对象属性和方法

7.2.1.2 属性

元数据库对象的属性如下：

- connection:GDBConnection—元数据库的连接信息,通过此信息打开元数据库。

7.2.1.3 方法

元数据库对象的方法如下：

- Create():Boolean—创建元数据库。
- Open():Boolean—打开元数据库。
- Close():Boolean—关闭元数据库。
- Connect():Boolean—连接元数据库。主要用于数据库连接断开后重新连接。
- IsConnected():Boolean—元数据库是否已连接。
- ImportMetadata(cFilePath:CharacterString):CharacterString—导入指定路径的数据为元数据。导入成功返回元数据的 UUID,否则返回空字符串。
- ExportMetadata(cPath:CharacterString,cMetaUUID:CharacterString):Boolean—导出元数据。将 UUID 为 cMetaUUID 的元数据导出到 cPath 文件,cPath 是全路径名称。
- GetMetadata(cMetadataUUID:CharacterString):GDBMetadata—获取指定 UUID 的元数据集对象。
- DropMetadata(cMetadataUUID:CharacterString):Boolean—移除指定 UUID 的元数据集。
- Query(querydefs:Set<GDBMetaQueryDef>):Set<CharacterString>—查询元数据。返回满足条件的元数据集的 UUID 集合。

- GetMetadata(OgdcInt nMetadataID):GDBMetadata—获取元数据对象。
- FetchMetadataIDs(aryMetadataUUIDs:Set<CharacterString>):Boolean—获取元数据库中所有的元数据的 UUID。

7.2.2 元数据库查询对象

元数据库查询对象(GDBMetaQueryDef)用于构造元数据的查询条件,1 个或多个元数据库查询对象构成查询条件,对元数据库查询后返回符合条件的元数据集的 UUID 集合。见图 21 ,其属性描述如下:

- tag:CharacterString—要查询的标签名称;
- type:CharacterString—标签在数据库中存储的类型,如整型、浮点型、字符串型等;
- value:CharacterString—标签值;
- operator:CharacterString—标签名称和标签值之间的比较操作符号,如大于、小于、等于等;
- conjunction:CharacterString—元数据库查询对象与下一个查询对象之间的连接关系,如 and、or 等。

7.2.3 元数据集

7.2.3.1 对象描述

元数据集是元数据库组织的基本单元,一个元数据集由多个元数据项集合而成。一个元数据集存储一个数据集的相关元数据或一个数据源的元数据。同一元数据库中的同一个数据集只能存储一份元数据集。元数据集对象的 UML 图见图 22。

