Capítulo III

OpenGIS

(OpenGIS® Simple Feature Specification For SQL Revisión 1.1)

Liberado el 5 de Mayo de 1999

En este capítulo se describe la especificación de SQL92 para OpenGIS.

3. Definición de OpenGIS

OpenGIS es definido como un acceso transparente a Geodatos hetereogeneos y al geoprocesamiento de recursos en un ambiente de redes[14].

La especificación es OepnGIS es hecha por el "Open GIS Consortium (OGC)"[14]

Las compañías que participan para la creación de dicha especificación son:

- Environmental Systems Research Intitute, Inc. (ESRI)
- IBM Corporation.
- Informix Software, Inc.
- MapInfo Corporartion.

Hay que mencionar que la especificación OpenGIS se encuentra bajo revisión y esta sujeta a cambios.

3.1 Objetivo de OpenGIS

El propósito de esta especificación es definir un estándard para un esquema en SQL que soporte el almacenamiento, recuperación, interrogación y actualización de una colección geoespacial de características simples. Una característica simple es definida por la especificación abstracta de OpenGIS para tener atributos espaciales y no espaciales. Los atributos espaciales son los valores de las geometrías y las características simples están basadas en una geometría en 2 dimensiones (X,Y) con interpolación Lineaal entre vértices.

Una colección simple de características geoespaciales esta conceptualmente almacenada en tablas cuyas columnas representan a las geometrías y los renglones las características, tabla 3.1. Esta especificación esta pensada para usar una base de datos relacional (RDBMS).

GID	ETYPE	X	Y
100	1	12.5	34.56

GID	ETYPE	X	Y
101	1	45.67	898.9

Tabla 3.1 Geometrías y características

En la tabla 3.1 se muestra el campo GID, el cual representa la llave foránea de la tabla de carcterísticas, el campo ETYPE se refiere el tipo de elemento, como es el caso del 1, el cual representa a la geometría del punto.La X y Y representan las coordenadas del punto.

3.2 Modelo de Objetos Geométricos

Este modelo esta pensado para ser usado en una plataforma de cómputo distribuida y emplea la notación OMT. La clase base es “Geometría” la cual tiene como subclases ,”Punto”, “Curva”, “Superficie” y “Colección Geométrica”. Cada objeto geométrico esta asociado con un Sistema de Referencia Espacial (Spatial Reference System), el cual describe las coordenadas del espacio en las cuales el objeto geométrico esta definido.

\

Figura 3.1 Jerarquía de los tipos de Geometría

3.3 Breve descripción de las clases

3.3.1 Clase Geometría

Es la clase raíz de la jerarquía y es una clase abstracta (No instanciable). Las subclases de Geometría en esta especificación están restringidas a 0,1 y a dos dimensiones. Todas las clases que se definen en esta especificación son definidas como instancias válidas de una geometría si son topológicamente cerradas.

3.3.1.1 Métodos de la clase Geometría

Esta clase cuenta con tres tipos de métodos:

1. Métodos básicos: (Dimension( ), GeometríaType( ), SRID( ), Envelope( ), AsText( ), AsBinary( ), IsEmpty( ), IsSimple( ), Boundary( ))
2. Métodos para probar las relaciones espaciales entre objetos geométricos: (Equals( ), Disjoint( ), Intersects( ), Touches( ), Crosses( ), Within( ), Contains( ), Overlaps( ), Relate( ).
3. Métodos para el análisis espacial: Distance( ), Buffer( ), ConvexHull( ), Intersection( ), Union( ), Difference( ), SymDifference( ).

Para obtener mayor información sobre estos métodos y su razón de ser referirse a las secciones 3.12.3.2 y 3.12.2 de [15]

3.3.2 Clase ColecciónGeométrica

Esta clase es una geometría que es una colección de una o más geometrías. Las geometrías que forman la colección restringen a la clase Colección Geométrica. Así como también lo hace la sobreposición de los elementos.

3.3.2.1 Métodos de la clase ColecciónGeométrica

1. NumGeometries( ): Regresa un entero el cual es el número de geometrías en este Colección Geométrica.

2. GeometríaN(int N): Regresa la N-ésima geometría de la Colección Geométrica.

3.3.2 Clase Punto

Un Punto es una clase de 0 dimensiones y representa un simple punto en el espacio. Un punto consta de una coordenada en el eje X y otra en el eje Y. Del tipo Doble. El punto se encuentra limitado por un conjunto vacío.