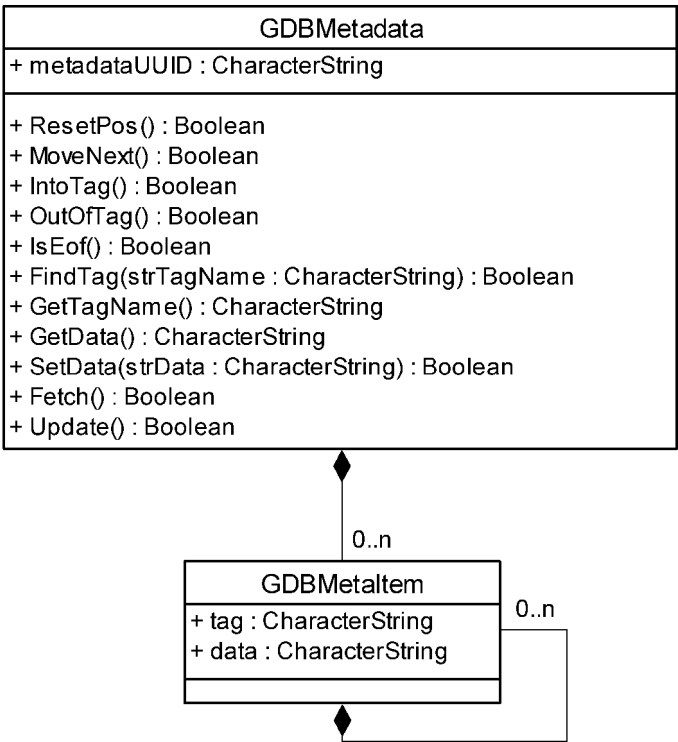


图 22 元数据集对象属性和方法

7.2.3.2 属性

元数据集对象的属性如下：

- metadataUUID:CharacterString—元数据集的唯一标识。

7.2.3.3 方法

元数据集对象的方法如下：

- ResetPos():Boolean—回到当前级别标签的顶层；
- MoveNext():Boolean—在当前层级中下移到下一个标签；
- IntoTag():Boolean—进入下一级标签；
- OutOfTag():Boolean—返回上一级标签；
- IsEOF():Boolean—是否已经到达最后一个标签；
- FindTag(cTagName:CharacterString):Boolean—按标签名查找标签项,并定位到该标签；
- GetTagName():CharacterString—获取标签名；
- GetData():CharacterString—获取标签值；
- SetData(cData:CharacterString):Boolean—设置标签值；
- Fetch():Boolean—从数据库中读取该元数据集的详细内容；
- Update():Boolean—将当前元数据中的内容更新到数据库中。

7.2.4 元数据子项

元数据子项是元数据中同类描述信息的集合或是元数据的一个条目,由元数据标题和元数据值构成。元数据子项可嵌套,对元数据包、元数据类和元数据元素的操作提供统一接口。元数据项对象的UML图见图22,具体描述如下：

- tag:CharacterString—元数据项的标题,标识此元数据子项包含的标题内容；
- data:CharacterString—元数据项的值。值本身可以是一个元数据子项格式化后的内容,以自嵌套形式描述元数据包或元数据类；也可以是没有嵌套、不可拆分的元数据元素的值内容。

8 空间参照系

8.1 空间参照系对象逻辑关系

空间参照系用于建立地图要素与实际地物之间的关系,是地物描述的基础。空间参照系通常有地理参照系和投影参照系两种,GDB-CLI的空间参照系对象参考ISO 19111:2007的内容,在此基础上遵循简洁通用的原则对空间参照系的访问对象进行描述,见图23。

地理参照系由大地水平参照系、椭球体及本初子午线组成；投影坐标系由地理参照系和投影变换参数组成,据此确定球面坐标向平面坐标转换的方式。

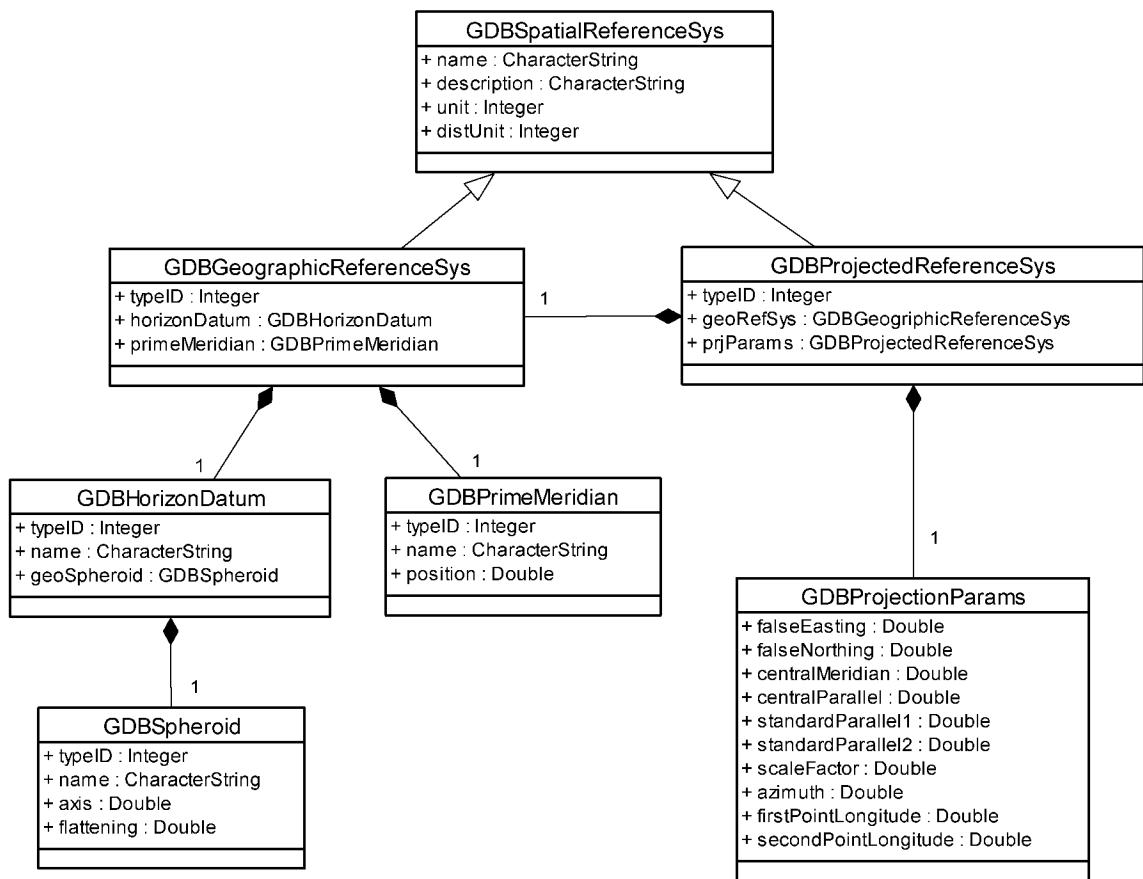


图 23 空间参考系对象 UML 图

8.2 空间参考系对象定义

8.2.1 空间参照系

空间参照系对象(GDBCoordReferenceSys)的属性描述如下：

- name:CharacterString—参照系名称；
- description:CharacterString—参照系描述信息；
- unit:Integer—参照系单位,可以是长度单位或者角度单位；
- distUnit:Integer—参照系距离单位,只能为长度单位。

8.2.2 大地参照系

大地参照系对象(GDBGeographicReferenceSys),是以经纬度来量算的球面参照系统,该坐标系统用来确定地球表面上点的位置。其属性描述如下：

- typeID:Integer—地理参照系类型；
- horizonDatum:GDBHorizonDatum—大地水平参照系；
- primeMeridian:GDBPrimeMeridian—本初子午线。

8.2.3 投影坐标系

投影参照系对象(GDBProjectedReferenceSys),是地理参照系下的地理数据经过投影变换后在二维平面上的表现。

- typeID: Integer—投影坐标系类型;
- geoRefSys: GDBGeographicReferenceSys—地理坐标系;
- prjParams: GDBProjectedReferenceSys—投影参数。

8.2.4 水平基准面

水平基准面对象(GDBHorizonDatum),是定位地表要素的参照物,它由椭球体与地球的位置关系来定义。其属性信息描述如下:

- typeID: Integer—水平基准面类型;
- name: CharacterString—水平基准面名称;
- geoSpheroid: GDBSpheroid—参考椭球体。

8.2.5 椭球体

椭球体对象(GDBSpheroid),是模拟地球形状的规则几何体。其属性信息描述如下:

- typeID: Integer—椭球体类型;
- name: CharacterString—椭球体名称;
- axis: Double—椭球体长半轴(赤道半径);
- flattening: Double—椭球体扁率。

8.2.6 本初子午线

本初子午线对象(GDBPrimeMeridian),是地理参照系统中经度坐标的起算点。

- typeID: Integer—本初子午线类型;
- name: CharacterString—本初子午线名称;
- position: Double—经线起算点位置。

8.2.7 投影参数

投影参数对象(GDBProjectionParams),定义了投影计算必备的一些参数,包括水平和垂直偏移量,中央经线,原点纬线,第一、第二标准纬线,第一、第二点经度以及比例因子和方位角。其属性信息描述如下:

- falseEasting: Double—False easting (X0),水平偏移量;
- falseNorthing: Double—False northing (Y0),垂直偏移量;
- centralMeridian: Double—Central meridian, Longitude of origin (LON0),中央经线;
- centralParallel: Double—Central parallel, Latitude of origin (LAT0),原点纬线;
- standardParallel1: Double—Standard parallel 1, Latitude of 1st point (LAT1),第一标准纬线;
- standardParallel2: Double—Standard parallel 2, Latitude of 2nd point (LAT2),第二标准纬线;
- scaleFactor: Double—Scale factor,比例因子;
- azimuth: Double—Azimuth,方位角;
- firstPointLongitude: Double—Longitude of 1st point (LON1),第一点经度;
- secondPointLongitude: Double—Longitude of 2nd point (LON2),第二点经度。