3.3.2.1 Métodos de la clase Punto

1. X( ): Coordenada X del Punto de tipo Doble.
2. Y( ): Coordenada Y del Punto de tipo Doble.

3.3.3 Clase MultiPunto

La clase MultiPunto es una geometría de 0 dimensiones. Los elementos de la clase MultiPunto estan restringidos por la clase Punto. Los puntos no están conectados ni tampoco tienen un orden. Se dice que el MultiPunto es simple si no existen dos puntos con coordenadas idénticas. El límite de un MultiPunto es un conjunto vacío.

3.3.4 Clase Curva

La clase Curva, es una clase no instanciable, la cual representa un objeto geométrico de una dimensión que usualmente es almacenado como una secuencia de puntos. Este tipo de especificación solamente define un tipo de subclase de Curva, LíneaString el cual usa interpolación lineal entre puntos. La Curva tiene una imagen similar a la gráfica genereda por una función R^2, es simple si no tiene elementos que pasen por un mismo punto dos veces, es cerrada si el punto inicial es igual al final, es un anillo si es simple y cerrada. El límite de una curva no cerrada esta dada por dos puntos finales. Un objeto Curva esta definido como topológicamente cerrado.

3.3.4.1 Métodos de la clase Curva

1. Length ( ): Regresa un Doble el cual representa la longitud del Curva asociado a la referencia espacial (SRID).
2. StartPunto ( ): Regresa un Punto el cual es el punto inicial de la Curva
3. EndPunto ( ): Regresa un Punto el cual es el Punto final de la Curva
4. IsClosed ( ): Regresa un entero el cual es 1 (TRUE) si (StartPunto( )=EndPunto( ) de otra forma es falso (FLASE)
5. IsRing ( ): Regresa un entero el cual es 1 (TRUE) si esta Curva es cerrada (IsClosed()) y si es simple, es decir que no pasa por un punto dos veces.

Para mayor referencia consultar la sección 3.12.7.3 y 3.12.3.2 de [15].

Figura 3.2 (1) un simple LíneaString, (2) un LíneaString que no es simple, (3) un LíneaString cerrado y simple, (3) un LíneaString cerrado, pero no simple.

3.3.5 Clases LíneaString, Línea y LínearRing

Una LíneaString es un Curva con interpolación lineal entre puntos. Cada par de puntos consecutivos define un segmento de línea.

Una Linea es una LineaString con exactamente dos puntos.

Un LinearRing es una LineaSrting que es cerrada y simple.

De la figura 3.2 podemos destacar que el (3) es una LíneaString cerrado que también es un LinearRing, la Curva de la figura 3.2 (4) es cerrada y es una LíneaString que no es un LinearRing.

3.3.5.1 Métodos de LíneaString, Línea y LinearRing

1. NumPuntos( )
2. PuntoN ( )

3.3.5.2 MultiCurva

Una MultiCurva es un Colección Geométrica cuyos elementos son Curvas. En esta especificación la MultiCurva es una clase no instanciable, en ella se definen métodos para sus subclases y se incluye por razones de extensibilidad.

Una MultiCurva es simple si y solo si todos sus elementos son simples, la única intersección ocurre cuando dos puntos cualquiera coinciden en el mismo punto y son puntos terminales del segmento.

Para mayor referencia sobre la MultiCurva ver la sección 3.12.3.2 de [15].

3.3.5.3 Métodos de la MultiCurva

1. IsClosed( ): Regresa un entero, 1 (TRUE) si todos los elementos son cerrados.
2. Length ( ): Regresa la longitud del MultiCurva (la suma de longitudes de sus elementos), la cual es un Doble.

3.3.6 Clase MultiLíneaString

Una MultiLíneaString es una MultiCurva cuyos elementos son LíneaStrings

Figura 3.3 (1)MultiLineaString simple,MultiLineaString con dos elementos que no es sencillo, multiLineaString cerrado, con dos elementos que no es simple.

3.3.7 Clase Superficie

Una Superficie es una clase no instanciable que representa un objeto geométrico de dos dimensiones. Un Superficie simple es simple "patch" que esta asociado con uno o más vecindarios exteriores y cero o más vecindarios interiores. Superficies simples en tres dimensiones son isomórficos a las superficies planas. Las superficies de los Polihedros están formados por "stitching" , simples superficies juntas a lo largo de su vecindario.

El vecindario (boundary) de una superficie simple es el conjunto de curvas cerradas que corresponden a sus vecindarios interiores y exteriores.

La única subclase instanciable de esta especificación , es el Polígono el cual es una superficie plana.

3.3.7.1 Métodos de la clase Superficie

1. Area( ): Regresa un Doble el cual es el área de este Superficie
2. Centroid ( ): Regresa un Punto el cual es el centro matemático de esta superficie.
3. PuntoOnSuperficie( ): Regresa un Punto, el cual se garantiza que esta en la superficie.

3.3.8 Clase Polígono

Un Polígono es una superficie plana, definida por un vecindario exterior y cero o más vecindarios interiores. Cada vecindario interior define un hueco en el Polígono.

Las reglas para decir que un Polígono es válido son:

1. Los Polígonos son topológicamente cerrados
2. El vecindario de un Polígono consiste en conjunto de LinearRings que dividen el vecindario interior del exterior.
3. No se puden cruzar dos anillos dentro del vecindario del Polígono, el único caso en el que se pueden intersectar es cuando el punto es tangente.
4. Un Polígono puede no tener líneas cortdas, "spikes" o "punctures".
5. El interior del Polígono es un conjunto de puntos conectados
6. El exterior de un Polígono con uno o más huecos no esta conectado. Cada hueco define un componente conectado al exterior.

La combinación de los puntos 1 y 3 hacen que el Polígono sea regular y cerrado.

Los Polígono son geometrías simples

3.3.8.1 Métodos de Polígono

1. ExteriorRing( ): Regresa el anillo exterior del Polígono como LineaString.
2. NumInteriorRings( ): Regresa el número de anillos interiores del Polígono como entero
3. InteriorRingN(N: Integer): Regresa el anillo interior N

Figura 3.4 Ejemplos de Polígonos con 1, 2 y 3 anillos respectivamente.

Figura 3.4 Ejemplos de objetos que no pueden representarse como una sola instancia de Polígono. (1) y (4) pueden ser representados como dos Polígonos separados.

3.3.9 Clase MultiSuperficie ( )

Una MultiSuperficie es una colección geométrica de dos dimensiones cuyos elementos son superficies. El interior de dos superficies cualquiera dentro de una MultiSuperficie no deben intersectarse. El vecindario de dos elementos cualquiera dentro de un MultiPolígono pueden intersectarse.

La MultiSuperficie es una clase no instanciable en esta especificación. Esta define un conjunto de métodos para sus subclases y es incluida por razones de extensibilidad. La clase instanciable de la MultiSuperficie es el MultiPolígono que corresponde a una colección de Polígonos.

3.3.9.1 Métodos de MultiSuperficie

1. Area( ): Regresa el área de este MultiSuperficie, como Doble.

2. Centroid ( ): Regresa un punto el cual representa el centro matemático de este MultiSuperficie.
3. PuntoOnSuperficie( ): Regresa un Punto, el cual se asegura que esta dentro de este MultiSuperficie.

3.3.10 MultiPolígono

Un MultiPolígono es un MultiSuperficie cuyos elementos son Polígonos.

3.3.10.1 Reglas de un MultiPolígono

1. El interior de dos Polígono que son elementos de un MultiPolígono no se intersectan.
2. Los vecindarios de dos Polígono cualquiera que son elementos de un MultiPolígono no se pueden cruzar y se pueden tocar en solamente un número finito de puntos.
3. Un MultiPolígono es definido como topológicamente cerrado.
4. Un MultiPolígono no puede tener "cut Lineas", "spikes" o "punctures", un MultiPolígono es regular y cerrado.
5. El número de componentes conectados del interior del MultiPolígono es igual al número de Polígonos en el MultiPolígono.

El vecindario de un MultiPolígono es un conjunto cerrado de Curvas (LineaStrings) correspondientes a los vecindarios de sus elementos Polígonos. Cada Curva en el vecindario de un MultiPolígono esta exactamente en el vecindario de un Polígono del MultiPolígono. Cada Curva en el vecindario de un elemento Polígono esta en el vecindario de un MultiPolígono.

Para ahondar más en la especificación de los MuliPolígonos referirse a la especificación [14].