9 数据提供者加载与卸载

9.1 概述

数据提供者管理器(GDBProviderManager)是数据提供者(GDBProvider)在对外提供空间数据访

问能力前的注册点,它负责加载和卸载相关的数据提供者,并根据相关信息和要求选择对应的数据提供者,以满足外部的数据访问请求。同时,它还提供统一的开放接口,以辅助并简化对数据提供者访问方式的配置和管理。

数据提供者(GDBProvider)的加载与卸载由数据提供者管理器(GDBProviderManager)管理。数据提供者管理器也是 GDB-CLI 应用的总的入口,它加载数据提供者有两种:指定全路径的单个数据提供者和应用程序当前路径下的所有数据提供者,加载的过程中可以对数据提供者的信息进行检查,成功加载的数据提供者通过 CreateGDBDataSource()方法创建出对应数据提供者的数据源(GDBDataSource)实例对象。

数据提供者管理器、数据提供者和数据源的 UML 关系图见图 24。

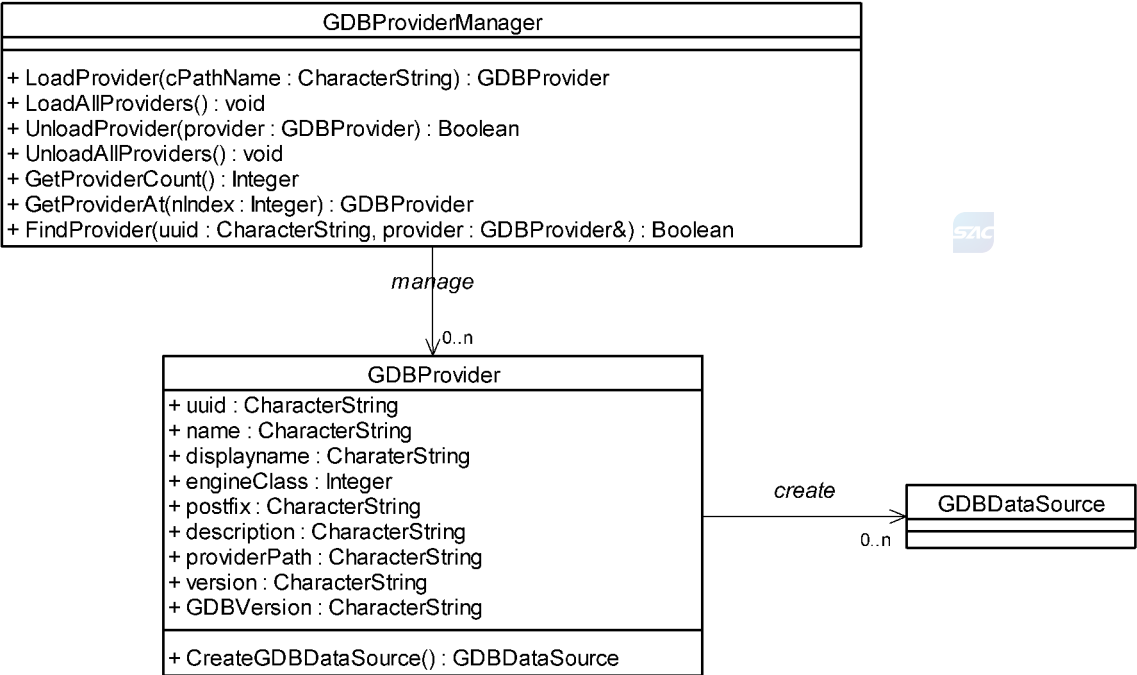


图 24 数据提供者管理器类图

9.2 数据提供者管理器

数据提供者管理器对象的方法如下:

- LoadProvider(cPathName:CharacterString):GDBProvider—根据指定的全路径名称 cPathName 加载数据提供者动态库,加载成功则返回可用的 GDBProvider,否则返回的是空对象;
- LoadAllProviders():void—加载当前应用程序所在路径下的所有的数据提供者动态库,加载完成后可以通过 GetProviderCount()方法获取成功加载的动态库个数;通过 GetProviderAt(nIndex:Integer)方法获取数据提供者对象;
- UnloadProvider(provider:GDBProvider):Boolean—卸载指定的 provider,卸载后该 provider 将不可用;
- UnloadAllProviders():void—卸载所有已成功加载的数据提供者;
- GetProviderCount():Integer—获取当前已成功加载的数据提供者个数;
- GetProviderAt(nIndex:Integer):GDBProvider—获取第 nIndex 个数据提供者,nIndex 从 0 开始,小于 GetProviderCount()方法获取的整数值;
- FindProvider(uuid:CharacterString):GDBProvider—通过唯一标识 uuid 获取数据提供者。每

个数据提供者都应有一个 uuid 值,以标志该提供者。

9.3 数据提供者

9.3.1 数据提供者的应用形式

以 GDB-CLI 的 C++ 实现为例,Provider 以标准动态库的形式提供。动态库以.odx 为扩展名,例如:CompanyAProviderX.odx。

9.3.2 数据提供者的加载和卸载

GDB-CLI 包含两个加载数据提供者的接口,用于加载指定全路径的数据提供者或加载应用程序当前路径下的所有的以合法扩展名作为后缀的数据提供者。

GDB-CLI 包含两个卸载数据提供者的接口,用于卸载指定路径的数据提供者或卸载所有已加载的数据提供者。

GDBProvider 的加载和卸载通过 GDBProviderManager 对象管理,参见 9.2。

9.3.3 属性

数据提供者对象的属性如下:



- uuid:CharacterString—数据提供者的唯一标识,在创建数据源和打开数据源时需要指定此参数以找到匹配的数据提供者。
- name:CharacterString—数据提供者名称标识,例如:"CompanyA/ProductX/Raster.Provider.1.0.11.3"。
- displayName:CharacterString—便于识别的、友好的数据提供者名称。例如:"CompanyA's OGDC Provider for DatabaseX"。对于这个名称所用的语言,可以由用户的系统来确定。
- engineClass:Integer—数据提供者分类,主要依据数据提供者针对的数据来源:engineClass = 1 为网络数据库,engineClass = 2 本地文件型,engineClass = 3 为网络服务型。
- postfix:CharacterString—主要针对 engineClass = 2 的本地文件型数据提供者,给出该数据提供者所能访问的数据的文件后缀名。
- description:CharacterString—对数据提供者的简单扼要的描述,如说明其支持的功能及数据格式,例如"由 CompanyA 提供的基于 DatabaseX 数据源的驱动程序,具有读和写权限"。
- providerPath:CharacterString—数据提供者动态库的全路径名,包含动态库文件的全名。
- version:CharacterString—数据提供者动态库的版本信息,版本号格式为:<VersionMajor>.<VersionMinor>.<BuildMajor>.<BuildMinor>,例如,1.0.11.3。
- GDBVersion:CharacterString—GDB-CLI 的版本,说明数据对象对应的 GDB-CLI 版本,版本号格式为:<VersionMajor>.<VersionMinor>.<BuildMajor>.<BuildMinor>,例如,1.0.1.0。

9.3.4 方法

数据提供者对象的方法如下:

- CreateGDBDataSource():GDBDataSource—创建数据源对象(GDBDataSource),创建成功则返回可用的 GDBDataSource,否则返回空对象。

参 考 文 献

- [1] GB/T 23707—2009 地理信息 空间模式(Geographic information—Spatial schema)
 - [2] ISO 19101 地理信息 地理信息参考模型(Geographic information—Reference model)
-



中 华 人 民 共 和 国
国 家 标 准
地理空间数据库访问接口

GB/T 30320—2013

*

中国标准出版社出版发行
北京市朝阳区和平里西街甲2号(100029)
北京市西城区三里河北街16号(100045)

网址: www.gb168.cn

服务热线: 400-168-0010

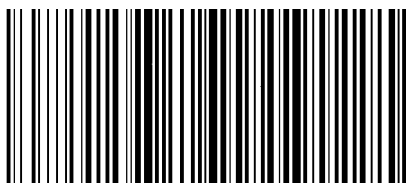
010-68522006

2014年6月第一版

*

书号: 155066 • 1-49318

版权专有 侵权必究



GB/T 30320-2013