Figura 3.4 Ejemplos de MultiPolígonos

4. Formas de implementación de la especificación

1.1 Usando los tipos numéricos de SQL para almacenamiento de las geometrías usando ODBC para el acceso.

1.2 Usando los tipos binarios de SQL para el almacenamiento de las geometrías usando ODBC para el acceso

2. Usando SQL92 con tipos geométricos para el almacenamiento de las características de las tablas ya sea de forma binaria o de texto.

Nota: el empleo de ODBC corresponde a que la especificación está pensada en el uso de ODBC/SQL

Tabla 3.1 Esquema de las características de las tablas.bajo SQL92

La implementación de OpenGIS bajo SQL92 define un esquema de almacenamiento de las tablas de características, geometrías y referencias espaciales de los sistemas de información. La implementación bajo SQL92 no define funciones de SQL para acceso, mantenimiento o indexación de las geometrías, ya que estas especificaciones no pueden ser implementadas de manera uniforme a través de los sistemas de bases de datos que emplean el estándar de SQL92.

La figura 3.1 describe el esquema de base de datos necesario para soportar el modelo simple de datos de OpenGIS. En dicho esquema se aprecian las relaciones existentes entre las tablas.

4.1 Tabla de características de las vistas de Metadatos (Feature Table Metadata Views)

Una tabla de características es una tabla que tiene una o más llaves foráneas que hacen referencia a las tablas de geometrías o de vistas. Un conjunto de tablas de características en una base de datos pueden ser creadas usando las tablas de la figura 3.1.

4.2 Columnas Geométricas de las vistas de Metadatos (Geometry Columns Metadata Views)

Esta tabla consiste de un renglón por cada columna de geometría de la base de datos. Los datos almacenados por cada geometría incluyen

- Identificador de la tabla de características de la cual es miembro
- Identificador del sistema espacial de referencia
- Un tipo de geometría por columna
- En que plano esta la información de la columna (1D,2D)
- Identificación de las tablas de geometrías que almacenan sus instancias
- Información necesaria para navegar en las tablas.

4.3 Información del Sistema de Referencia Espacial (Spatial Reference Systems)

Cada columna de las geometrías está asociada con un "Sistema de Referencia Espacial", la cual identifica el tipo de sistemas de coordendas para todas las geometrías almacenadas en las columnas y da significado a los valores geométricos para cualquier instancia de las geometrías almacenadas en la tabla.

La tabla de Información del Sistema de Referencia Espacial cuenta con los siguientes campos:

- SRID: Identificador del sistema de referencia espacial

- AUTH_NAME: Nombre de la autoridad del sistema de referencia espacial
- AUTH_SRID: Identificador del Sistema de Referencia Espacial de la Autoridad
- SR_TEXT: provee una representación textual del Sistema de Referencia Espacial

El SRID es la única llave de tipo enetro para todo el Sistema de Referencia Espacial de la base de datos.

4.4 Geometrías y elementos geométricos de las vistas

En la tabla se hace el almacenamiento de las geometrías y el formato de almacenamiento se da según el tipo de almacenamiento seleccionado, los cuales se encuentran en el punto 4 de este capítulo y sobre la implementación de todos los tipos de almacenamiento consultar la sección 3 de [ ].

5 Clases comunes o similares entre la jerarquía de clases de los Shapefiles y OpenGIS

En base a las descripciones de las clases tanto de los “shapefiles” y OpenGIS encontramos las siguientes clases comunes.

Tabla 3.2 Clases comunes entre los “Shapefiles” y OpenGIS

De la tabla 3.2 se aprecian las clases comunes en esencia entre los “Shapefiles” y OpenGIS, para mayor referencia entre las clases consultar el capítulo II y III de este documento.

Tabla 3.3 Clases de los “Shapefiles” que absorben clases de OpenGIS

La clase Polilínea de los “Shapefiles” es un caso general al igual que la clase LíneaString, que por consiguiente abarca los casos particulares de LinearRing y Línea.

La clase Polígono es una clase que por su estructura general abarca las estructuras de la clase Polígono y MultiPolígono.

.También es posible obtener información de manera indirecta de otras clases para cubrir huecos de las especificaciones como lo es buscar recuperar o generar nueva información a través de tomar información de varias clases o pedirsela directamente al usuario.

Conclusión

Aunque es posible recuperar toda la información que proveen los “Shapefiles”, no es posible cubrir todas los puntos necesarios para la especificación de OpenGIS, dentro los puntos importantes que se necesitan cubrir, se encuentran las unidades que tiene la información, la proyección y el color de los objetos.